Министерство образования и науки Российской Федерации

федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт ИнЭО
Направление подготовки Информатика и вычислительная техника
Кафедра Автоматики и компьютерных систем

БАКАЛАВРСКАЯ РАБОТА

Тема работы
Автоматизированное тестирование пользовательского интерфейса крос-сплатформенного мобильного приложения "Machining Cloud"

УДК 004.512:004.451:004.415.53

Студент

Группа	ФИО	Подпись	Дата
3-8В2В3	Брагина Ольга Олеговна		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Цапко Сергей Геннадьевич	К.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Рахимов Тимур Рустамович	К.э.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Инженер	Маланова Наталья Викторовна	К.т.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

И.о. зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АиКС	Суходоев Михаил Сергеевич	К.т.н.		

Томск, 2017

ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЕ ПО ООП

Код результата	Результат обучения (выпускник должен быть готов)
Профессиональные компетенции	
ПК1	Разрабатывать бизнес-планы и технические задания на оснащение отделов, лабораторий, офисов компьютерным и сетевым оборудованием.
ПК2	Осваивать методики использования программных средств для решения практических задач
ПК3	Разрабатывать интерфейсы "человек - электронно-вычислительная машина".
ПК4	Разрабатывать модели компонентов информационных систем, включая модели баз данных.
ПК5	Разрабатывать компоненты программных комплексов и баз данных, использовать современные инструментальные средства и технологии программирования.
ПК6	Обосновывать принимаемые проектные решения, осуществлять постановку и выполнять эксперименты по проверке их корректности и эффективности.
ПК7	Готовить презентации, научно-технические отчеты по результатам выполненной работы, оформлять результаты исследований в виде статей и докладов на научно-технических конференциях.
ПК8	Готовить конспекты и проводить занятия по обучению сотрудников применению программно-методических комплексов, используемых на предприятии.
ПК9	Участвовать в настройке и наладке программно-аппаратных комплексов.
ПК10	Сопрягать аппаратные и программные средства в составе информационных и автоматизированных систем.
ПК11	Инсталлировать программное и аппаратное обеспечение для информационных и автоматизированных систем.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт ИнЭО
Направление подготовки Информатика и вычислительная техника
Кафедра Автоматики и компьютерных систем

УТВЕРЖДАЮ:
Зав. кафедрой

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

Бакалаврской работы

Студенту:

Группа	**ФИО**
З-8В2В3	Брагина Ольга Олеговна

Тема работы:

Автоматизированное тестирование пользовательского интерфейса крос-сплатформенного мобильного приложения "Machining Cloud"	
Утверждена приказом директора (дата, номер)	

Срок сдачи студентом выполненной работы:	

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе	*Требования от заказчика на автоматизацию тестирования программного обеспечения*
Перечень подлежащих исследованию, проектированию и разработке вопросов	*1) Анализ области автоматизации тестирования программного обеспечения;* *2) Выбор инструмента для автоматизации тестирования пользовательского интерфейса;* *3) Разработка проекта по автоматизации тестирования;* *5) Написание фреймворков под платформу iOS и Android с набором функций, реализующих взаимодействие с интерфейсом тестируемого приложения;* *4) Создание набора автоматических тестов для регрессионного функционального тестирования и тестирования пользовательского интерфейса;* *5) Создание набора автоматических тестов для тестирования производительности.*
Перечень графического материала	---
Консультанты по разделам выпускной квалификационной работы	
Раздел	**Консультант**
Финансовый менеджмент	Рахимов Тимур Рустамович, доцент
Социальная ответственность	Маланова Наталья Викторовна, инженер
Названия разделов, которые должны быть написаны на русском и иностранном языках:	

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	**ФИО**	**Ученая степень, звание**	**Подпись**	**Дата**
Доцент	Цапко Сергей	К.т.н.		

	Геннадьевич			

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8В2В3	Брагина Ольга Олеговна		

Министерство образования и науки Российской Федерации

федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Институт ИнЭО
Направление подготовки (специальность) Информатика и вычислительная техника
Уровень образования Бакалавр
Кафедра АиКС
Период выполнения весенний семестр 2016/2017 учебного года

Форма представления работы:

Бакалаврская работа

(бакалаврская работа, дипломный проект/работа, магистерская диссертация)

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
	Введение	
	Обзор литературы	
	Объект и методы исследования	
	Описание предметной области	
	Техническое задание	
	Проектирование	
	Руководство пользователя	
	Финансовый менеджмент	
	Социальная ответственность	

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Цапко Сергей Геннадьевич	К.т.н.		

СОГЛАСОВАНО:

И.о. зав. кафедрой	ФИО	Ученая степень, звание	Подпись	Дата
АиКС	Суходоев Михаил Сергеевич	К.т.н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
3-8В2В3	Брагина Ольга Олеговна

Институт	ИнЭО	**Кафедра**	АиКС
Уровень образования	Бакалавр	**Направление/ специальность**	Информатика и вычислительная техника

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. *Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих* 2. *Нормы и нормативы расходования ресурсов* 3. *Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования*	*Человеческие ресурсы: 1 чел.*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. *Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения*	*Проведена оценка коммерческого потенциала:* *1.Потенциальные потребители результатов исследования.* *2.Анализ конкурентных технических решений* *3.SWOT-анализ*
2. *Планирование и формирование бюджета научных исследований*	*Произведен расчет бюджета научных исследований.*
3. *Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования*	*Определена ресурсная, финансовая, бюджетная эффективность исследования посредством расчета интегрального финансового показателя, интегрального показателя ресурсоэффективности и эффективности.*
Перечень графического материала *(с точным указанием обязательных чертежей)***:**	
-	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Рахимов Тимур Рустамович	К.э.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
3-8В2В3	Брагина Ольга Олеговна		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
З-8В2В3	Брагина Ольга Олеговна

Институт	ИнЭО	**Кафедра**	АиКС
Уровень образования	Бакалавр	**Направление/ Специальность**	Информатика и вычислительная техника

Исходные данные к разделу «Социальная ответственность»:	
Описание рабочего места (рабочей зоны, технологического процесса, механического оборудования)	*Рабочее место программиста*
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Анализ выявленных вредных факторов проектируемой производственной среды в следующей последовательности: – *физико-химическая природа вредности, её связь с разрабатываемой темой;* – *действие фактора на организм человека;* – *приведение допустимых норм с необходимой размерностью (со ссылкой на соответствующий нормативно-технический документ);* – *предлагаемые средства защиты (сначала коллективной защиты, затем – индивидуальные защитные средства)*	*1) Повышенная напряженность электромагнитного поля;* *3) Умственное перенапряжение;* *4) Отсутствие или недостаток естественного освещения;* *5) Неоптимальный микроклимат помещения;* *6) Длительная работа в сидячем положении;* *7) Высокая нагрузка на зрительный аппарат.*
2. Анализ выявленных опасных факторов проектируемой произведённой среды в следующей последовательности – *механические опасности (источники, средства защиты;* – *электробезопасность;* – *пожаровзрывобезопасность*	*Обеспечение пожарной безопасности на рабочем месте*
3. Безопасность в чрезвычайных ситуациях	*Меры действия при возникновении пожара*
4. Правовые и организационные вопросы обеспечения безопасности	*1) Правовые вопросы обеспечения безопасности;* *2) Организационные вопросы обеспечения безопасности.*
Перечень графического материала:	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Инженер	Маланова Наталья Викторовна	К.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
З-8В2В3	Брагина Ольга Олеговна		

РЕФЕРАТ

Выпускная квалификационная работа 109 с., 23 рис., 25 табл., 24 источника, 1 прил.

Ключевые слова: автоматизация тестирования, тестирование пользовательского интерфейса, тестирование мобильных приложений, тестирование производительности, регрессионное тестирование.

Объектом исследования является организация тестирования пользовательского интерфейса кроссплатформенного мобильного приложения в процессе его разработки.

Цель работы – автоматизировать процесс функционального тестирования и тестирования пользовательского интерфейса программного продукта, тестирования производительности.

В процессе исследования проводились анализ тестируемого приложения на предмет выявления требований к автоматическим тестам, сравнительный анализ инструментов автоматизированного тестирования, методов и подходов, существующих в сфере организации автоматизации тестирования программного обеспечения.

В результате исследования был спроектирован и разработан проект автоматизации тестирования в среде TestComplete, созданы наборы автоматических тестов.

Степень внедрения: разработанные тесты в настоящий момент используются в процессе тестирования приложения «Machining Cloud».

Область применения: данная разработка представляет для компаний, занимающихся разработкой сложного, инженерного программного обеспечения, для внедрения автоматизации тестирования на долгосрочных, трудоемких проектах.

Экономическая эффективность/значимость работы заключается в возможности сократить расходы на ручное тестирование.

В будущем планируется написать автоматизированные тесты на базе разработанного фреймворка для нагрузочного тестирования, организовать возможность запуска тестов на облачных сервисах устройств.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире программное обеспечение используется практически во всех сферах нашей жизни, огромные средства тратятся на разработку разнообразных программ, востребованных в промышленности и бизнесе, в индустрии развлечений, в образовании и медицине. Задачи снижения стоимости разработки программного обеспечения и улучшения качества выпускаемой продукции являются одними из наиболее актуальных в индустрии информационных технологий.

Автоматизация тестирования позволяет значительно сократить расходы компаний-разработчиков, сэкономить время и ресурсы, затрачиваемые на тестирование, снизить риск выпуска на рынок некачественного продукта. Поэтому технологии автоматизации тестирования набирают все большую популярность среди компаний, связанных с разработкой программных продуктов. Это и определяет актуальность темы, выбранной для дипломного проекта.

Рынок мобильных приложений – один из самых перспективных и быстрорастущих. При этом автоматизация тестирования именно мобильных приложений является относительно молодой областью. Инструменты и технологии, использующиеся в этой области, относительно мало изучены и только завоевывают рынок и доверие разработчиков. При этом именно мобильные программные приложения, как никакие другие, нуждаются в автоматизированном тестировании в силу разнообразия линейки поддерживаемых устройств и версий операционных систем мобильных устройств, необходимости тестирования в портретном и альбомном режиме, при различных соединениях с интернетом и прочее. Это определяет практическую новизну проекта.

Предметом исследования дипломной работы является разработка проекта автоматизированного тестирования пользовательского интерфейса кроссплатформенного мобильного приложения на языке Jscript с использованием инструмента «Test Complete» на базе паттерна PageObject.

Целью работы является автоматизация процесса тестирования приложения «Machining Cloud», разработка тестов для тестирования пользовательского интерфейса под операционными системами iOS и Android, регрессионного функционального тестирования и тестирования производительности.

Практическая значимость результата дипломной работы: разработанные тесты в настоящий момент используются в процессе тестирования приложения «Machining Cloud», их внедрение позволило значительно сократить расходы компании на тестирование, ускорить время разработки и улучшить качество программного продукта.

1 АВТОМАТИЗАЦИЯ ТЕСТИРОВАНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА

1.1 Понятие автоматизированного тестирования и его значение для разработки программного обеспечения

Тестирование является важной частью процесса разработки программных продуктов и входит в число наиболее эффективных способов обеспечения их качества. Причем под качеством в сфере разработки программных средств подразумевается не только надежность программы или удобство пользования.

В соответствии с ГОСТ 28806–90, качеством программного средства считается совокупность его свойств, обуславливающих его пригодность удовлетворять заданные и подразумеваемые потребности в соответствии с его назначением [1]. Таким образом, качественным считается тот продукт, который удовлетворяет критериям качества, предъявляемым к нему заинтересованными лицами, заказчиками либо пользователями данного продукта. Программный продукт должен соответствовать определенным стандартам и ожиданиям для того, чтобы его можно было считать качественным.

Согласно определению Гленфорд Майерс, тестированием называется процесс исследования программы с целью нахождения в ней ошибок [2]. И здесь, опять же, под ошибками, или дефектами программы, понимаются изъяны в разработке программного продукта, следствием которых становится несоответствие ожидаемых результатов выполнения программного продукта и фактически полученных результатов [3]. То есть по сути задачей процесса тестирования является выявление фактов расхождения реального поведения приложения с требованиями, предъявляемыми к нему.

На практике тестирование осуществляется путем выполнения определенного набора действий в тестируемом приложении, получении результатов выполнения этих действий и дальнейшей сверки их с данными, которые определены как эталонные. Выполняться этот процесс может как вручную, специалистами по тестированию, так и автоматически, с использованием различных

программных средств. Именно такой процесс верификации программного обеспечения, в ходе которого основные функции и шаги теста, такие как запуск, инициализация, выполнение, анализ и выдача результата, выполняются автоматически при помощи инструментов для автоматизированного тестирования, и называется автоматизированным тестированием программного обеспечения [4].

У автоматизированного тестирования есть как свои преимущества, так и недостатки. К сильным сторонам автоматизированного тестирования относят:

- быстрая скорость выполнения, намного превосходящая возможности человека;
- отсутствие влияние «человеческого фактора» (невнимательность, усталость);
- возможность многократного выполнения тестов и снижение затрат через это;
- выполнение тест-кейсов особенной сложности, недоступных человеку;
- способность хранить, анализировать в удобной форме колоссальные объемы данных;
- способность выполнять низкоуровневые действия с приложением, с операционной системой и т.д.

К недостаткам автоматизированного тестирования относятся:

- необходимость привлечения высококвалифицированного персонала для автоматизации взамен возможности использовать низкоквалифицированный труд тестировщиков;
- затраты на средства автоматизации, на разработку и сопровождение тестов;
- финансовые затраты и риски, связанные с наличием большого количества средств автоматизации и сложностью выбора;

- устаревание тестов в случаях изменений требований, переработки интерфейсов тестируемых продуктов и т.п.

Автоматизация тестирования не позволяет полностью исключить ручное тестирование из процесса разработки, но значительно снижает его долю, помогает убрать рутину из процесса тестирования, снизить стоимость и значительно улучшить качество разрабатываемых программных продуктов.

Тестирование является одной из наиболее трудоемких фаз разработки программного обеспечения, занимающей большое количество времени. При этом цена исправления ошибки, как правило, напрямую зависит от времени, которое проходит с момента ее возникновения до момента обнаружения.

В своей книге «A Practitioner's Guide to Software Test Design» Ли Копланд приводит ответы, которые дают специалисты по тестированию на вопрос о том, с какими наиболее распространенными проблемами им приходится сталкиваться в их работе: «недостаточно времени, чтобы тестировать тщательно», «слишком много комбинаций входных данных», «недостаточно времени, чтобы протестировать хорошо», «слишком мало времени выделено на тестирование» [5].

Почти все называли проблемы, связанные с объемом работы, которую необходимо выполнить, и количеством времени, которое необходимо затратить для того, чтобы качественно протестировать программный продукт. Поэтому естественно, что идея об автоматизации процесса тестирования, позволяющей решить вопрос с недостатком времени для осуществления качественного тестирования, становилась все более актуальной по мере увеличения сложности разрабатываемых программных продуктов и роста требований к их качеству.

Попытки автоматизировать процесс тестирования совершались еще в 80-ых годах. В 90-ых появились первые инструментальные средства автоматизации тестирования. А в нулевые годы автоматизация тестирования воспринималась уже как неотъемлемая часть большинства проектов.

Сегодня сложно представить себе разработку крупного программного проекта без использования в процессе разработки автоматизированного тестирования. Современный этап развития тестирования характеризуется глубокой интеграцией тестирования с процессом разработки в целом, а также широчайшим использованием автоматизации, колоссальным набором технологий и инструментальных средств для автоматизации [6].

1.2 Уровни автоматизации. Место тестирования пользовательского интерфейса в общем процессе автоматизированного тестирования.

На сегодняшний день существуют различные подходы к автоматизации тестирования. Комплексная стратегия тестирования включает в себя 3 уровня автоматизации:

1. уровень модульного тестирования (Unit Tests Layer) – компонентные тесты, обычно пишутся разработчиками;
2. уровень функционального тестирования (Functional Test Layer, or Service / API Tests Layer) – тестирование через доступ к функциональному слою, минуя пользовательский интерфейс;
3. уровень тестирования пользовательского интерфейса, или через пользовательский интерфейс (GUI Test Layer) – дает возможность тестировать не только сам интерфейс пользователя, но и реализовывать функциональное тестирование, выполняя операции, использующие бизнес логику приложения.

Для обеспечения наилучшего качества приложения рекомендуется автоматизировать все три уровня. На рисунке 1 приведена классическая пирамида автоматизированного тестирования [7]. Данная пирамида показывает идеальное соотношение тестов различного уровня для среднестатистического проекта. Тесты верхнего уровня считаются более сложными в разработке и дорогостоящими. Соотношение может меняться в зависимости от специфики тестируемого продукта.

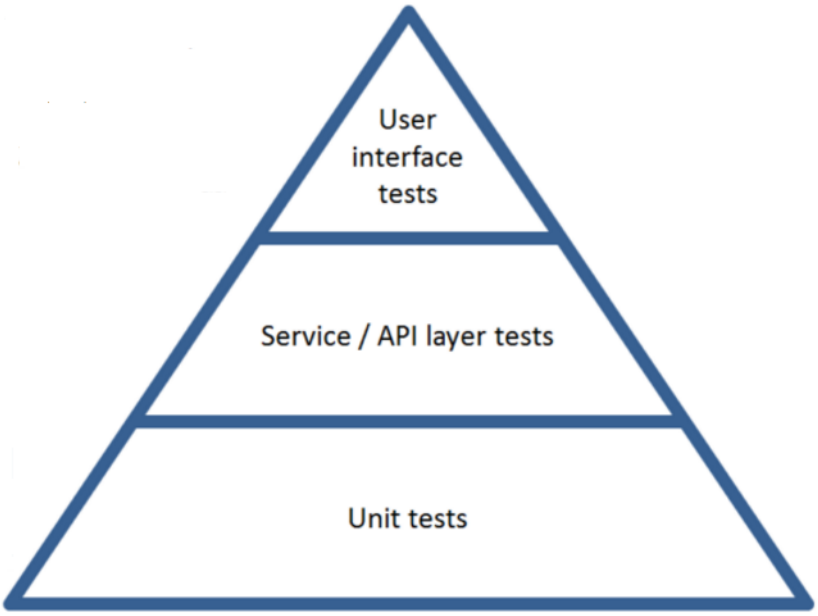


Рисунок 1 – Классическая пирамида автоматизации

Данная дипломная работа посвящена реализации третьего уровня тестирования – тестирования пользовательского интерфейса. Автоматизация тестирования программных продуктов через графический пользовательский интерфейс набирает всё большую популярность в настоящее время. И это неудивительно, поскольку только в этом случае приложение будет протестировано тем же самым способом, которым оно будет использоваться конечными пользователями.

Плюсами такого вида тестирования является то, что тестирование происходит на реальных устройствах с различной конфигурацией, с различными операционными системами, на разных браузерах. Такой вид тестирования является наиболее важным видом тестирования мобильных приложений в силу необходимости осуществлять тестирование на всей линейке поддерживаемых устройств.

К минусам данного вида автоматизированного тестирования можно отнести то, что тесты проходят долго, много усилий тратится на их поддержание (при изменении интерфейса требуется внесение изменений во все тесты, ис-

пользующие обновленные элементы), тесты могут быть нестабильными (могут «падать» из-за попыток обратиться к элементам интерфейса, которые еще не подгрузились, нестабильно работать с конфигурациями устройств для запуска тестов, отличными от той, на которой тесты создавались).

Реализуется такой вид автоматизации через использование специализированных инструментов для автоматизации тестирования, которые предоставляют возможность взаимодействия с интерфейсом тестируемой программы. Как правило, при этом тестируемое приложение «инструментируется», то есть при компиляции в приложение добавляется дополнительная библиотека (в случае инструментации приложения для работы в среде TestComplete это библиотека PatchServices.jar), а в код приложения добавляются методы вызова данной библиотеки. После этого разработчик тестов получает доступ к элементам интерфейса приложения, он может получать информацию об элементах интерфейса и их свойствах, изменять эту информацию и имитировать различные действия пользователя в приложении: клики мышкой по указанным объектам, нажатие клавиш на клавиатуре, жесты и т.п.

1.3 Инструменты для автоматизации тестирования пользовательского интерфейса

В качестве инструментов для автоматизации подобного рода тестирования можно использовать коммерческие продукты, бесплатные (или условно-бесплатные) инструменты, либо разработать собственные утилиты для тестирования.

У каждого из этих методов есть свои достоинства и недостатки.

К преимуществам использования коммерческих продуктов можно отнести:

- значительное снижение времени, затрачиваемого на разработку автотестов за счет использования функциональности, предоставляемой в поставке с продуктом, которую не нужно писать самостоятельно;

- предоставление вместе с инструментами собственной среды разработки, которая дает возможность удобного написания и отладки тестов;
- поддержка со стороны разработчиков, возможность запросить дополнительный функционал, попросить об исправлении ошибок.

В качестве недостатков использования коммерческих продуктов можно отметить:

- высокую стоимость;
- отсутствие доступа к исходному коду;
- отсутствие гибкости в выборе подхода к тестированию, необходимость следовать модели, на базе которой построен инструмент.

К преимуществам использования бесплатных или условно бесплатных инструментов следует отнести:

- разумную стоимость;
- доступность исходного кода.

В качестве недостатков использования бесплатных или условно бесплатных инструментов можно выделить:

- недостаточную функциональность, порой невозможность реализовать все поставленные задачи;
- риск отказа от поддержки со стороны разработчиков продукта.

Преимуществами использования собственных инструментов являются:

- возможность разработки инструмента как побочного результата процесса создания самого приложения;
- легкость использования за счет хорошего знания инструмента;
- доступность исходного кода.

В качестве недостатков использования собственных инструментов необходимо отметить:

- риски, связанные с тем, что функциональность инструмента может оказаться недостаточной для реализации полноценного проекта по тестированию;
- вероятность того, что цена создания и поддержки таких инструментов будет чрезмерно высокой;
- отсутствие каких-либо гарантий стабильности результатов.

Наиболее известными коммерческими продуктами для автоматизации тестирования пользовательского интерфейса мобильных приложений являются на сегодня такие инструменты, как TestComplete, Ranorex, SeeTest Automation, Squish. Интересные и многообещающие предложения есть и среди бесплатных инструментов: Appium, Selendroid, Robotium, Calabash.

В качестве оценочных критериев при выборе инструментов рекомендуется использовать следующие [8]:

- поддерживающие мобильные платформы (распознавание необходимых элементов интерфейса, технологии взаимодействия с ними, как реализован процесс взаимодействия);
- поддерживающие типы приложений (веб, гибридные, нативные);
- поддержка запуска тестов на эмуляторах, реальных устройствах;
- IDE (поддерживающие языки, удобство пользования, инструменты отладки);
- менеджеры тестов (возможности по конфигурированию тестовых наборов, возможность параллельного запуска тестов, настройка параметров запуска);
- логирование, создание отчетов (разрешения, в которых предоставляются отчеты, возможность настроек, информативность).

При выборе инструмента специалисты советуют принимать во внимание специфику приложения, поставленные задачи, квалификацию разработчиков и инженеров по тестированию, технические и финансовые ресурсы, планы на будущее, риски [9].

1.4 Существующие подходы к автоматизации. Паттерн Page Object.

На текущий момент к автоматизации тестирования пользовательского интерфейса существует 4 основных подхода:

1. Capture/Playback – Запись и воспроизведение.

Данный подход предполагает использование утилит записи и воспроизведения, записывающих определенную последовательность действий и затем ее воспроизводящих уже без участия человека. Такие тесты создаются легко и быстро, но при любых изменениях интерфейса требуют полной перезаписи. Считается наиболее неэффективной техникой, непригодной для масштабирования. Тем не менее является очень дешевой и доступной. Может использоваться для смок-тестирования, при необходимости разового прогона на различных устройствах.

2. Scripting – Написание сценария.

Методология заключается в написании тестовых сценариев на языках программирования, разработанных специально для автоматизации тестирования. Данный подход решает проблемы с масштабированием и поддержкой тестов, открывает большие возможности. Но при этом является более дорогостоящим, требующим большего количества ресурсов, поскольку разработкой занимаются программисты более высокого уровня.

3. Data-driven testing – Управление данными тестирования.

Методология создания скриптов и верификации их на основе данных, хранящихся в каком-либо хранилище или базе данных. Используется в случаях, если необходимо реализовывать однотипные виды проверок для множественных вариантов входных данных.

4. Keyword-based – Тестирование по ключевым словам.

Тест представляет собой не программный код, а последовательность действий с их параметрами, описанную с помощью ключевых слов, реализация данной последовательности выполняется фреймворком. Позволяет созда-

вать тесты людям, не имеющим навыков программирования. Является недостаточно эффективным методом для организации полноценного тестирования, но получил распространение за счет простоты и доступности, при этом частично решает проблемы первого подхода (тесты более стабильны, их легче поддерживать).

Для долгосрочных, сложных проектов наиболее подходящими являются вторая и третья техника (третья – при необходимости тестирования на большом объеме данных), они требуют больше ресурсов для реализации, но при этом окупаются в будущем, позволяя множественно использовать как написанный код, так и сами созданные тесты (для регрессионного тестирования, включать их в процесс непрерывной интеграции (CI), использовать для нагрузочного тестирования, тестирования производительности и т.д.).

При написании тестовых скриптов обычно применяют технологию фреймворков. Фреймворк – это организация проекта, позволяющая упростить разработку, модификацию и поддержку программного кода. В случае организации проекта по автоматизации тестирования пользовательского интерфейса приходится считаться с тем, что тесты потребуется поддерживать, а возможно и существенно переписывать в случае изменений в интерфейсе тестируемого приложения. Основная задача, решаемая через использование фреймворков – снижение количества изменений, которые требуется внести в тесты при изменениях в тестируемом приложении.

Учебник по TestComplete от компании SmartBear рекомендует использовать при организации проекта функциональную декомпозицию – разнесение кода в разные функции или модули в зависимости от их назначения [10]. Также многие авторы по автоматизации тестирования пользовательских интерфейсов рекомендуют организовывать код с использованием нескольких уровней абстракции, например, через пространства имен, карты объектов интерфейса и т.п. [11, 12].

В настоящий момент в области автоматизации тестирования существует один наиболее известный шаблон проектирования – Page Object. Подобная организация проекта позволяет значительно упростить поддержку тестов и снизить дублирование кода.

Основные идеи данного шаблона проектирования:

- четкое разделение между непосредственно кодом тестов и кодом, специализирующимся на работе с локаторами (путями к элементам интерфейса);
- наличие единого репозитория для всех служб и операций, представленных на странице, а не множества разбросанных по всему тестовому набору [13].

Мартин Фаулер объясняет данный паттерн на простом примере, приведенном на рисунке 2 [14].

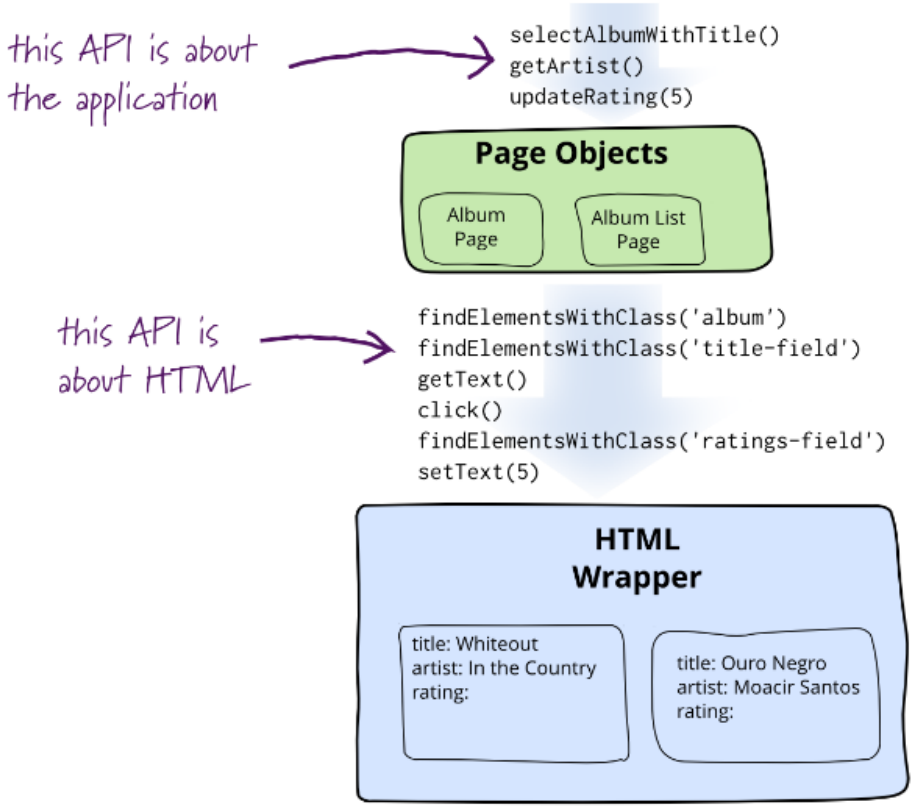


Рисунок 2 – Логика разделения кода в паттерне Page Object

В приведенном примере в качестве тестируемого приложения рассматривается сайт с музыкальными альбомами. Веб-страничка приложения содержит список альбомов, каждый из которых имеет заголовок, название исполнителя и рейтинг.

В нижней части рисунка изображен блок HTML Wrapper – обертка HTML. В качестве примеров функций этого уровня приводятся:

- findElementsWithClass(‘album’) – получить список всех элементов класса «альбом»;
- findElementsWithClass(‘title-field’) – получить список всех элементов класса «поле заголовка»;
- getText() – получить текст;
- click() – щелкнуть мышкой;
- findElementsWithClass(‘ratings-field’) – получить список всех элементов класса «поле рейтинга».

Все эти функции касаются непосредственно HTML кода. На этом уровне реализована работа с HTML кодом страницы.

В верхней части рисунка изображен блок Page Objects. Он содержит такие объекты, как страница альбома и страница списка альбомов. В качестве примера функций, реализуемых на этом уровне приведены следующие процедуры:

- selectAlbumWithTitle() – выделить альбом с заголовком, передаваемым в качестве аргумента функции;
- getArtist() – получить название исполнителя;
- updateRating(5) – установить значение рейтинга равным 5.

Эти функции касаются самого приложения, его бизнес-логики, а не кода.

Паттерн Page Object был разработан для тестирования веб-приложений, но его идеи используются сегодня во всех остальных видах автоматизирован-

ного тестирования, существуют различные разновидности реализации данного шаблона. Основным преимуществом его использования является то, что при изменениях в пользовательском интерфейсе достаточно модифицировать код тестов только в одном месте.

2 АНАЛИЗ ПРЕДМЕТНОЙ ОБЛАСТИ

2.1 Анализ тестируемого приложения

Приложение Machining Cloud [15] разрабатывается международной компанией Machining Cloud GmbH. Компания имеет несколько офисов, расположенных в различных странах мира. Томский офис, подразделение «DP Labs», является частью компании ООО «Рубиус» и занимается разработкой мобильной версии приложения Machining Cloud, предназначенной для использования на планшетах. Приложение разрабатывается под платформы iOS и Android. Разработка ведется в среде Visual Studio 2015 с использованием фреймворка для кроссплатформенной разработки мобильных приложений Xamarin на языке C#.

На сегодня Machining Cloud представляет из себя крупнейшую независимую базу знаний о продуктах, применяемых в сфере обработки материалов с использованием станков с числовым программным управлением. Приложение предоставляет пользователю различный функционал, помогающий подбирать инструменты для станков: доступ к электронным каталогам компаний-производителей инструментов, фильтры по каталогам, интеллектуальный подбор по описанию задачи. В качестве примера на рисунках 3-4 приведены поиск по каталогу, а также поиск с использованием инструмента Tool Advisor в приложении Machining Cloud.

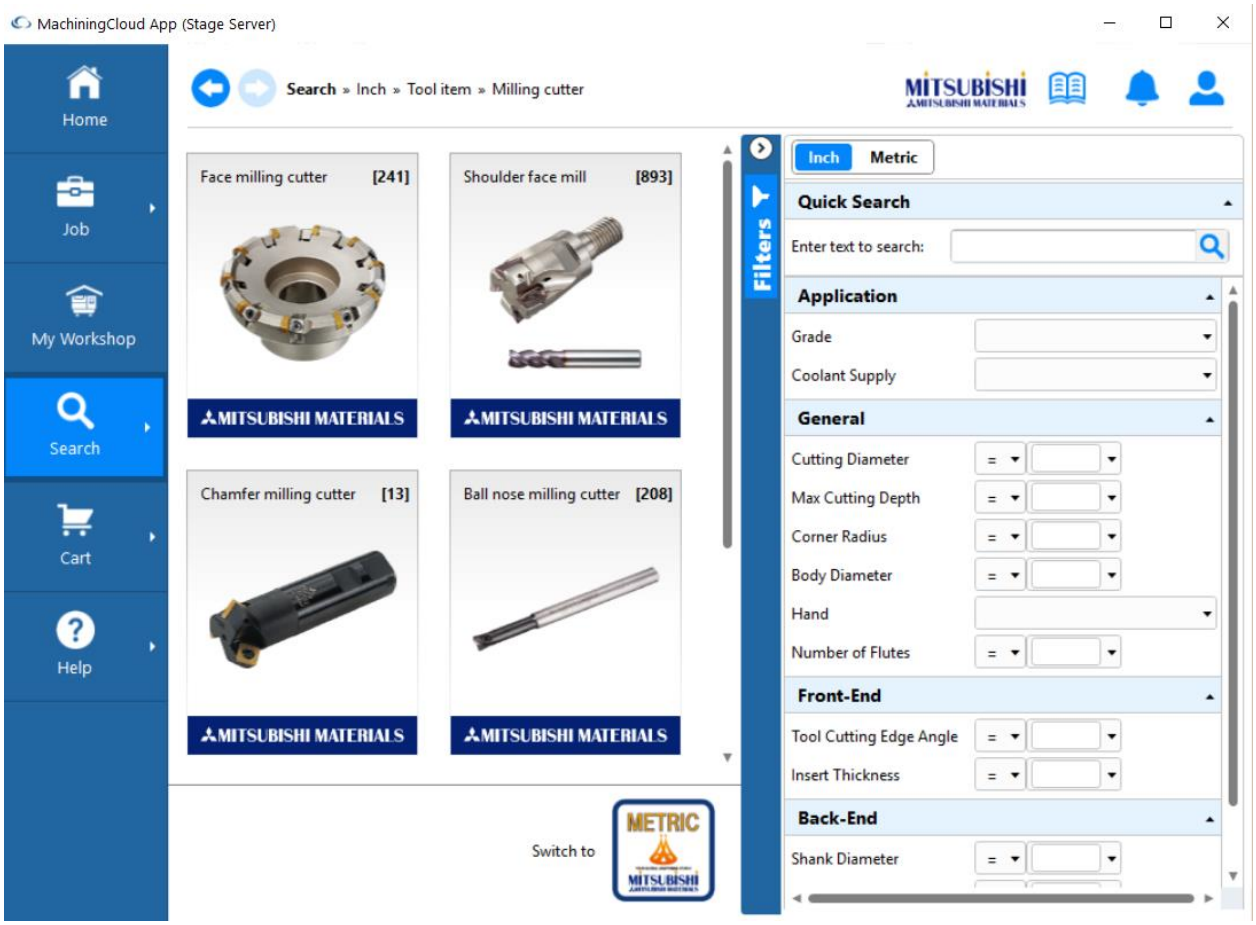


Рисунок 3 – Каталог и панель фильтров в программе Machining Cloud

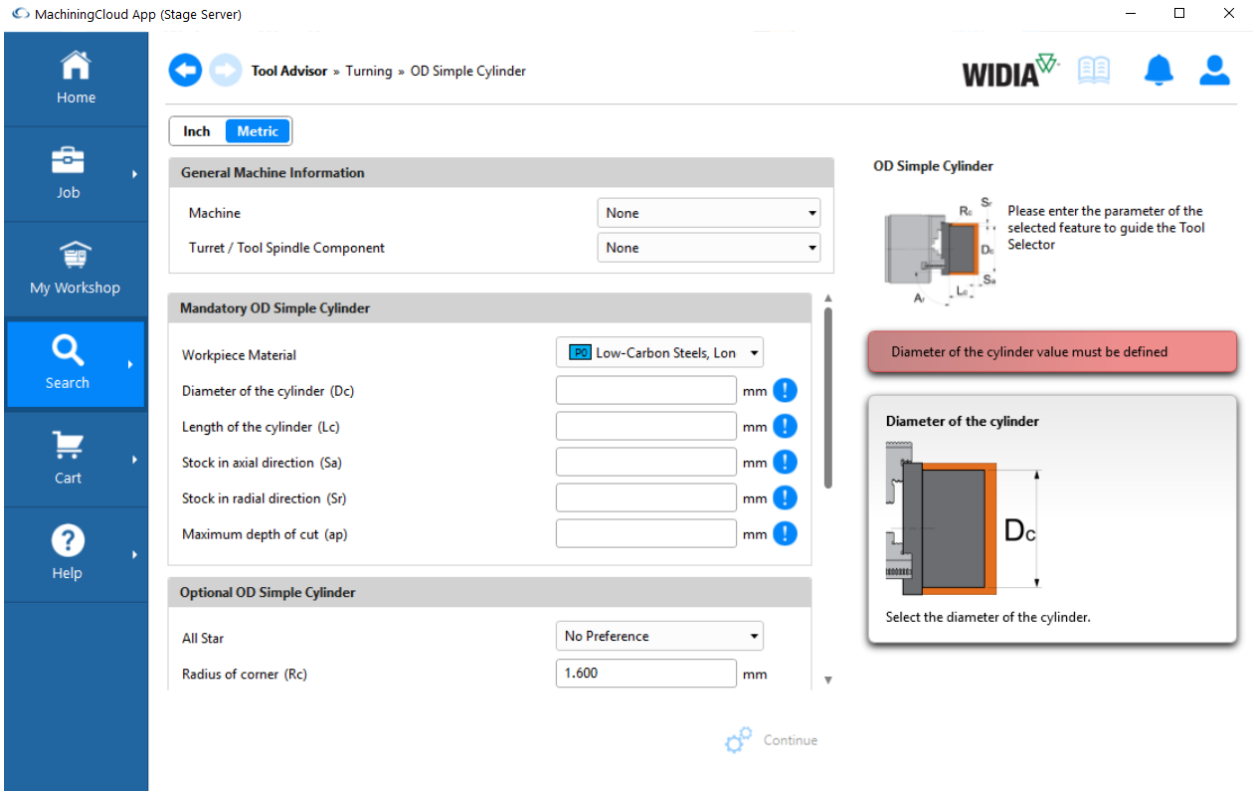


Рисунок 4 – Интеллектуальный подбор инструментов в Machining Cloud

Также приложение позволяет собирать рабочие комплекты инструментов, просматривать полученные результаты в виде 3D-моделей, выгружать данные по инструментам в виде отчетов, чертежей, готовых моделей (визуализированных, антиколлизионных и пр.) для дальнейшего использования в разных CAM-системах. На рисунке 5 представлена визуализация 3D-модели собранного рабочего комплекта инструмента.

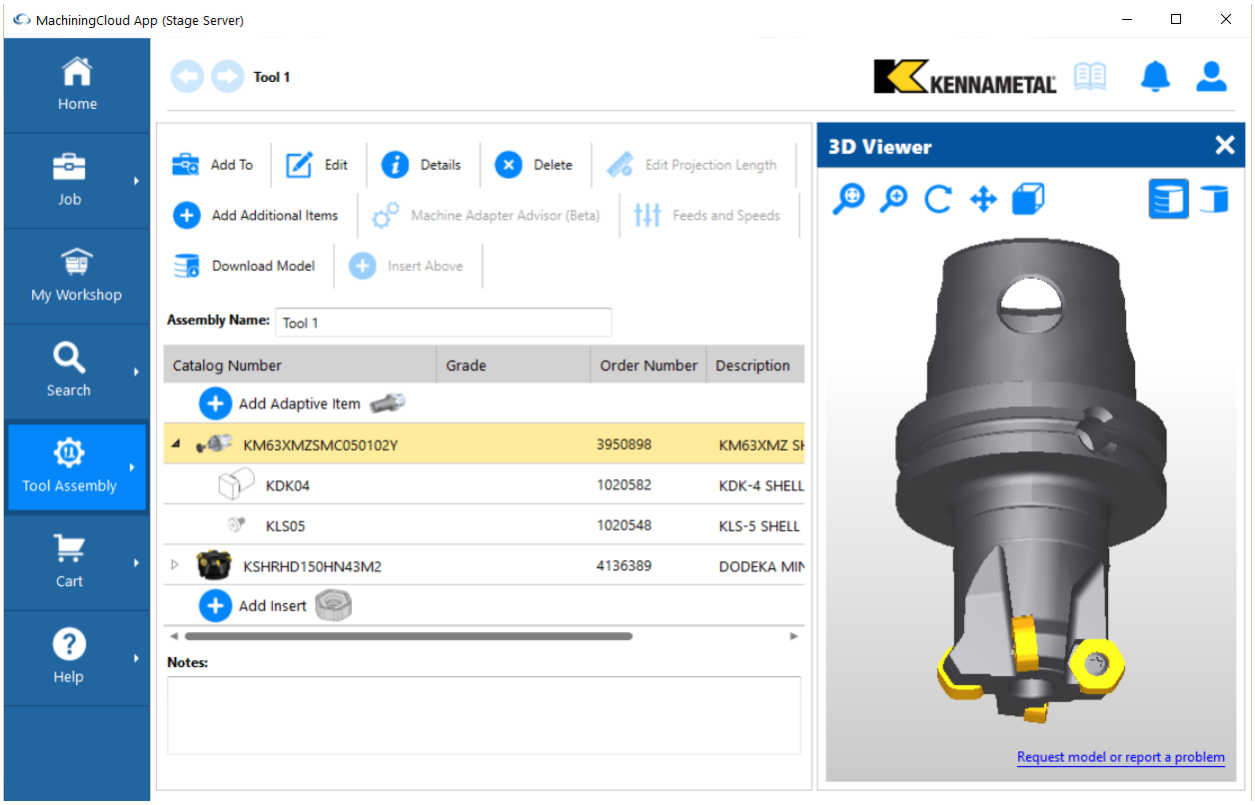


Рисунок 5 – Модель рабочего комплекта в Machining Cloud

Помимо функционала по конфигурированию инструментов, в приложении реализована возможность хранения созданных сборок инструментов, организации их по проектам, предоставления доступа к сборкам отдельным пользователям приложения и рабочим группам, пересылка данных по почте.

Система также содержит множество дополнительной информации об инструментах: рекомендации по использованию, информацию о наличии на складе пользователя (подключается к электронным системам учета инструментов ToolBOSS, MATRIX), рейтинги инструментов, автоподбор замены для устаревших компонентов. На рисунке 6 представлена диалоговое окно приложения, содержащее информацию о скоростях и подачах для выбранного комплекта инструмента.

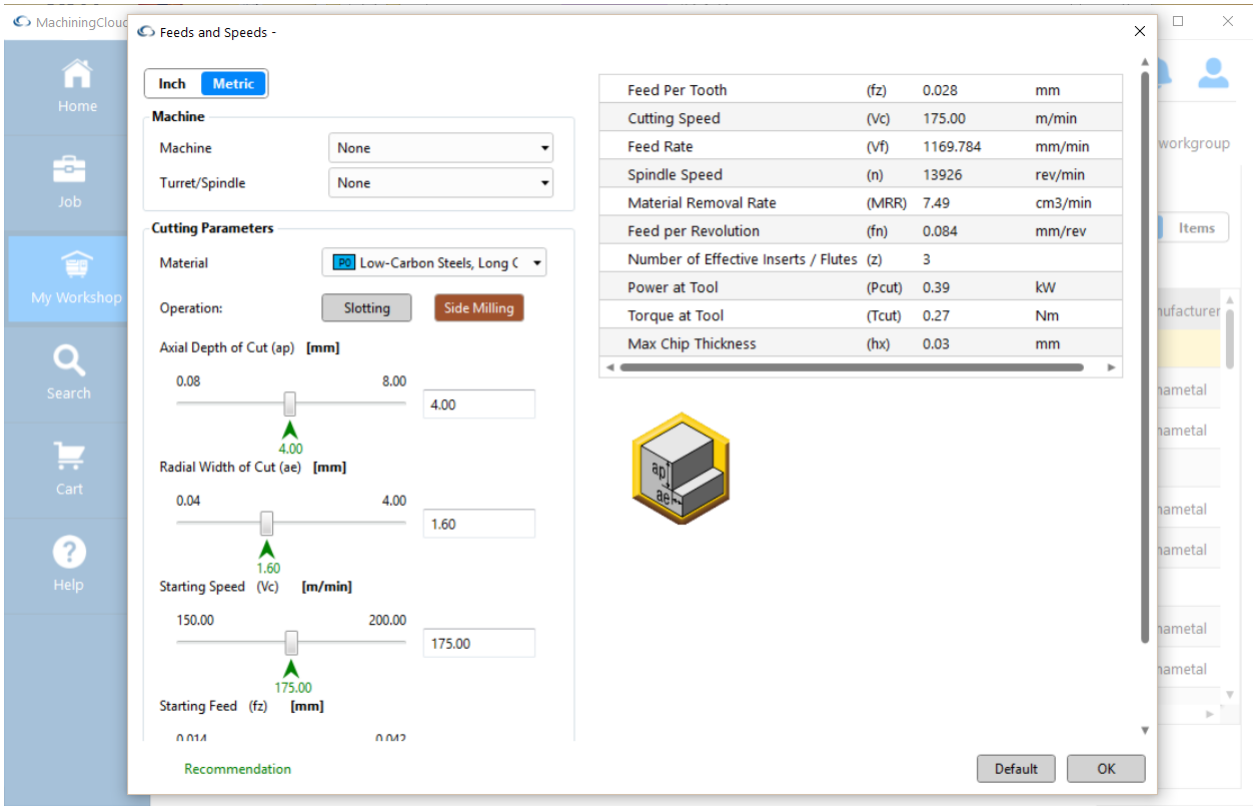


Рисунок 6 – Диалог Feeds and Speeds приложения Machining Cloud

Приложение включает в себя систему электронной продажи и менеджер по работе с заказами и платежными аккаунтами. Сервис E-Commerce приложения представлен на рисунке 7.

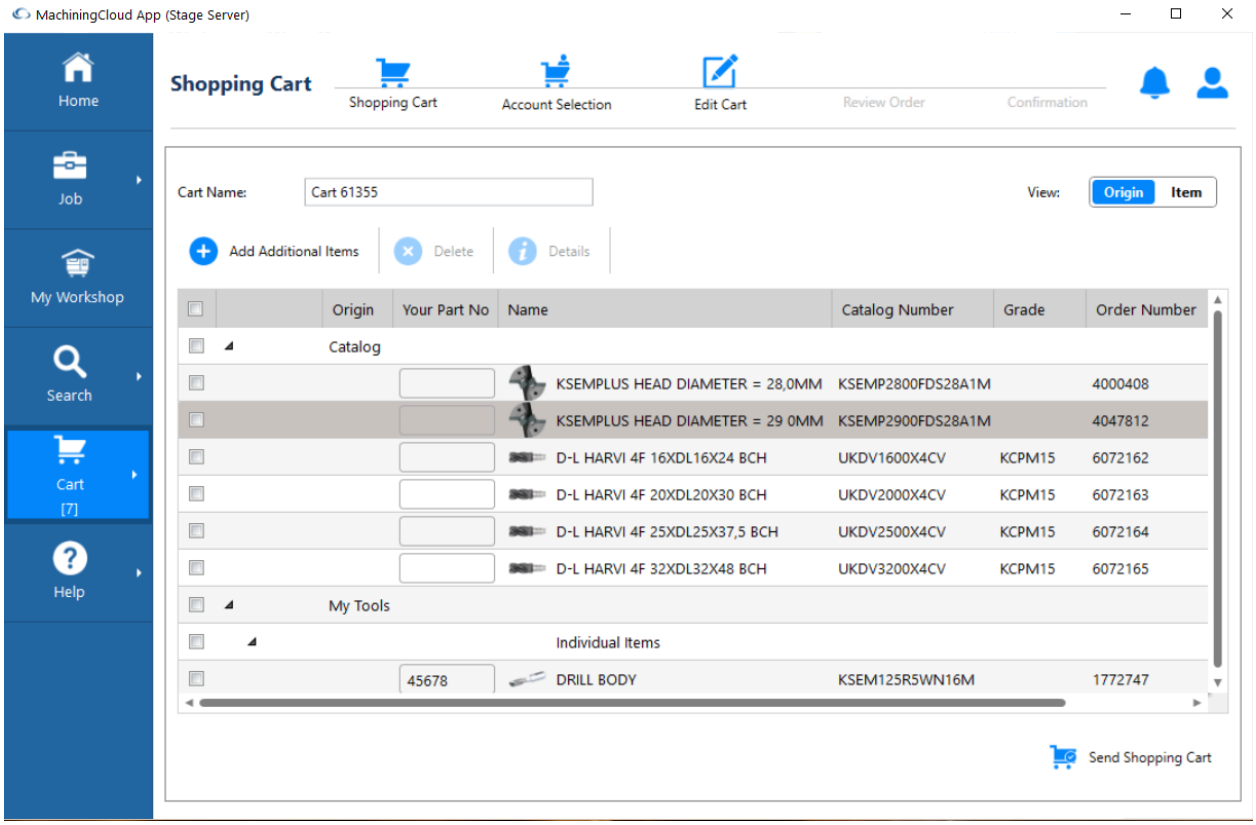


Рисунок 7 – E-Commerce сервис приложения Machining Cloud

Разрабатываемое приложение является долгосрочным, многолетним проектом, объемным и сложным по своей структуре. Проект постоянно дорабатывается и меняется: появляется новый функционал, интерфейс приложения адаптируется под потребности пользователей, добавляются новые каталоги, расширяется линейка поддерживаемых форматов экспорта данных.

2.2 Формулирование требований к проекту автоматизированного тестирования

Поскольку приложение предназначено для использования в сфере высокоточного, дорогостоящего производства, где неправильно подобранный инструмент, неточные данные о режимах его использования приводят к поломке дорогостоящего оборудования, а иногда и к травмам на производстве, к низкому качеству производимых товаров, и как следствие – к экономическим

потерям, большое внимание при разработке уделяется качеству программного обеспечения, а соответственно и организации тестирования этого качества.

Приложение поддерживает около 15 языков, используется на большой линейке устройств, заявлена поддержка операционных систем iOS 8 и старше, Android 4.0 и старше, соответственно должна обеспечиваться возможность запуска тестов на всей линейке устройств во всех требуемых конфигурациях, причем как на эмуляторах, так и на реальных устройствах. Запуск на эмуляторах упрощает включение тестов в процесс непрерывной интеграции (Continuous Integration), а запуск на реальных устройствах дает больше гарантий, что все ошибки будут действительно воспроизведены.

Заказчиком была поставлена задача реализовать тестирование пользовательского интерфейса, функциональное регрессионное тестирование, а также тестирование производительности. Должна быть возможность включить тесты в процесс непрерывной интеграции (CI), формировать из созданных тестов различные наборы сетов для параллельного запуска на разных билд-агентах с целью сокращения времени прохождения тестов.

Запуск тестов и сохранение результатов должны происходить автоматически. Причем для тестов производительности результаты с измерениями времени выполнения отдельных операций должны записываться дополнительно в отдельные файлы и сохраняться для возможности дальнейшего анализа данных.

Проект необходимо реализовать под 2 платформы, в которых пользовательский интерфейс значительно различается, но при этом бизнес-логика работы приложения идентична. Интерфейс приложения, для тестирования которого разрабатывается проект, подвержен частым изменениям, что может потребовать значительных усилий для поддержания написанных тестов в будущем.

В приложении есть функционал, отвечающий за визуализацию 3D-моделей, для просмотра моделей в приложении используются стандартные жесты мобильных устройств, такие как Drag and Drop (нажатие и перетаскивание), Swipe (свайп), Pinch и Spread (стягивание и растягивание). Для тестирования данного функционала необходимо реализовать имитацию всех необходимых жестов, а также возможность сравнивать скриншоты.

Учет этих факторов позволяет сформулировать следующие требования к разрабатываемым автотестам:

1. тесты должны требовать минимальных затрат на их поддержание в будущем;
2. тесты должны быть стабильными и по возможности максимально быстрыми;
3. тесты не должны быть зависимыми от конфигурации устройства, на котором они запускаются;
4. тесты должны поддерживать использование жестов на устройстве;
5. тесты должны поддерживать сравнение картинок с использованием настроек толерантности и масок для тестирования графики;
6. результаты тестов должны храниться в удобном для анализа виде и при этом не занимать много места на диске;
7. должна быть возможность запускать тесты в различных комбинациях и сочетаниях.

В соответствии с этими требованиями были сформулированы следующие задачи:

1. выбор подходящего инструмент для автоматизации тестирования;
2. разработка структуры проекта автоматизированного тестирования пользовательского интерфейса с использованием выбранного инструмента;
3. разработка сценариев для тестов;

4. создание двух репозиториев (для платформ iOS и Android) с основными функциями, реализующими: имитацию действий пользователя в тестируемом приложении (взаимодействие с пользовательским интерфейсом через нажатие кнопок, выбор элементов из списка, ввод текста в поля, жесты и т.д.) и осуществление различного типа проверок через получение информации из пользовательского интерфейса (сравнение значений различных свойств элементов, сравнение картинок для проверки работы приложения с графикой и т.д.);
5. реализация записи результатов тестирования;
6. написание тестов для функционального регрессионного тестирования, а также тестирования производительности на базе созданных фреймворков.

3 ОСОБЕННОСТИ РЕАЛИЗАЦИИ ПРОЕКТА

3.1 Обоснование выбора инструмента и подходов

Первостепенной задачей при организации автоматизации тестирования является **выбор инструмента**. В рамках тестирования приложения «Machining Cloud» нецелесообразно использовать бесплатные инструменты, поскольку они не могут предоставить функционал достаточный для поставленных задач, а написание собственных инструментов займет много времени и может стать более дорогим, чем покупка готового решения.

Инструмент должен поддерживать мобильные платформы iOS и Android, поддерживаемый тип приложений – нативные. При этом критичным фактором является своевременное добавление поддержки для новых версий операционных систем. Инструмент должен распознавать все используемые в интерфейсе виды контролов.

Инструмент должен предоставлять возможность запуска тестов на эмуляторах и реальных устройствах. Поддерживать автоматическую установку тестируемого приложения на устройства. Важно наличие возможности параллельного запуска тестов на разных устройствах в целях сокращения времени прогона тестов. Легкость интеграции с системами управления версиями (а конкретно с Team Foundation Server).

IDE инструмента должна поддерживать современные, знакомые разработчикам тестов языки, иметь инструменты для отладки. Наличие встроенного менеджера тестов также является плюсом. Инструмент должен иметь средства для записи результатов тестирования, причем результаты должны сохраняться в удобном виде, доступном для дальнейшего анализа, обязательно наличие возможности сохранения скриншотов. Важна поддержка жестов для работы с интерфейсом, а также наличие инструментов для интеллектуального сравнения изображений с настройками толерантности и использованием масок (для тестирования работы приложения с 3д-графикой).

В настоящее время на рынке инструментов автоматизированного тестирования пользовательских интерфейсов такими возможностями обладают два инструмента: Ranorex и TestComplete. Стоимость инструментов примерно одинакова, предоставляемый функционал сходен. При этом Ranorex является достаточно молодым инструментом на рынке инструментов для тестирования, тогда как TestComplete является признанным авторитетом, имеет развитую базу знаний по своим продуктам и хорошую службу поддержки. Таким образом, с целью минимизировать риски выбор был остановлен на фреймворке TestComplete, несмотря на его высокую цену.

Поскольку основной сложностью автоматизированного тестирования пользовательских интерфейсов является сложность поддержания тестов, выбор в вопросе используемых подходов был остановлен на написании скриптовых тестов с использованием паттерна Page Object. TestComplete поддерживает все существующие виды создания тестов: Capture/Playback, Data-driven testing, Keyword-based тесты, тем не менее скриптовые тесты являются наиболее гибкими, масштабируемыми и открывают больше возможностей для использования.

TestComplete поддерживает несколько языков разработки: VBScript, JScript, C++Script, C#Script, DelphiScript. В качестве языка разработки был выбран JScript – скриптовый язык от компании Microsoft. Он более компактен с точки зрения написания кода, более современный и обладает большими возможностями. В настоящий момент TestComplete добавил поддержку языков Python и Java, но на момент начала работы над проектом эти языки не поддерживались.

В качестве паттерна для организации проекта был выбран Page Object, являющийся по сути стандартом для организации скриптового подхода автоматизированного тестирования. TestComplete накладывает определенные ограничения на организацию проектов внутри своей среды, поэтому паттерн реализован не в своем изначальном виде, а адаптирован под среду разработки.

3.2 Организация проекта в среде TestComplete

Особенностью данного проекта является необходимость организовать тестирование приложения одновременно для 2 платформ (iOS и Android). При этом пользовательский интерфейс тестируемых приложений значительно отличается, но бизнес-логика совпадает. Здесь как нельзя удобным оказывается использование паттерна Page Object, позволяющего отделить работу с бизнес-логикой приложения от работы с интерфейсом. Это дает возможность использовать одни и те же модули тестов, содержащих бизнес-логику, одновременно для обоих проектов.

На рисунке 8 приведена структура созданного проекта автоматизированного тестирования в среде TestComplete.

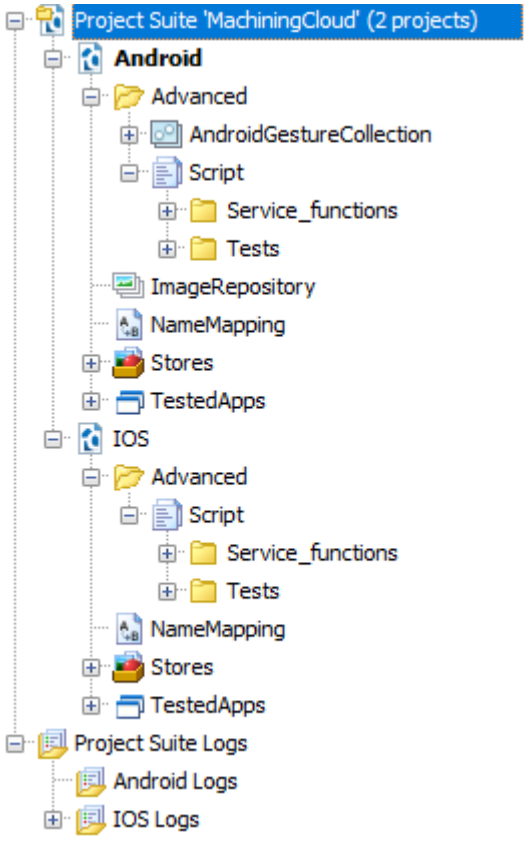


Рисунок 8 – Структура проекта в TestComplete

Комплект проектов содержит в себе 2 отдельных полноценных проекта: один для организации тестирования iOS-приложения, а второй для приложения под Android. Каждый проект содержит:

- NameMapping – иерархическую карту объектов тестируемого приложения;
- Script – пакет модулей с исполняемыми скриптами. Пакет разбит на 2 отдельных пакета: репозиторий Service functions (содержит модули с функциями, симулирующими взаимодействие с пользовательским интерфейсом приложения) и пакет Tests со скриптами, содержащими бизнес-логику тестов;
- Gesture Collection – коллекцию жестов;
- Stores – сохраненные объекты, использующиеся в качестве эталонных для осуществления проверок во время выполнения тестов;
- TestedApps – тестируемые приложения, используемые для запуска на них тестов (хранятся как в виде готовых сборок, так и в виде ссылки для скачивания);
- LogFiles – результаты запуска тестов.

Модули пакета Tests, содержащие тестовые скрипты, представляют собой особый тип модулей – shared project items, данные модули являются общими («расшариваются») для обоих проектов, при изменении одного модуля, данные изменения подхватываются в другом.

3.2.1 Карта элементов интерфейса Name Mapping.

Для создания дополнительного уровня абстракции, а также сокращения времени, затрачиваемого на поиск элементов интерфейса во время выполнения тестов, TestComplete предлагает использовать Name Mapping – иерархическую карту объектов пользовательского интерфейса тестируемого приложения.

NameMapping создается разработчиком с использованием встроенных инструментов TestComplete: Object Browser или Map Object. Данные инструменты позволяют выбрать нужный элемент интерфейса и создать для него соответствующую запись в карте имен, задав определяющие его параметры. NameMapping используется для идентификации объектов во время выполнения тестов, он содержит имена объектов и ассоциированные с ними пары свойство-значение свойства, заданные разработчиком в качестве уникальных для распознавания данных объектов.

NameMapping должен содержать все UI объекты тестируемого приложения, которые будут использоваться автоматизированными тестами. При этом сохранение лишних элементов будет засорять карту и увеличивать время, затрачиваемое на нахождение нужных объектов в тестируемом приложении.

На рисунке 9 приведен пример карты NameMapping, созданной для приложения «Machining Cloud». Выделенный объект JobTabs представляет в данной карте вкладку страницы Job тестируемого приложения, приведенную на рисунке 10. Данный объект распознается по двум уникальным свойствам ObjectType со значением ScrollView и accessibilityHint со значением TabsScrollView (свойство добавлено в код тестируемого приложения разработчиками специально для того, чтобы облегчить процедуру идентификации объектов в процессе автоматизированного тестирования). В тестах обратиться к данному объекту теперь можно по пути NameMapping.Mobile.Device.process.MainWindow.JobTabs.

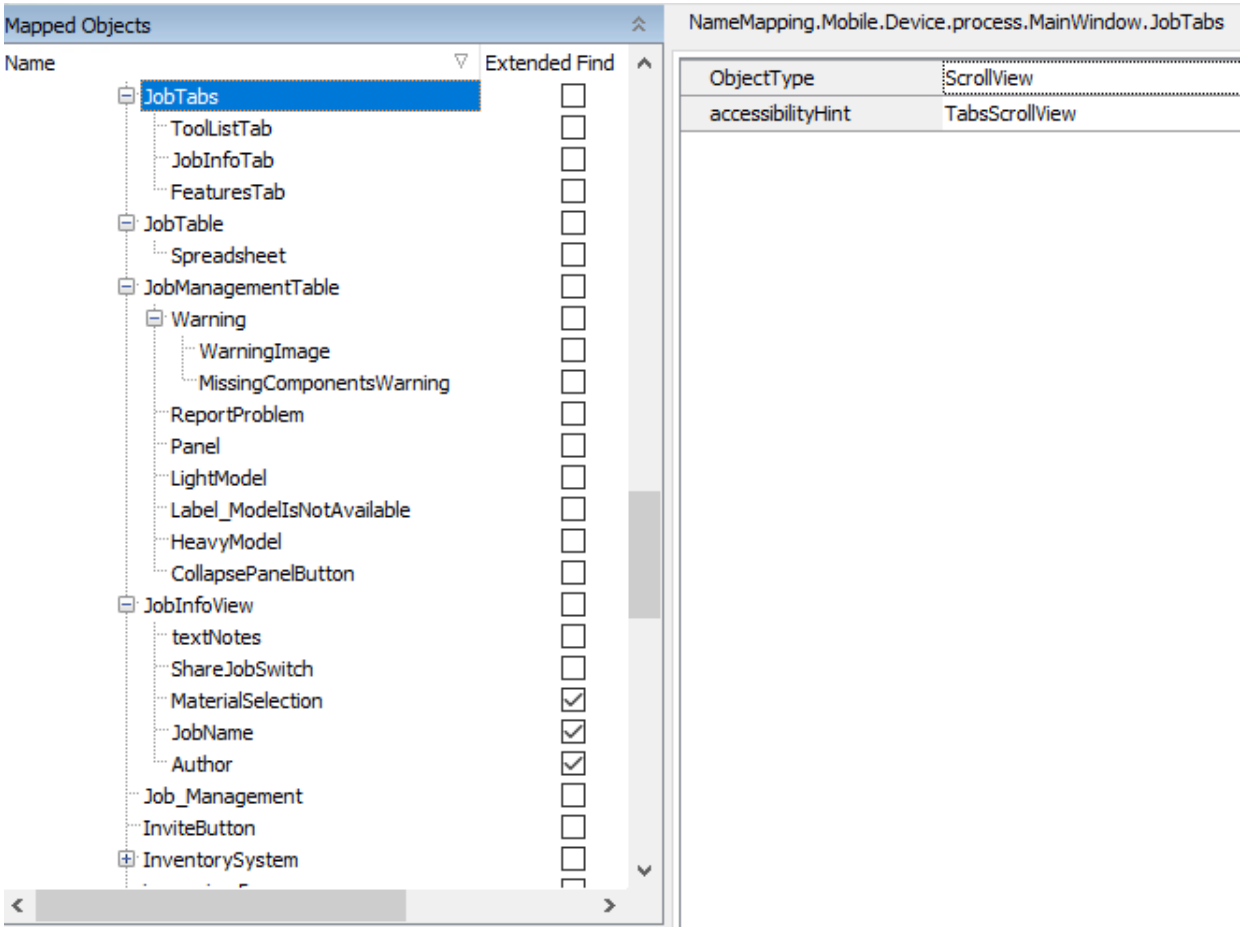


Рисунок 9 – Карта NameMapping

Рисунок 10 – Вкладки на странице Job тестируемого приложения

TestComplete позволяет записывать объекты в карту NameMapping автоматически, либо в ручном режиме. В ходе реализации данного проекта все объекты были записаны вручную. Это сделано для того, чтобы избежать проблем распознавания в будущем. Кроме того, плохо записанная карта объектов значительно снижает скорость выполнения тестов.

Свойства объектов выбраны действительно уникальные, объектам даны значимые имена для простоты поддержания тестов и читаемости кода. В свойствах использованы выражения условия (or), wildcards, в некоторых случаях применен метод extended search.

В качестве дополнения к NameMapping TestComplete позволяет использовать Aliases – карту псевдонимов. Это упрощенное иерархическое дерево объектов, куда можно добавить только те объекты, которые непосредственно используются в тестах, исключив все промежуточные узлы. Использование псевдонимов позволять давать объектам короткие имена и использовать их в коде тестов, что делает код более компактным и понятным. Карта Aliases также была создана в процессе реализации проекта. Использование одинаковых псевдонимов в проектах под iOS и Android позволило в дальнейшем переиспользовать значительную часть кода самих тестов.

На рисунке 11 приведена карта псевдонимов. Имя, по которому будет происходить обращение в тестах к данной вкладке, сократилось до Aliases.Device.Job.JobTabs.

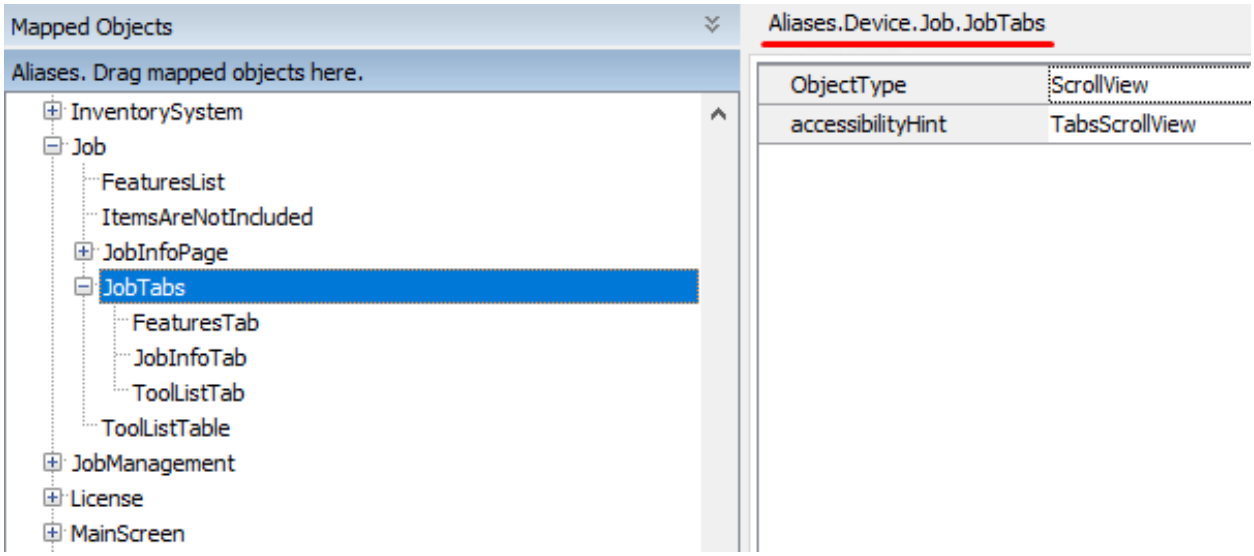


Рисунок 11 – Aliases – карта псевдонимов

Объекты пользовательского интерфейса тестируемого приложения, подверженные частым изменениям в ходе разработки, не были сохранены в карту NameMapping, был реализован их поиск непосредственно из кода с использованием методов FindChild() – поиск дочерних элементов.

Например, для выбора отдельного пункта из выпадающего списка меню используется следующий код:

```
function OpenJobMenuItem(item_name)
{
    Prop = new Array("ViewID", "ControlText");
```

```
    Value = new Array("text1", item_name);
    Aliases.Device.JobMenu.FindChild(Prop, Value, 3).Touch();
}
```

Нужный нам элемент интерфейса, соответствующий отдельному пункту меню, в карту имен не записан, поскольку пункты меню являются элементами интерфейса, подверженными изменениям (они могут выпадать в разном порядке, изменяются их названия и т.д.). В карте имен записан элемент интерфейса Aliases.Device.JobMenu – выпадающий список меню, являющийся родительским объектом по отношению к отдельным пунктам меню.

При выполнении данной функции будет осуществлен поиск дочернего элемента на глубину равную 3 от элемента Aliases.Device.JobMenu, удовлетворяющего следующим параметрам: свойство ViewID искомого элемента должно быть равным text1, а свойство ControlText – аргументу функции item_name (название пункта меню). Для найденного элемента интерфейса будет выполнена процедура Touch() – имитация прикосновения пальцем к элементу на экране устройства.

3.2.2 Организация пакета модулей Scripts.

Пакет Script содержит непосредственно разработанные программные скрипты. Как уже говорилось выше, он разбит на 2 компонента: Service functions и Tests.

В репозитории Service functions на каждую страницу тестируемого приложения (а также на элементы приложения, которые повторно используются на разных страницах, такие как панель навигации) созданы отдельные модули (units) с функциями, каждая из которых симулирует какой-то вид взаимодействия с UI этой страницы.

На рисунке 12 показан пример модулей пакета Service functions, созданных в ходе реализации проекта.

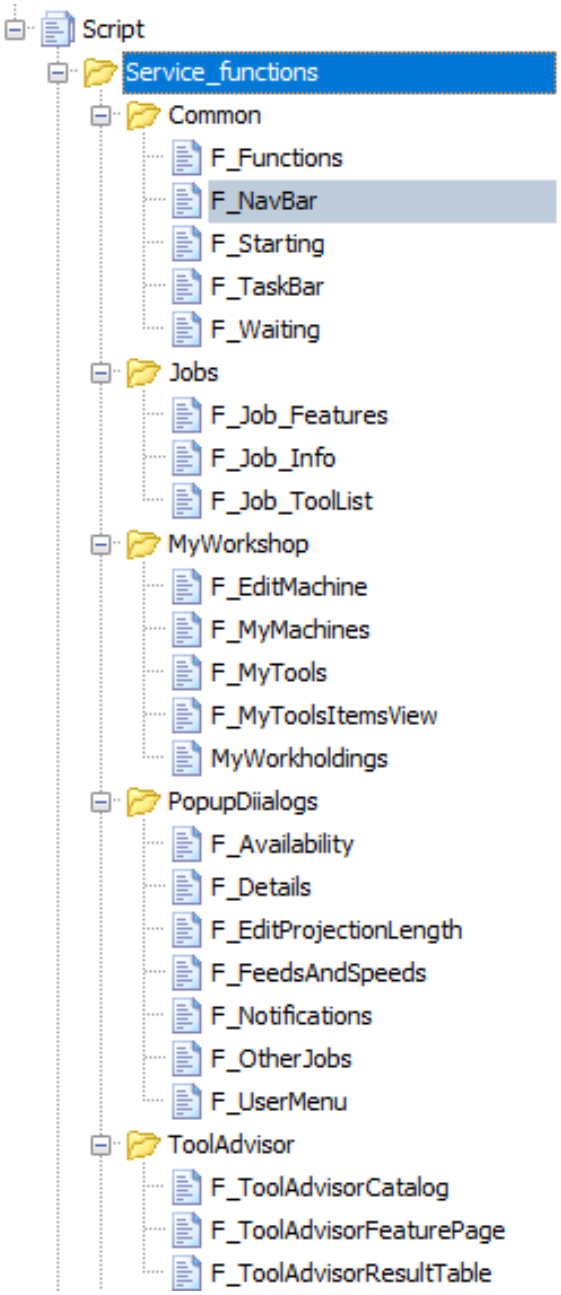


Рисунок 12 – Репозиторий вспомогательных функций Service functions

Примеры некоторых функций модуля репозитория F_NavBar, содержащего функции для взаимодействия с интерфейсом навигационной панели, приведены в приложении А1. Там же приведены примеры дальнейшего использования данных функций в тестах.

Пакет «Tests» также содержит разные модули, в которые объединены тесты, имеющие отношение к определенному функционалу приложения. В тестах нет прямого обращения к элементам интерфейса, используются функции

из пакета Service Functions, что позволяет использовать код самих тестов в проектах для разных платформ. На рисунке 13 – пакет реализованных модулей Tests.

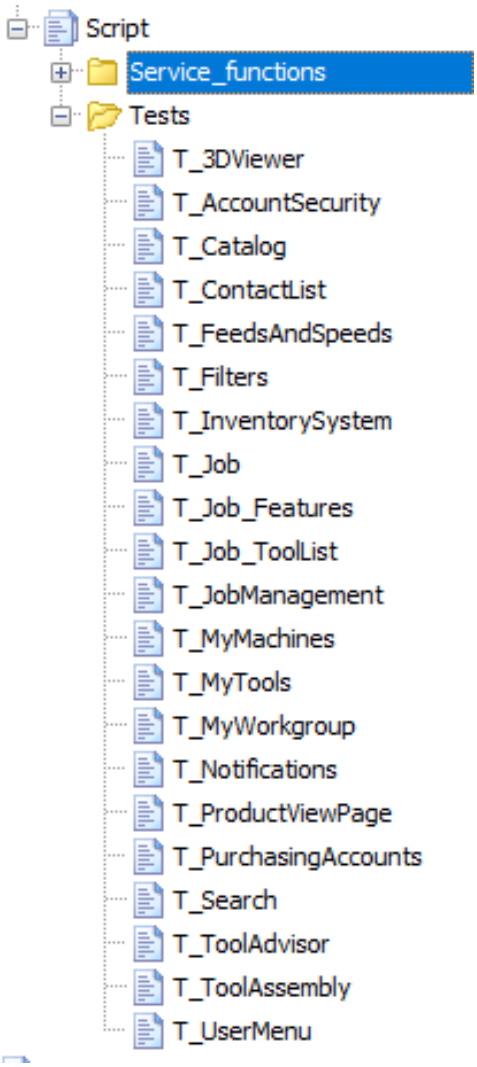


Рисунок 13 – Пакет модулей с тестовыми скриптами

Содержание одного из таких модулей приведено в приложении А2. Тесты разработаны на базе стандартных методик, предлагаемых для дизайна автоматизированных тестов для функционального тестирования.

В приложении также реализовано тестирование производительности. Тесты производительности используют те же функции из репозитория Service functions. В ходе выполнения тесты замеряют время, проходящее от клика на какой-то элемент страницы до загрузки соответствующего окна (элемента), это время записывается в отдельный файл. Пример тестов производительности

приведен в приложении А3. На рисунке 14 показан пример файла, полученного в результате прохождения тестов.

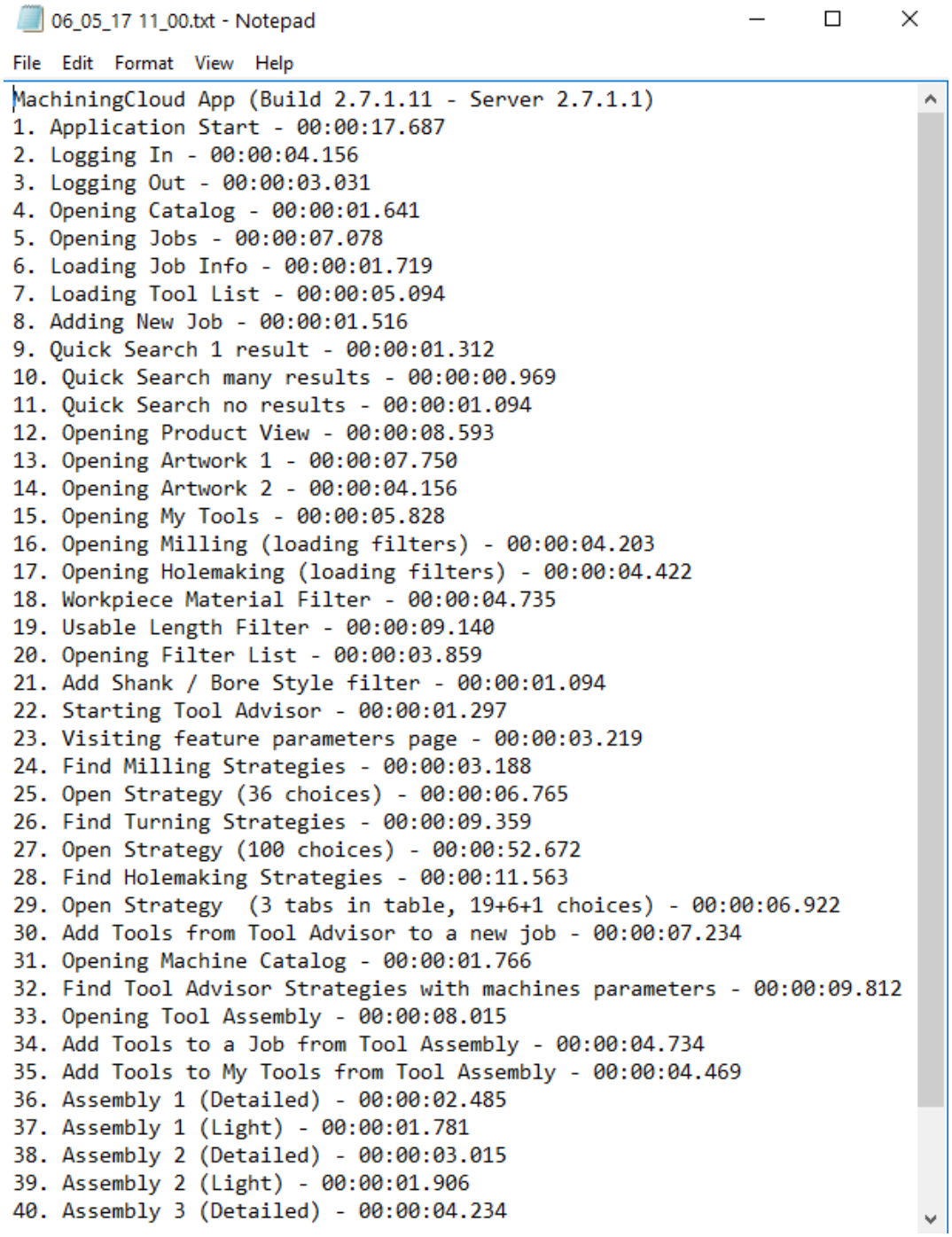


Рисунок 14 – Файл с результатами тестирования производительности

3.2.3 Реализация работы с жестами и сравнение по картинкам.

Для тестирования функционала приложения, связанного с визуализацией 3D-моделей каталога выбран метод сравнения изображений как наиболее

простой и доступный. Работа с моделями в мобильном приложении осуществляется с использованием стандартных жестов. TestComplete позволяет создать собственную библиотеку жестов и применять ее в будущем для имитации этих жестов во время выполнения тестирования.

В ходе реализации проекта была создана библиотека стандартных жестов, приведенная на рисунке 15.

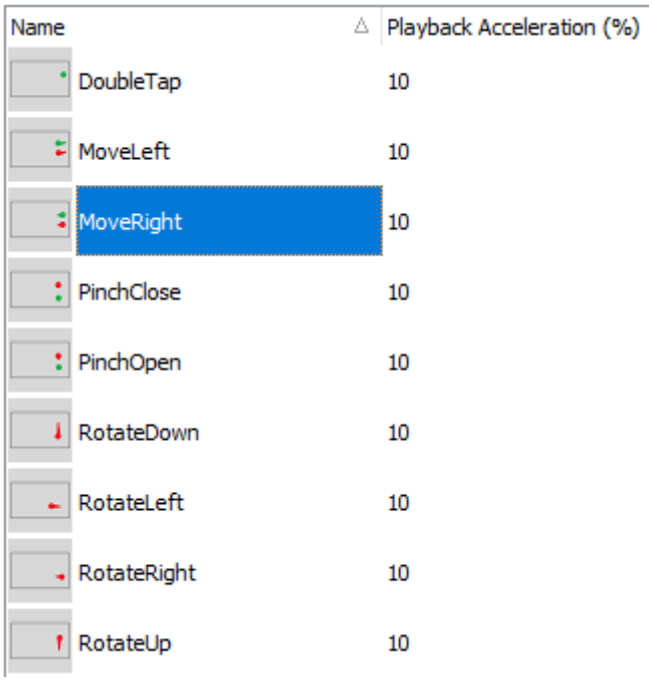
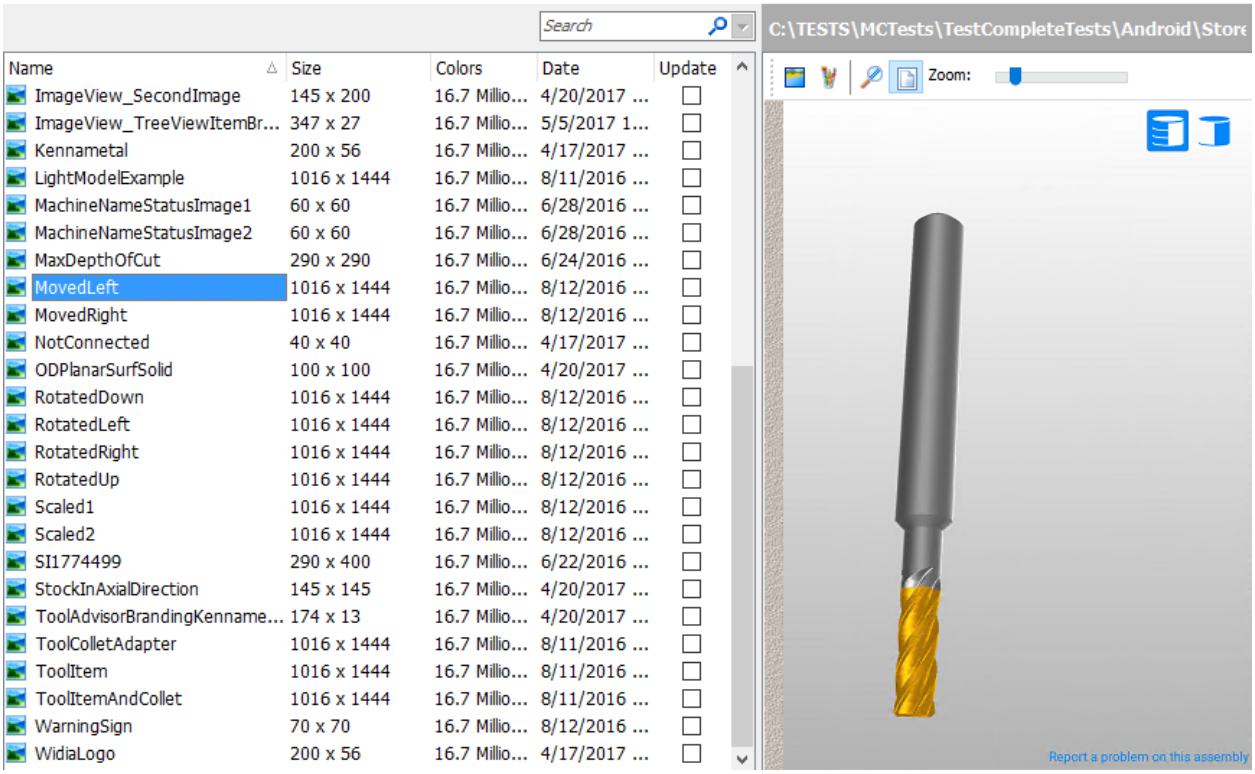


Рисунок 15 – Коллекция жестов в TestComplete

Также были сохранены изображения, использующиеся как эталонные для проверки отображения 3д-моделей. Содержание пакета Stores приведено на рисунке 16.



Name	Size	Colors	Date	Update
ImageView_SecondImage	145 x 200	16.7 Millio...	4/20/2017 ...	
ImageView_TreeViewItemBr...	347 x 27	16.7 Millio...	5/5/2017 1...	
Kennametal	200 x 56	16.7 Millio...	4/17/2017 ...	
LightModelExample	1016 x 1444	16.7 Millio...	8/11/2016 ...	
MachineNameStatusImage1	60 x 60	16.7 Millio...	6/28/2016 ...	
MachineNameStatusImage2	60 x 60	16.7 Millio...	6/28/2016 ...	
MaxDepthOfCut	290 x 290	16.7 Millio...	6/24/2016 ...	
MovedLeft	1016 x 1444	16.7 Millio...	8/12/2016 ...	
MovedRight	1016 x 1444	16.7 Millio...	8/12/2016 ...	
NotConnected	40 x 40	16.7 Millio...	4/17/2017 ...	
ODPlanarSurfSolid	100 x 100	16.7 Millio...	4/20/2017 ...	
RotatedDown	1016 x 1444	16.7 Millio...	8/12/2016 ...	
RotatedLeft	1016 x 1444	16.7 Millio...	8/12/2016 ...	
RotatedRight	1016 x 1444	16.7 Millio...	8/12/2016 ...	
RotatedUp	1016 x 1444	16.7 Millio...	8/12/2016 ...	
Scaled1	1016 x 1444	16.7 Millio...	8/12/2016 ...	
Scaled2	1016 x 1444	16.7 Millio...	8/12/2016 ...	
SI1774499	290 x 400	16.7 Millio...	6/22/2016 ...	
StockInAxialDirection	145 x 145	16.7 Millio...	4/20/2017 ...	
ToolAdvisorBrandingKenname...	174 x 13	16.7 Millio...	4/20/2017 ...	
ToolColletAdapter	1016 x 1444	16.7 Millio...	8/11/2016 ...	
ToolItem	1016 x 1444	16.7 Millio...	8/11/2016 ...	
ToolItemAndCollet	1016 x 1444	16.7 Millio...	8/11/2016 ...	
WarningSign	70 x 70	16.7 Millio...	8/12/2016 ...	
WidiaLogo	200 x 56	16.7 Millio...	4/17/2017 ...	

Рисунок 16 – Сохраненные изображения в модуле Stores

Для проверки соответствия изображения эталонному создана простая функция CheckModelIn3DViewer (проверка модели, размещенной в просмотрщике 3D), в качестве аргумента использующая сохраненные картинки из Stores:

```
function CheckModelIn3DViewer(ExpectedImage)
{
    ExpectedImage.Check(Aliases.Device.ModelViewer.PaintingView, false,
false, 342300, 60);
}
```

В качестве аргумента в данную функцию передается название сохраненного эталонного изображения. Данное изображение сравнивается с изображением, полученным из элемента интерфейса, записанного в карту имен под псевдонимом `Aliases.Device.ModelViewer.PaintingView`. В данном случае сравнение производится без использования масок, но с настройками толерантности по количеству отличающихся пикселей и по их цвету.

Для тестирования 3D Viewer написаны тесты, использующие жесты из библиотеки жестов и функцию сравнения картинок. Пример подобного теста:

```
function ModelMoving()
{
    FindToolAndOpenAssembly("2391783");
    Open3DViewer();

    AndroidGestureCollection.Default.MoveRight.Execute();
    CheckModelIn3DViewer(Regions.MovedRight);

    AndroidGestureCollection.Default.MoveLeft.Execute();
    CheckModelIn3DViewer(Regions.MovedLeft);

    AndroidGestureCollection.Default.DoubleTap.Execute();
    CheckModelIn3DViewer(Regions.DefaultPosition);

    GoHome();
    AgreeAssemblyExit();
}
```

Тест проверяет перемещение моделей в просмотрщике 3д тестируемого приложения. В начале теста выполняется ряд подготовительных действий: поиск нужного инструмента по его номеру в каталоге, открытие страницы сборки данного инструмента и открытие просмотрщика 3д для данного инструмента. Затем для данной страницы осуществляется выполнение жеста из библиотеки MoveRight, соответствующего «перетаскиванию» элемента интерфейса по экрану вправо, производится сравнение картинки, отображаемой в просмотрщике с сохраненной эталонной, соответствующей изображению можели, перемещенной вправо. Те же операции осуществляются с перетаскиванием модели влево и с осуществлением жеста DoubleTap – двойное прикосновение к экрану устройства, при котором модель в приложении занимает начальное положение. В случае несоответствия картинок в лог будет записана ошибка и тест будет помечен как «не пройденный». После выполнения теста выполняется возврат на начальный экран приложения и закрытие окна просмотра инструментальной сборки инструмента.

3.2.4 Ведение логирования, визуализация результатов.

При автоматическом запуске тестов важно иметь возможность быстро и легко проанализировать результаты тестов. TestComplete предоставляет

удобные инструменты для логирования, но этого не всегда оказывается достаточно. При дефолтных настройках TestComplete делает скриншоты на каждый шаг теста и записывает их в лог-файл. Это очень удобно для того, чтобы проанализировать результаты, но в случае включения тестов в процесс непрерывной интеграции, когда тесты запускаются с каждым собранным билдом приложения, хранить такое количество больших лог-файлов становится проблематичным.

В данном проекте было решено хранить скриншоты последних 5 шагов до возникновения ошибки, но при этом писать в лог дополнительную текстовую информацию, позволяющую более продуктивно читать лог. В настройках TestComplete был выбран соответствующий режим, а функции репозитория были разработаны с реализацией записи дополнительной информации в логи непосредственно из скриптов. Это позволило не просто «ронять» тест в функциях проверок в случае несоответствия ожидаемому результату, но и писать при этом дополнительные данные в лог-файл, полезные для дальнейшего анализа.

Например, функция проверки заголовка продукта реализована следующим образом:

```
function CheckProductTitle(expected_title)
{
    if (Aliases.Device.ProductView.Title.ControlText ==
    expected_title) {
        Log.Message("Product title is correct")}
    else{
        Log.Error("Wrong product title: " + Aliases.De-
vice.ProductView.Title.ControlText + "is not equal to expected " + ex-
pected_title);
    }
}
```

В случае, если свойство ControlText элемента интерфейса заголовка страницы ProductView совпадает с переданным в функцию в качестве аргумента expected_title, в лог пишется информационное сообщение «Product title is correct» (Название продукта правильное). Если название не совпадает, пи-

шется сообщение об ошибке в формате: Неправильное название продукта: *[заголовок продукта, полученный из интерфейса]* не совпадает с ожидаемым *[аргумент, переданный в функцию]*.

На рисунке 17 приведен пример лога теста, обнаружившего ошибку (при нажатии на кнопку добавления системы инвентаризации добавлялось 2 инвентарные системы). Лог читаем, ошибку легко увидеть и идентифицировать.

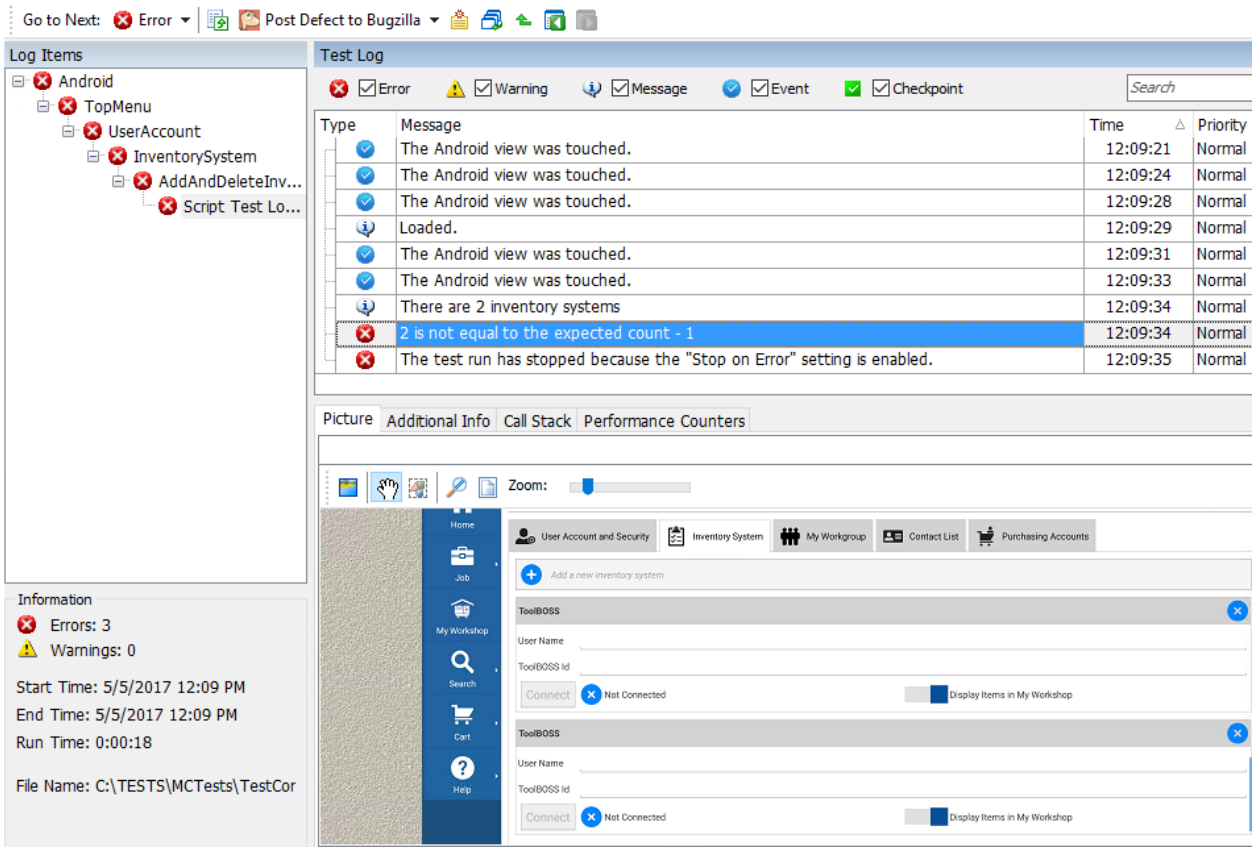


Рисунок 17 – Лог-файл упавшего теста

На рисунке 18 приведен отчет об успешном прохождении сета тестов.

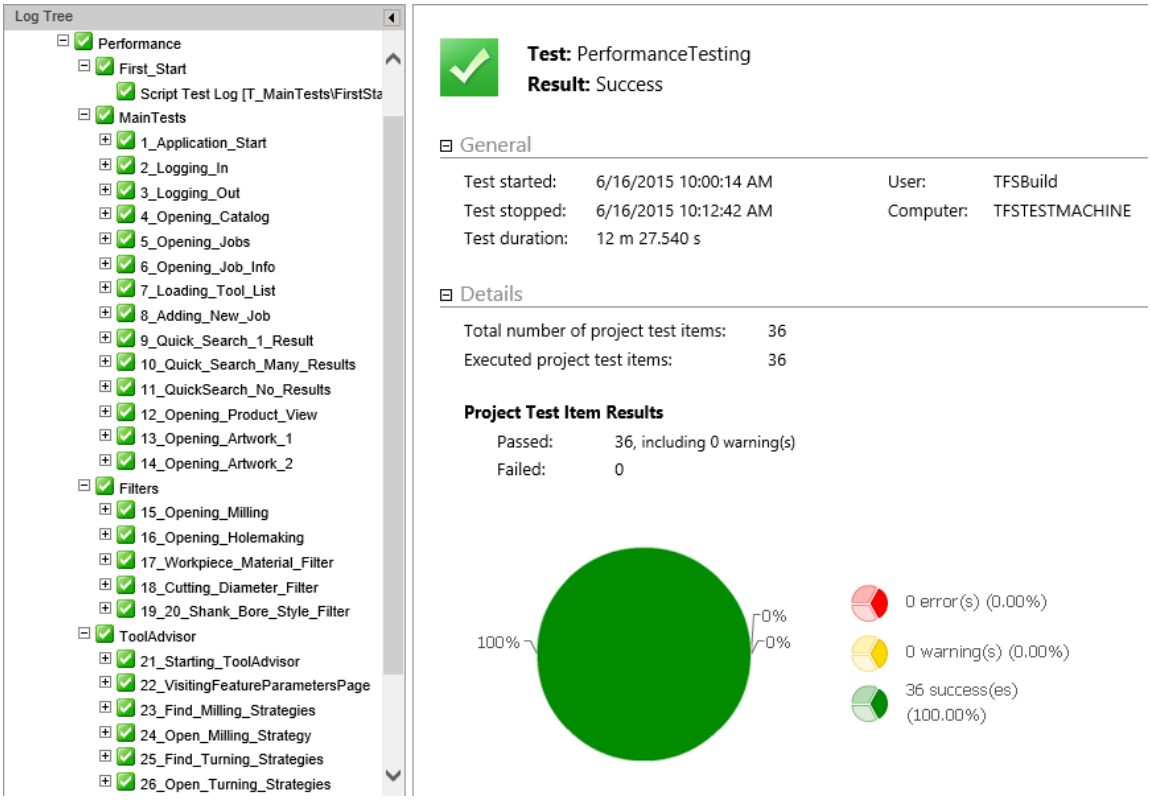


Рисунок 18 – Лог-файл успешного прохождения сета тестов

3.2.5 Организация тестов и конфигурация тестовых наборов.

Поскольку прохождение полного комплекта тестов занимает длительное время, целесообразно разбить тесты на отдельные сеты, в зависимости от проверяемого ими функционала. Это позволит в дальнейшем более гибко использовать тесты.

В менеджере тестов TestComplete была разработана структура тестовых пакетов, приведенная на рисунке 19.

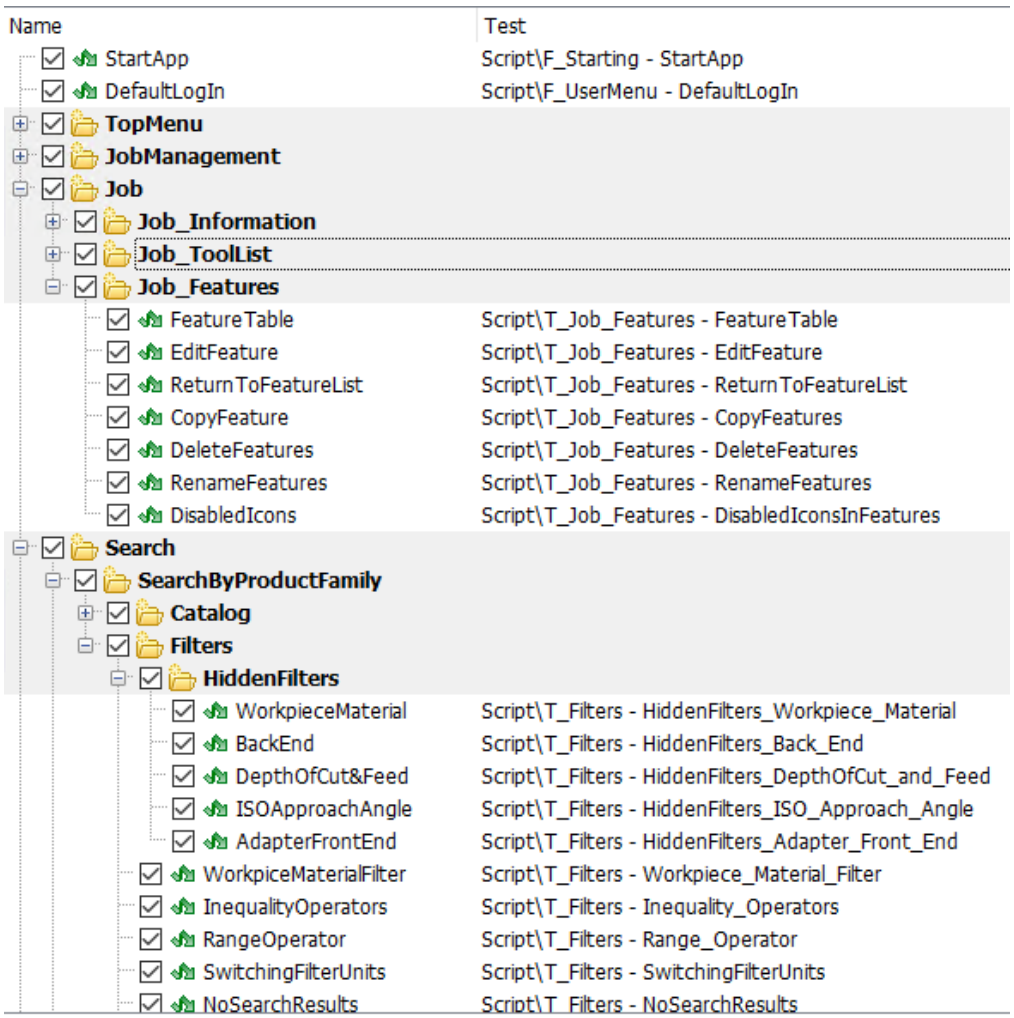


Рисунок 19 – Организация тестов в тест-сеты

Подобные сеты тестов можно использовать для запуска на разных этапах тестирования приложения.

4 РЕЗУЛЬТАТЫ ВЫПОЛНЕННОЙ РАБОТЫ И ВЫВОДЫ

В ходе выполнения квалификационной работы были разработаны следующие пакеты тестов:

- UserMenu (панель пользователя). Содержит разбитые на более мелкие пакеты тесты для тестирования диалогов авторизации и регистрации пользователей, центра оповещений, экранов с настройками пользователя (настройки безопасности, подключение систем инвентаризации, рабочие группы пользователя, списки контактов, платежные аккаунты).
- JobManagement (менеджер проектов). Тесты, проверяющие работу менеджера проектов: создание, удаление, копирование, пересылка проектов и т.п.
- Job (проект). Пакет тестов для проверки функционала, связанного с проектами. Включает в себя отдельные пакеты с тестами, тестирующими окно настроек проекта, окна для работы со списками инструментов и фичерсов внутри проекта.
- Search (поиск). Включает пакеты для тестирования Поиска по каталогу, фильтров, Поиска с помощью Советника по подбору инструмента, Быстрого поиска и Поиска по индустрии.
- ProductView (страница продуктов). Тестирует функционал, связанный со страницей, содержащей информацию о продуктах: таблицы с данными о продукте, переключение единиц измерения, всплывающие подсказки, добавление продуктов в проекты, мои инструменты и т.д.
- ToolAssembly (инструментальная сборка). Тестирует функционал экрана сборки комплектов инструмента: редактирование сборок, добавление различного вида компонентов, редактирование вылета инструмента и т.д.
- 3D Viewer (просмотрщик 3д). Тестирует использование жестов для взаимодействия с моделями (масштабирование, перемещение, вращение и

т.д.), переключение между антиколлизионными моделями и моделями для визуализации, изменение моделей при добавлении компонентов в инструментальную сборку и т.д.

- FeedsAndSpeeds (скорости и подачи). Пакет тестов диалогового окна скоростей и подач инструмента. Тестирует работу с разными единицами измерения, работу с дефолтными значениями и рекомендациями, задание параметров с использованием слайдеров и ручной ввод, предупреждения, выводимые в диалоге в случаях отсутствия информации, неверных параметрах , превышении допустимых значений для выбранного станка и т.д.

- MyWorkshop (моя мастерская). Содержит пакеты тестов для тестирования функционала MyMachines (Мои станки), MyTools (Мои инструменты), MyWorkholdings (Мои держатели).

- Performance (производительность). Пакет тестов для тестирования производительности приложения.

Все разработанные в результате выполнения квалификационной работы тесты в настоящее время используются в процессе тестирования приложения Machining Cloud в компании DP Labs.

В первую очередь тесты используются для регрессионного тестирования с целью обнаружения ошибок, возникающих в уже проверенном ранее, рабочем функционале приложения, который может быть поврежден в ходе работы над новыми задачами. Часть тестов включена в процесс непрерывной интеграции (Continuous Integration) и запускается для каждого нового билда приложения вместе с unit-тестами на симуляторах, установленных на сервере. Эта практика позволяет выявлять ошибки на самых ранних стадиях возникновения, когда их проще локализовать и исправить. Произвольные, уже более полные пакеты тестов запускаются тестировщиками при проверке нового функционала на реальных устройствах, чтобы убедиться, что добавленные фичи не повредили созависимые с ним области.

При предрелизном тестировании весь комплект тестов прогоняется на большом количестве различных реальных устройств и симуляторов.

Тесты оказалось удобным использовать и программистам в процессе отладки. При запуске на слабых девайсах в циклическом режиме тесты помогают выявлять утечки памяти.

Тесты производительности также запускаются автоматически на сервере. Этот комплект тестов запускается по расписанию один раз в день с последним собранным билдом приложения в одно и то же время. Собранные данные анализируются раз в две недели и позволяются отследить регрессию в производительности приложения.

Введение автоматизации тестирования пользовательского интерфейса сделало процесс тестирования более эффективным, быстрым и качественным, значительно сократило время, затрачиваемое на ручное тестирование. При этом написание и поддержка автоматизированных тестов сами по себе требуют достаточно больших временных затрат и не высвобождают человеческие ресурсы, занятые на тестировании, но зато наличие автоматизированных тестов дает возможность более гибкого распоряжения этими ресурсами, уменьшает количество ситуаций дедлайнов на проекте, расширяет возможности тестирования, за счет чего значительно улучшает качество самого разрабатываемого продукта и делает процесс разработки более организованным и продуктивным.

5 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

ВВЕДЕНИЕ

Целью данного раздела является оценка эффективности автоматизации тестирования пользовательских интерфейсов в процессе разработки программного обеспечения, обоснование целесообразности использования для автоматизации тестирования коммерческих средств разработки, выявление сильных и слабых сторон такого решения.

При разработке сложных, многолетних программных продуктов при добавлении даже незначительной части изменений в код программы высок риск возникновения ошибок в старом функционале, поэтому процесс регрессионного тестирования (проверки работоспособности ранее протестированных участков программы) занимает на таких проектах огромное количество времени и человеческих ресурсов.

Кроме того, современный рынок требует от разрабатываемых приложений поддерживать всю линейку современных устройств в огромном количестве конфигураций, тестирование работы приложения на таком количестве конфигураций без использования автоматизированных средств практически нереально.

Внедрение автоматизации тестирования на предприятиях, разрабатывающих программное обеспечение, позволит снизить количество ручного тестирования, улучшить качество приложения и сделать сам процесс разработки быстрым и высокотехнологичным.

Актуальность разработки

1. Экономическая актуальность

Реализация проекта автоматизации позволит значительно сократить время, затрачиваемое на осуществление процесса тестирования в процессе разработки программного продукта, и тем самым уменьшить его стоимость.

Кроме того, чем меньше времени проходит с момента возникновения ошибки до момента ее обнаружения, тем меньше времени потребуется разработчикам для локализации ошибки и ее исправления. Таким образом, сокращается и время, затрачиваемое программистами на исправление ошибок в программе.

2. Социальная актуальность

Данная разработка позволит улучшить качество и надежность разрабатываемого продукта, а соответственно и удовлетворенность пользователя данным продуктом. Также улучшаются условия работы специалистов по тестированию в самой компании, где внедряется данный проект, исключив из работы рутинные операции.

3. Техническая актуальность

Разработанная система позволит осуществлять запуск тестов одновременно на большом количестве устройств разных конфигураций, осуществлять тестирование производительности.

Снижает риск поломки оборудования и возникновения аварий на производстве, где используется программное обеспечение, автоматизации тестирования которого посвящена разработка.

Цели и задачи разработки

Целью разработки является автоматизация процесса функционального тестирования и тестирования пользовательского интерфейса программного продукта, тестирования производительности.

Для достижения поставленной цели нужно решить следующие задачи:

- анализ области автоматизации тестирования программного обеспечения;
- выбор инструмента для автоматизации тестирования;
- разработка проекта по автоматизации тестирования;
- создание нескольких наборов автоматических тестов;

- реализовать автоматический запуск тестов на эмуляторах и реальных устройствах.

Критерии эффективности

Таблица 1 – Критерии эффективности разработки

Тип показателя	Показатель
экономические	- снижение временных затрат на осуществление тестирования; - снижение временных затрат на исправление ошибок в программном обеспечении.
Социальные	- увеличение удовлетворенности пользователя программным продуктом; - улучшение условий работы тестировщиков.
Технические	- увеличение скорости процесса тестирования; - увеличение количества устройств, которые можно охватить в процессе тестирования; - увеличение надежности разрабатываемого программного продукта.

5.1 Оценка коммерческого потенциала и перспективности проведения научных исследований с позиции ресурсоэффективности и ресурсосбережения

5.1.1 Потенциальные потребители результатов исследования

Для анализа потребителей результатов исследования необходимо рассмотреть целевой рынок и разбить его на сегменты.

В первую очередь результаты данной разработки будут интересны компаниям производителям программного обеспечения, занимающимся долгосрочными, сложными, дорогими проектами. К таким типам программного обеспечения можно отнести профессиональный софт – инженерный, медицинский, образовательный.

В силу высокой стоимости и трудозатратности реализации предложенного решения данную разработку целесообразно применять в крупных и средних по размеру компаниях, и нецелесообразно в мелких.

Как видно из таблицы 2, основной сегмент потребителей, на которых нацелена данная разработка – крупные компании, занимающиеся производством профессионального программного обеспечения, либо крупные производители мобильных приложений, требующих тестирования на большом количестве устройств.

Таблица 2 – Карта сегментирования рынка

		Программное обеспечение				
		развле-кательное	общего назначения	образова-тельное	професси-ональное	мобильные приложения
Потребители	Крупные компании					
	Средние компании					
	Стартапы					

В будущем возможно применение данной технологии для автоматизации тестирования в компаниях, разрабатывающих образовательный софт и программное обеспечение общего назначения.

5.1.2 Анализ конкурентных технических решений

В качестве конкурентных решений можно рассмотреть:

1) процесс ручного тестирования, без использования средств автоматизации, при котором тестирование продукта производится командой низкоквалифицированных тестировщиков;

2) процесс тестирования с разработкой автоматических тестов с использованием бесплатных инструментов для тестирования, например Appium.

В таблице 3 приведено сравнение разработанного в ходе выполнения выпускной квалификационной работы решения с двумя данными альтернативными решениями.

Анализ конкурентных технических решений определяется по формуле:

$$К = \sum Вi \cdot Бi, \quad (1)$$

, где К – конкурентоспособность научной разработки или конкурента;

Вi – вес показателя (в долях единицы);

Бi – балл i-го показателя.

Таблица 3 – Оценочная карта сравнения конкурентных решений

Критерии оценки	**Вес критерия**	**Баллы**			**Конкурентоспособность**		
		$Б_ф$	$Б_{к1}$	$Б_{к2}$	$К_ф$	$К_{к1}$	$К_{к2}$
1	2	3	4	5	7	8	9
Технические критерии оценки ресурсоэффективности							
1. Функциональные возможности	0.1	5	3	1	0.5	0.3	0.1
2. Скорость выполнения	0.3	5	1	3	1.5	0.3	0.9
3. Стабильность результатов	0.25	4	4	3	1	1	0.75
4. Простота создания тестов	0.05	3	5	3	0.15	0.25	0.15
5. Простота поддержания тестов	0.2	4	5	1	0.8	1	0.2
Экономические критерии оценки эффективности							
6. Цена	0.07	3	3	5	0.21	0.21	0.35
7. Поддержка продукта	0.03	5	5	2	0.15	0.15	0.06
Итого	1				4.31	3.21	2.51

Сильными сторонами разработанного решения являются скорость выполнения тестов, стабильность их результатов и простота поддержания написанных тестов. Сильной стороной конкурентного решения использовать ручное тестирование является отсутствие необходимости прилагать усилия для поддержания тестов в случае использования ручного тестирования.

Интересно, что в результате анализа получилось, что использование ручного тестирования выигрывает в сравнении с использованием бесплатных продуктов для автоматизации, предоставляющих небогатые возможности и требующих больших усилий для поддержания тестов.

5.1.3 SWOT-анализ

На основе анализа конкурентных решений была составлена матрица SWOT-анализа, в которой показаны сильные и слабые стороны проекта, возможности и угрозы для разработки. Матрица SWOT представлена в таблице 4.

Таблица 4 – SWOT-анализ

	Сильные стороны научно-исследовательского проекта: С1. Снижение затрат на сотрудников, занимающихся тестированием. С2. Улучшение качества разрабатываемого продукта. С3. Легкость поддержания тестов. С4. Широкие возможности для переиспользования тестов, гибкость и масштабируемость. С5. Возможность параллельного запуска тестов на разных девайсах.	**Слабые стороны научно-исследовательского проекта:** Сл1. Высокая стоимость разработки. Сл2. Зависимость от стороннего инструмента. Сл3. Сложность внедрения.
Возможности: В1. Развитие IT-бизнеса, рост доходов в сегменте компаний среднего размера, высвобождение средств, которые могут быть использованы на автоматизацию тестирования. В2. Увеличение спроса на разработку сложных программных продуктов и мобильных приложений на рынке.	**Направления развития:** В1С2, В2С1С2С5 Добавлять новый функционал, позволяющий задействовать сферы, где еще задействовано ручное тестирование. Организовать запуск в облачных хранилищах, организовать нагрузочное тестирование, чтобы улучшить качество тестируемого продукта.	**Сдерживающие факторы:** В1Сл1, В2Сл1 Написать фреймворк с готовыми функциями, используя которые разработкой сможет заниматься менее квалифицированный сотрудник. Искать новые проекты, в которых возможно применить ту же технологию автоматизации, через это минимизировать разработку общих для обоих продуктов частей кода.
Угрозы: У1. Появление конкурентных продуктов или технологий для автоматизации. У2. Прекращение поддержки среды разработки производителем.	**Угрозы развития:** У2С3С4 Подготовиться к возможному варианту перехода на другой инструмент, усилить декомпозицию кода, использование абстракций, чтобы в случае угрозы была	**Уязвимости:** У1Сл1, У2Сл2 Попытаться снизить стоимость разработки за счет частично использования дешевых продуктов конкурентов.

	возможность переиспользовать бизнес-логику тестов даже при смене инструмента.	Попытаться быть менее зависимыми от среды разработки и быть готовыми к прекращению поддержки.

На втором этапе проведения SWOT-анализа проводится составление интерактивных матриц проекта, в которых производится анализ соответствия параметров SWOT каждого с каждым. Соотношения параметров представлены в таблицах 5-8.

Таблица 5 – Интерактивная матрица для сильных сторон и возможностей

Сильные стороны проекта						
Возможности проекта		С1	С2	С3	С4	С5
	В1	-	+	-	-	-
	В2	+	+	-	-	+

Таблица 6 – Интерактивная матрица для слабых сторон и возможностей

Слабые стороны проекта				
Возможности проекта		Сл1	Сл2	Сл3
	В1	+	-	-
	В2	+	-	-

Таблица 7 – Интерактивная матрица для сильных сторон и угроз

Сильные стороны проекта							
Угрозы проекта		С1	С2	С3	С4	С5	С6
	У1	-	-	-	-	-	-
	У2	-	-	+	+	-	-

Таблица 8 – Интерактивная матрица для слабых сторон и угроз

Слабые стороны проекта				
Угрозы проекта		Сл1	Сл2	Сл3
	У1	+	-	-
	У2	-	+	-

5.1.4 Определение возможных альтернатив проведения научных исследований

Для определения альтернативных путей проведения научных исследований и вариантов реализации технической части был использован морфологический подход. Созданная морфологическая матрица представлена в таблице 9.

Таблица 9 – Морфологическая матрица

	1	**2**	**3**
А. Фреймворк для разработки	Ranorex	Appium	TestComplete
Б. Технология реализации	Script	Capture/Playback	Keyword-based
В. Язык разработки	C#	Java	JScript
Г. Место запуска тестов	Реальные устройства	Эмуляторы	Облачные хранилища
Д. Разработчик	Высоко-квалифицированный программист	Программист низкой квалификации	Тестировщик без навыков программирования

Из полученной морфологической матрицы можно выделить 3 варианта реализаций проекта:

- Исполнение 1. А1Б3В1Г3Д3
- Исполнение 2. А2Б2В2Г3Д2
- Исполнение 3. А3Б1В3Г1Д2

5.2 Планирование научно-исследовательских работ

Структура работ в рамках научного исследования

Для организации работы в рамках научного исследования необходимо составить полный перечень работ и определить занятость каждого участника в проекте. Такой перечень приведен в таблице 10.

Таблица 10 – Перечень этапов, работ и распределение исполнителей

Основные этапы	№ работ	Содержание работ	Должность исполнителя

Разработка задания	1	Постановка задач	Руководитель
Выбор направления исследования	2	Анализ тестируемого приложения	Студент
	3	Изучение инструмента автоматизации, технологий	Студент
Реализация	4	Написание тестового плана	Студент
	5	Настройка инструмента/ создание проекта	Студент
	6	Написание тестов внутри созданного проекта	Студент
	7	Настройка среды для запуска	Студент
Тестирование системы	8	Отладка тестов в среде для запуска тестов	Студент
Ввод системы в эксплуатацию	9	Настройка автозапуска тестов, включение их в CI	Студент
Анализ и оформление результатов	10	Оценка полученных результатов	Руководитель

Определение трудоемкости выполнения работ

Трудоемкость выполнения работ рассчитывается по следующим формулам:

$$t_{\text{ожі}} = \frac{3t_{\min i} + 2t_{\max i}}{5}$$

Где $t_{ожі}$ – ожидаемая трудоемкость выполнения і-ой работы чел.-дн.;

$t_{\min i}$ – минимально возможная трудоемкость выполнения заданной і-ой работы (оптимистическая оценка), чел.-дн.;

$t_{\max i}$ – максимально возможная трудоемкость выполнения заданной і-ой работы (пессимистическая оценка), чел.-дн.

Исходя из ожидаемой трудоемкости работ, определяется продолжительность каждой работы в рабочих днях $T_р$, учитывающая параллельность выполнения работ несколькими исполнителями. Такое вычисление необходимо для обоснованного расчета заработной платы, так как удельный вес зарплаты в общей сметной стоимости научных исследований составляет около 65 %.

$$T_{\text{р}i} = \frac{t_{\text{ожі}}}{\text{Ч}_i}$$

Где $T_{\text{р}i}$ – продолжительность одной работы, раб. дн.;

$t_{\text{ожі}}$ – ожидаемая трудоемкость выполнения одной работы, чел.-дн.;

Ч_i – численность исполнителей, выполняющих одновременно одну и ту же работу на данном этапе, чел.

Расчеты продолжительности работ представлены в таблице 11.

Разработка графика проведения научного исследования

Для сравнительно небольших по объему научных работ, наиболее удобным и наглядным является построение ленточного графика проведения научных работ в форме диаграммы Ганта.

Диаграмма Ганта – горизонтальный ленточный график, на котором работы по теме представляются протяженными во времени отрезками, характеризующимися датами начала и окончания выполнения данных работ.

Для удобства построения графика, длительность каждого из этапов работ из рабочих дней следует перевести в календарные дни. Для этого необходимо воспользоваться следующей формулой:

$$T_{\text{к}i} = T_{\text{р}i} \cdot k_{\text{кал}},$$

где $T_{\text{к}i}$ – продолжительность выполнения *i*-й работы в календарных днях;

$T_{\text{р}i}$ – продолжительность выполнения *i*-й работы в рабочих днях;

$k_{\text{кал}}$ – коэффициент календарности.

Коэффициент календарности определяется по следующей формуле:

$$k_{\text{кал}} = \frac{T_{\text{кал}}}{T_{\text{кал}} - T_{\text{вых}} - T_{\text{пр}}},$$

где $T_{\text{кал}}$ – количество календарных дней в году;

$T_{\text{вых}}$ – количество выходных дней в году;

$T_{\text{пр}}$ – количество праздничных дней в году.

Рассчитанные значения в календарных днях по каждой работе $T_{\text{к}i}$ необходимо округлить до целого числа.

Для расчета коэффициент календарности подсчитаем количество рабочих и выходных дней в 2017 году. Всего в году 247 рабочих дней и 118 выходных и праздничных дней. Исходя из полученных данных, рассчитывается коэффициент календарности:

$$k_{\text{кал}} = \frac{T_{\text{кал}}}{T_{\text{кал}} - T_{\text{вых}} - T_{\text{пр}}} = \frac{365}{247} = 1{,}4778$$

На основе таблицы 11 построен календарный план-график проведения работ (табл. 12).

Таблица 11 – Временные показатели проведения научного исследования

Название Работы	**Исполнители**	**Трудоёмкость работ**									**Длительность работ в рабочих днях** T_{pi}			**Длительность работ в календарных днях** T_{ki}		
		t_{min}, чел-дни			t_{max}, чел-дни			$t_{ожi}$, чел-дни								
		Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3
Постановка задач	руководитель	1,00	1,00	1,00	2,00	2,00	2,00	1,40	1,40	1,40	1,40	1,40	1,40	2,07	2,07	2,07
Анализ тестируемого приложения	тестировщик	3,00	4,00	3,00	3,00	4,00	3,00	3,00	4,00	3,00	3,00	4,00	3,00	4,43	5,91	4,43
Изучение инструмента автоматизации, технологий	тестировщик	5,00	10,00	5,00	6,00	15,00	6,00	5,40	12,00	5,40	5,40	12,00	5,40	7,98	17,73	7,98
Написание тестового плана	тестировщик	7,00	9,00	7,00	11,0	13,00	11,00	8,60	10,60	8,60	8,60	10,60	8,60	12,71	15,66	12,71
Настройка инструмента/ создание проекта	тестировщик	1,00	6,00	3,00	3,00	8,00	5,00	1,80	6,80	3,80	1,80	6,80	3,80	2,66	10,05	5,62
Написание тестов внутри созданного проекта	тестировщик	15,00	10,00	20,00	20,00	15,00	25,00	17,00	12,00	22,00	17,00	12,00	22,00	25,12	17,73	32,51
Настройка среды для запуска	тестировщик	3,00	1,00	1,00	5,00	2,00	2,00	3,80	1,40	1,40	3,80	1,40	1,40	5,62	2,07	2,07
Отладка тестов в среде для запуска тестов	тестировщик	3,00	1,00	1,00	5,00	2,00	2,00	3,80	1,40	1,40	3,80	1,40	1,40	5,62	2,07	2,07
Настройка автозапуска тестов, включение их в CI	тестировщик	4,00	6,00	6,00	6,00	8,00	8,00	4,80	6,80	6,80	4,80	6,80	6,80	7,09	10,05	10,05
Оценка полученных результатов	тестировщик	1,00	1,00	1,00	2,00	2,00	2,00	1,40	1,40	1,40	1,40	1,40	1,40	2,07	2,07	2,07
Итого								51,0	57,8	55,2	51,0	57,8	55,2	75,4	85,4	81,6

Таблица 12 – Календарный план-график проведения работ

№ работ	Вид работ	Исполнители	$T_{кi}$, кал. дн.	Продолжительность выполнения работ											
				Февраль			март			Апрель			май		
				1	2	3	1	2	3	1	2	3	1	2	3
1	Постановка задач	Руководитель	2												
2	Анализ тестируемого приложения	Тестировщик	6												
3	Изучение инструмента автоматизации, технологий	Тестировщик	18												
4	Написание тестового плана	Тестировщик	16												
5	Настройка инструмента/ создание проекта	Тестировщик	10												
6	Написание тестов внутри созданного проекта	Тестировщик	18												
7	Настройка среды для запуска	Тестировщик	2												
8	Отладка тестов в среде для запуска тестов	Тестировщик	2												
9	Настройка автозапуска тестов, включение их в CI	Тестировщик	10												
10	Оценка полученных результатов	Руководитель	2												

– Руководитель

– Тестировщик

5.3 Бюджет научно-технического исследования

В состав бюджета работ по научно-технической работе входит стоимость всех расходов, необходимых для их выполнения. При формировании бюджета используется группировка затрат по следующим статьям:

- основная заработная плата исполнителей темы;
- дополнительная заработная плата исполнителей темы;
- отчисления во внебюджетные фонды (страховые отчисления);
- накладные расходы.

Расчет материальных затрат НТИ

Данная статья включает стоимость всех материалов, используемых при разработке проекта.

Расчет материальных затрат осуществляется по следующей формуле:

$$З_м = (1 + k_T) \cdot \sum_{i=1}^{m} Ц_i \cdot N_{расхi}$$

где m - количество видов материальных ресурсов, потребляемых при выполнении научного исследования;

Nрасхi – количество материальных ресурсов *i*-го вида, планируемых к использованию при выполнении научного исследования (шт,, кг, м, м2 и т.д.);

$Ц_i$ – цена приобретения единицы *i*-го вида потребляемых материальных ресурсов (руб./шт., руб./кг, руб./м, руб./м^2 и т.д.);

k_T – коэффициент, учитывающий транспортно-заготовительные расходы.

Величина коэффициента (k_T), отражающего соотношение затрат по доставке материальных ресурсов и цен на их приобретение, зависит от условий договоров поставки, видов материальных ресурсов, территориальной удаленности поставщиков и т.д. Транспортные расходы принимаются в пределах 15-25% от стоимости материалов. Материальные затраты, необходимые для данной разработки, заносятся в таблицу 13.

Таблица 13 – Материальные затраты

Наименование	Ед. измерения	Количество			Цена за ед., Тыс.руб.			Затраты на материалы, (**Зм**), тыс.руб.		
		Исп. 1	Исп. 2	Исп. 3	Исп. 1	Исп. 2	Исп .3	Исп.1	Исп. 2	Исп.3
Амортизация оборудования	шт.	1	2	2	3.3	3.3	3.3	3.3	6.6	6.6
TestComplete	шт.	0	0	1	0	0	170	0	0	170
Ranorex	шт.	1	0	0	147	0	0	147	0	0
Облачное хранилище	мес.	1	0	0	4	0	0	4	0	0
							Итого	154.3	6.6	176.6

В ходе разработки использовалось имеющееся оборудование, поэтому в материальные расходы внесены затраты на его амортизацию за 4 месяца.

Также в материальные расходы занесены затраты на приобретение лицензий для используемого программного обеспечения и на оплату подписки на облачный парк девайсов для запуска тестов на 4 месяца. Эмуляторы для запуска тестов используются бесплатные.

Расчет основной заработной платы исполнителей системы

В данную статью расходов включается заработная плата научного руководителя и студентов, а также премии и доплаты. Расчет выполняется на основе трудоемкости выполнения каждого этапа и величины почасовой оплаты работы исполнителей.

Основной расчет фонда заработной платы выполняется по формуле:

$$З_{зп} = З_{осн} + З_{доп}$$

где $З_{осн}$ – основная заработная плата;

$З_{доп}$ – дополнительная заработная плата (12-20 % от $З_{осн}$).

Основная заработная плата ($З_{осн}$) руководителя (лаборанта, инженера) от предприятия (при наличии руководителя от предприятия) рассчитывается по следующей формуле:

$$З_{осн} = З_{дн} \cdot T_p ,$$

где $З_{осн}$ – основная заработная плата одного работника;

$T_р$ – продолжительность работ, выполняемых научно-техническим работником, раб. часов.

$З_{дн}$ – среднедневная заработная плата работника, руб.

Таблица 14 – Расчет основной заработной платы

Название Работы	**Исполнители**	**Трудоемкость, чел.-дн.**			**Заработная плата, приходящаяся на один чел.-дн., тыс. руб.**			**Всего зара-ботная плата по тарифу (окладам), тыс. руб.**		
		Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3	Исп.1	Исп.2	Исп.3
Постановка задач	руководитель	1,40	1,40	1,40	1,7	1,7	1,7	2,38	2,38	2,38
Анализ тестируемого приложения	тестировщик	3,00	4,00	3,00	4,5	2	4,5	13,5	8	13,5
Изучение инструмента автоматизации, технологий	тестировщик	5,40	12,00	5,40	4,5	2	4,5	24,3	24	24,3
Написание тестового плана	тестировщик	8,60	10,60	8,60	4,5	2	4,5	38,7	21,2	38,7
Настройка инструмента/ создание проекта	тестировщик	1,80	6,80	3,80	4,5	2	4,5	8,1	13,6	17,1
Написание тестов внутри созданного проекта	тестировщик	17,00	12,00	22,00	4,5	2	4,5	76,5	24	99
Настройка среды для запуска	тестировщик	3,80	1,40	1,40	4,5	2	4,5	17,1	2,8	6,3
Отладка тестов в среде для запуска тестов	тестировщик	3,80	1,40	1,40	4,5	2	4,5	17,1	2,8	6,3
Настройка автозапуска тестов, включение их в CI	тестировщик	4,80	6,80	6,80	4,5	2	4,5	21,6	13,6	30,6
Оценка полученных результатов	руководитель	1,40	1,40	1,40	1,7	1,7	1,7	2,38	2,38	2,38
Итого								221,7	114,8	240,6

Среднедневная заработная плата рассчитывается по формуле:

$$З_{дн} = \frac{З_м \cdot М}{F_д},$$

где $З_м$ – месячный должностной оклад работника, руб.;

М – количество месяцев работы без отпуска в течение года (для научного руководителя – 11.2 месяца; для студента – 10.4 месяцев);

$F_д$ – действительный годовой фонд рабочего времени научно-технического персонала, раб. дн. (для научного руководителя – 190 раб дн., для студента – 178 раб. дн.)

Отчисления во внебюджетные фонды (страховые отчисления)

В данной статье расходов отражаются обязательные отчисления по установленным законодательством Российской Федерации нормам органам государственного социального страхования (ФСС), пенсионного фонда (ПФ) и медицинского страхования (ФФОМС) от затрат на оплату труда работников. Величина отчислений во внебюджетные фонды определяется исходя из следующей формулы:

$$З_{внеб} = k_{внеб} \cdot (З_{осн} + З_{доп})$$

где $k_{внеб}$ – коэффициент отчислений на уплату во внебюджетные фонды (пенсионный фонд, фонд обязательного медицинского страхования и пр.).

На 2017 г. в соответствии с Федеральным законом от 24.07.2009 №212-ФЗ установлен размер страховых взносов равный 30%.

Отчисления во внебюджетные фонды представлены в таблице 18.

Таблица 18 – Отчисления во внебюджетные фонды

Исполнитель	**Основная заработная плата, руб.**		
	Исп.1	Исп.2	Исп.3
Руководитель проекта	4250	4250	4250
Тестировщик	217450	110500	236350
Итого	221700	114800	240600
Коэффициент отчислений во внебюджетные фонды	0,3		

Итого	
Исполнение 1	**66510**
Исполнение 2	**34440**
Исполнение 3	**72180**

Расчет накладных расходов

Накладные расходы учитывают прочие затраты организации, не попавшие в предыдущие статьи расходов: печать и ксерокопирование материалов исследования, оплата услуг связи, электроэнергии, размножение материалов и т.д. Их величина определяется по следующей формуле:

$$З_{накл} = (\text{сумма статей } 1 \div 7) \cdot k_{нр}$$

где $k_{нр}$ – коэффициент, учитывающий накладные расходы.

Величину коэффициента накладных расходов можно взять в размере 16%. В рассматриваемом случае были использованы лишь 4 статьи, поэтому деление производится на 4.

Для исполнения 1: $З_{накл} = 66510 * 0,16 = 10641,6$

Для исполнения 2: $З_{накл} = 34440 * 0,16 = 5510,4$

Для исполнения 3: $З_{накл} = 72180 * 0,16 = 11548,8$

Формирование бюджета затрат научно-исследовательского проекта

Сумма затрат по всем статьям расходов рассчитывается заносится на данном этапе в таблицу 19.

Таблица 19 – Бюджет затрат научно-исследовательского проекта

Наименование статьи	**Сумма, руб.**			Примечание
	Исп.1	Исп.2	Исп.3	
1. Материальные затраты НТИ	154300	6600	176600	Пункт 4.1
2. Затраты по основной заработной плате исполнителей темы	221700	114800	240600	Пункт 4.2
3. Отчисления во внебюджетные фонды	66510	34440	72180	Пункт 4.4

4. Накладные расходы	10641,6	5510,4	11548,8	16 % от суммы ст. 1-4
5. Бюджет затрат НТИ	453151,6	161350,4	500928,8	Сумма ст. 1- 5

5.4 Определение ресурсной, финансовой, бюджетной, социальной и экономической эффективности исследования

Определение эффективности происходит на основе расчета интегрального показателя эффективности научного исследования. Его нахождение связано с определением двух средневзвешенных величин: финансовой эффективности и ресурсоэффективности.

$$I_{\text{финр}}^{\text{исп.}i} = \frac{\Phi_{\text{p}i}}{\Phi_{\max}}$$

где $I_{\text{финр}}^{\text{исп.i}}$ – интегральный финансовый показатель разработки;

$\Phi_{\text{p}i}$ – стоимость *i*-го варианта исполнения;

$\Phi_{\max}$ – максимальная стоимость исполнения научно-исследовательского проекта (в т.ч. аналоги).

Исполнение 1: $I_{\text{финр}} = 453151,6/500928,8 = 0,9$

Исполнение 2: $I_{\text{финр}} = 161350,4/500928,8 = 0,3$

Исполнение 3: $I_{\text{финр}} = 500928,8/500928,8 = 1$

Полученная величина интегрального финансового показателя разработки отражает соответствующее численное удешевление стоимости разработки в разах.

Интегральный показатель ресурсоэффективности вариантов исполнения объекта исследования можно определить следующим образом:

$$\text{I}_{\text{pi}} = \sum a_i \cdot b_i ,$$

где I_{pi} – интегральный показатель ресурсоэффективности для i-го варианта исполнения разработки;

a_i – весовой коэффициент *i*-го варианта исполнения разработки;

b_i^a, b_i^p – бальная оценка *i*-го варианта исполнения разработки, устанавливается экспертным путем по выбранной шкале оценивания;

n – число параметров сравнения.

Расчет интегрального показателя ресурсоэффективности рекомендуется проводить в форме таблицы (табл. 20).

Таблица 20 – Сравнительная оценка характеристик вариантов исполнения проекта

Объект исследования / Критерии	Весовой коэффициент параметра	Исп.1	Исп.2	Исп.3
1. Функциональные возможности	0,1	4	2	5
2. Скорость выполнения	0,2	3	4	5
3. Стабильность результатов	0,2	3	2	4
4. Простота создания тестов	0.1	4	5	4
5. Простота поддержания тестов	0,2	1	1	5
6. Гибкость и масштабируемость	0,1	1	1	5
7. Кроссплатформенность	0,1	4	2	5
ИТОГО	1			

$I_{р\text{-}исп1} = 0,1 * 4 + 0,2 * 3 + 0,2 * 3 + 0,1 * 4 + 0,2 * 1 + 0,1 * 1 + 0,1 * 4 = 2,7$

$I_{р\text{-}исп2} = 0,1 * 2 + 0,2 * 4 + 0,2 * 2 + 0,1 * 5 + 0,2 * 1 + 0,1 * 1 + 0,1 * 2 = 2,4$

$I_{р\text{-}исп3} = 0,1 * 5 + 0,2 * 5 + 0,2 * 4 + 0,1 * 4 + 0,2 * 5 + 0,1 * 5 + 0,1 * 5 = 13,7$

Интегральный показатель эффективности вариантов исполнения разработки ($I_{испi.}$) определяется на основании интегрального показателя ресурсоэффективности и интегрального финансового показателя по формуле:

$$I_{исп.1} = \frac{I_{р-исп1}}{I_{финр}^{исп.1}}$$

$I_{исп1} = 2,7 / 0,9 = 3$

$I_{исп2} = 2,4 / 0,3 = 8$

$I_{испЗ} = 13,7 / 1 = 13,7$

Полученное значение интегрального показателя эффективности исполнения разработки значительно выше исполнений конкурентных решений. Таким образом, результат работы можно считать положительным, так как оценка интегрального показателя ресурсоэффективности очень высокая и это оправдывает использование такого дорогого решения.

Сравнение интегрального показателя эффективности вариантов исполнения разработки позволит определить сравнительную эффективность проекта (см.табл.18) и выбрать наиболее целесообразный вариант из предложенных.

Сравнительная эффективность проекта ($Э_{ср}$):

$$Э_{ср} = \frac{I_{исп.i}}{I_{исп.\min}}$$

Сравнительная эффективность разработки представлена в таблице 21.

Таблица 21 – Сравнительная эффективность разработки

№ п/п	Показатели	Исп.1	Исп.2	Исп.3
1	Интегральный финансовый показатель разработки	0,9	0,3	1
2	Интегральный показатель ресурсоэффективности разработки	2,7	2,4	13,7
3	Интегральный показатель эффективности	3	8	13,7
4	Сравнительная эффективность вариантов исполнения	1	2,6	4,6

ЗАКЛЮЧЕНИЕ

В ходе оценки перспективности и альтернатив проведения научного исследования с позиции ресурсоэффективности и ресурсосбережения коммерческого потенциала для выпускной квалификационной работы были определены

потенциальные потребители – крупные и среднего размера компании, занимающиеся разработкой мобильных приложений или инженерного программного обеспечения.

Был сделан анализ конкурентных решений, выявлены сильные и слабые стороны разработанного решения, его возможности и угрозы, а также корреляция этих показателей, определенных в ходе SWOT-анализа.

Исходя из полученных данных и проведенного анализа эффективности, можно сделать вывод, что выбранный вариант исполнения является наиболее эффективным за счет своих возможностей, что делает его разработку целесообразной, несмотря на то, что выбранное решение достаточно дорогое.

6 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

ВВЕДЕНИЕ

В данном разделе рассмотрены вопросы, связанные с организацией рабочего места и условий в которых будет реализовываться разработка системы автоматизации процесса тестирования пользовательского интерфейса кросс-платформенного мобильного приложения "Machining Cloud".

Рабочим местом является офисное помещение открытого типа, арендуемое компанией ООО «Рубиус Групп». Так называют вариант планировки офисного помещения, при котором отсутствуют отдельные кабинеты специалистов, а в большом по площади помещении располагают несколько рабочих мест для сотрудников, не отгороженных друг от друга, и отдельные кабинеты для руководителей высшего звена, закрытые переговорные комнаты и помещение для отдыха.

В данном разделе рассмотрены такие вредные производственные факторы, оказывающие негативное влияние на организм и здоровье сотрудников предприятия, как электромагнитное излучение, неоптимальный микроклимат помещения, недостаточность естественного освещения, повышенный уровень шума и такой опасный фактор, как электрический ток. Также указан характер вредного воздействия данных факторов на организм и последствия их длительного или чрезмерного воздействия.

Также рассмотрены чрезвычайные ситуации, которые могут произойти на рабочем месте и действия, которые необходимо выполнить в случае их возникновения.

6.1 Производственная безопасность

Перечень опасных и вредных факторов, характерных для рабочего места, оборудованного ПЭВМ представлен в таблице 1.

Таблица 22 – Опасные и вредные факторы при выполнении работ с использованием рабочего места на базе персонального компьютера, оснащенного жидкокристаллическим дисплеем.

Источник фактора, наименование видов работ	**Факторы (по ГОСТ 12.0.003-2015)**		**Нормативные документы**
	Вредные	Опасные	
Разработка системы автоматизации процесса тестирования пользовательского интерфейса кросс-платформенного мобильного приложения "Machining Cloud", связанная с длительным пребыванием на рабочем месте, оборудованном ПЭВМ	1. Повышенная напряженность электромагнитного поля [1]; 2. Умственное перенапряжение [1]; 3. Отсутствие или недостаток естественного света [1]; 4. Аномальные микроклиматические параметры воздушной среды [1] 4. Длительная работа в сидячем положении [1];	1. Электрический ток.	Гигиенические требования к персональным электронно-вычислительным машинам и организации работы устанавливаются СанПиН 2.2.2/2.4.1340-03I [2]. Параметры электромагнитного излучения устанавливаются СанПиН 2.2.4.3359-16 Требования к естественному и искусственному освещению устанавливаются СП 52.13330.2011 Параметры микроклимата устанавливаются СанПиН 2.2.4.548-96

	5. Высокая нагрузка на зрительный аппарат [1].		Требования по пожарной безопасности устанавливаются ГОСТ 12.1.004-91 [6]. Требования к организации труда устанавливаются Трудовым кодексом Российской Федерации от 30.12.2001 N 197-ФЗ [8]. Требования при выполнении работ сидя устанавливаются ГОСТ 12.2.032-78 [9].

Повышенная напряженность электромагнитного поля

Источниками повышенной напряженности электромагнитного поля в данном случае могут являться персональный компьютер в составе системного блока и двух ЖК-мониторов. Благоприятным фактором является отсутствие у ЖК-мониторов накопления статического электрического заряда на рабочей поверхности экрана и отсутствие тормозного ионизирующего излучения, характерного для дисплеев на основе электронно-лучевых трубок.

Электромагнитная волна, распространяясь от источника в неограниченном пространстве со скоростью света, создает электромагнитное поле (ЭМП), способное воздействовать на заряженные частицы и токи, в результате чего происходит превращение энергии поля в другие виды энергии. Электромагнитное поле представляет собой особую форму материи, состоящую из взаимосвязанных электрического и магнитного полей.

Физические причины существования переменного электромагнитного поля связаны с тем, что изменения во времени электрического поля порождают магнитное поле, а изменения магнитного поля – вихревое электрическое поле.

Характеристикой электрической составляющей ЭМП является напряженность электрического поля в вольтах на метр, а характеристиками магнитной – напряженность магнитного поля в амперах на метр, и магнитная индукция в теслах.

Длительное воздействие электромагнитного поля промышленной частоты (50 Гц), излучаемое электрической сетью, обеспечивающей энергоснабжение помещения, приводит к расстройствам в головном мозге и центральной нервной системе. У человека могут наблюдаться головная боль в височной и затылочной областях, вялость, ухудшение памяти, боли в области сердца, угнетенное настроение, апатия, своеобразная депрессия с повышенной чувствительностью к яркому свету и интенсивному звуку, расстройство сна, сердечно-сосудистой системы, органов пищеварения, дыхания, повышенная раздражительность. Могут наблюдаться функциональные нарушения в центральной нервной системе, а также изменения в составе крови.

Воздействие переменного магнитного поля с частотой 50 Гц на человека заключается в индуцировании в теле человека вихревых токов. При длительном систематическом воздействии могут возникнуть расстройства функционального состояния нервной системы, иммунной системы и сердечно-сосудистой системы. Длительное воздействие ЭМП промышленной частоты может спровоцировать онкологические заболевания.

В соответствии с СанПиН 2.2.4.3359-16, предельно допустимые уровни электромагнитных полей на рабочих местах пользователей персональными компьютерами (ПК) и другими средствами информационно-коммуникационных технологий (ИКТ) представлены в таблице 4.2 [2].

Таблица 23 – ПДУ электромагнитных полей на рабочих местах пользователей ПК и другими средствами ИКТ

Наименование параметров		**ВДУ**
Напряженность электрического поля	в диапазоне частот 5 Гц-2 кГц	25 В/м
	в диапазоне частот 2 кГц-400 кГц	2,5 В/м
Плотность магнитного потока	в диапазоне частот 5 Гц-2 кГц	250 нТл
	в диапазоне частот 2 кГц-400 кГц	25 нТл
Напряженность электростатического поля		15 кВ/м

Для уменьшения уровня электромагнитного поля от персонального компьютера рекомендуется соблюдать комфортное расстояние до монитора, так как напряженность электрической составляющей убывает пропорционально расстоянию в третьей степени [3]. При хронических заболеваниях органов зрения (близорукость, астигматизм и т.д.) следует применять соответствующие, подобранные врачом-офтальмологом, средства коррекции, так как отсутствие коррекции вынуждает сокращать расстояние до монитора, что приводит как к перегрузке органов зрения, так и к расположению головы в зоне электромагнитных полей более высокой интенсивности.

К средствам индивидуальной защиты при работе на компьютере относят спектральные компьютерные очки для улучшения качества изображения, защиты от избыточных энергетических потоков видимого света и для профи-

лактики «компьютерного зрительного синдрома». Очки уменьшают утомляемость глаз на 25-30 %. Их рекомендуется применять всем операторам при работе более 2 ч в день, а при нарушении зрения на 2 диоптрии и более – независимо от продолжительности работы [4].

Умственное перенапряжение

Умственное перенапряжение возникает в результате того, что в процессе работы приходится изучать и усваивать большие объемы информации. Разработка и реализация алгоритмов также требует большого психоэмоциональных усилий, направленных на концентрацию внимания. Для устранения влияния данного негативного фактора можно применять общепринятые методики – чередование видов умственной нагрузки, перерывы в работе, дыхательная гимнастика.

Отсутствие или недостаток естественного освещения

Выполнение зрительной работы, как с монитором персонального компьютера, так и с бумажными документами при недостаточной освещенности рабочего места может привести к развитию некоторых дефектов глаз: близорукость ложная и истинная; дальнозоркость истинная и старческая, а такжну к высокой утомляемости и снижению трудоспособности.

К офисному освещению независимо от источника света предъявляются следующие требования:

– достаточная освещенность, т. е. освещенность объекта должна обеспечить комфортные условия для общей работоспособности;

– равномерность освещения, т. е. освещенность должна быть равномерной во времени и пространстве;

– отсутствие блескости в поле зрения работающих.

Рассмотрим некоторые характеристики освещения.

Освещенность – поверхностная плотность светового потока; единица освещенности – люкс.

Яркость – поверхностная плотность силы света в данном направлении, которая определяется из отношения силы света излучаемой поверхности в этом направлении к проекции светящейся поверхности на плоскость, перпендикулярную данному направлению. Единица яркости – кандела на квадратный метр.

Коэффициент отражения p характеризует способность поверхности отражать падающий на нее световой поток. Если $p < 0{,}2$ – фон считается темным; если $0{,}2 < p < 0{,}4$ – средним; при $p > 0{,}4$ – светлым.

Контраст объекта с фоном K определяется из соотношения яркостей рассматриваемого объекта и фона.

Контраст объекта с фоном считается малым, если $K < 0{,}2$, средним – при $0{,}2 < K < 0{,}5$ и большим при $K > 0{,}5$.

Освещение в производственных помещениях в светлое время суток осуществляется естественным источником света – небосводом. Естественное освещение может быть боковым (через окна), верхним (через зенитные фонари) и комбинированным. Применение той или иной системы естественного освещения зависит от назначения и размеров помещения, расположения его в плане здания, а также от светового климата местности.

Интенсивность естественного освещения оценивается коэффициентом естественного освещения (КЕО), показывающего, во сколько раз освещенность в помещении меньше освещенности наружной.

В соответствии с СП 52.13330.2011 к освещению помещений промышленных предприятий применяются требования, представленные в приложении А [5].

Неоптимальный микроклимат помещения

Показатели микроклимата должны обеспечивать сохранение теплового баланса человека с окружающей средой и поддержание оптимального или допустимого теплового состояния организма.

Показателями, характеризующими микроклимат в производственных помещениях, являются:

а) температура воздуха;

б) температура поверхностей;

в) относительная влажность воздуха;

г) скорость движения воздуха;

д) интенсивность теплового облучения.

Оптимальные микроклиматические условия установлены по критериям оптимального теплового и функционального состояния человека. Они обеспечивают общее и локальное ощущение теплового комфорта в течение 8-часовой рабочей смены при минимальном напряжении механизмов терморегуляции, не вызывают отклонений в состоянии здоровья, создают предпосылки для высокого уровня работоспособности и являются предпочтительными на рабочих местах.

Оптимальные величины показателей микроклимата необходимо соблюдать на рабочих местах производственных помещений, на которых выполняются работы операторского типа, связанные с нервно-эмоциональным напряжением (в кабинах, на пультах и постах управления технологическими процессами, в залах вычислительной техники и др.).

В соответствии с СанПиН 2.2.4.548-96, оптимальные величины параметров микроклимата на рабочих местах применительно к выполнению работ различных категорий в холодный и теплый периоды года приведены в таблице 3 [6].

Таблица 24 Оптимальные величины показателей микроклимата на рабочих местах производственных помещений

Период года	Категория работ по уровню энергозатрат, Вт	Температура воздуха, °С	Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с
Холодный	Ia (до 139)	22-24	21-25	60-40	0,1
Теплый	Ia (до 139)	23-25	22-26	60-40	0,1

Допустимые величины показателей микроклимата на рабочих местах должны соответствовать значениям, приведенным в таблице 4.5 применительно к выполнению работ различных категорий в холодный и теплый периоды года.

Таблица 25 – Допустимые величины показателей микроклимата на рабочих местах производственных помещений

Период года	Категория работ по уровню энерготрат, Вт	Температура воздуха, °С		Температура поверхностей, °С	Относительная влажность воздуха, %	Скорость движения воздуха, м/с	
		диапазон ниже оптимальных величин	диапазон выше оптимальных величин			для диапазона температур воздуха ниже оптимальных величин, не более	для диапазона температур воздуха выше оптимальных величин, не более
Холодный	Ia (до 139)	20,0-21,9	24,1-25,0	19,0-26,0	15-75	0,1	0,1
Теплый	Ia (до 139)	21,0-22,9	25,1-28,0	20,0-29,0	15-75	0,1	0,2

Отклонение параметров микроклимата от нормативных значений существенно влияет на здоровье и производительность труда. Высокая температура вызывает интенсивное потоотделение, что приводит к обезвоживанию организма, потере минеральных солей и водорастворимых витаминов С, В1, В2. Нарушение кратности воздухообмена приводит к росту концентрации углекислого газа, что, в свою очередь, негативно сказывается на здоровье сотрудников.

В офисных помещениях ООО «Рубиус Групп» микроклимат поддерживается системой вентиляции и кондиционирования, регулярно проходящей техническое обслуживание и замер показателей эффективности работы и кратности воздухообмена.

Длительная работа в сидячем положении

Основными повреждающими здоровье при любой сидячей работе, являются следующие неспецифичные (т.е. не связанные именно с работой за компьютером) факторы:

- Длительная гиподинамия. Любая поза при длительной фиксации вредна для опорно-двигательного аппарата, кроме того, ведет к застою крови во внутренних органах и капиллярах.
- Нефизиологическое положение различных частей тела.
- Длительно повторяющиеся однообразные движения. Здесь вредна не только усталость тех групп мышц, которые эти движения выполняют, но и психологическая фиксация на них (образование устойчивых очагов возбуждения ЦНС с компенсаторным торможением других ее участков). Хотя наиболее вредны именно повторяющиеся однообразные нагрузки. Через усталость они могут вести к физическому повреждению суставов и сухожилий. Наиболее известен в среде пользователей PC тендовагинит запястных сухожилий, связанный с вводом информации посредством мыши и клавиатуры.

Минимизировать вышеперечисленные негативные факторы можно с помощью производственной гимнастики, а также правильной организации рабочего места.

Конструкция рабочего стола должна обеспечивать оптимальное разме-

щение на рабочей поверхности используемого оборудования с учетом его количества и конструктивных особенностей, характера выполняемой работы. При этом допускается использование рабочих столов различных конструкций, отвечающих современным требованиям эргономики. Используемый стул (кресло) должно обеспечивать подгонку по высоте и углу наклона спинки.

Поверхность сиденья, спинки и других элементов стула (кресла) должна быть полумягкой, с нескользящим, слабо электризующимся и воздухопроницаемым покрытием, обеспечивающим легкую очистку от загрязнений.

Высокая нагрузка на зрительный аппарат

Длительные зрительные нагрузки (а в рассматриваемом рабочем процессе зрительная нагрузка составляет практически весь рабочий день) могут вызывать ряд патологических состояний зрительного аппарата. Возникновение ложной близорукости, нарушение тонуса глазодвигательных мышц, компьютерный зрительный синдром, прогрессирование миопии.

Профилактические меры представляют собой элементарные упражнения, помогающие предотвратить развитие спазма аккомодации. Важно периодически переводить взгляд с экрана монитора вдаль, выбрав какой-то предмет в комнате.

Через каждые 40-60 мин работы необходимо делать перерыв на 10-15 минут. При этом следует встать со своего рабочего места, выполнить несколько простых физических упражнений, которые снимут усталость с мышц.

6.2 Экологическая безопасность

Так как основным объектом исследования данной работы являются электрические приборы, серьезной проблемой является электропотребление. Это влечет за собой общий рост объема потребляемой электроэнергии. Для

удовлетворения потребности в электроэнергии, приходиться увеличивать мощность и количество электростанций. Это приводит к нарушению экологической обстановки, так как электростанции в своей деятельности используют различные виды топлива, водные ресурсы, а также являются источником вредных выбросов в атмосферу.

На данный момент во многих странах используются альтернативные источники энергии (солнечные батареи, энергия ветра). Еще одним способом решения данной проблемы является использование энергосберегающих систем. Дополнительно можно отметить, что автоматизация тестирования, являющаяся основной темой ВКР, также может рассматриваться как способ снижения энергозатрат путем уменьшения используемого машинного времени.

В офисных помещениях не ведется никакого производства. К отходам, производимым в помещении можно отнести сточные воды и бытовой мусор.

Сточные воды здания относятся к бытовым сточным водам. За их очистку отвечает городской водоканал.

Основной вид мусора – это отходы печати, бытовой, коробки от техники, использованная бумага. Утилизация отходов печати вместе с бытовым мусором происходит в обычном порядке.

6.3 Безопасность в чрезвычайных ситуациях

Чрезвычайными ситуациями в рассматриваемых помещениях офисного типа могут быть пожары. Требования по пожарной безопасности устанавливаются ГОСТ 12.1.004-91 [8].

Пожар представляет собой неконтролируемое горение вне специального очага, причиняющее материальный ущерб, вред жизни и здоровью граждан, интересам общества и государства.

В пространстве, где развивается пожар, можно выделить три зоны: горения; теплового воздействия, где нельзя находиться без специальной тепловой защиты; задымления с опасностью для жизни и здоровья. Интенсивность

горения при пожаре зависит от скорости поступления в зону горения кислорода из окружающей среды.

С ростом энергооснащенности производства в значительной степени увеличивается опасность пожара в механических мастерских, местах хранения техники и транспортных средств при эксплуатации в них электроустановок. Короткое замыкание, перегрузка, большие переходные сопротивления, взрывы колб и ламп накаливания, замыкания фазных проводов на заземленные конструкции. Чаще всего причиной пожара становится короткое замыкание в электрических установках.

По пожарной и взрывопожарной опасности помещения производственного и складского назначения независимо от их функционального назначения подразделяются на следующие категории:

– повышенная взрывопожароопасность (А);

– взрывопожароопасность (Б);

– пожароопасность (В1-В4);

– умеренная пожароопасность (Г);

– пониженная пожароопасность (Д) [7].

Противопожарная защита должна достигаться применением одного из следующих способов или их комбинацией:

– применением средств пожаротушения и соответствующих видов пожарной техники;

– применением автоматических установок пожарной сигнализации и пожаротушения;

– применением основных строительных конструкций и материалов, в том числе используемых для облицовок конструкций, с нормированными показателями пожарной опасности;

– применением пропитки конструкций объектов антипиренами и нанесением на их поверхности огнезащитных красок (составов);

– устройствами, обеспечивающими ограничение распространения пожара;

– организацией с помощью технических средств, включая автоматические, своевременного оповещения и эвакуации людей;

– применением средств коллективной и индивидуальной защиты людей от опасных факторов пожара;

– применением средств противодымной защиты.

– переходом на пожаробезопасные источники освещения, преимущественно светодиодные.

Для обеспечения эвакуации необходимо:

– установить количество, размеры и соответствующее конструктивное исполнение эвакуационных путей и выходов;

– обеспечить возможность беспрепятственного движения людей по эвакуационным путям;

– организовать при необходимости управление движением людей по эвакуационным путям (световые указатели, звуковое и речевое оповещение и т.п.).

В производственных помещениях должно быть не менее двух эвакуационных выходов. Здание корпуса по адресу ул. Нахимова 13/1 соответствует требованиям пожарной безопасности. В здании установлена система охранно-пожарной сигнализации, имеются в наличии порошковые огнетушители, а также установлен план эвакуации с указанием направлений к основному и запасному эвакуационным выходам. Доступность эвакуационных выходов регулярно контролируется.

На рисунке 1 представлен план эвакуации при возникновении пожара и других ЧС.

Рисунок 20 – План эвакуации

Порошковые огнетушители ОП-8, пригодные для тушения электрооборудования под напряжением до 1 кВ с расстояния не менее 1 м (то есть, пригодные для тушения ПЭВМ), располагаются в доступных, отмеченных на плане эвакуации местах, пример размещения показан на рисунке 2.

Рисунок 21 – Расположение огнетушителей ОП-8

6.4 Правовые и организационные вопросы обеспечения безопасности

Специальные правовые нормы трудового законодательства

Государственный надзор и контроль в организациях независимо от организационно-правовых форм и форм собственности осуществляют специально уполномоченные на то государственные органы и инспекции в соответствии с федеральными законами. Согласно трудовому кодексу РФ:

– продолжительность рабочего дня не должна превышать 40 часов в неделю;

– во время регламентированных перерывов целесообразно выполнять комплексы упражнений и осуществлять проветривание помещения [9].

Существуют также специализированные органы, осуществляющие государственный контроль и надзор в организациях на предмет соблюдения существующих правил и норм.

К таким органам относятся:

– Федеральная инспекция труда;

– Государственная экспертиза условий труда Федеральной службы по труду и занятости населения;

– Федеральная служба по надзору в сфере защиты прав потребителей и благополучия человека и др.

Организационные мероприятия при компоновке рабочей зоны

Рабочее место для выполнения работ сидя организуют при легкой работе, не требующей свободного передвижения работающего, а также при работе средней тяжести в случаях, обусловленных особенностями технологического процесса.

Конструкция рабочего места и взаимное расположение всех его элементов (сиденье, органы управления, средства отображения информации и т.д.) должны соответствовать антропометрическим, физиологическим и психологическим требованиям, а также характеру работы.

Очень часто используемые средства отображения информации, требующие точного и быстрого считывания показаний, следует располагать в вертикальной плоскости под углом ± 15° от нормальной линии взгляда и в горизонтальной плоскости под углом ± 15° от сагиттальной плоскости, рисунки 4 и 5.

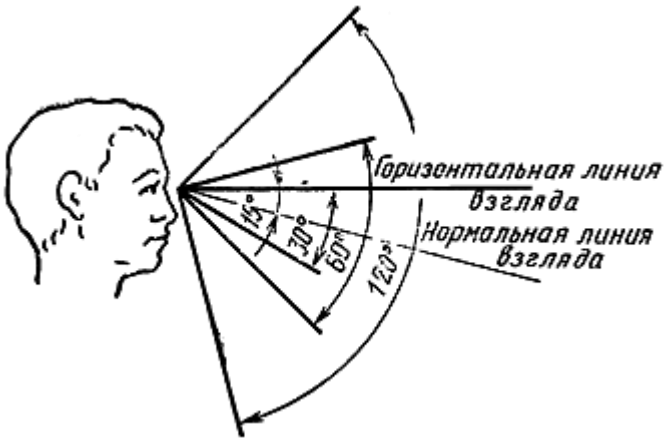


Рисунок 22 – Зоны зрительного наблюдения в вертикальной плоскости

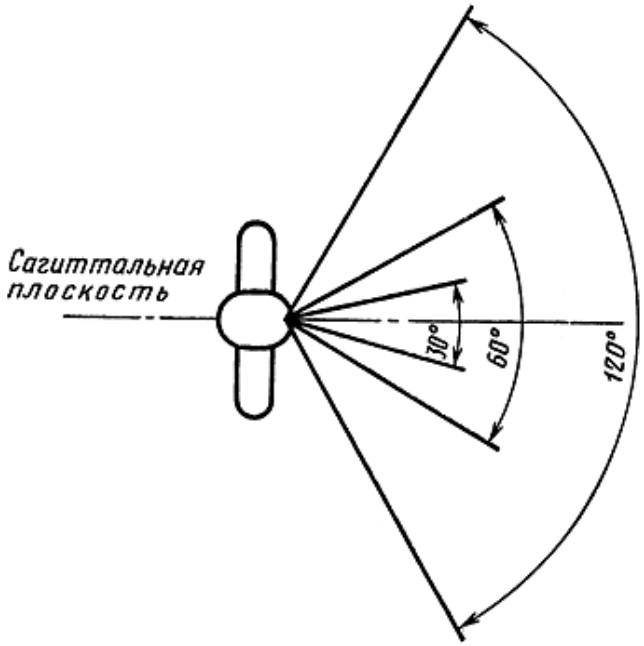


Рисунок 23 – Зоны зрительного наблюдения в горизонтальной плоскости

ЗАКЛЮЧЕНИЕ

В ходе выполнения выпускной квалификационной работы были изучены существующие на сегодня методы автоматизации тестирования пользовательского интерфейса и на основании этих данных, а также проведенного анализа тестируемого приложения, был выбран подход, использованный при реализации работы. В соответствии с выбранными методиками была подобрана инструментальная среда – тестовый фреймворк TestComplete. Структура проекта была разработана на базе паттерна Page Object.

Были написаны сценарии тестов, и на их основе создано около 20 пакетов с наборами тестов для автоматизированного тестирования пользовательского интерфейса, функционального тестирования, а также тестирования производительности. Разработанные тесты покрывают основные случаи использования тестируемого приложения и отвечают всем нуждам заказчика.

Также в ходе выполнения выпускной квалификационной работы был проведен сравнительный анализ возможных решений поставленной задачи, выявлены сильные и слабые стороны выбранного решения, проведена оценка трудоемкости и материальных затрат.

В настоящее время весь комплект разработанных тестов используется в процессе тестирования приложения Machining Cloud в компании DP Labs, часть тестов включена в процесс непрерывной интеграции. Внедрение автоматизации помогло значительно сократить время, затрачиваемое на процесс тестирования, улучшило качество выпускаемого продукта, сократило расходы на осуществление ручного тестирования.

CONCLUSION

During the execution of final qualifying work, the existing methods of automated user interface testing were examined and the approach used in the implementation of this work was selected based on this data and on the results of analysis of the application under test. In accordance with the chosen methods, TestComplete framework was selected as the development environment. The structure of the project was developed based on the Page Object pattern.

Test scripts were written, and about 20 packages with test suites for automated user interface testing, functional testing, and performance testing were created based on them. The developed tests cover the main use-cases of the tested application and meet all the needs of the customer.

Also, during the implementation of the final qualifying work, a comparative analysis of possible solutions to the task was carried out, the strengths and weaknesses of the chosen solution were identified, the labor intensity and finansial costs were estimated.

Currently, the whole set of developed tests is used in the process of testing the Machining Cloud application in DP Labs, part of the tests was included in the process of continuous integration. The introduction of automation helped significantly reduce the time spent on the testing process, improved the quality of the product, reduced the cost of manual testing.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. ГОСТ 28806–90. Качество программных средств. Термины и определения. [Электронный ресурс] URL: http://docs.cntd.ru/document/5200224, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 05.05.2017.

2. The Art of Software Testing / Glenford J. Myers, Revised and Updated by Tom Badgett, Todd M.Thomas, Corey Sandler. - 2nd ed. - Hoboken, New Jersey.: John Wiley & Sons, Inc., 2004 - 234 p.

3. Р. Калбертсон, К. Браун, Г. Кобб. Быстрое тестирование. - Вильямс, 2004. – 379 с.

4. Автоматизированное тестирование программного обеспечения - основные понятия. // ПроТестинг.RU. - [Электронный ресурс URL: http://www.protesting.ru/automation/, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.04.2017.

5. L. Copland. A Practitioner’s Guide to Software Test Design. - London: STQE Publishing, 2004. - 270 c.

6. С. Куликов. Тестирование программного обеспечения. Базовый курс. - Минск: Четыре четверти, 2015. - 293 с.

7. Пирамида автоматизации и другие геометрические фигуры // AT.Info [Электронный ресурс URL: http://automated-testing.info/t/piramida-avtomatizacz.., свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 13.05.2017.

8. Д. Химион. Анализ инструментов автоматизации мобильного тестирования. // SQA Days-21 [Электронный ресурс] URL: http://sqadays.com/ru/talk/43383, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.05.2017.

9. Автоматизация тестирования: выбор инструмента. // OpenQuality.ru. Качество программного обеспечения. - [Электронный ресурс] URL: http://blog.openquality.ru/tool-choice/, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.05.2017.

10. Использование фреймворков // Учебник по TestComplete. - [Электронный ресурс] URL: http://tctutorial.ru/frameworks/, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.04.2017.
11. Linda G. Hayes. Automated Testing Handbook. – Software Testing Inst; 2nd edition, 2004. – 182 p.
12. Mark Fewster. Software Test Automation: Effective Use of Test Execution Tools / Mark Fewster, Dorothy Graham. – Addison-Wesley & Son, Ltd, 2000. – 574 p.
13. Test Design Considerations // Selenium HQ. Browser automation. [Электронный ресурс] URL: http://www.seleniumhq.org/docs/06_test_design_conside.., свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.05.2017.
14. Page Object // martinfowler.com [Электронный ресурс] URL: https://martinfowler.com/bliki/PageObject.html, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.05.2017.
15. Machining Cloud. Smart Manufacturing. [Электронный ресурс] URL: https://www.machiningcloud.com, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 12.05.2017.
16. Межгосударственный стандарт. Система стандартов безопасности труда. Опасные и вредные производственные факторы. Классификация. [Электронный ресурс] URL: http://docs.cntd.ru/document/5200224, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 15.05.2017.
17. Постановление от 13 июня 2003 года №118 «О введении в действие санитарно-эпидемиологических правил и нормативов СанПиН 2.2.2/2.4.1340-03» (с изменениями на 21 июня 2016 года). [Электронный ресурс] URL: http://docs.cntd.ru/document/901865498, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 15.05.2017.
18. Давыдов, Борис Ильич. Биологическое действие, нормирование и защита от электромагнитных излучений / Б. И. Давыдов, В. С. Тихончук, В. В. Антипов. — Москва: Энергоатомиздат, 1984. — 177 с.: ил.: 21 см.

19. Краткий курс лекций по дисциплине «Безопасность жизнедеятельности» // Безопасность жизнедеятельности) [Электронный ресурс] URL: http://studme.org/1584072013070/bzhd/bezopasnost_zhiznedeyatelnosti, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 15.05.2017.
20. Свод правил: СП 52.13330.2011 Естественное и искусственное освещение. М.: Минрегион России, 2011. – 74 с.
21. Санитарно-эпидемиологические правила и нормативы: СанПиН 2.2.4.548-96 Гигиенические требования к микроклимату производственных помещений. – М.: Минздрав России, 1997. – 20 с.
22. ГОСТ 12.1.004-91 ССБТ. Пожарная безопасность. Общие требования. – М.: Изд-во стандартов, 2006. – 67 с.
23. Государственный стандарт Российской Федерации «Безопасность в чрезвычайных ситуациях. Защита населения». [Электронный ресурс] URL: http://docs.cntd.ru/document/1200001521, свободный. - Загл. С экрана. – Яз. рус. Дата обращения: 07.05.2017.
24. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ. Официальный текст. - М. : Пропаганда : Омега- Л, 2002. - 176 с.

Приложение А

(справочное)

Код основных частей программы

А.1 Реализация тестирования навигационной панели

Реализация вспомогательных функций, имитирующих возможные действия пользователя в отношении навигационной панели в репозитории F_NavBar:

```
function GetPath()
{
    Prop = new Array("ObjectType", "accessibilityHint", "VisibleOnScreen");
    Values = new Array("Button", "navigationBar*", "True");
    PathObjects = Aliases.Device.FindAll(Prop, Values, 10);
    if (PathObjects == null){
    Log.Error("There are no path objects");
    }
    else {
    PathObjects = (new VBArray(PathObjects)).toArray();
    s = "";
    for (i=0; i<PathObjects.length; i++) {
        s = s + PathObjects[i].ObjectText + "/";
    }
    s = s.slice(0,(s.length-1));
    Log.Message(s);
    return s;
    }
}

function CheckPath(path)
{
    s = GetPath();
    if (s == path) {
    Log.Message("Navigation Path is correct")}
    else{
    Log.Error("Wrong Navigation Path");
```

```
    }
}

function ClickForwardArrow()
{
    Aliases.Device.MainScreen.ForwardArrow.TouchButton();
}

function ClickOnPath(PathName)
{
    Aliases.Device.MainScreen.Breadcrumbs.Find("ObjectText", PathName, 2).Touch();
}
```

Реализация тестов навигационной панели с использованием функций репозитория, приведенных выше:

```
//USEUNIT F_Catalog
//USEUNIT F_Search
//USEUNIT F_NavBar
//USEUNIT F_TaskBar

function CatalogNavigation_Arrows()
{
    SearchByProductFamily();
    First_Catalog = GetCatalog();
    OpenCatalogItemByName("Milling");
    Milling_Catalog = GetCatalog();
    OpenCatalogItemByName("Indexable Milling");
    IndMil_Catalog = GetCatalog();

    ClickBackArrow();
    CheckPath("Search»Milling");
    CheckCatalog(Milling_Catalog);

    ClickBackArrow();
    CheckPath("Search");
    CheckBackButtonDisabled();
    CheckCatalog(First_Catalog);

    ClickForwardArrow();
    CheckPath("Search»Milling");
    CheckCatalog(Milling_Catalog);


    ClickForwardArrow();
    CheckPath("Search»Milling»Indexable Milling");
    CheckForwardButtonDisabled();
    CheckCatalog(IndMil_Catalog);

    GoHome();
}
```

```
function CatalogNavigation_Path()
{
    SearchByProductFamily();
    First_Catalog = GetCatalog();
    OpenCatalogItemByName("Threading");
    Threading_Catalog = GetCatalog();
    OpenCatalogItemByName("Thread Turning");
    Thread_Turning_Catalog = GetCatalog();
    OpenCatalogItemByName("LT Threading");

    ClickOnPath("Thread Turning");
    CheckPath("Search»Threading»Thread Turning");
    CheckCatalog(Thread_Turning_Catalog);
    OpenCatalogItemByName("LT Threading");

    ClickOnPath("Threading");
    CheckPath("Search»Threading");
    CheckCatalog(Threading_Catalog);

    ClickOnPath("Search");
    CheckPath("Search");
    CheckCatalog(First_Catalog);

    GoHome()
}
```

А.2 Пример некоторых тестов, содержащихся в модуле T_JobToolList:

```
//USEUNIT F_Job_ToolList
//USEUNIT F_JobManagement
//USEUNIT F_ToolAssembly
//USEUNIT F_NavBar
//USEUNIT F_TaskBar
//USEUNIT F_Details
//USEUNIT F_Waiting
//USEUNIT F_Functions
//USEUNIT F_Availability

function ToolListTable()
{
    FindToolAndOpenAssembly("1291085");
    var Tool1 = GetPath();
    AddToolToNewJob();
    WaitToolListLoading();

    Table_columns = new Array ("","Description", "Designation", "Grade", "Manufac-
turer Part #", "Customer Part #", "Manufacturer", "Last Modified Date", "Status");
    CheckTableColumns(Table_columns);

    Table_values = new Array ("TRHOR1615D", "", "1291085");
    CheckToolListTableValuesForTool("DEEP GROOVING TOOLHOLDERS", Table_values);

    GoJob();
    DeleteAllJobs();
    GoHome();
}

function OpenToolsInToolList()
{
```

```
    FindToolAndOpenAssembly("5593149");
    var ToolName = GetPath();
    AddToolToNewJob();
    WaitToolListLoading();

    SelectToolInToolList(ToolName);
    ClickButtonOnScreen("Open");
    WaitAssemblyLoading();

    CheckPath(ToolName);
    var AssemblyToolList = new Array("TM24D14L26Z1");
    CheckAssemblyToolList(AssemblyToolList);
    ReturnToJob();

    GoJob();
    DeleteAllJobs();
    GoHome();
}

function CopyToolsInToolList()
{
    FindToolAndOpenAssembly("2981197");
    var Tool1 = GetPath();
    AddToolToNewJob();
    DropDownToolInToolList("1");

    CopyToolToCurrentJobInToolList(Tool1);
    Tool2 = Tool1 + " (1)";
    var ToolList = new Array(Tool1,Tool2);
    CheckToolList(ToolList);
    var JobName = GetOpenedJobName();
    GoHome();

    GoJob();
    OpenJob(JobName);
    OpenToolList();
    CheckToolList(ToolList);

    GoJob();
    DeleteAllJobs();
    GoHome();
}

function RenameToolsInToolList()
{
    FindToolAndOpenAssembly("5544366");
    var Tool1 = GetPath();
    AddToolToNewJob();

    RenameToolinToolList(Tool1, "Renamed Tool");
    var ToolList = new Array("Renamed Tool");
    CheckToolList(ToolList);

    var JobName = GetOpenedJobName();
    GoHome();

    GoJob();
    OpenJob(JobName);
    OpenToolList();
    CheckToolList(ToolList);
```

```
    GoJob();
    DeleteAllJobs();
    GoHome();
}

function DeleteToolsFromToolList()
{
    FindToolAndOpenAssembly("1184115");
    var Tool1 = GetPath();
    AddToolToNewJob();

    FindToolAndOpenAssembly("1774499");
    var Tool2 = GetPath();
    AddToolToCurrentJob();
    var ToolList = new Array(Tool1,Tool2);

    SelectToolInToolList(Tool1);
    ClickButtonOnScreen("Delete");
    CheckDeleteAssemblyWarning(Tool1);
    ClickButtonOnScreen("No");
    WaitToolListLoading();
    CheckToolList(ToolList);

    DeleteToolFromToolList(Tool2);
    var ToolList = new Array(Tool1);
    CheckToolList(ToolList);

    var JobName = GetOpenedJobName();
    GoHome();

    GoJob();
    OpenJob(JobName);
    OpenToolList();
    CheckToolList(ToolList);

    GoJob();
    DeleteAllJobs();
    GoHome();
}

function ItemsAreNotIncludedInToolList()
{
    FindToolAndOpenAssembly("5337806");
    AddToolToNewJob();
    WaitToolListLoading();
    CheckITemsNotIncludedWarninginToolList();
    GoJob();
    DeleteAllJobs();
    GoHome();
}
```

A.3 Пример тестов производительности:

```
//USEUNIT F_Common
//USEUNIT F_UserMenu
//USEUNIT F_MainMenu
//USEUNIT F_JobManagement
//USEUNIT F_JobDetails
//USEUNIT F_QuickSearch
//USEUNIT F_Catalog
```

```
var StopWatch = HISUtils.StopWatch;

//initialize all the caches and warm up the application
function FirstStart()
{
    CleanUpCache();
    CreateLogFile();
    FirstStartApp();
    FillSignInInfo(Project.Variables.MC_account, Project.Variables.password, false);
    SignIn();
    aqFile.WriteToTextFile(LogFile, GetAppVersion() + "\r\n", aqFile.ctANSI, false);
    SearchByProductFamily();
    GoHome();
    SearchWithTheToolAdvisor();
    CloseApp();
}

function Application_Start()
{
   StopWatch.Start();
   NoFirstStartApp();
   StopWatch.Stop();
   Log.Message("Test time - " + StopWatch.ToString());
   aqFile.WriteToTextFile(LogFile, "1. Application Start - " + StopWatch.ToString() +
   "\r\n", aqFile.ctANSI, false);
   StopWatch.Reset();

}

function Logging_In()
{
   FillSignInInfo(Project.Variables.MC_account, Project.Variables.password, false);
   StopWatch.Start();
   SignIn();
   StopWatch.Stop();
   Log.Message("Test time - " + StopWatch.ToString());
   aqFile.WriteToTextFile(LogFile, "2. Logging In - " + StopWatch.ToString() +
   "\r\n", aqFile.ctANSI, false);
   StopWatch.Reset();
}

function Logging_Out()
{
    OpenUserMenu();
    StopWatch.Start();
    LogOut();
    StopWatch.Stop();
    Log.Message("Test time - " + StopWatch.ToString());
    aqFile.WriteToTextFile(LogFile, "3. Logging Out - " + StopWatch.ToString() +
    "\r\n", aqFile.ctANSI, false);
    StopWatch.Reset();
}

function Opening_Catalog()
{
    LogIn();
    StopWatch.Start();
    SearchByProductFamily();
    StopWatch.Stop();
    Log.Message("Test time - " + StopWatch.ToString());
    aqFile.WriteToTextFile(LogFile, "4. Opening Catalog - " + StopWatch.ToString() +
```

```
    "\r\n", aqFile.ctANSI, false);
    StopWatch.Reset();
    GoHome();
}

function Opening_Jobs()
{
    StopWatch.Start();
    OpenJobManagement();
    StopWatch.Stop();
    Log.Message("Test time - " + StopWatch.ToString());
    aqFile.WriteToTextFile(LogFile, "5. Opening Jobs - " + StopWatch.ToString() +
    "\r\n", aqFile.ctANSI, false);
    StopWatch.Reset();
    GoHome();
}
```