Т. 12. №9 2026
https://doi.org/10.33619/2414-2948/130

УДК 004.415.2 https://doi.org/10.33619/2414-2948/130/17

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ СПЕЦИФИКАЦИИ ТРЕБОВАНИЙ К ИГРОВОМУ ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ

©Ельчанинов М. Н., *ORCID: 0009-0000-0383-7357, SPIN-код: 8808-0691,*
Российская академия народного хозяйства и государственной службы при Президенте Российской Федерации (РАНХиГС), г. Москва, Россия, slcrmmbr@yandex.com

COMPARATIVE ANALYSIS OF REQUIREMENTS SPECIFICATION METHODS FOR GAME SOFTWARE

©Elchaninov M., *ORCID: 0009-0000-0383-7357, SPIN-code: 8808-0691,*
Russian Presidential Academy of National Economy and Public Administration (RANEPA), Moscow, Russia, slcrmmbr@yandex.com

Аннотация. Цель работы состоит в сравнительном анализе существующих методов спецификации требований, применяемых при разработке игрового программного обеспечения. Актуальность исследования обусловлена тем, что до 65% проблем в разработке игр возникают на этапе предпроизводства по причине неоднозначных или неполных требований. Это указывает на концептуальный разрыв между творческим языком гейм-дизайнерской документации и формальными требованиями для программной реализации. Исследование выполнено методами анализа, сравнительного анализа и систематизации на основе 8 публикаций из баз данных IEEE Xplore, Springer, ACM Digital Library и eLIBRARY за период 2005–2022 годов. Рассмотрены методы формализации гейм-дизайнерской документации на основе стандарта IEEE 830 (шаблон iGDD), UML-моделирование игровых систем, гибкие методологии управления требованиями, а также нефункциональные требования, значимость которых существенно зависит от жанра игры. На основе сопоставления пяти методов по критериям степени формальности, применимости к нефункциональным требованиям, совместимости с гибкими методологиями и порога освоения установлено, что ни один из методов не является универсальным. Однако совместное применение iGDD и адаптированной методологии Scrum снижает нормализованный объём переработки с 11,50% до 2,73%. Сформулированы практические рекомендации по поэтапной трансляции творческих требований в формальные спецификации применительно к малым и средним студиям разработки.

Abstract. The aim of this study is to provide a comparative analysis of existing requirements specification methods applied in game software development. The relevance of the research is determined by the fact that up to 65% of problems in game development arise at the pre-production stage due to ambiguous or incomplete requirements. This indicates a conceptual gap between the creative language of game design documentation and the formal requirements needed for software implementation. The study was conducted using methods of analysis, comparative analysis, and systematization based on 8 publications from the IEEE Xplore, Springer, ACM Digital Library, and eLIBRARY databases for the period of 2005–2022. The paper examines methods of game design document formalization based on the IEEE 830 standard (the iGDD template), UML modeling of game systems, agile requirements management methodologies, and non-functional requirements, the significance of which significantly depends on the game genre. Based on a comparative analysis of five methods across the criteria of formality degree, applicability to non-functional requirements,

Т. 12. №9 2026
https://doi.org/10.33619/2414-2948/130

compatibility with agile methodologies, and learning threshold, it is established that no single method is universal. However, the application of iGDD combined with an adapted Scrum methodology reduces the normalized rework volume from 11.50% to 2.73%. Practical recommendations for the step-by-step translation of creative requirements into formal specifications are formulated for small and medium development studios.

Ключевые слова: спецификация требований, гейм-дизайнерская документация, программное обеспечение, IEEE 830, унифицированный язык моделирования, гибкая разработка, нефункциональные требования.

Keywords: requirements specification, game design document, software, IEEE 830, Unified Modeling Language, agile development, non-functional requirements.

Индустрия игрового программного обеспечения является одной из наиболее динамично развивающихся отраслей современной цифровой экономики. Совокупные продажи программных и аппаратных продуктов игровой индустрии превышают показатели кинематографической отрасли, что свидетельствует о масштабности и экономической значимости данного сектора [1].

Вместе с тем эмпирические исследования в области разработки игрового программного обеспечения фиксируют высокий уровень провалов проектов: как отмечается в работе, 65% проблем в разработке игр возникают на данном этапе и связаны с неспецифицированными или неоднозначными требованиями в гейм-дизайнерской документации [2].

Актуальность настоящего исследования определяется противоречием между растущей сложностью игровых программных систем и отсутствием устоявшейся методологии трансформации творческих проектных документов в формальные технические спецификации. Игровое программное обеспечение относится к классу сложных интерактивных программных комплексов реального времени, объединяющих мультимедийные подсистемы, подсистемы искусственного интеллекта, физического моделирования и пользовательского взаимодействия. Специфика данного класса систем предъявляет особые требования к процессу инженерии требований (Requirements Engineering), поскольку наряду с функциональными требованиями значительную роль играют субъективные нефункциональные характеристики: вовлечённость пользователя, игровой баланс, эстетическое восприятие.

Объектом исследования является процесс спецификации требований к игровому программному обеспечению. Предметом исследования выступают методы и подходы к формализации требований, обеспечивающие переход от гейм-дизайнерской документации к техническим спецификациям программного обеспечения.

Целью работы является аналитический обзор и сравнительный анализ существующих методов спецификации требований, применяемых при разработке игрового программного обеспечения.

Проблема перехода от творческой документации к инженерным спецификациям впервые была системно исследована в работе Callele, Neufeld и Schneider [1].

На основании анализа 50 ретроспективных отчётов о завершённых проектах (post-mortem) из журнала Game Developer авторы установили, что слабое управление переходом от этапа предпроизводства к этапу производства является источником значительной части проектных неудач. Авторы идентифицировали три уровня имплицитной информации в гейм-дизайнерских документах: информация, извлекаемая непосредственно из представленных материалов; информация, требующая общего знания предметной области; информация,

Т. 12. №9 2026
https://doi.org/10.33619/2414-2948/130

выводимая только при наличии знаний о целевой архитектуре реализации. Каждый последующий уровень требует привлечения всё более квалифицированных специалистов, что порождает организационные и экономические трудности.

Salazar, Mitre и соавт. предложили улучшенный шаблон гейм-дизайнерской документации (improved Game Design Document, iGDD), разработанный на основе сравнительного анализа существующих шаблонов ГДД с лучшими практиками спецификации требований к программному обеспечению по стандарту IEEE 830 [3]. De Lope и Medina-Medina исследовали применение диаграмм унифицированного языка моделирования для формализации игровых систем [4]. Mitre-Hernández и соавт. провели эмпирическую проверку комбинированного подхода, интегрирующего iGDD с адаптированной методологией Scrum, и показали статистически значимое снижение объёма переработки [2].

Paschali и соавт. выполнили обзор нефункциональных требований, влияющих на игровой опыт [5].

Franch, Glinz и соавт. провели масштабное исследование знания и использования стандартов инженерии требований в промышленности [6].

В отечественной литературе вопросы разработки игрового программного обеспечения рассматриваются в контексте управления проектами и выбора инструментальных средств. Рындина и Данилушкина исследовали применение референтных моделей бизнес-процессов для управления разработкой компьютерных игр [7].

Гордиенко выполнила обзор технологий, методов и инструментальных систем разработки программного обеспечения, включая вопросы спецификации требований на основе диаграмм и текстовых описаний [8].

При этом в русскоязычной научной литературе вопрос формализации гейм-дизайнерской документации с позиций инженерии требований остаётся недостаточно освещённым.

Характерной особенностью игрового программного обеспечения является наличие двух принципиально различных видов проектной документации. На этапе предпроизводства создаётся гейм-дизайнерская документация (ГДД), представляющая собой творческий документ, описывающий концепцию, механики и эстетику игры на естественном языке с использованием предметно-специфической терминологии. На этапе производства команде разработчиков необходима формальная спецификация требований к программному обеспечению (Software Requirements Specification, SRS), на основании которой формируются архитектурные решения, проводится верификация и планируются трудозатраты [1].

Callele и соавт. продемонстрировали масштаб информационной трансформации на конкретном примере: из одной страницы нарративного текста, описывающего посещение игроком персонажа для получения предмета, формируются требования к аватару игрока, неигровому персонажу, предмету инвентаря, состоянию игрового мира и условиям прогрессии. При этом детальная спецификация этих элементов (с учётом художественного стиля, анимации, звукового сопровождения) может легко достичь 50 страниц, когда включаются вопросы художественного стиля, анимации и состояний игрового мира [1].

Подход iGDD, предложенный Salazar, Mitre и соавт., структурирует содержание ГДД по пяти разделам: обзор, механика, динамика, эстетика, допущения и ограничения [3].

Каждый раздел сопоставлен с характеристиками качества SRS-документа по стандарту IEEE 830. Раздел «Механика» соответствует организации функциональных требований к объектам игры, раздел «Динамика» описывает требования к взаимодействию элементов и их соответствие профилю игрока, раздел «Допущения и ограничения» фиксирует технические ограничения и граничные условия системы. Данная структура обеспечивает прослеживаемость требований от творческого замысла до программной реализации.

Т. 12. №9 2026
https://doi.org/10.33619/2414-2948/130

Эмпирическая проверка подхода iGDD была проведена в рамках контролируемого эксперимента с участием двенадцати инженеров-программистов, разделённых на две группы. Группа, использовавшая iGDD совместно с адаптированной методологией Scrum, продемонстрировала нормализованный объём переработки 2,73%, тогда как контрольная группа, применявшая стандартный шаблон ГДД, показала 11,50% [2].

Статистическая значимость различий подтверждена критерием Вилкоксона ($p = 0,0085$). Авторы указывают, что определение разделов iGDD позволяет участникам прояснять информацию о целях, обоснованиях, особенностях игрового процесса и характеристиках игрока уже на ранних этапах, что упрощает интерпретацию требований на последующих стадиях разработки.

Применение унифицированного языка моделирования (Unified Modeling Language, UML) к игровым системам исследовано в работе De Lope и Medina-Medina [4]. Авторы предложили методологию, включающую несколько типов диаграмм, каждый из которых соответствует определённому аспекту игровой системы. Диаграммы прецедентов использования (Use Case Diagrams) моделируют взаимодействие игрока с системой и позволяют формализовать основные игровые действия в виде прецедентов со строго определёнными предусловиями, основным потоком событий и постусловиями. Диаграммы классов (Class Diagrams) описывают структуру игровых объектов и их отношения. Диаграммы состояний (State Diagrams) формализуют дискретное поведение игровых сущностей: боевые состояния персонажей, логику конечных автоматов неигровых персонажей, управление прогрессией. Диаграммы последовательностей описывают порядок обмена сообщениями между компонентами системы в рамках конкретных сценариев взаимодействия.

Практическая значимость диаграмм состояний подтверждается примером из работы Callele и соавт. [1]: описание одной головоломки (Pyramid Puzzle) в виде одностраничного документа гейм-дизайнера потребовало создания четырёх конечных автоматов для валидации пользовательского ввода и трёх конечных автоматов для отдельных частей головоломки, а также управления взаимодействием с состоянием игрового мира, состоянием игрока, инвентарём и подсистемой сохранения. Как отмечают авторы, ни один из этих элементов не был явно указан дизайнером; все они были имплицитно заложены в описании задачи.

Нефункциональные требования играют в игровом программном обеспечении принципиально иную роль, чем в традиционных программных системах. Paschali, Ampatzoglou и соавт. провели опрос более 110 игроков для оценки нефункциональных требований и выделили семь ключевых факторов, влияющих на удовлетворённость: сценарий (Scenario), графика (Graphics), звук (Sound), скорость игры (Game Speed), управление (Game Control), целостность персонажа (Character Solidness) и игровое сообщество (Community) [5].

Авторы подчёркивают, что приоритет этих нефункциональных требований строго зависит от жанра игры: для спортивных игр критична целостность персонажа, тогда как для ролевых игр на первый план выходят игровое сообщество и сценарий. Авторы заключают, что эти факторы не являются единообразными для всех компьютерных игр, а существенно зависят от жанра [5].

Данное наблюдение указывает на то, что в игровой инженерии различие функциональных и нефункциональных требований выражено менее чётко, чем в традиционных программных системах, поскольку многие нефункциональные требования реализуются через строго определённые функциональные механизмы: формулы расчёта урона, таблицы прогрессии, параметры физической симуляции. Вопрос применения стандартов инженерии требований в промышленной практике исследован в работе Franch, Glinz и соавт. на основании анкетирования 90 специалистов [6].

Т. 12. №9 2026
https://doi.org/10.33619/2414-2948/130

Результаты показали, что около 47% респондентов, работающих в роли инженеров по требованиям или бизнес-аналитиков, не знакомы с актуальным стандартом ISO/IEC/IEEE 29148, в то время как его предшественник IEEE 830-1998 по-прежнему остаётся наиболее известным и используемым стандартом в данной области. Авторы отмечают, что «участники исследования преимущественно используют стандарты по личному решению, а не по предписанию компании, заказчика или регулятора» (Participants in our study mostly use standards by personal decision rather than being imposed by their respective company, customer, or regulator) [6].

Данный результат указывает на разрыв между наличием нормативной базы и её фактическим применением, что особенно характерно для игровой индустрии, где формализация процессов традиционно уступает место итеративному экспериментированию.

В отечественной научной литературе вопросы управления разработкой игрового программного обеспечения рассмотрены Рындиной и Данилушкиной, предложившими применение референтных моделей бизнес-процессов для структурирования процесса разработки компьютерных игр [7].

Гордиенко выполнила обзор технологий и инструментальных систем разработки программного обеспечения, отмечая роль спецификаций в виде диаграмм и текстовых описаний для формулирования внешних требований и связей между моделями системы [8].

На основании проведённого анализа в Таблице представлено сравнение рассмотренных методов спецификации требований по выделенным критериям.

Таблица

СРАВНИТЕЛЬНЫЙ АНАЛИЗ МЕТОДОВ СПЕЦИФИКАЦИИ ТРЕБОВАНИЙ К ИГРОВОМУ ПРОГРАММНОМУ ОБЕСПЕЧЕНИЮ

Метод	*Степень формальности*	*Применимость к нефункциональным требованиям*	*Совместимость с Agile*	*Порог освоения*
Стандартный ГДД	низкая	ограниченная	слабая	низкий
iGDD (IEEE 830)	средняя	умеренная	высокая	средний
UML-моделирование	высокая	частичная	средняя	высокий
Agile-бэклог (User Stories)	низкая–средняя	умеренная	высокая	низкий
Формальные методы (конечные автоматы)	высокая	низкая	слабая	очень высокий

Из Таблицы следует, что ни один из рассматриваемых методов не является универсальным решением. Стандартный ГДД обеспечивает минимальный порог входа, поскольку составляется на естественном языке без необходимости владения инженерными нотациями, однако его низкая формальность порождает неоднозначность и имплицитность требований [1].

Применимость стандартного ГДД к нефункциональным требованиям ограничена: эстетические и эмоциональные аспекты фиксируются в виде общих пожеланий, не допускающих верификации. Совместимость с Agile оценена как слабая, так как монолитный документ ГДД плохо поддаётся декомпозиции на итерации и не предусматривает механизмов управления изменениями. Подход iGDD на основе IEEE 830 обладает средней степенью формальности: он структурирует содержание ГДД по разделам с определёнными характеристиками качества, но сохраняет естественный язык описания [3].

Применимость к нефункциональным требованиям умеренная: раздел «Эстетика» позволяет фиксировать часть таких требований, однако систематическая операционализация

Т. 12. №9 2026
https://doi.org/10.33619/2414-2948/130

субъективных требований в шаблоне не предусмотрена. Совместимость с Agile высокая, что подтверждено экспериментальной интеграцией iGDD с адаптированной методологией Scrum [2].

Порог освоения средний: от команды требуется знакомство со стандартом IEEE 830 и понимание принципов инженерии требований. UML-моделирование характеризуется высокой степенью формальности благодаря использованию стандартизированных нотаций с чётко определённой семантикой [4].

Применимость к нефункциональным требованиям частичная: диаграммы эффективно формализуют функциональное поведение системы, однако такие факторы, как графика, звук или целостность персонажа [5], не поддаются непосредственному представлению в виде UML-моделей. Совместимость с Agile средняя: диаграммы могут создаваться итеративно, но требуют значительных трудозатрат на поддержание актуальности при частых изменениях требований. Порог освоения высокий, так как корректное применение UML предполагает владение нотацией и навыки объектно-ориентированного проектирования.

Подход на основе бэклога пользовательских историй (User Stories), принятый в гибких методологиях, позволяет декомпозировать требования ГДД до атомарных задач разработки с определёнными критериями приёмки. Степень формальности оценена как низкая или средняя, поскольку пользовательские истории формулируются на естественном языке, но дополняются структурированными критериями приёмки. Применимость к нефункциональным требованиям умеренная: нефункциональные требования могут включаться в критерии приёмки, но не имеют выделенного механизма систематического учёта. Совместимость с Agile высокая по определению. Порог освоения низкий, так как формат пользовательских историй интуитивно понятен и не требует специализированной подготовки.

Формальные методы на основе конечных автоматов, применяемые для строгой верификации поведения игровых сущностей, обеспечивают высокую степень строгости. Применимость к нефункциональным требованиям низкая: конечные автоматы моделируют дискретное поведение системы и не предназначены для формализации субъективных качественных характеристик. Совместимость с Agile слабая, поскольку формальные модели трудно поддаются частым изменениям, свойственным итеративной разработке, и требуют полной переверификации при модификации требований. Порог освоения очень высокий: построение и верификация конечных автоматов предполагают глубокие знания в области теории автоматов и дискретной математики.

Оптимальным подходом для малых и средних студий разработки представляется поэтапная схема трансляции требований. На этапе предпроизводства разрабатывается ГДД с применением шаблона iGDD, обеспечивающего формализацию разделов «Механика» и «Динамика» в соответствии с характеристиками качества IEEE 830. Для ключевых игровых механик строятся UML-диаграммы прецедентов и диаграммы состояний. Нефункциональные требования операционализируются через измеримые метрики и включаются в критерии приёмки пользовательских историй. Управление изменениями требований осуществляется в рамках итеративного процесса с привязкой бэклога к разделам iGDD и проведением ревизии требований по завершении спринта с участием как геймдизайнеров, так и программистов.

Итак, проведён аналитический обзор методов спецификации требований к игровому программному обеспечению. Установлено, что основная проблема данной предметной области состоит в концептуальном разрыве между творческим языком геймдизайнерской документации и формальными требованиями, необходимыми для инженерии программного обеспечения. Данный разрыв является структурной причиной высокого уровня переработки в индустрии разработки игр. Анализ показал, что рассмотренные методы (формализация ГДД на

основе IEEE 830, UML-моделирование игровых систем, гибкие методологии управления требованиями) каждый в отдельности не решают проблему в полной мере, однако их комплексное применение позволяет существенно снизить неоднозначность требований. Эмпирические данные свидетельствуют о том, что применение улучшенной геймдизайнерской документации (iGDD) в сочетании с адаптированной методологией Scrum снижает нормализованный объём переработки более чем в четыре раза. Дальнейшее развитие данного направления представляется перспективным в части разработки автоматизированных средств трансформации геймдизайнерской документации в формальные спецификации, а также в части создания метрик для количественной оценки нефункциональных требований, специфичных для игровых систем. Систематическое внедрение методов инженерии требований в процесс разработки игрового программного обеспечения является необходимым условием повышения качества и снижения стоимости программных продуктов данного класса.

Список литературы:

1. Callele D., Neufeld E., Schneider K. Requirements engineering and the creative process in the video game industry // 13th IEEE International Conference on Requirements Engineering (RE'05). IEEE, 2005. P. 240-250. https://doi.org/10.1109/RE.2005.58

2. Mitre-Hernández H. A., Lara-Alvarez C., González-Salazar M., Martín D. Decreasing rework in video games development from a software engineering perspective // Trends and Applications in Software Engineering: Proceedings of the 4th International Conference on Software Process Improvement CIMPS'2015. Cham: Springer International Publishing, 2015. P. 295-304. https://doi.org/10.1007/978-3-319-26285-7_25

3. Salazar M. G., Mitre H. A., Olalde C. L., Sánchez J. L. G. Proposal of Game Design Document from software engineering requirements perspective // 2012 17th International Conference on Computer Games (CGAMES). IEEE, 2012. P. 81-85.

4. De Lope R. P., Medina-Medina N. Using UML to model educational games // 2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES). IEEE, 2016. P. 1-4. https://doi.org/10.1109/VS-GAMES.2016.7590373

5. Paschali M. E., Ampatzoglou A., Chatzigeorgiou A., Stamelos I. Non-functional requirements that influence gaming experience: a survey on gamers satisfaction factors // Proceedings of the 18th International Academic MindTrek Conference. ACM, 2014. P. 195–202. https://doi.org/10.1145/2676467.2676471

6. Franch X., Glinz M., Mendez D., Seyff N. A study about the knowledge and use of requirements engineering standards in industry // IEEE Transactions on Software Engineering. 2021. V. 48. №8. P. 2901–2916. https://doi.org/10.1109/TSE.2021.3087792

7. Рындина С. В., Данилушкина Э. И., Плаксина В. С. Управление разработкой компьютерных игр на основе референтных моделей бизнес-процессов // Модели, системы, сети в экономике, технике, природе и обществе. 2019. №4. С. 32-39.

8. Гордиенко Е. П. Обзор технологий, методов и инструментальных систем разработки программного обеспечения // Актуальные проблемы и перспективы развития транспорта, промышленности и экономики России (ТрансПромЭк 2021). 2021. С. 23-29.

References:

1. Callele, D., Neufeld, E., & Schneider, K. (2005, August). Requirements engineering and the creative process in the video game industry. In *13th IEEE International Conference on Requirements Engineering (RE'05)* (pp. 240-250). IEEE. https://doi.org/10.1109/RE.2005.58

2. Mitre-Hernández, H. A., Lara-Alvarez, C., González-Salazar, M., & Martín, D. (2015, October). Decreasing rework in video games development from a software engineering perspective. In *Trends and Applications in Software Engineering: Proceedings of the 4th International Conference on Software Process Improvement CIMPS'2015* (pp. 295-304). Cham: Springer International Publishing. https://doi.org/10.1007/978-3-319-26285-7_25

3. Salazar, M. G., Mitre, H. A., Olalde, C. L., & Sánchez, J. L. G. (2012, July). Proposal of Game Design Document from software engineering requirements perspective. In *2012 17th International Conference on Computer Games (CGAMES)* (pp. 81-85). IEEE.

4. De Lope, R. P., & Medina-Medina, N. (2016, September). Using UML to model educational games. In *2016 8th International Conference on Games and Virtual Worlds for Serious Applications (VS-GAMES)* (pp. 1-4). IEEE. https://doi.org/10.1109/VS-GAMES.2016.7590373

5. Paschali, M. E., Ampatzoglou, A., Chatzigeorgiou, A., & Stamelos, I. (2014). Non-functional requirements that influence gaming experience: a survey on gamers satisfaction factors. In *Proceedings of the 18th International Academic MindTrek Conference, ACM,* 195–202. https://doi.org/10.1145/2676467.2676471

6. Franch, X., Glinz, M., Mendez, D., & Seyff, N. (2021). A study about the knowledge and use of requirements engineering standards in industry. *IEEE Transactions on Software Engineering, 48*(8), 2901–2916. https://doi.org/10.1109/TSE.2021.3087792

7. Ryndina, S. V., Danilushkina, Eh. I., & Plaksina, V. S. (2019). Upravlenie razrabotkoj komp'yuternyh igr na osnove referentnyh modelej biznes-processov. *Modeli, sistemy, seti v ehkonomike, tehnike, prirode i obshchestve,* (4), 32-39. (In Russian).

8. Gordienko, E. P. (2021). Obzor tehnologij, metodov i instrumental'nyh sistem razrabotki programmnogo obespecheniya. *Aktual'nye problemy i perspektivy razvitiya transporta, promyshlennosti i ehkonomiki Rossii (TransPromEhk 2021),* 23-29. (In Russian).

Поступила в редакцию
01.07.2026 г.

Принята к публикации
11.07.2026 г.

Ссылка для цитирования:

Ельчанинов М. Н. Сравнительный анализ методов спецификации требований к игровому программному обеспечению // Бюллетень науки и практики. 2026. Т. 12. №9. С. 168-175. https://doi.org/10.33619/2414-2948/130/17

Cite as (APA):

Elchaninov, M. (2026). Comparative Analysis of Requirements Specification Methods for Game Software. *Bulletin of Science and Practice, 12*(9), 168-175. (in Russian). https://doi.org/10.33619/2414-2948/130/17