УДК 004.054

РОЛЬ ФОРМАЛИЗАЦИИ ТРЕБОВАНИЙ В ТЕСТИРОВАНИИ ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ

Бахвалова Зинаида Андреевна
к.т.н., доцент центра программной инженерии
Института информационных технологий и анализа данных
e-mail: zinand@inbox.ru
Иркутский национальный исследовательский технический университет,
664074, Российская Федерация, г. Иркутск, ул. Лермонтова, 83.
Камышова Елена Александровна,
Специалист по обеспечению качества отдела разработки и тестирования,
АО ISPsystem,,
e-mail leka96@mail.ru,

Аннотация. В статье рассмотрена роль формализации требований в тестировании программного обеспечения (ПО). Исследованы понятия «требование к ПО» и «тестирование ПО», сделан вывод об их тесной взаимосвязи. На основе выполненного анализа влияния некачественных требований на процесс тестирования сделан вывод о важности высокого качества требований для процесса разработки ПО. Предложено формализовать требования для повышения их качества. Рассмотрены несколько методов формализации требований, по каждому из которых сделан вывод о влиянии подобной формализации на ручное и автоматизированное тестирование. Выявлено, что рассмотренные методы предлагают формализацию поведения системы, но с точки зрения упрощения автоматизации тестирования (а именно подготовки наборов данных для тестов) формальная модель должна описывать структуру данных. Подобная модель может быть организована в виде иерархии.

Ключевые слова: требования к ПО, тестирование ПО, формализация требований, качество требований/

Цитирование: Бахвалова З.А., Камышова Е.А.. Роль формализации требований в тестировании программного обеспечения // Информационные и математические технологии в науке и управлении. 2020. № 1 (17). С. 120–129. DOI: 10.38028/ESI.2020.17.1.009

Введение. Неотъемлемой составляющей процесса создания программного обеспечения (ПО) является разработка требований, которые непосредственно влияют как на выполнение дальнейших этапов создания ПО, таких, как проектирование, кодирование и т.д., так и на качество итогового продукта, оценить которое позволяет тестирование ПО.

Однако требования не всегда являются качественными, то есть не отражают целевое состояние разрабатываемого ПО. Проблема формирования качественных требований является важной, поскольку любые ошибки на этом этапе, в зависимости от своевременности их обнаружения, могут иметь значительные последствия для самого ПО, заказчика и разработчика. Кроме того, чем ниже качество требований, тем сложнее и дольше проходит

процесс тестирования, как ручного, так и автоматизированного: растет время, затрачиваемое тестировщиком, повышается количество возвратов для исправления и доработки задач

В статье рассматриваются стандарты и взгляды различных авторов на определение таких ключевых понятий, как «требования» и «тестирование», и на основании анализа этих источников формулируются характерные свойства указанных терминов. Вследствие этого анализа выявляется зависимость тестирования от требований, и впоследствии эта связь рассматривается более подробно, как с точки зрения влияния хорошо сформированных требований на тестирования, так и с обратной – как некачественные требования влияют на процесс обеспечения качества программных продуктов.

В качестве одного из методов повышения качества требований рассматривается их формализация и анализируются различные подходы к ней, в результате чего выявляется отсутствие подхода, который можно было бы использовать для описания требований в входным данным. Поэтому выдвигается предположение о том, что можно разработать формальную модель, направленную на описание входных данных, и приводится простейший пример структуры таких требований.

Основные понятия. Единого определения, объясняющего термин «требования к программному обеспечению » не существует, несмотря на важность этого понятия.

Согласно С.Куликову, «требование – это описание того, какие функции и с соблюдением каких условий должно выполнять приложение в процессе решения полезной для пользователя задачи» [5; стр. 28], особенно выделяется роль пользователя и его задач.

Международная организация по сертификации тестирования программного обеспечения ISTQB в своем терминологическом словаре приводит следующее определение требований: «Это условия или возможности, необходимые пользователю для решения определенных задач или достижения определенных целей, которые должны быть достигнуты для выполнения контракта, стандартов, спецификации, или других формальных документов» [3].

В книге Карла И. Вигерса установлено, что требования должны подразделяться на типы: «Требования – это спецификация того, что должно быть реализовано. В них описано поведение системы, свойства системы или ее атрибуты. Они могут быть ограничены процессом разработки системы» [2; стр. 6].

Д. Леффенгуэл и Д. Уидриг, в свою очередь, определяют требование как «возможность, которую должна обеспечивать система; некое свойство ПО, необходимое пользователю для решения проблемы при достижении поставленной цели; некое свойство ПО, которым должна обладать система или её компонент, чтобы удовлетворить требования контракта, стандарта, спецификации либо иной формальной документации» [6].

Анализ приведенных определений позволяет установить, что универсальной трактовки термина «требование к ПО» не существует, однако можно выделить следующие общие черты:

- требование может являться условием, возможностью (функцией) или, напротив, ограничением;
- требование должно способствовать решению полезной для пользователя задачи;
- требования должны быть представлены в виде документов и связаны с другими документами: контрактами, стандартами, спецификациями и т.д.

Термин «тестирование программного обеспечения» также не является однозначно сформулированным.

Например, С. Куликов предлагает следующее определение: «тестирование программного обеспечения – процесс анализа программного средства и сопутствующей документации с целью выявления дефектов и повышения качества продукта» [5; стр. 5].

С. Канер предлагает похожее определение: «тестирование — это процесс поиска ошибок» [4; стр. 39]. В то же время автор критикует другое распространенное определение: «тестирование – проверка правильности работы программы». По его мнению, такое определение является бессмысленным, так как заключение о том, что программа работает правильно, сделать невозможно.

В глоссарии ISTQB такого термина, как «тестирование программного обеспечения» нет, так как он используется в русском языке. Глоссарий содержит только термин «тестирование (testing)»: это процесс, содержащий в себе все активности жизненного цикла, как динамические, так и статические, касающиеся планирования, подготовки и оценки программного продукта и связанных с этим результатов работ с целью определить, что они соответствуют описанным требованиям, показать, что они подходят для заявленных требований и для определения дефектов [3].

Другой стандарт, ISO/IEC TR 19759:2005, определяет тестирование как процесс исследования, испытания программного продукта, имеющий своей целью проверку соответствия между реальным поведением программы и её ожидаемым поведением на конечном наборе тестов, выбранных определенным образом [12].

На основе всех приведенных выше определений можно выделить схожие, ключевые элементы и сформулировать основные тезисы, определяющие понятие «тестирование ПО»:

- тестирование заключается в анализе реализованного поведения программного продукта на соответствие ожиданиям (требованиям, которые были описаны до начала разработки);
- цель тестирования состоит не в проверке правильности работы программы, а в поиске дефектов, при этом программа должна быть для этого пригодна;
- процесс непосредственного тестирования сопровождается предварительным планированием, проектированием, а также последующей оценкой выполненной работы.

Следует обратить внимание на то, что тестирование ПО напрямую связано с требованиями, о чем говорится по крайней мере в одном из приведенных определений.

Влияние некачественных требований на тестирование. Согласно С. Куликову, в зависимости от модели разработки ПО, выбранной для проекта, тестирование может появиться на различных этапах: при «водопадной» модели тестирование следует начинать с середины проекта, в то время как при итерационной – в определенные моменты каждой итерации на протяжении всего процесса реализации [5; стр. 24]. При этом зачастую к тестированию приступают уже на этапе формирования требований, на основании которых определяются метрики качества и составляется тест-план [5; стр. 19]. Соответственно, от качества требований будет зависеть качество тестирования и, как следствие, качество итогового продукта.

При этом необходимо принимать во внимание, что, несмотря на отсутствие однозначной системы классификации ПО, имеется несколько общепринятых классов программных продуктов, и, следовательно, при разработке требований необходимо учитывать, что важность каждой характеристики требований меняется в зависимости от

класса продукта [8]. Например, «безопасность» наиболее важна для программного обеспечения, работающего с важными пользовательскими данными, и в этом случае тестировщик должен будет провести тестирование безопасности. Чем правильнее определены приоритеты характеристик и назначение продукта при формировании требований, тем полнее и эффективнее будет тестирование [11].

В таблице 1 рассмотрены риски, связанные с требованиями к ПО, которые выделяет К. Вигерс [2, стр. 16], и их последствия при последующем тестировании.

В соответствии с таблицей можно сделать вывод, что процессу разработки требований необходимо уделять особое внимание, так как он непосредственно влияет на последующий процесс тестирования.

В свою очередь, формирование высококачественных требований, в соответствии с К. Вигерсом [2; стр. 21], имеет следующий ряд преимуществ:

- уменьшается количество дефектов в самих требованиях;
- уменьшается количество исправлений кода и повторного тестирования;
- ненужные функции не реализуются и не тестируются;
- стоимость модификации продукта снижается за счет предварительной оценки этой модификации;
- процесс разработки и тестирования ускоряется;
- уменьшается разобщенность требований;
- границы требований указываются более точно;
- уменьшается беспорядок в проекте;
- оценки тестирования становятся точнее; повышается удовлетворенность заказчиков и разработчиков.

Таблица 1. Последствия некачественных требований для тестирования ПО

Риски	Характеристика	Последствия для тестирования
Недостаточное вовлечение пользователей	Зачастую заказчики не понимают, для чего необходимо вовлекать конечных пользователей в процесс разработки, предполагая, что разработчики знают их потребности, а разработчики предпочитают написание кода общению с клиентами	Вследствие того, что тестировщик проверяет программный продукт на соответствие описанным требованиям, он зачастую может одобрить ПО, которое не соответствует ожиданиям пользователя. Если же он выявит ошибки в требованиях, это приведет к задержке завершения проекта и повтору уже пройденных этапов разработки
«Разрастание» требований пользователей	Объем требований постепенно увеличивается, проект может выйти за установленные рамки сроков и бюджета, что не сразу может быть обнаружено на этапе формирования требований	В связи с выросшим объемом требований растет и объем тестирования, что может привести к превышению сроков и бюджета (при добавлении новых требований тестированию подвергаются не только вносимые изменения, но и связанные с ними функции)
Двусмысленность требований	Пользователь имеет возможность интерпретировать одно и то же положение требований по-разному, при разработке программисты зачастую восполняют пробелы в соответствии со своим личным представлением, которое может быть ошибочным	Если тестировщик поймет требования так же, как и программист, ошибка не будет замечена, и пользователь может получить некачественный продукт. Если тестировщик поймет требования иначе и предложит внести исправления, то будет потрачено дополнительное время на исправление и повторное тестирование

«Золочение» продукта	Разработчики добавляют функции, которых нет в спецификации, но им кажется, что это понравится пользователям	Тестировщик тратит дополнительное время на проверку тех функций, которые внес разработчик, или предлагает убрать их в соответствии со спецификацией, вследствие чего будет потрачено время на исправление и повторное тестирование
Минимальная спецификация	Вместо полноценной спецификации создается сокращенный вариант, который дорабатывается разработчиками по их представлениям	Тестировщик вынужден выяснять у разработчика подробности добавленных функций, так как по урезанной спецификации он не способен качественно проверить продукт
Небрежное планирование	Пока требования не детализированы и не проанализированы, нельзя верно оценить время на их выполнение	При небрежно оцененном требовании время на тестирование, как и на разработку, будет превышено, и заказчик останется недоволен
Пропуск классов пользователей	Требования зачастую разрабатываются для групп пользователей, имеющих определенный опыт работы с ПО и могут применять различные наборы функций. Иногда эти классы могут быть пропущены	Тестировщик может заметить пропущенные группы, если имеет соответствующий опыт. В этом случае требования будут перепроектированы, код дополнен, и тестирование будет повторено заново. В противном случае продукт будет выпущен с неполной функциональностью

Формализация требований. Одним из способов формирования качественных требований может являться их формализация: представление в виде некоторого математического формализма, которое часть свойств требований обеспечивает за счет выбранной формы их представления, а другую часть позволяет проконтролировать, затратив приемлемые усилия [10]. Иначе говоря, это преобразование вербальной модели требований в формальную модель [7].

Формализация требований позволит снизить вероятность возникновения следующих рисков:

- «разрастание» требований пользователей – вследствие необходимости подробно описывать все новые требования в соответствии с определенной моделью, чрезмерное «разрастание» может быть замечено ранее;
- двусмысленность требований – формализация обеспечивает полноту, однозначность и непротиворечивость требований;
- «золочение» продукта – поскольку почти каждая деталь ПО может быть описана формализованными требованиями, более вероятно, что разработчик будет придерживаться описанных функций;
- минимальная спецификация – формализация предусматривает подробное полное описание продукта.

Кроме того, формализация требований окажет положительное влияние на тестирование, а именно поможет упростить работу тестировщика и повысить скорость разработки тестов.

При исследовании вопроса формализации требований можно встретить различные подходы.

Ю. Липко описывает универсальный подход к формализации, а именно преобразование требований к программному продукту в промежуточную модель, которая затем трансформируется в формальную [7]. В частности, в своей статье она описывает правила такого преобразования. Предлагаемый ею алгоритм формализации можно описать следующим образом:

1. Трансформация предложений на формальном языке в табличную форму следующего вида: словосочетание в начале предложения, подлежащее, сказуемое, дополнение, словосочетание в конце предложения.
2. Заполнение пустых ячеек таблицы в соответствии с определенными правилами.
3. Составление листа акторов – лиц, взаимодействующих с программным продуктом.
4. Добавление в лист акторов действий в порядке их появления в тексте.
5. Проектирование графа элементов и добавление путей между последовательно выполняемыми действиями.

Таким образом, удается осуществить формализацию требований путем их приведения к виду графа. Такой подход следует использовать в проектах, для которых характерно частое изменение требований, отслеживание выполнения требований или привязка к ним задач, людей, кода и т.д. Однако графы не всегда удобно использовать для автоматизации: они могут быть применены для описания поведения автоматизированного теста. но не приспособлены для описания данных, с которыми работает этот тест.

Такой метод формализации также может повлиять на составление тест-кейсов для ручного тестирования программного обеспечения. Тест-кейс – это набор входных данных, условий выполнения и ожидаемых результатов, разработанный с целью проверки того или иного свойства или поведения программного средства [5; стр. 114]. Чем более подробно это свойство или поведение описывается требованиями к продукту, тем больше проверок может быть выполнено и тем меньше вероятность упустить какие-либо дефекты ПО.

С. Минюров предлагает более узконаправленный подход к формализации требований на основе базы знаний [9]. Автор утверждает, что результатом разработки требований должен являться не набор утверждений на естественном языке, а набор связей между информационными элементами. Единицей информации становится не слово или предложение, а именно связь. Для удобства хранения и обработки информации должна быть разработана особая структура; каждое неформализованное требование должно подвергаться семантическому анализу. С. Минюров предлагает структуру, состоящую из двух частей: атрибутов (необходимы для упорядоченного хранения данных, таких, как тип информационного элемента, описание, источник, дата) и связей (необходимы для взаимосвязи данного требования с другими: описание связанных процессов, атрибутов, правил). Однако автор указывает, что структура требований может отличаться в зависимости от проекта, различной степени детализации и формализации данных. Также он отмечает, что разные члены команды по разработке ПО могут структурировать требования по-разному, вследствие чего может возникнуть проблема несогласованности и дублирования данных.

Метод формализации путем приведения требований к виду базы знаний может быть использован, например, для описания интерфейса ПО и, как следствие, тестирования самого интерфейса или его прототипов, элементы которых могут быть описаны в виде иерархической модели. Однако метод, который предлагает автор, описывает поведение системы, но не структуру данных. Поэтому он может быть использован для автоматизации тестирования, но только в виде сценария для написания тестов. Для того, чтобы данный подход мог использоваться для автоматизации тестирования с минимальными усилиями на подготовку тестовых наборов путем разделения входных данных на классы эквивалентности, его необходимо преобразовать для создания иерархической структуры другого типа. Такая формализованная модель должна определять структуру данных, то есть конкретные значения

любых параметров, при которых ПО должно выполнять запросы или, напротив, обрабатывать ошибочные действия.

Еще один автор, А. Веряев, посвятил свою статью вопросам формализации требований безопасности информации к средствам анализа защищенности [1]. Несмотря на то, что его работа является узкоспециализированной, его подход также следует рассмотреть. Автор подразделяет информационные системы на классы в зависимости от требуемой защиты, а также выделяет основные функциональные возможности средств анализа защищенности и присваивает им уникальные обозначения. Таким образом, любую систему подобного типа можно определенно описать с указанием класса защиты и перечня стандартных функциональных возможностей. Такой подход удобно использовать для описания определенного класса систем, обладающих схожей функциональностью. Соответственно, для каждого класса таких систем могут быть составлены стандартные чек-листы (структурированные наборы идей для тестирования), которые с небольшими дополнениями будут применимы к любым системам соответствующего класса. Это позволит ускорить процесс тестирования и снизить вероятность упущения тех или иных проверок.

Авторами предлагается, в частности, формальная модель, которая должна описывать поля, предназначенные для ввода входных данных, а также взаимосвязи этих полей между собой – например, если одни входные данные зависят от других, или если несколько входных данных должны обладать одинаковыми параметрами.

В общем виде формализованные требования к полю могут иметь следующий вид:

```
"Поле - общее описание": {
   "Тип": "поле",
   "Группа": "имя_группы",
   "Зависимое": "статус_зависимости",
   "Обязательное": "статус_обязательности",
   "Формат": "тип_поля",
   "Значение поля": "значение_поля"
 }
```

Имя поля является его идентификатором среди других полей (указывается перед фигурными скобками). Тип элемента необходимо указывать, так как, помимо полей, описанию подлежат и другие сущности. Поля могут быть зависимыми от других полей (то есть значение данного поля зависит от значения другого поля) или независимыми; обязательными для ввода и необязательными (поле может принимать пустое значение). Значения в поле могут быть заданы в каком-то определенном формате, помимо того, могут быть наложены различные ограничения. В дальнейшем модель описания требований должна расширяться в зависимости от выбранного формата вводимых данных: строка, число, файл и т.д.

Заключение. Рассмотрев несколько методов формализации и их влияние на процесс тестирования, можно сделать вывод, что формализация требований не только повышает их качество и снижает вероятность возникновения рисков, связанных с неточной формулировкой требований, но также положительно влияет на работу тестировщика, снижая затрачиваемое им время, в то же время повышая скорость разработки чек-листов и тест-кейсов. При этом необходимо отметить, что все перечисленные методы применяются для описания поведения системы, и их сложно применять для описания входных данных этой системы. Однако возможно разработать такую формальную модель требований, которая позволит значительно

упростить и ускорить подготовку входных данных для автоматизированного тестирования (например, осуществлять автоматическое преобразование требований в готовые тестовые наборы) (фрагмент формальной модели, предложенной авторами, приведен выше)

СПИСОК ЛИТЕРАТУРЫ

1. Веряев А.С., Фадин А.А. Формализация требований безопасности информации к средствам анализа защищенности // Вопросы кибербезопасности. 2015. №4(12). С. 23-27.
2. Вигерс Карл. Разработка требований к программному обеспечению / Пер, с англ. М.: Издательско-торговый дом «Русская Редакция». 2004. 576 с.
3. Глоссарий международной организации по сертификации тестирования программного обеспечения ISTQB. Режим доступа: http://glossary.istqb.org/ (дата обращения: 03.01.2018).
4. Канер Сэм, Джек Фолк, Енг Кек Нгуен. Тестирование программного обеспечения. Фундаментальные концепции менеджмента бизнес-приложений: Пер. с англ. Киев: Издательство «ДиаСофт». 2001. 544 с.
5. Куликов С.С. Тестирование программного обеспечения. Базовый курс: практическое пособие. Минск: Четыре четверти, 2015. 294 с.
6. Леффенгуэл Д., Уидриг Д. Принципы работы с требованиями к программному обеспечению. Унифицированный подход.: Пер. с англ.- М.: «Вильямс», 2002. 448с.
7. Липко Ю.Ю. Алгоритм формализации требований при разработке информационных систем [Текст] / Ю.Ю. Липко // Известия Южного федерального университета. Технические науки. 2014. №6(155). С. 153-158.
8. Мейрманова Д.М. Процессы управления качеством программного обеспечения [Текст] / Д.М. Мейрманова // Наука, образование и культура. 2016. №9(12). С. 11-16.
9. Минюров С., Минюрова А. Формализация требований на основе базы знаний. Разработка требований для программного обеспечения. М. 2012. 55 с.
10. Росс Дугласс Т. Качество начинается с определения требований // Проблемы информатики. 2017. №2(35). С. 65-70.
11. Формализация требований на практике. В.В. Кулямин и др. [Электронный ресурс] // Сайт Института системного программирования им. В.П. Иванникова РАН. Режим доступа:: http://panda.ispras.ru/~kuliamin/docs/Req-2006-ru.pdf (дата обращения: 05.01.2018).
12. ISO/IEC TR 19759:2005. Software Engineering – Guide to the Software Engineering Body of Knowledge.

UDK 004.054

THE ROLE OF REQUIREMENTS FORMALIZATION IN SOFTWARE TESTING

Zinaida A. Bakhvalova
PhD, associate professor of Software Engineering Center,
Irkutsk National Research Technical University,
83 Lermontov Str., Irkutsk 664074, Russian Federation
e-mail zinand@inbox.ru,
Elena A. Kamyshova,
Quality Assurance specialist of development and testing department,
ISPsystem Hosting Software
125, Dekabrskih sobytiy st., Irkutsk 664007 Russian Federation
e-mail leka96@mail.ru,

Abstract. The role of formalization of requirements in software testing is considered in this article. The authors investigate such terms as “requirement to the software” and “testing of the software” and point out their interrelation. The analysis of the influence of low-quality requirements on software testing process is made and its results obtained show that the quality of requirements is important for all software development process. The requirements formalization is offered as a way of improvement of their quality. Several methods of formalization of requirements are considered, on each of them the conclusion is drawn on influence of the formalization on the manual and automated testing, but the considered methods offer formalization of behavior of system. The formal model has to describe the structure of data from the point of view of testing automation simplification (regarding preparation of data sets for tests). The similar model can be organized in the form of hierarchy.

Keywords: requirements to the software, software testing, requirements formalization, quality of requirements.

References

1. Veryaev A.C. Formalizatsiya trebovaniy bezopasnosti informatsii k sredstvam analiza zashchishchennosti [Security Requirements Formalization for Security Assessment Tools] // Voprosi ciberbezopasnosti = Cybersecurity issues]. 2015. №4(12). Pp. 23–27 (in Russian).
2. Vigers Karl. Razrabotka trebovaniy k programmnomu obespecheniyu [Development of software requirements] //M.: Izdatel'sko-torgovyy dom «Russkaya Redaktsiya»= Publishing and trading house "Russian Edition". 2004. 576 p. (in Russian).
3. Glossariy mezhdunarodnoy organizatsii po sertifikatsii testirovaniya programm-nogo obespecheniya [ISTQB Glossary]. Availabe at http://glossary.istqb.org/ (accessed 03.01.2020)
4. Kaner Sem, Dzhek Folk, Yeng Kek Nguyen. Testirovaniye programmnogo obespecheniya. Fundamental'nyye kontseptsii menedzhmenta biznes-prilozheniy [Software testing. Fundamental Business Application Management Concepts]. Kiyev: Izdatel'stvo «DiaSoft»= DiaSoft Publishing House. 2001. 544 p. (in Russian).
5. Kulikov C.C. Testirovaniye programmnogo obespecheniya [Software testing]. // Minsk: Chetyre chetverti= “Four quarters”, 2015. 294 p. (in Russian).
6. Leffenguel D., Uidrig D. Printsipy raboty s trebovaniyami k programmnomu obes-pecheniyu. Unifitsirovannyy podkhod [Principles of working with software requirements. A unified approach]. // M.: «Vil'yams»= "Williams", 2002. 448 p. (in Russian).

7. Lipko YU.YU. Algoritm formalizatsii trebovaniy pri razrabotke informatsionnykh sistem [An algorithm for formalizing requirements in the development of information systems]. // Izvestiya Yuzhnogo federal'nogo universiteta. Tekh-nicheskiye nauki = Bulletin of the Southern Federal University. Technical science. 2014. №6 (155). Pp. 153-158. (in Russian).
8. Meyrmanova D.M. Protsessy upravleniya kachestvom programmnogo obespecheniya [Software Quality Management Processes]. // Nauka, obrazovaniye i kul'tura = Science, Education and Culture . 2016. №9 (12). Pp. 11-16. (in Russian).
9. Minyurov C. Formalizatsiya trebovaniy na osnove bazy znaniy. Razrabotka trebova-niy dlya programmnogo obespecheniya [Formalization of requirements based on a knowledge base. Development of software requirements]. Availabe at https://minyurov.files.wordpress.com/2014/02/formal_sw_rqr.pdf (accessed 13.02.2020)
10. Ross Duglass T. Kachestvo nachinayetsya s opredeleniya trebovaniy [Quality begins with defining requirements]. // Problemy informatiki = Problems of Computer Science. 2017. №2 (35). Pp. 65-70.
11. Formalizatsiya trebovaniy na praktike. V.V. Kulyamin i dr. [Formalization of requirements in practice]. Availabe at http://panda.ispras.ru/~kuliamin/docs/Req-2006-ru.pdf (accessed: 05.01.2020).
12. ISO/IEC TR 19759:2005. Software Engineering – Guide to the Software Engineering Body of Knowledge.