МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра Системы управления аэрокосмической техники

«Допущен к защите»
Заведующий кафедрой СУАТ
Шимербаев Н.К., к.т.н. проф. ВАК
(Ф.И.О., ученая степень, звание)
_____ « 14 » 04 **20 16 г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: Разработка методов оценки качества пиротехнических средств космического назначения

Специальность (5В074600) Космическая техника и технологии

Выполнил (а) Мендибаева М.А. СУКПА-12-1.
(Фамилия и инициалы) группа

Научный руководитель Исмаил Е.Е., к.т.н., проф.
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Бекищева А.И., к.э.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ « 16 » 06 20 16 г.
(подпись)

по безопасности жизнедеятельности:
Мазалов К.Ф., к.х.н., проф
(Фамилия и инициалы, ученая степень, звание)
_____ « 14 » 06 20 16 г.
(подпись)

Нормоконтролер: Шимербаев Н.К. к.т.н., профессор ВАК
(Фамилия и инициалы, ученая степень, звание)
_____ « 14 » 04 20 16 г.
(подпись)

Рецензент: Бекмуханбетов Б.Т.
(Фамилия и инициалы, ученая степень, звание)
_____ « 14 » 06 20 16 г.
(подпись)

Алматы 2016 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Аэрокосмические и информационные технологии
Специальность 5В074600 - Космическая техника и технологии
Кафедра Системы управления аэрокосмической техникой

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Мадибрахимова Инабат Ахатовна
(фамилия, имя, отчество)

Тема проекта Разработка методов оценки качества программных средств космического назначения

утверждена приказом ректора № 21 от «10» марта 2016 г.
Срок сдачи законченной работы «___» _____ 20___ г.
Исходные данные к проекту требуемые параметры результатов проектирования (исследования) и исходные данные объекта

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:

Особенности оценки качества программных средств космического назначения

Анализ методов оценки показателей качества ПСКН.

Выбор и обоснование методов оценки надежности ПСКН.

Анализ видов последствий и критичности отказов ПСКН.

Перечень графического материала (с точным указанием обязательных чертежей)

Множество разных в тексте

Графические изображения

Таблицы

Рекомендуемая основная литература

Богданов А.А. Ильин И.А., Ермаков И.Е. Пути повышения надежности и качества программных обеспечений в космической отрасли. Управления большими системами Выпуск 59.

Липаев В.В. Выбор и оценивание характеристик качества программных средств методов и стандартов. Серия „Информационные технологии.

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
ОБЖ	Мазалов И.Ф	14.06.16	
Экономическая час.	Рекашова Ал. Ал	16.06.16	
Основная часть	Шеман Е.Е	16.06.16	
Нормоконтроль	Шамирбаев Н.К.	14.06.2016г.	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
	Введение	25. 03. 2016	
	Особенности оценки качества.	11. 03. 2016	
	Содержание и сущность...	18. 03. 2016	
	Основные требования...	21. 03. 2016	
	Постановка задачи...	25. 04. 2016	
	Анализ методов оценки...	26. 04. 2016	
	Методы функционального...	27. 04. 2016	
	Методы структурного...	29. 04. 2016	
	Методы статистического...	1. 05. 2016	
	Выбор и обоснования...	3. 05. 2016	
	Анализ особенностей...	10. 05. 2016	
	Анализ моделей оценки...	10. 05. 2016	
	Выбор модели оценки...	12. 05. 2016	
	Прогнозирования показ...	12. 05. 2016	
	Анализ видов последствий...	13. 05. 2016	
	Основные принципы и...	14. 05. 2016	
	Идентификация видов...	14. 05. 2016	
	Оценка критичности...	15. 05. 2016	
	Экономический раздел...	16. 05. 2016	
	Безопасность жизнедеят...	16. 05. 2016	
	Заключение...	18. 05. 2016	

Дата выдачи задания «03» 11 2015 г.

Заведующий кафедрой СУАТ М. Шимирбаев
(подпись) (Фамилия и инициалы)

Руководитель Чолаев Е.Е.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент Мадибрашова М.А.
(подпись) (Фамилия и инициалы)

Аннотация

Данная дипломная работа выполнена на актуальные темы: «Разработка методов оценки качества программных средств космического назначения»

Тема основывается тем фактом и сложностям объектов, ракетно-космической техники и обусловлена многообразием решаемых ими задач научного, социально-экономического и оборонного характера.

Основная цель дипломной работы — это разработка методов оценки качества программных средств космического назначения-выполнено.

Аннотация

Дипломдық жұмыс «Космостық құрылғылардың бағыттарын жоспарларының сапасын бағалау ерекшеліктері»- тақырыбына орындалған.

Атқарылған жұмыс қазіргі кездегі күрделі космосты-ракеталы құрылғыларға техникалық есептеулер мен алға койылған міндеттерді ғылыми, әлеуметті-экономикалық және қорғаныс мінездемелері тұрғысына сәйкестендіріліп көп салалалы бағыттарға негізделген.

Дипломдық жұмыстың негізгі мақсаты бұл бағалау ерекшеліктері мен сапасын анықтау болды. Ал алға қойылған мақсат орындалған.

Annotation

This is work is done to current topics: "Development of quality assessment methods for space application software"

The theme is based fact and complexity of the object, rocket and space technology, and due to the diversity of scientific problems solved by them, socio-economic and defensive nature.

The main aim of the thesis - is the development of quality assessment methods cosmic purpose software-implemented.

Содержание

Введение……7

1 Особенности оценки качества программных средств космического назначения ……9

1.1Содержание и сущность задачи оценки качества ПСКН……9

1.2Основные требования к оценке качества ПСКН……11

1.3 Постановка задачи оценки качества ПСКН……...16

2 Анализ методов оценки показателей качества ПСКН……19

2.1 Методы функционального тестирование программного обеспечения …19

2.2 Методы структурного тестирования программного обеспечения……20

2.3Методы статического анализа программного обеспечения ……27

3 Выбор и обоснование методов оценки надежности ПСКН……29

3.1 Анализ особенностей и требовании к оценке надежности ПСКН……29

3.2 Анализ вводимых данных на тестирования надежности ПС……34

3.3 Принципы оценки показателей качества ПСКН……35

4 Анализ видов, последствие и критичности отказов ПСКН……37

4.1 Основные принципы и положение анализа видов, последствии и критичности отказов ПСКН……37

4.2 Идентификация видов причин и влияния отказов……38

4.3Оценка критичности и вероятности отказов ……40

5 Экономический раздел……44

6 Безопасность жизнедеятельности……55

Заключение……64

Список литературы……66

Введение

Согласно статистическим данным, около 25 % авариных ситуаций в ракетно-космической технике связано с отказом программно-технических комплексов и дефектами в программных средствах [1]. Данное обстоятельство порождает необходимость для применяемых в космических системах гарантированно работающих программных средств (так называемые программные средства космического назначения - ПСКН). Они должны обладать заданными качественными свойствами, в том числе надежностью и безопасностью, а также устойчивостью по отношению к сбоям и погрешностям различного происхождения.

В современных практических подходах к описанию и оценке качества программного средства (ПС) используют понятие «моделей качества», которые являются основами как для формального описания характеристики качества и их взаимоотношений, так и для проведения оценки качества программных средств. Модели качества ПС представляют собой структурированные множества взаимосвязанных характеристик и отношений между собой. Структуру моделей качества программных средств описывают при помощи иерархии, элементы которой - это множество характеристик (подхарактеристики, атрибуты) и отношения подчиненности между ними.

Пользователи и разработчики программных средств в космической отрасли испытывают потребность в разработке различных моделей качества, которые применяются для описания и оценки качества конкретных ПС с учетом их функционального назначения, специфики и условий использования. Однако, для оценки качества конкретной программной системы, стандартизованная модель не всегда подходи в полной мере. Для конкретной разрабатываемой программной системы необходимо, исходя из её функционального предназначения, особенности, степени важности отдельных требований, формирование адаптированной модели качества.

Большое количество требований к качеству ПСКН, в зависимости от их функционального предназначения, принципиальной особенности и условий эксплуатации, привело к потребности проведения адаптации и детализации рекомендаций уже имеющихся базовых стандартов, которые регламентируют качество программных систем. В первую очередь, это относится к ПКСН для критического применения.

Актуальность темы дипломной работы обосновывается тем фактом, что сложность объектов, применяемых в ракетно-космической технике, обуславливается многообразием возложенных на них научных, социально-экономических и оборонных задач. В ближайшей перспективе

многофункциональные объекты ракетно-космической техники по своим возможностям будут напоминать автоматические летающие роботы. Группировки и комплексы управления такими объектами будут представлять из себя большие, пространственно-распределенные интеллектуальные системы.

Цель дипломной работы состоит в разработке методов оценки качества программных средств космического назначения.

Объектом исследования данной работы являются программные средства космического назначения.

Для достижения цели работы поставлены следующие задачи:

1. Изучить предметную область ПСКН;
2. Рассмотреть основные требования к оценке качества ПСКН;
3. Провести анализ методов оценки показателей качества ПСКН;
4. Выбрать и обосновать методы оценки надежности ПСКН;
5. Сделать выводы;

Теоретической базой для дипломной работы является литература по проектированию, реализации и управлению программными средствами, нормативно-правовые документы, учебная и периодическая литература.

1 Особенности оценки качества программных средств космического назначения

1.1 Содержание и сущность задачи оценки качества ПСКН

Основная задача построения модели качества программных средств - это формирование, учитывая назначение, особенности, условия эксплуатации, степень важности отдельных требований и др., номенклатуру актуальных характеристик их свойств и отношений между ними, которые обеспечивают основу при определении требований к качеству класса программных средств и надежной оценки уровня их качества.

Модель качества для программных средств обычно строят, основываясь на международные стандарты, которыми регламентируются показатели качества. Однако при оценке качества данных программных средств, определенные стандартами характеристики, не всегда соответствуют полностью. При разработке конкретного программного средства необходимо учитывать его функциональное назначение, особенности, степень важности отдельных требований и, следовательно, сформировать полный перечень актуальных характеристик качества.

В соответствии с анализом данных, практически все существующие в настоящее время подходы и методы построения модели качества ПС основываются на формировании иерархической структуры характеристик. Вверху располагаются характеристики качества, которые детализированы показателями более низких уровней, и так до тех пор, пока декомпозиция не приведет к измеримым и атомарным атрибутам. Отличия состоят в том, что предлагается разное число уровней иерархии (от 2 и более), а также сами характеристики верхнего уровня иерархии, многие из которых все-таки совпадают.

Предпочтение иерархической структуре для модели качества программного средства можно объяснить, во-первых, тем, что с помощью многоуровневой структуры показателей качества возможно системное описание требований к программному средству, позволяя заинтересованным сторонам задавать свойства (характеристики) программных продуктов, которые они хотят видеть.

Во-вторых, многие базовые показатели качества, как функциональное соответствие, надежность, безопасность и др., нельзя непосредственно измерить и оценить. При получении оценок этих показателей их можно представить в виде совокупности более узких характеристик нижнего уровня (оценочные показатели), которые, в свою очередь, также можно детализировать. Такую детализацию осуществляют до тех пор, пока

характеристики нижнего уровня иерархии можно будет легко оценить и получить объективные количественные оценки. Характеристика нижнего уровня, выраженная в физических или относительных единицах, называется единичным показателем качества.

Построить модель качества программного средства возможно, если последовательно и детально сверху вниз описать многоуровневую структуру показателей от характеристик верхнего уровня иерархии до оценочных элементов (единичных показателей). При этом оценочные элементы должны обеспечивать непосредственное определение наличия тех или иных свойств в программном средстве.

Строить модель качества следует сверху вниз от характеристик до оценочных элементов, а оценивать достигнутые показатели качества следует в обратном направлении: от оценивания единичных оценочных элементов до агрегированного оценивания вышестоящих показателей качества в моделях качества ПО.

В настоящее время отсутствует общепринятая методика построения модели качества программного средства, посредством которой факторы качества сводятся к конечному набору количественных оценочных показателей, которые можно легко и объективно оценить.

Но на практике выработали рекомендации и общие требования для формирования системы характеристик качества программных средств. Для системы характеристик качества программных средств космического назначения эти требования сформулированы таким образом:

- модель качества программного средства космического назначения рекомендуется строить, основываясь на современные международные стандарты, регламентирующие показатели качества программного средства, учитывая функциональное назначение, специфику и требования области применения;

- систему характеристик качества программных средств космического назначения необходимо формировать, основываясь на стандартные характеристики верхнего уровня иерархии, учитывая требования всех заинтересованных сторон;

- рекомендуется сделать проект обобщенной базовой модели качества для нескольких групп (классов) программных средств космического назначения, имеющим максимально возможный перечень показателей качества, и в каждом конкретном случае, основываясь на базовую модель, строить модель качества для определенной группы или конкретного программного средства космического назначения, расширять или ссужать диапазон базовых показателей качества;

- системой характеристик качества программных средств космического назначения должны описываться все особенности и основные свойства программных средств конкретного класса, должна быть возможность в дальнейшем уточнять и детализировать;

- рекомендуется формировать оценочные характеристики качества, основываясь на принципе измеряемости и понятности значений;

- большая группа характеристик качества программного средства космического назначения должна быть разбита на логически связанные подгруппы, количество подхарактеристик в которых не должно быть выше 5-7;

- каждой характеристике качества должно соответствовать важное свойство программного средства конкретного класса;

- характеристики качества не должны перекрываться или дублироваться;

- единичный показатель качества, выраженный в физических единицах, должен быть преобразован в относительные единицы, находящиеся в интервале от 0 до 1 (нормированы).

Основываясь на результаты сравнительного анализа современных моделей качества программного средства [2], показана целесообразность построения базовой модели качества для программных средств космического назначения на основе стандартной модели ISO/IEC 25010 [3]. Но при применении к конкретному типу программных средств космического назначения ее необходимо адаптировать, учитывая особенности, предъявляемые требования, условия применения путем выбора актуальных характеристик и подхарактеристик, а также возможное добавление дополнительных атрибутов качества.

1.2 Основные требования к оценке качества ПСКН

Базовые системы характеристик качества для различных групп (классов) программных средств космического назначения основываются на стандартных характеристиках верхнего уровня иерархии для стандартной модели качества программного средства ISO/IEC 25010. Учитывая особенности программных средств космического назначения, обосновано введение в базовую систему характеристик качества дополнительной характеристики «качество требований», аналогичной модели качества SATC НАСА, в общем случае может иметь следующие подхарактеристики [5]:

- однозначные требования;
- полный охват требований и потребностей;
- понятные требования;

- изменчивые требования;
- прослеживаемые требования;
- полные и корректные требования к условиям эксплуатации.

Таким образом, в базовую систему характеристик качества ПСКН входят:

– функциональное соответствие;
– эффективность функционирования;
– защищенность;
– удобство использования;
– надежность;
– совместимость;
– мобильность или переносимость;
– сопровождаемость;
– качество требований.

Созданная с учетом изложенных выше предложений таблицу применяемости показателей качества ПСКН приведем в таблице 1. В ней укажем только некоторые группы ПСКН, которые актуальны с точки зрения разработчика (при необходимости число групп ПСКН можно увеличить или сократить с учетом специфики конкретных программных проектов).

Таблица 1- Таблица применяемости для показателей качества ПСКН

Показатель качества ПСКН	Применяемость показателя качества по группе					
	Групп а 1	Гру ппа 2	Груп па 3	Груп па 4	Груп па 5	Груп па 6
1	2	3	4	5	6	7
1. Соответствие функциональное:	+	+	-	+	+	+
1.1 Полнота функциональная;	+	+	+	+	+	+
1.2 Точность функционирован ия правильность;	+	±	+	-	+	+
1.3 пригодность функциональная.	+	+	+	+	+	+
2. Надежность:	+	+	±	+	±	+
2.1 отказоустойчиво	+	+	+	+	±	+

сть						

Продолжение таблицы 1

1	2	3	4	5	6	7
2.2 завершенность	-	+	+	+	±	-
2.3 восстанавливаемость	+	+	±	+	±	+
2.4 готовность	+	±	+	+	±	+
3. Удобство использования:	+	+	+	+	+	+
3.1 Защита от ошибок пользователя	-	±	+	+	+	-
3.2 удобство работы	+	+	+	+	±	+
4 Эффективность:	+	+	+	+	±	+
4.1 эффективность использования ресурсов	-	+	+	+	±	-
4.2 временная эффективность	+	+	+	+	±	+
4.3 Мощность (производительность)	+	+	+	+	±	+
5 Совместимость	+	+	-	+	+	+
5.1 Способность к взаимодействию	+	±	+	+	+	±
5.2 способность к сосуществованию	+	+	+	+	±	+
6 Защищенность:	+	+	±	-	±	+
6.1 Целостность	+	+	+	+	±	+

Продолжение таблицы 1

1	2	3	4	5	6	7
6.2 Конфиденциальность (защита от НСД)	+	+	+	+	±	±
6.3 способность учета (формирование контрольного следа)	+	±	+	+	-	+
6.4 Доказуемость действий	+	+	+	+	-	+
7 Сопровождаемость	+	+	+	+	±	+
7.1возможность многократного использования	+	+	+	±	±	+
7.2 модульность	+	±	+	+	+	+
7.3 изменяемость	+	±	+	+	-	+
7.4 анализируемость	+	+	+	+	+	+
7.5 тестируемость	+	+	±	-	±	+
8 Переносимость	+	+	+	-	+	±
8.1 устанавливаемость	+	+	+	+	+	+
8.2 адаптируемость	+	+	+	+	+	+
8.3 Качество требований	+	+	+	-	±	+
9 удобство замены	+	+	+	+	+	+
9.1 однозначность	+	-	+	+	+	+
9.2 понятность	+	+	+	-	+	+
9.3 полнота	+	+	+	+	+	+
9.4 изменчивость	+	+	-	+	-	+
9.5 охват потребностей	±	+	+	+	+	+
9.6 прослеживаемость	+	+	±	+	±	-
9.7 условия эксплуатации	+	+	+	+	+	+
9.8 сложность логики	+	±	+	±	+	-

В данной таблице используются обозначения

1) «+» значит применимость соответствующего показателя качества ПСКН, знак «-» - означает неприменяемость показателя качества ПСКН, знак «±» - это ограниченная применяемость показателя.

2) группы показателей качества ПСКН, которые выделены в соответствии с классификацией обладают следующими наименованиями:

- Группа 1 – программные системы для бортовых комплексов управления космическими аппаратами;

- Группа 2 – программные системы для вычислительных комплексов ракет-носителей и разгонных блоков;

- Группа 3 – программные системы для технических и стартовых комплексов, наземных автоматизированных комплексов управления космическими аппаратами, а также наземного оборудования и сооружений;

- Группа 4 – программные системы для полезных нагрузок;

- Группа 5 – программные системы для космических экспериментов и моделирования;

-Группа 6 – остальные критические ПСКН

Приоритетное требование качества для ПСКН критического применения состоит в гарантии качества или гарантоспособности (dependability), под которой понимают доказанную уверенность способности программных систем для надежного и безопасного выполнения необходимого функционала в течение заданного периода времени, не принимая во внимание возникающие внутренние и внешние возмущения [6]. В связи с этим предлагают базовую модель качества ПСКН критического применения дополнять комплексными характеристиками качества – «гарантоспособность», в которую в общем случае могут включаться следующие подхарактеристики [6]:

- Категория безотказности (reliability);
- Категория функциональной безопасности (functional safety);
- готовность (availability);
- Категория живучести (survivability);
- Категория целостности (integriity;
- Категория конфиденциальности (confidentiality);
- Достоверность (high confidence, trusyworthiness);
- Обслуживаемость (maintainability);
- Категория диагностируемости;
- Категория восстанавливаемости;
- Категория многоверсионности;
- наличие резервирования.

Как правило, в требования к качеству критических ПСКН должны обязательно включаться такие подхарактеристики гарантоспособности:

- безотказности;
- функциональной безопасности;
- живучести.

Отличительная особенность базовой модели качества ПСКН состоит в том, что её строят на основании системы стандартных характеристик верхнего уровня иерархии модели качества ISO/IEC 25010 с дополнением ее характеристикой качества требований из модели качества SATC НАСА и комплексной характеристикой «гарантоспособности», которая рекомендована стандартом ECSS-Q-80-03-2006.

1.3 Постановка задачи оценки качества ПСКН

Создание моделей качества ПСКН необходимо проводить с учетом их предназначения, типов (классов), стадий жизненного цикла, на котором они будут применяться.

На первом этапе за основу следует взять всю базовую номенклатуру характеристик, подхарактеристик и атрибутов качества программной системы по стандарту ISO 25010. Её описание желательно предварительно упорядочивать по приоритету с учетом предназначения и сфер применения конкретных проектов программных средств. Дальше следует выделять и проранжировать по приоритету потребителей, для которых необходимыми являются определенные показатели качества проектов программных средств с учетом профессиональных интересов. Подготовку исходных данных завершают выделением номенклатуры базовых или приоритетных показателей качества, которые определяют функциональную пригодность программных средств для определенного потребителя.

На втором этапе, после установления исходных данных, следует проводить ранжирование характеристик и подхарактеристик для конкретных проектов. Далее подготовленными специалистами для каждого из отобранных показателей следует установить и согласовать метрику и шкалу оценок подхарактеристик и их атрибутов. Выбранные значения характеристик качества программного средства и их атрибутов разработчиками следует предварительно проверить на реализуемость с учетом доступных ресурсов для конкретных проектов и при необходимости откорректировать.

При создании номенклатуры показателей качества нужно указать приоритет для каждого из показателей. Наивысший приоритет следует интерпретировать как обязательное выполнение разработчиками соответствующий требований к указанным свойствам или атрибутам качества. Наименьшие значения приоритета означает, что данные показатели могут не учитываться в данном проекте. Промежуточное значение приоритета должно отражать относительное влияние соответствующего атрибута на качество программного средства и его свойства с учетом

доступных ресурсов на его реализацию. Для конкретных программных проектов ПСКН состав и значение приоритета следует адаптировать и уточнить с учетом его назначения и функции.

Номенклатуру показателей качества для каждого из типов (классов) ПСКН представляют в виде таблиц применяемости показателя качества, в которых кроме обязательных или рекомендуемых показателей качества, следует также устанавливать коэффициенты весомости (или значимости) каждого показателя. При определении коэффициента весомости показателя качества обычно применяют либо методы стоимостной регрессионной зависимости (стоимостные методы), либо методы предельного и номинального значения (вероятностно-статистические методы) [4]. Но их применение затрудняется из-за отсутствия необходимых исходных данных. Поэтому на практике наиболее распространенными являются экспертные методы определения коэффициента весомости.

Таблица применяемости является основой для выбора рабочей номенклатуры показателя качества конкретного программного средства космического назначения.

Рабочую номенклатуру ПС устанавливают с учетом:

- назначение и условия эксплуатации ПСКН;
- результат анализа требований пользователей (заказчиков);
- состав, структура и специфика характеризуемого свойства.

При выделении атрибутов и соответствующего показателя качества ПСКН следует применять следующие основные принципы:

1) следует выделять совокупности свойств, которые характеризуют качества оцениваемых ПСКН, и упорядочивать их по четко определенным принципам в виде многоуровневых иерархических структур - деревьев свойств;

2) дерева свойств должны отражать все основные особенности применения и функционирования оцениваемых ПСКН;

3) для каждого из выделенных свойств программных средств космического назначения следует определить соответствующие показатели качества. Для сложных свойств необходимо установить группу показателей, которые необходимы и достаточны для определения этих свойств;

4) для выбранных свойств и показателей качества программных средств должны существовать возможности выражения их в шкалах «хуже - лучше», «меньше -больше»;

5) выбранный показатель качества должен быть скоррелирован с соответствующим свойством ПСКН. Это означает, что между выделенным свойством и характеризующим его показателем должно быть установлено

взаимно однозначное соответствие. Установление такого соответствия дает возможность вместо деревьев свойств применять деревья показателей качества программных продуктов;

6) исходную (рабочую) номенклатуру показателей следует создавать открытой, т.е. должна быть допущена возможность внести или исключить из нее отдельные элементы. Такое требование обуславливается, с одной стороны, большим разнообразием программных средств и условий их применения, а с другой недостаточно большим опытом проведения оценки качества ПС;

7) для отобранных показателей качества ПС следует обеспечить выполнение принципа непротиворечивости, т.е. улучшение одних показателей не должно привести к ухудшению других.

2 Анализ методов оценки показателей качества ПСКН

2.1 Методы функционального тестирования программного обеспечения

Функциональным тестированием называют проверку программного обеспечения на правильность функционирования в идеальных условиях. В случае нефункционального тестирования, либо условия неидеальны (нагрузочное тестирование), либо тестирование проводится не на правильность функционирования, а на что-то другое (тест на удобство пользования или структуру программы).

Для большинства приложений производительность и удобство пользования некритичны. Как правило, в требованиях к программному обеспечению содержится только функциональная часть. И практически нет требований к программному обеспечению без функциональной части. Следовательно, функциональное тестирование нужно проводить для любого программного обеспечения.

Обычно функциональное и нефункциональное тестирование программного обеспечения проводятся параллельно, и, как правило, это делается разными командами или людьми. Большинство источников указывает на то, что функциональное тестирование является синонимом black-box тестирования (программу рассматривают как черный ящик).

В зависимости от степени доступа к коду различаются два типа тестирования:

1. тестирование "черного ящика" (black box), или функциональное тестирование - без доступа к коду;
2. тестирование "белого ящика" (white box), или структурное тестирование - тестирование кода.

В случае " black box " программу рассматривают как конечный автомат, имеющую входные и выходные данные, набор внутренних состояний и переходов между ними. Проводится тестирование правильности поведения программы с различными входными данными и внутренним состоянием. Правильность определяется, опираясь, главным образом, на спецификацию, а также разными другими способами, исключая изучение кода.

В случае " white box " тестировщиком пишутся тесткейсы, исходя исключительно из кода программы (тесты на правильность кода).

Расширение black-box тестирования, когда также применяют изучение кода, - это тестирование "серого ящика" (grey box). В таком случае правильность поведения определяют любым подходящим способом, включая изучение кода. Таким образом, это позволяет написать более эффективный black-box тест (то есть тест на правильность поведения).

Функциональный тест базируется на взаимодействии с другими системами, а также на функциях и особенностях, и может быть представлен на следующих уровнях тестирования:

- компонентном или модульном (Component/Unit testing);
- интеграционном (Integration testing);
- системном (System testing);
- приемочном (Acceptance testing).

Функциональными видами тестирования рассматривается внешнее поведение системы. Самые распространенные виды функциональных тестов:

- Функциональное тестирование (Functional testing);
- Тестирование безопасности (Security and Access Control Testing);
- Тестирование взаимодействия (Interoperability Testing).

2.2 Методы структурного тестирования программного обеспечения

Структурное тестирование - это также тестирование по «маршрутам». В этом случае тестовые наборы формируются с помощью анализа маршрутов, которые предусмотрены алгоритмом. Маршруты - это последовательность операторов программы, которая выполняется при определенных исходных данных.

Структурное тестирование основывается на концепцию проведения максимально полного тестирования всех маршрутов программы. Так, в случае алгоритма программы с ветвлением, при одних исходных данных выполняется последовательность операторов, которая реализует действия, предусмотренные одной ветвью, а при втором наборе исходных данных - другая. Следовательно, для программы существуют маршруты, различающиеся выбранным при ветвлении вариантом.

Считается, что если с помощью тестов осуществляется выполнение программы по всем возможным маршрутам передач управления, то программа проверена полностью. Однако легко заметить, что даже в случае программы среднего уровня сложности количество неповторяющихся

маршрутов бывает очень велико, и, таким образом, тестирование маршрутов полностью или исчерпывающе, как правило, невозможно.

В соответствии с методом структурного тестирования (рисунок 3) предполагается создание тестов, основываясь на структуру системы и ее реализацию. Такой подход иногда называется «тестирование методом «белого ящика», «стеклянного ящика» или «прозрачного ящика», отличающийся от тестирования методом «черного ящика» [3].

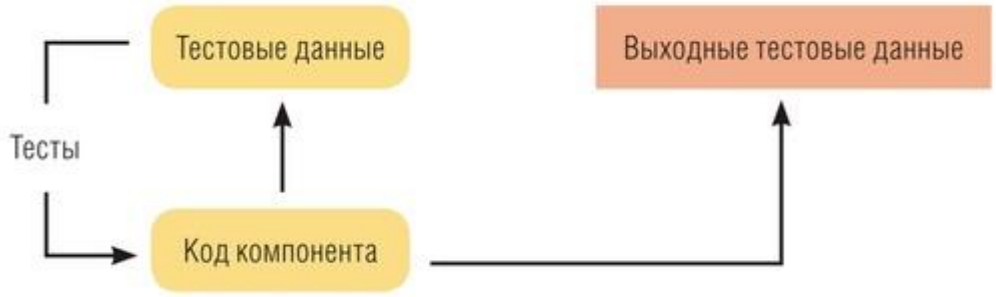

Рисунок 1- Структурное тестирование

Обычно структурное тестирование могут применить к сравнительно небольшому программному элементу, например, к подпрограмме или к методу, ассоциированному с объектом. В данном подходе испытателем анализируется программный код и для получения тестовых данных пользуются знаниями о структуре компонента. Например, по анализу кода можно получить информацию о том, сколько контрольных тестов должны быть выполнены для того, чтобы при тестировании все операторы выполнились, по меньшей мере, один раз [2-4].

Тестирование ветвей в случае метода структурного тестирования проверяются все независимо выполняемые ветви компонента или программы. Если все независимые ветви выполняются, следовательно, все операторы должны выполняться не меньше одного раза. Кроме того, условные операторы тестируют со всеми значениями условий, как с истинными, так и с ложными. В объектно-ориентированных системах используют тестирование ветвей при тестировании методов, ассоциированных с объектами.

Число ветвей в программе, как правило, пропорционально ее размеру. Методы структурного тестирования, после внедрения программных модулей в систему, оказываются невыполнимыми. В связи с этим, методы тестирования ветвей обычно используют для тестирования отдельных программных элементов и модулей [2,3].

В процессе тестирования ветвей все возможные комбинации ветвей программы не проверяются. Кроме наиболее тривиальных программных компонентов без циклов, проведение подобной полной проверки компонентов оказывается нереальным, ведь в программах с циклами число возможных комбинаций ветвей бесконечно. Программа может иметь дефекты, проявляющиеся только при определенных комбинациях ветвей, хотя все операторы программы были протестированы (т.е. выполнялись) хотя бы один раз.

Метод тестирования ветвей осуществляется на основе графов потоков управления программы. Этот граф является скелетной моделью всех ветвей программы. Граф потоков управления представляет собой узлы, соответствующие ветвлениям решений, и дуг, которые показывают потоки управления. Если в программе отсутствуют операторы безусловного перехода, то создание графа является достаточно простым процессом. Для построения графа потоков можно проигнорировать все последовательные операторы (операторы присвоения, вызова процедур и ввода-вывода). Все ветвления операторов условного перехода (if-then-else или case) представлены отдельными ветвями, а циклы обозначают стрелками с замкнутыми концами на узле с условием цикла. На рисунке 2 изображены циклы и ветвления в графе потоков управления программы бинарного поиска [3].

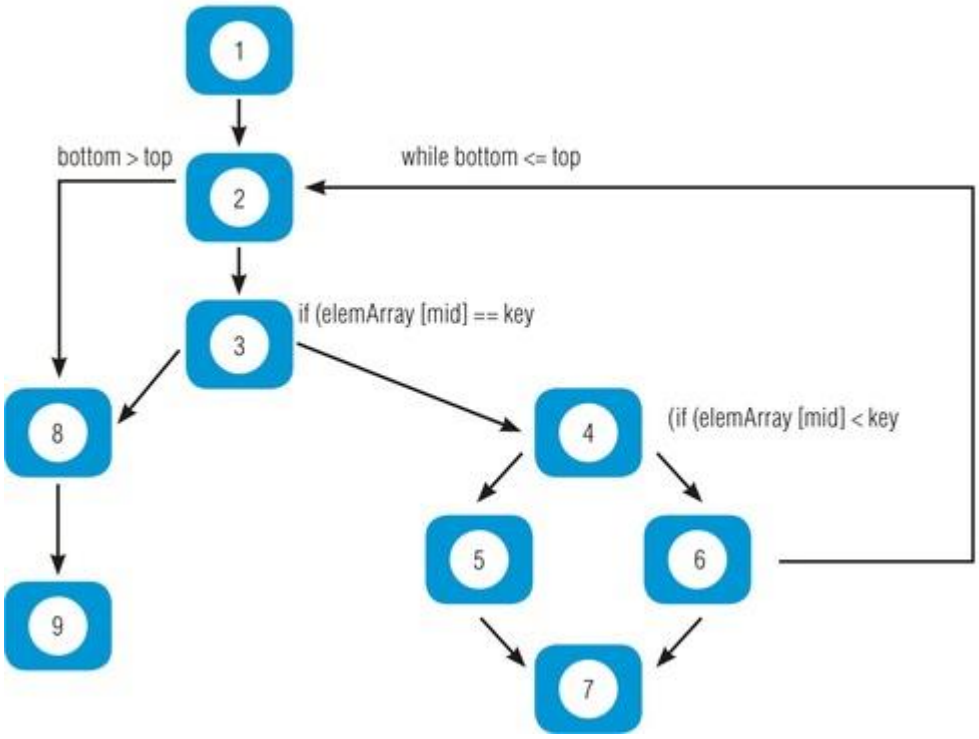

Рисунок 2-Граф потоков управления бинарного поиска

Целью структурного тестирования является удостоверение в том, что все независимые ветви программы выполняются хотя бы один раз. Независимой ветвью программы называется ветвь, проходящая, по крайней мере, по одной новой дуге графа потоков. В соответствии с терминологией программы это означает, что она выполняется при новых условиях. В графе потоков управления программы бинарного поиска с помощью трассировки могут быть выделены следующие независимые ветви [3]:

- 1, 2, 3, 8, 9
- 1, 2, 3, 4, 6, 7, 2
- 1, 2, 3, 4, 5, 7, 2
- 1, 2, 3, 4, 6, 7, 2, 8, 9

Если выполняется каждая из этих ветвей, то можно сказать, что:

1. Каждый оператор выполняется, по крайней мере, один раз;
2. Каждая ветвь выполняется при условиях, принимающих как истинные, так и ложные значения.

Число независимых ветвей в программе определяется, если вычислить цикломатическое число графа потоков управления программы [1-4]. Дипломатическое число C любого связанного графа G можно вычислить по формуле:

C (G) = количество дуг – количество узлов + 2

Программы, не содержащие операторы безусловного перехода, всегда имеют значение цикломатического числа больше, чем количество проверяемых условий. Для составных условий, в которых содержатся более одного логического оператора, необходимо учитывать каждый из них. Например, для программы с шестью операторами if и одним циклом while, цикломатическое число равно 8. В случае условного выражения в виде составного выражения с двумя логическими операторами (объединенными операторами and или or), цикломатическое число будет равно 10. Цикломатическое число для программы бинарного поиска соответствует 4.

Цикломатическое число вычисляется для того, чтобы определить количество независимых ветвей в программе, затем разрабатываются контрольные тесты для проверки каждой из ветвей. Минимальное количество тестов, которое требуется для проверки всех ветвей программы, равно цикломатическому числу [3,4].

Спроектировать контрольные тесты для программы бинарного поиска не составляет трудностей. Но для программы со сложной структурой ветвлений, трудно предсказать, как будет проходить выполнение какого-либо

отдельного контрольного теста. В таком случае пользуются динамическим анализатором программ при составлении рабочего профиля программы.

Динамическими анализаторами программ являются инструментальные средства, работающие совместно с компиляторами. В процессе компилирования в сгенерированный код добавляют дополнительные инструкции, которые подсчитывают, сколько раз выполняется каждый из операторов программы. Чтобы в процессе выполнения отдельных контрольных тестов определить, выполнялись ли все ветви в программе, распечатывают рабочий профиль программы, где можно увидеть непроверенные участки [3].

Тестирование сборки после проведения тестирования всех отдельных программных компонентов, выполняют сборку системы, в итоге чего будет создана частичная или полная система. В процесс интеграции системы включены сборка и тестирование полученной системы, в ходе которого могут быть выявлены проблемы, которые возникают при взаимодействиях компонентов. Тесты, с помощью которых проверяется сборка системы, необходимо разрабатывать, основываясь на системную спецификацию, причем тестировать сборку следует сразу после того, как созданы работоспособные версии компонентов системы.

В процессе проведения тестирования сборки возникают проблемы с локализацией выявленных ошибок. Компоненты системы имеют между собой сложные связи, и в случае обнаружения аномальных выходных данных бывает сложно выявить источник ошибки. Для облегчения локализации ошибок рекомендуется использование пошагового метода сборки и тестирования системы. Сначала создают минимальную конфигурацию системы и тестируют ее. Затем в минимальную конфигурацию добавляют новые компоненты и снова протестируют, и так далее до полной сборки системы [2,3,4].

На рисунке 3 показан пример выполнения последовательности тестов T1, T2 и T3, которые сначала выполняются в системе, состоящей из модулей A и B (минимальная конфигурация системы). Если в процессе тестирования обнаружен дефект, он исправляется. Затем в систему добавляют модуль C. Тесты T1, T2 и T3 повторяются, чтобы убедиться, что в новой системе нет никаких неожиданных взаимодействий между модулями A и B. Если в ходе тестирования появились какие-то проблемы, то, вероятно, они возникли во взаимодействиях с новым модулем C. Источник проблемы локализован, таким образом упрощается определение дефекта и его исправление. Затем система запускается с тестами T4. На последнем шаге добавляется модуль D

и система тестируется еще раз выполняемыми ранее тестами, а затем новыми тестами T5 [3,4].

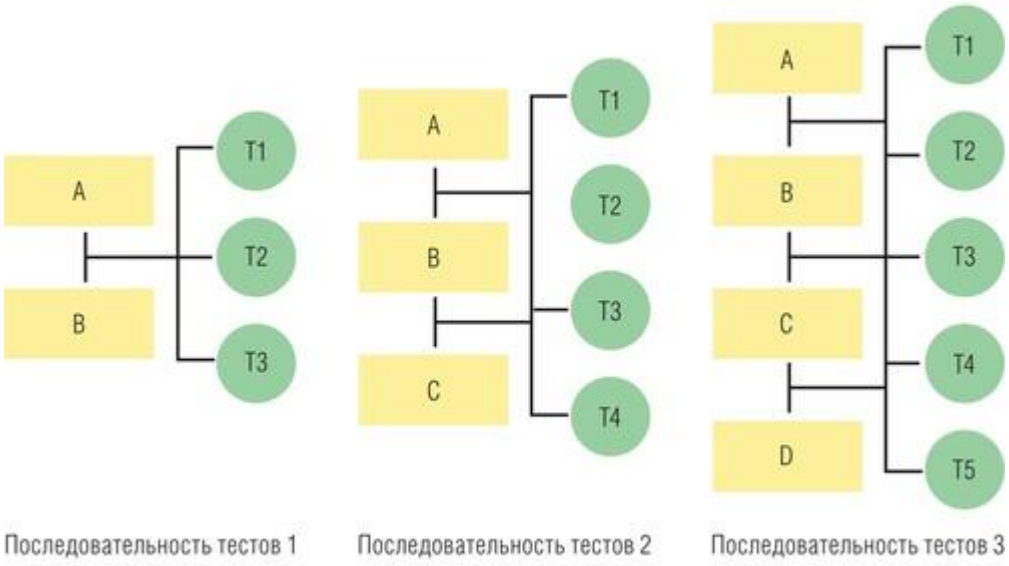

Рисунок 3-Тестирование сборки

Конечно, на практике редко встречаются такие простые модели. Функции системы могут быть реализованы в нескольких компонентах. Тестирование новой функции, таким образом, требует интеграции сразу нескольких компонентов. В этом случае тестирование может выявить ошибки во взаимодействиях между этими компонентами и другими частями системы. Исправление ошибок может оказаться сложным, так как в данном случае ошибки влияют на целую группу компонентов, реализующих конкретную функцию. Более того, при интеграции нового компонента может измениться структура взаимосвязей между уже протестированными компонентами. Вследствие этого могут выявиться ошибки, которые не были выявлены при тестировании более простой конфигурации [2-4].

Инструментальные средства тестирования - дорогой и трудоемкий этап разработки программных систем. Поэтому создан широкий спектр инструментальных средств для поддержки процесса тестирования, которые значительно сокращают расходы на него.

На рисунке 5 показаны возможные инструментальные средства тестирования и отношения между ними.

1. **Организатор тестов.** Управляет выполнением тестов. Он отслеживает тестовые данные, ожидаемые результаты и тестируемые функции программы.

2. **Генератор тестовых данных**. Генерирует тестовые данные для тестируемой программы. Он может выбирать тестовые данные из базы

данных или использовать специальные шаблоны для генерации случайных данных необходимого вида.

3. **Оракул.** Генерирует ожидаемые результаты тестов. В качестве оракулов могут выступать предыдущие версии программы или исследуемого объекта. При тестировании параллельно запускаются оракул и тестируемая программа и сравниваются результаты их выполнения.

4. **Компаратор файлов.** Сравнивает результаты тестирования с результатами предыдущего тестирования и составляет отчет об обнаруженных различиях. Компараторы особенно важны при сравнении различных версий программы. Различия в результатах указывают на возможные проблемы, существующие в новой версии системы.

5. **Генератор отчетов.** Формирует отчеты по результатам проведения тестов.

6. **Динамический анализатор.** Добавляет в программу код, который подсчитывает, сколько раз выполняется каждый оператор. После запуска теста создает исполняемый профиль, в котором показано, сколько раз в программе выполняется каждый оператор.

7. **Имитатор.** Существует несколько типов имитаторов. Целевые имитаторы моделируют машину, на которой будет выполняться программа. Имитатор пользовательского интерфейса — это программа, управляемая сценариями, которая моделирует взаимодействия с интерфейсом пользователя. Имитатор ввода/вывода генерирует последовательности повторяющихся транзакций [4,5].

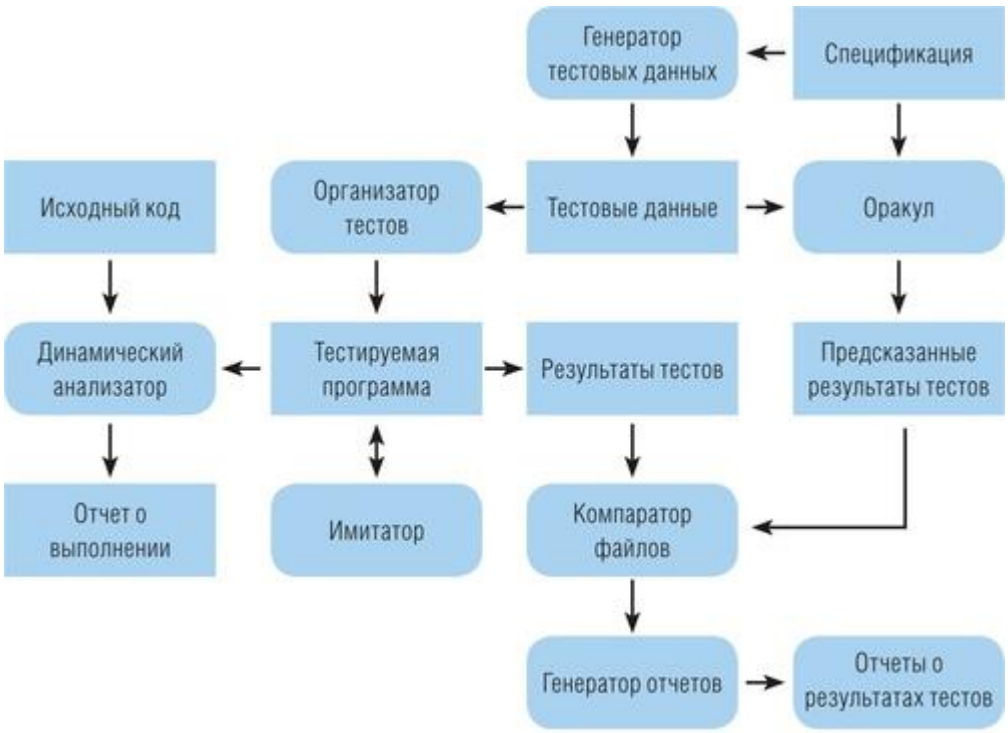

Рисунок 4-Инструментальные средства тестирования

Требования, предъявляемые к процессу тестирования больших систем, зависят от типа разрабатываемого приложения. Поэтому инструментальные средства тестирования неизменно приходится адаптировать к процессу тестирования конкретной системы.

Для создания полного комплекса инструментального средства тестирования, как правило, требуется много сил и времени. Весь набор инструментальных средств, показанных на рис. 5, используется только при тестировании больших систем. Для таких систем полная стоимость тестирования может достигать 50% от всей стоимости разработки системы. Вот почему выгодно инвестировать разработку высококачественных и производительных CASE-средств тестирования [4, 5, 6].

2.3 Методы статического анализа программного обеспечения

Общая модель процесса тестирования дефектов показана на рисунке 6. Тестовые сценарии — это спецификации входных тестовых данных и ожидаемых выходных данных плюс описание процедуры тестирования. Тестовые данные иногда генерируются автоматически. Автоматическая генерация тестовых сценариев невозможна, поскольку результаты проведения теста не всегда можно предсказать заранее.

Рисунок 5- Процесс тестирования дефектов

Полное тестирование, когда проверяются все возможные последовательности выполнения программы, нереально. Поэтому тестирование должно базироваться на некотором подмножестве всевозможных тестовых сценариев. Существуют различные методики выбора этого подмножества. Например, тестовые сценарии могут предусмотреть выполнение всех операторов в программе, по меньшей мере, один раз. Альтернативная методика отбора тестовых сценариев базируется на опыте использования подобных систем, в этом случае тестированию подвергаются только определенные средства и функции работающей системы.

Из опыта тестирования (и эксплуатации) больших программных продуктов, таких как текстовые процессоры или электронные таблицы, вытекает, что необычные комбинации функций иногда могут вызывать ошибки, но наиболее часто используемые функции всегда работают правильно [2,3].

3 Выбор и обоснование методов оценки надежности ПСКН

3.1 Анализ особенностей и требований к оценке надежности ПСКН

Сложность объектов ракетно-космической техники обусловлена многообразием решаемых ими задач научного, социально-экономического и оборонного характера. В перспективе многофункциональные объекты РКТ по своим возможностям будут приближаться к автоматическим летающим роботам, а их группировки и комплексы управления - к большим пространственно распределенным интеллектуальным системам. Такие системы топологически могут быть представлены в виде наземно-космической интеллектуальной информационной сети. Интеллектуальность сети, а также ее эффективность во многом определяются степенью развития средств программного обеспечения. Объектами внедрения и развития программного обеспечения в рамках такой сети являются:

- информационные системы технических и стартовых комплексов, командных пунктов, информационно-аналитические центры и узлы связи космодромов;
- автоматизированные системы управления космодромами;
- наземный автоматизированный комплекс управления (НАКУ) космическими аппаратами в составе отдельных командно-измерительных комплексов (ОКИК), центров управления и контроля полета (ЦУП и ЦКП), баллистических центров (БЦ), средств связи и передачи данных;
- информационные системы правительственных ведомств, занимающихся космической деятельностью;
- информационные системы корпоративного управления космических центров, конструкторских организаций и предприятий ракетно-космической отрасли;
- бортовые вычислительные комплексы космических транспортных средств (ракет-носителей, разгонных блоков, воздушно-космических самолетов, межорбитальных буксиров и т.д.);
- бортовые комплексы управления пилотируемых КА и орбитальных станций;

- бортовые комплексы управления автоматических КА и межпланетных станций;
- технические средства телекоммуникационных систем.

Создаваемое и внедряемое для таких систем и объектов программное обеспечение подразделяется на общее, общесистемное и специальное.

Общее программное обеспечение (ОПО) предназначено для организации вычислительного процесса в локальных и глобальных вычислительных сетях (фрагментах наземно-космической информационной сети), ведения распределенных и пользовательских баз данных, обеспечения защиты информации от несанкционированного доступа. В состав ОПО входят сетевые и пользовательские операционные системы (ОС), распределенные и пользовательские системы управления базами данных (СУБД), программные средства обеспечения защиты информации. Основное назначение общесистемного программного обеспечения (ОсПО) заключается в организации:

- комплексного функционирования взаимосвязанных вычислительных средств фрагментов наземно-космической информационной сети;
- телекоммуникационного обмена данными;
- поддержки "дружественного" интерфейса человека с интеллектуальными программными комплексами (системами);
- функционирования средств отображения информации.

В состав ОсПО включаются программные средства:

- поддержки сетевой технологии и телекоммуникаций;
- электронной почты;
- геоинформационных систем;
- обработки графической и текстовой информации, аудио и видеоинформации;
- проблемно-ориентированных экспертных систем;
- автоматизированного проектирования элементов РКТ;
- систем, реализующих формальную модель нейронной сети (нейрокомпьютеров).

Специальное программное обеспечение (СПО) - это совокупность программных средств, реализующих конкретные функции задач, решаемых в процессе управления объектами и системами. СПО функционирует в среде ОПО и, как правило, информационно и программно сопрягается со средствами ОсПО. СПО создается для автоматизации тех задач, которые невозможно реализовать, опираясь только на средства ОПО и ОсПО.

В укрупненном виде можно выделить ряд групп СПО, предназначенных для решения таких важнейших функциональных задач, как:
навигационно-баллистическое обеспечение;
информационно-телеметрическое обеспечение;
планирование и командно-программное управление объектами РКТ и их группировками;
обработка, обмен и отображение телеметрической и другой информации.

Не прибегая к детализации и перечислению всего состава программных средств, включающих сотни упомянутых групп СПО, укажем, что в совокупности они являются интеллектуальным ядром технических систем и обеспечивают автоматизированную работу взаимосвязанных комплексов аппаратуры объектов РКТ, их группировок и центров управления.

Требования к функциональным возможностям перспективного ПО вытекают прежде всего из характера решаемых задач, а также из экономических критериев, используемых на этапах его создания и эксплуатации. К важнейшим общим требованиям, предъявляемым к перспективному ПО, следует отнести:

- мобильность, т.е. возможность переноса ПО с одной вычислительной платформы на другую, а также из одной операционной среды в другую;
- возможность работы в открытых вычислительных сетях;
- наличие "дружественного" интерфейса, в том числе на естественном языке;
- совместимость, т.е. способность работать во взаимодействии с широким спектром общепризнанных программных средств общего и специального назначения;
- удовлетворение требованиям отечественных и международных стандартов в области программной инженерии.

Выполнение этих требований позволит обеспечить не только работоспособность ПО с широким спектром смежных программных

продуктов, но и возможности использования при его разработке прогрессивных инструментальных средств, зарекомендовавших себя как высокоэффективные. Следует подчеркнуть, что в перспективе нужно обращать самое серьезное внимание на технологию разработки ПО, которая должна базироваться на стандартах. Именно в стандартах, число которых в области программного обеспечения уже приближается к тысяче, заключен огромный отечественный и зарубежный опыт разработки ПО, при использовании которого обеспечивается выполнение упомянутых общих требований.

Для разработки ПО космической деятельности используются инструментальные лингвистические и программные средства технологического программного обеспечения (ТПО). В состав ТПО входят:

- алгоритмические языки низкого и высокого уровней, проблемно и объектно ориентированные алгоритмические языки;
- трансляторы и компиляторы алгоритмических языков;
- средства автоматизированного проектирования специального программного обеспечения - приложений (средства CASE-технологии).

В связи с широким использованием объектов РКТ многоцелевого назначения и активным управлением ими возрастает актуальность применения программного обеспечения, позволяющего реализовать вербально-командное управление их состоянием и функционированием, основанное на распознавании голосовых команд, что позволяет существенно облегчить и повысить оперативность и качество управления. При этом в связи с широким международным сотрудничеством в области космической деятельности может быть преодолен языковой барьер в рамках сотрудничающих сторон на основе использования автоматического распознавания голосовых команд и перевода с одного языка на другой. Проблемным вопросом при создании такого интерфейса является разработка соответствующих физических датчиков, алгоритмов и программных средств, обеспечивающих в совокупности распознавание смысла голосовых команд или текста.

Одной из важнейших проблем является проблема обеспечения безопасности. Средства безопасности предусматривают использование целого ряда механизмов для защиты передаваемых управляющих сигналов, речи и данных. В их число входят средства аутентификации, обеспечения конфиденциальности, управление ключами (включая передачу последних по эфиру) и блокировка (разблокировка) терминалов. Кроме того,

обеспечивается сквозное шифрование, при котором задействуется технология синхронного шифрования потока информации, что позволяет достичь высокого уровня защиты трафика пользователя. Криптографические методы и средства защиты информации в настоящее время экономически более выгодны, чем другие технические и организационные меры, а в ряде случаев только они способны дать существенные результаты. Уже сейчас производители предлагают широкий спектр криптосредств для сокрытия документальной, речевой и другой информации, которые работают в сетях при скоростях передачи информации от десятков бит до сотен мегабит в секунду.

В перспективе должна быть налажена индустриальная технология разработки программного обеспечения с использованием упомянутого ТПО. Экономические данные свидетельствуют о том, что уже в настоящее время доходы от разработок программного обеспечения вносят существенный вклад в валовой национальный продукт ведущих стран, который с углублением информатизации будет только нарастать. Это обстоятельство является залогом самого пристального внимания к созданию программного обеспечения и серьезного отношения к его развитию, в том числе и применительно к ракетно-космической технике.

В РКТ программное обеспечение находит применение и развитие в области научных исследований, проектирования и производства, испытаний и управления полетом, т.е. на всех этапах жизненного цикла ракетно-космической техники. Сегодня и в будущем эффективное проведение научных исследований не мыслится без программного обеспечения поддержки банков данных и баз знаний. Современное проектирование элементов РКТ, ракетно-космических комплексов осуществляется с широким применением систем автоматизированного проектирования (САПР). Основу САПР составля-ют развитые аппаратно-программные средства в составе локальных вычислительных сетей (ЛВС). В ходе производства РКТ программное обеспечение используется как интеллектуальное ядро систем контроля качества продукции и диагностического контроля. В заводских условиях, на контрольно-испытательных станциях и космодромах эффективное проведение испытаний возможно только при наличии высокоинтеллектуальных аппаратно-программных комплексов (АПК). На базе таких АПК создаются перспективные наземные проверочно-пусковые комплексы (НППК). Один из таких комплексов разработки МОКБ "Марс" показан на рис.

Он представляет собой многопроцессорную вычислительную систему с резервированием, работающую в синхронном режиме с БЦВК бортовой

системы управления летательным аппаратом (РН, РБ, КА). НППК обладает высокой информативностью и универсальностью, обеспечивает оперативное внесение изменений в собственное программное обеспечение и в программное обеспечение БЦВК. На предприятиях ракетно-космической отрасли создаются и внедряются программные подсистемы,объединяемые в так называемую информационную систему корпоративного управления. Такая система призвана решать задачи контроля и управления документопотоком, материальными и людскими ресурсами, имеет связь с технологическими АСУ.

Создание мощных информационных систем управления видится в качестве пути развития космодромов. Информационная система управления космодромом относится к классу сложных систем и состоит из следующих фрагментов:

- АСУ управления космодромом по видам деятельности на базе ЛВС органов управления и подчиненных подразделений;
- АСУ подготовкой к пуску транспортных средств выведения (РН, РБ и др.) и запуску КА на техническом комплексе;
- АСУ подготовкой и пуском транспортных средств выведения на стартовом комплексе;
- информационно-аналитического центра в составе измерительного комплекса космодрома;
- компьютеризированных узлов связи и телекоммуникационных средств сбора и передачи данных.

Программные средства перечисленных фрагментов обеспечивают автоматизированное решение функциональных задач, информационное взаимодействие между фрагментами, а также с центром управления полетом, информационными системами корпоративного управления предприятий и ведомств ракетно-космической отрасли, конструкторских организаций и научно-исследовательских учреждений. Без применения программных комплексов на объектах НАКУ, в ЦУПах и баллистических центрах невозможны контроль и управление полетом транспортных средств выведения и космических аппаратов. Дальнейшее развитие космической деятельности будет сопровождаться повышением роли, степени интеллектуализации и расширением областей применения программного обеспечения.

3.2 Анализ вводимых данных на тестирования надежности ПС

Тестирование методом черного ящика базируется на том, что все тесты основываются на спецификации системы или ее компонентов. Система представляется как «черный ящик», поведение которого можно определить только посредством изучения ее входных и соответствующих выходных данных. Другое название этого метода — функциональное тестирование, связано с тем, что испытатель проверяет не реализацию ПО медиаобразовательного портала, а только его выполняемые функции [2,3].

На рисунке 6 показана модель системы, тестируемая методом черного ящика. Этот метод также применим к системам, организованным в виде набора функций или объектов. Испытатель подставляет в компонент или систему входные данные и исследует соответствующие выходные данные. Если выходные данные не совпадают с предсказанными, значит, во время тестирования ПО успешно обнаружена ошибка (дефект).

Основная задача испытателя — подобрать такие входные данные, чтобы среди них с высокой вероятностью присутствовали элементы множества 1е. Во многих случаях выбор тестовых данных основывается на предварительном опыте испытателя. Однако дополнительно к этим эвристическим знаниям можно также использовать систематический метод выбора входных данных, обсуждаемый в следующем разделе [2,3].

Рисунок 6-Тестирование методом черного ящика

3.3 Принципы оценки показателей качества ПСКН

При выделении свойств и соответствующих показателей качества ПСКН необходимо руководствоваться следующими основными принципами:

1) необходимо выделить совокупность свойств, характеризующих качество оцениваемого ПСКН, и упорядочить ее по четко определенному принципу в виде многоуровневой иерархической структуры - дерева свойств;

2) дерево свойств должно отражать все основные особенности использования и функционирования, оцениваемого ПСКН;

3) для каждого из выделенных свойств ПСКН должен быть определен соответствующий показатель качества. Для сложного свойства должна быть установлена группа показателей, необходимых и достаточных для определения этого свойства;

4) для каждого из выделенных свойств и показателей качества ПС должна существовать возможность выражения их в шкалах «лучше – хуже», «больше – меньше»;

5) выбранные показатели качества должны быть скоррелированны с соответствующими свойствами ПСКН. Это значит, что между каждым из выделенных свойств и характеризующими его показателями должно быть установлено однозначное соответствие. Установление такого соответствия позволяет вместо дерева свойств использовать дерево показателей качества программного продукта;

6) исходная (рабочая) номенклатура показателей должна быть открытой, т.е. должна допускать возможность внесения или исключения из нее отдельных элементов. Это требование обусловлено, с одной стороны, большим разнообразием ПС и условий их применения, а с другой недостаточным опытом оценки качества ПС;

7) для выбранных показателей качества ПС должен быть обеспечен принцип непротиворечивости, т.е. улучшение одного показателя не должно приводить к ухудшению другого.

4. Анализ видов, последствий и критичности отказов ПСКН

4.1 Основные принципы и положения анализа видов, последствий и критичности отказов ПСКН

Основными причинами, вызывающими нарушения нормального функционирования ПО, являются:

- ошибки, скрытые в самой программе;
- искажение входной информации;
- неверные действия пользователя;
- неисправность аппаратных средств ИС, на которой реализуется вычислительный процесс.

Ошибки, скрытые в программе. При разработке сложного ПО возможно возникновение ошибок, которые не всегда удается обнаружить и ликвидировать в процессе отладки. В силу этого в программах остается некоторое количество скрытых ошибок. Они являются причиной неверного функционирования этих программ. Среди ошибок подобного рода можно выделить следующие характерные группы.

Ошибки вычислений. Ошибки этой группы связаны с некорректной записью или программированием математических выражений, а также неверное преобразование типов переменных. Вследствие этого получаются неправильные результаты.

Логические ошибки. Эта группа ошибок является причиной искажения алгоритма решения задачи. К ошибкам подобного рода можно отнести неверную передачу управления, неверное задание диапазона изменения параметра цикла, неверное условие и другие ошибки.

Ошибки ввода-вывода. Эти ошибки связаны с неправильным управлением ввода-вывода, формированием выходных записей, определением размера записей и другими неправильно свершенными действиями.

Ошибки манипулирования данными. К числу таких ошибок относятся: неверное определение числа элементов данных; неверные начальные значения, присвоенные данным; неверное указание длины операнда или имени переменной и другие ошибки.

Ошибки совместимости связаны с отсутствием совместимости разрабатываемого или применяемого ПО с операционной системой или другими прикладными программами.

Ошибки сопряжений. группа этих ошибок вызывает неверное взаимодействие ПО с другими программами или подпрограммами, с системными программами, устройствами ЭВМ или входными данными.

Искажение входной информации. Указанная причина вызывает нарушение функционирования ПО, когда входные данные не попадают в допустимую область значения переменных. В этом случае возникает несоответствие между исходной информацией и возможностями программы.

Неверные действия пользователя связаны с неправильной интерпретацией сообщений, с неправильными действиями пользователя при работе в диалоговом режиме. Часто эти ошибки являются следствием некачественной программной документацией.

Неисправность аппаратных средств ИС. Эти неисправности оказывают определенное влияние на характеристики надежности ПО. Появление отказов или сбои в работе аппаратуры приводят к нарушению хода обработки информации и, как следствие, могут искажать как исходные данные, так и саму программу.

появления ошибок в программе является ее отказ. Последствия отказов ПО можно разделить на:

· полное прекращение выполнения функций программы;

· кратковременное нарушение хода обработки информации в ИС.

Степень серьезности последствий отказов ПО оценивается соотношением между временем восстановления программы после отказа и динамическими характеристиками объектов, использующих результаты работы этой программы.

Аварийное завершение работы прикладного ПО легко идентифицируется, так как операционная система выдает сообщения, содержащие аварийный код. Характерными причинами появления аварийного завершения являются ошибки при выполнении макрокоманды, неверное использование методов доступа, нарушение защиты памяти, нехватка ресурсов памяти, неверное использование макрокоманды, возникновение программных прерываний, для которых не указан обработчик, и другие причины.

4.2 Идентификация видов, причин и влияния отказов

Верификация обозначает проверку того, что ПО разработано в соответствии со всеми требованиями к нему, или что результаты очередного этапа разработки соответствуют ограничениям, сформулированным на предшествующих этапах.

Валидация — это проверка того, что сам продукт правилен, т.е. подтверждение того, что он действительно удовлетворяет потребностям и ожиданиям пользователей, заказчиков и других заинтересованных сторон.

Эффективность верификации и валидации, как и эффективность разработки ПО в целом, зависит от полноты и корректности формулировки требований к программному продукту.

Решение любой задачи, выполнение любой функции, возложенной на ЭВМ, работающей в сети или локально, возможно при взаимодействии аппаратных и программных средств. Поэтому при анализе надежности выполнения ЭВМ заданных функций следует рассматривать единый комплекс аппаратных и программных средств.

По аналогии с терминами, принятыми для обозначения показателей надежности ТУ, под надежностью программного обеспечения (ПО) понимается свойство этого обеспечения выполнять заданные функции, сохраняя свои характеристики в установленных пределах при определенных условиях эксплуатации.

Надежность ПО определяется его безотказностью и восстанавливаемостью. Безотказность ПО – это свойство сохранять работоспособность при использовании его для обработки информации в ИС. Безотказностью программного обеспечения оценивается вероятность его работы без отказов при определенных условиях внешней среды в течение заданного периода наблюдения.

В приведенном определении под отказом ПО понимается недопустимое отклонение характеристик функционирования этого обеспечения от предъявляемых требований. Определенные условия внешней среды – это совокупность входных данных и состояние самой ИС. Заданный период наблюдения соответствует времени, необходимому для выполнения на ЭВМ решаемой задачи.

Безотказность ПО может характеризоваться средним временем возникновения отказов при функционировании программы. При этом предполагается, что аппаратные средства ЭВМ находятся в исправном состоянии. С точки зрения надежности, принципиальное отличие ПО от аппаратных средств состоит в том, что программы не изнашиваются и их выход из строя из-за поломки невозможен. Следовательно, характеристики функционирования ПО зависят только от его качества, предопределяемого процессом разработки. Это означает, что безотказность ПО определяется его корректностью и зависит от наличия в нем ошибок, внесенных на этапе его создания. Кроме того, проявление ошибок ПО связано еще и с тем, что в некоторые моменты времени на обработку могут поступать ранее не встречавшиеся совокупности данных, которые программа не в состоянии корректно обработать. Поэтому входные данные в определенной мере влияют на функционирование ПО.

4.3 Оценка критичности и вероятности отказов

Источниками ошибок в программном обеспечении являются специалисты – конкретные люди с их индивидуальными особенностями, квалификацией, талантом и опытом. Вследствие этого плотность потоков ошибок и размеры необходимых корректировок в модулях и компонентах при разработке и сопровождении программного обеспечения могут различаться в десятки раз. Однако в крупных комплексах программ статистика и распределение ошибок и типов выполняемых изменений, необходимых для их исправления, для коллективов разных специалистов нивелируются и проявляются общие закономерности, которые могут использоваться как ориентиры при выявлении ошибок и их систематизации. Этому могут помогать оценки типовых ошибок, модификаций и корректировок путем их накопления и обобщения по опыту создания определенных классов программного обеспечения.

Оценка качества программного обеспечения могут проводиться с двух позиций: с позиции положительной эффективности и непосредственной адекватности их характеристик назначению, целям создания и применения, а также с негативной позиции возможного при этом ущерба – риска от использования ПС или системы. Показатели качества преимущественно отражают положительный эффект от применения программного обеспечения, и основная задача разработчиков проекта состоит в обеспечении высоких значений качества. Риски характеризуют возможные негативные последствия проявившихся в ходе эксплуатации ошибок или ущерб для пользователя при применении и функционировании программного обеспечения.

Согласно [ГОСТ 9126] качество программного обеспечения это весь объем признаков и характеристик программного обеспечения, который относится к ее способности удовлетворять установленным или предполагаемым потребностям.

Качество программного обеспечения оценивается следующими характеристиками:

Функциональные возможности (Functionality). Набор атрибутов, относящихся к сути набора функций и их конкретным свойствам. Функциями являются те, которые реализуют установленные или предполагаемые потребности.

Надежность (Reliability). Набор атрибутов относящихся к способности программного обеспечения сохранять свой уровень качества

функционирования при установленных условиях за установленный период времени.

Практичность (Usability). Набор атрибутов, относящихся к объему работ, требуемых для использования и индивидуальной оценки такого использования определенным и предполагаемым кругом пользователей.

Эффективность (Efficiencies). Набор атрибутов, относящихся к соотношению между уровнем качества функционирования программного обеспечения и объемом используемых ресурсов при установленных условиях.

Сопровождаемость (Maintainability). Набор атрибутов, относящихся к объему работ, требуемых для проведения конкретных изменений (модификаций).

Мобильность (Portability). Набор атрибутов, относящихся к способности программного обеспечения быть перенесенным из одного окружения в другое.

В общем случае под ошибкой подразумевается неправильность, погрешность или неумышленное искажение объекта или процесса, что может быть причиной ущерба – риска при функционировании или применении программы [10]. При этом предполагается, что известно правильное, эталонное состояние объекта или процесса по отношению к которому может быть определено наличие отклонения. Исходным эталоном для любого программного обеспечения являются спецификации требований заказчика или потенциального пользователя, предъявляемых к программам и ожидаемый пользователем или заказчиком эффект от использования программного обеспечения. Важной особенностью при этом является отсутствие полностью определенной программы – эталона, которой должны соответствовать текст и результаты функционирования разрабатываемой программы. Поэтому определить качество программного обеспечения и наличие ошибок в нем путем сравнения разрабатываемой программы с эталонной программой невозможно.

Риски проявляются как негативные последствия проявления ошибок в программном обеспечении в ходе его применения и функционирования, которые могут нанести ущерб системе, в которой применяется это программное обеспечение, внешней среде или пользователям этой системы в результате отклонения характеристик программного обеспечения заданных или ожидаемых пользователем или заказчиком.

Исходя из определения ошибки в программном обеспечении, приведенном выше, можно сделать вывод, что ошибки, проявляющиеся в ходе функционирования программного обеспечения, могут влиять на все

показатели качества. В работе рассматриваются ошибки, проявления которых влияют на надежность функционирования программного обеспечения.

По определению, установленному в [12], надежность – свойство объекта выполнять заданные функции, сохраняя во времени значения установленных эксплуатационных показателей в заданных пределах, соответствующим заданным режимам и условиям использования, технического обслуживания, ремонта, хранения и транспортирования.

Рисунок 1-Надежность по ГОСТ 27.002 – 89

При этом надежность является комплексным свойством, которое в зависимости от назначения объекта и условий его применения может включать безотказность, долговечность, ремонтопригодность и сохраняемость или определенные сочетания этих свойств (рис. 10). Поскольку программное обеспечение в процессе эксплуатации не изнашивается, его поломка и ремонт в общепринятом смысле не производится, то надежность программного обеспечения имеет смысл характеризовать только с точки зрения безотказности его функционирования и возможности восстановления функционирования после отказов вызванных проявлениями ошибок.

В [13] надежность программного обеспечения предлагается характеризовать с помощью следующих характеристик (рис. 9): стабильность, устойчивость и восстанавливаемость.

Рисунок 8-Надежность программного обеспечения

В этом случае стабильность и устойчивость характеризуют безотказность программного обеспечения, а восстанавливаемость – возможность восстановления функционирования программного обеспечения после его отказа. Для количественной оценки надежности программного обеспечения необходимо определить показатели надежности для каждого свойства и методику их определения (оценки).

Для оценки стабильности программного обеспечения возможно использование показателей, характеризующих безотказность технических устройств [12] (рис. 10).

Рисунок 2-Показатели безотказности

5 Экономический раздел

Основной целью создания и внедрения алгоритма разработки является оптимизация и улучшение качества работы инженеров-программистов. Данный алгоритм позволит в кратчайшие сроки импортировать приложение на любые платформы, что позволит сократить трудозатраты.

Организационно-экономическая часть позволит выяснить затраты на проект, определить график выполнения проекта, а также обосновать целесообразность проекта.

Для реализации программной разработки будут задействованы следующие специалисты:

Руководитель проекта

Инженер-программист

Рисунок 3- Структура разработки

- Разработка ТЗ (ГОСТ 34.602)
- Анализ требований
- Составление ТЗ
- Согласование ТЗ
- Разработка алгоритма
- Создание алгоритма
- Генерация алгоритма
- Разработка интерфейса
- Разработка интерфейса пользователя
- Реализация интерфейса пользователя
- Тестирование и отладка
- Написание документации по ИС

- Написание руководства пользователя

Трудоемкость и продолжительность этапов представлена в Таблице 2.

Таблица 2-Трудоемкость и продолжительность этапов

№ раб	Наименование работы	Должность	Трудоемкость, чел./дни	Кол-во раб. Дней
1	2	3	4	5
1.	Разработка ТЗ			5
1.1	Анализ требований	Руководитель проекта Инженер-программист	1 2	2
1.2	Составление ТЗ	Руководитель проекта Инженер-программист	1 2	2
1.3	Согласование ТЗ	Руководитель проекта	1	1
2.	Разработка алгоритма			13
2.1	Разработка алгоритма	Руководитель проекта Инженер-программист	2 5	5
2.2	Генерация алгоритма	Инженер-программист	8	8
3.	Разработка интерфейса			21
3.1	Разработка интерфейса	Руководитель проекта	2 7	7

	пользователя	Инженер-программист		

Продолжение таблицы 2

1	2	3	4	5
3.2	Реализация интерфейса пользователя	Руководитель проекта Инженер-программист	2 7	7
3.3	Тестирование и отладка	Инженер-программист	7	7
4.	Написание документации по ИС			7
4.1	Написание руководства пользователя	Руководитель проекта Инженер-программист	1 7	7
Итого				46

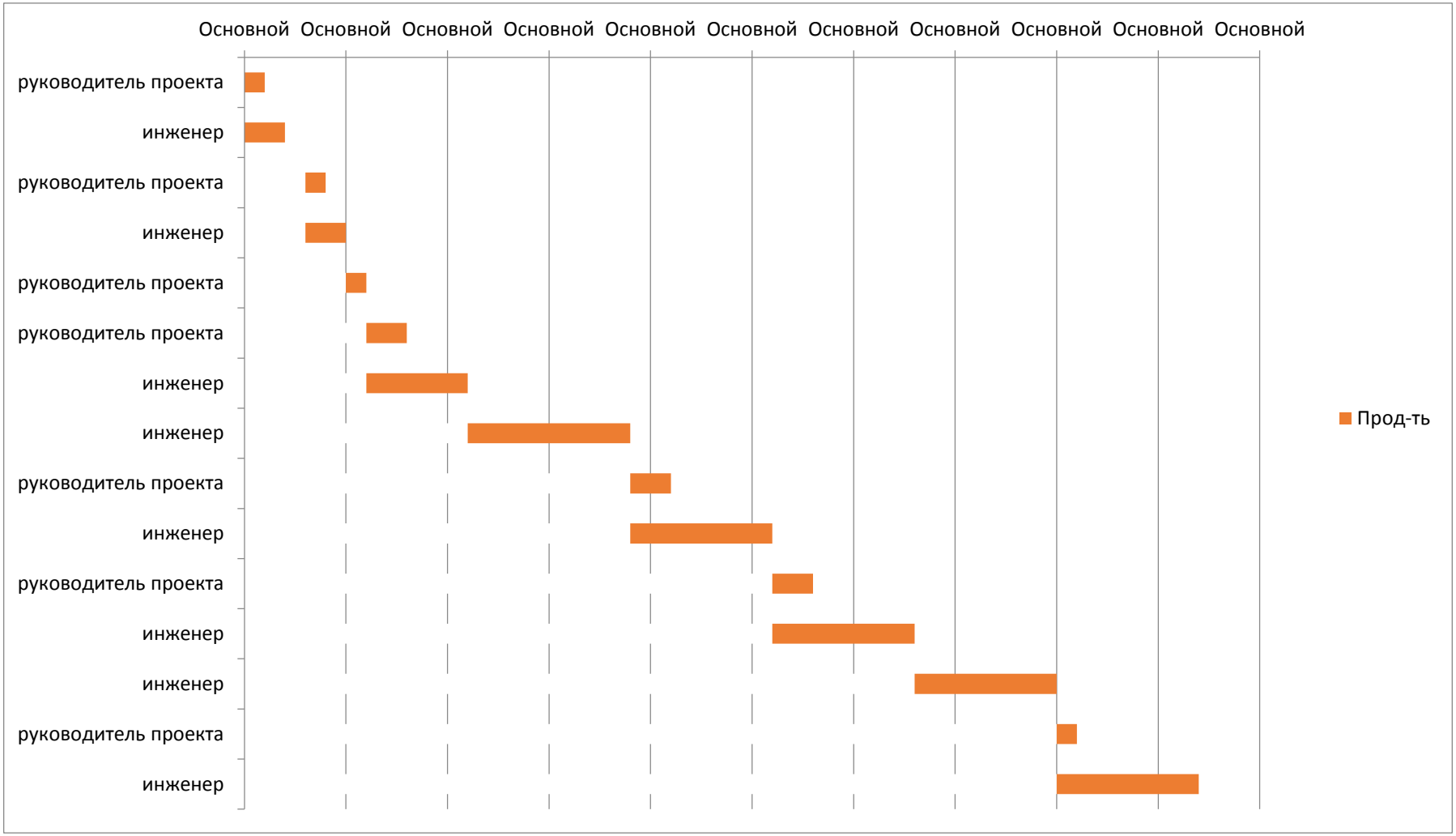

Рисунок 11-Диаграмма Ганта

Затраты определяются по следующим статьям калькуляции:

- материалы и приобретенные изделия;
- транспортно-заготовительные издержки;
- основная заработная плата;
- дополнительная заработная плата;
- отчисления на социальные нужды;
- накладные расходы.

Данная статья включает в себя затраты на приобретение расходных материалов, необходимых для процесса разработки, затраты на приобретение программных средств разработки, библиотек ресурсов.

Потребность в материальных ресурсах определяется на основе спецификаций, в которых указывается наименование и количество используемых материалов. Цена материальных ресурсов определяется по соответствующим прейскурантам.

Таблица 3- Затраты на расходные материалы

№	Наименование материала	Расход (шт.)	Цена (тг/шт.)	Сумма (тг)
1	Ручка гелевая	4	75	300
	Карандаш обычный	5	50	250
	Ластик	3	75	230
2	Упаковка бумаги для принтера (формат А4, 500 лист)	5	750	3900
3	Папка для документов	5	624	3120
4	Картридж для принтера	1	2500	2500
5	Диски CD, DVD	5	100	500
Итого:				10800

На основании финансовой политики предприятия транспортные расходы составляют 15% стоимости материальных затрат. Таким образом: 10800 * 1.15 = 12420 тг.

Среднее количество рабочих дней в месяце составляет – 22 дня.

Значит, дневную заработную плату определяем при делении размера оклада на количество рабочих дней в месяце.

Произведение величины трудоемкости на сумму дневной заработной платы дает величину затрат по зарплате на каждого работника на все время разработки программного продукта.

Таблица 4- Расчет основной заработной платы (ОЗП)

№ п/п	Наименование этапа	Исполнители	Трудо-ем-кость, чел/дней	Мес. оклад, тг.	Оплата за день, тг.	Сумма за работу, тг.
1	2	3	4	5	6	7
1	Разработка ТЗ	Руководитель проекта	3	150000	6818	20454
		Инженер-программист	4	60000	2727	10909
2	Разработка базы данных	Руководитель проекта	2	150000	6818	13636
		Инженер-программист	13	60000	2727	35451

Продолжение таблицы 4

1	2	3	4	5	6	7
3	Разработка интерфейса	Руководитель проекта	4	150000	6818	27272
		Инженер-программист	21	60000	2727	57267
4	Написание документации	Руководитель проекта	1	150000	6818	6818
		Инженер-программист	7	60000	2727	19089
Итог:			55	840000	38180	190896

В дополнительную заработную плату (ДЗП) входят выплаты, которые предусмотрены законодательством о труде за неотработанное по уважительной причине время (оплата очередного и дополнительного отпуска и т.д.), а также вознаграждение за сверхурочную работу и т.п. Основываясь на финансовой политике предприятия, дополнительную заработную плату определяют в процентном соотношении от ОЗП и составляет 20%.

Таким образом:

ДЗП= ОЗП × 20 % = 190896 тг × 0,2 = 38179 тг.

Сумма ОЗП и ДЗП составляет фонд оплаты труда (ФОТ):

ФОТ = 190896 тг +38179 тг = 229075 тг.

ПО = 229075 * 0.1 = 22907.5

Размер отчислений на социальные нужды определяют в процентном отношении к затратам на оплату труда (ФОТ) и составляют 11 %. ПО не облагается социальным налогом. Таким образом :

Отчисл. На соц. Нужды = (ФОТ – ПО) ×11 % = (229075 – 22907.5)×0,11 = 22678,425 тг.

По этой статье относятся расходы по управлению научно-исследовательскими организациями, выполняющие данную разработку, а также затраты по содержанию и ремонту здания, сооружений, оборудования, коммунальные услуги и т.п. В настоящее время накладные расходы на предприятии исчисляются в процентном соотношении к основной заработной плате и составляют 50 % от ФОТ.

НР = ФОТ × 50% =229075 × 0,5= 114537 тг.

Амортизация основных фондов.

Общая сума амортизационных отчислений определяется по формуле:

$$З_{АМ} = summ((Ф_i * Н_{Ai} * Т_{НИРi}) / (100 * Т_{ЭФi}))$$

Где:

$Ф_i$ - стоимость I – го ОФ, тенге.

$Н_{Ai}$ – годовая норма амортизации I – го ОФ, %

$Т_{НИРi}$ - Время работы I – го ОФ за весь период разработки ПП, часов

$Т_{ЭФi}$ - Эффективный фонд времени работы I – го ОФ за год, ч/год;

I - Вид ОФ.

При определнии стоимости ОФ необходимо учесть также затраты на доставку и монтаж, установку ПО. Годовые нормы амортизации ОФ принимаются по налоговому кодексу РК или определяются, исходя из возможного срока полезного использования ОФ.

Наименование оборудования и ПО	Стоимость оборудования и ПО, тг.	Годовая норма амортизации, %	Эффективный фонд времени работы оборудования,дней	Время работы оборудовании, ПО для разработки ПП, дней	Сумма, тг.
Персональный компьютер	170000	20%	365	55	5123.29
Операционная система Windows	13000	20%	365	55	39.17

Принтер	34000	20%	365	55	1024
Программный продукт	35000	20%	365	55	1054
Итого:					7240.46

Таблица 5-Затраты на разработку проекта

№	Наименование	Стоимость, тг.
1.	Материальные затраты с учетом транспортных расходов	12420
2.	Основная заработная плата научного персонала	190896
3.	Дополнительная з/п научного персонала	38179
4.	Соц.налог	22678
5.	Амортизация	7240.46
6.	Накладные расходы	114537
7.	Себестоимость	1349885
8.	Прибыль	404965
9.	НДС	251070
10.	Договорная цена	2070723

Договорная цена разработки складывается из суммы затрат, прибыли, которая установлена в размере 30 % от затрат, и НДС.

П = затраты × 30 % = 1349885× 0,3 % = 404966 тг

Оптовая цена разработки = затраты + П = 1349885 + 404965 = 1754850 тг.

НДС = Оптовая цена разработки × 12 % = 1754850 × 0,12= 210582 тг.

Таким образом, договорная цена на разработку составляет 2070723 тг.

Оценка целесообразности проекта (оценка его характеристик) будет проводиться как для инновационного проекта. Для оценки характеристики используется система рейтинговых оценок. Каждой характеристике присваивается вес, обозначающий важность данной

характеристики. Используя значения характеристики, определяется её балл. Диапазон значений шкалы баллов – от 1 до 10 (чем выше балл, тем лучше характеристика).

Таблица 6-Характеристики разрабатываемого проекта

Характеристика	Вес	Значение	Балл
Цена	0,2	1645920 тг	8
Спрос	0,11	Средний	5
Удобство настройки	0,16	Высокое	8
Гибкость настройки	0,18	Высокая	7
Удобство интерфейса	0,14	Высокое	7
Наличие самотестирования	0,08	Нет	8
Сложность освоения	0,13	Низкая	8
Итог	1		7.35

Алгоритм разработки прикладного программного обеспечения для мобильных операционных систем позволит сократить трудозатраты на разработку приложений. Исходя из этого, можно сделать вывод, что разработка данного алгоритма экономически целесообразна.

Оценка рынка сбыта является, очевидно, важнейшей частью бизнес-планирования на подготовку этого этапа не стоить жалеть ни средств, ни усилий, ни времени. Проанализированный опыт показал, что неудачи многих провалившихся коммерческих проектов были связаны именно со слабым изучением рынков и переоценкой их емкости.

Самыми первыми необходимыми сведениями является: кто будет приобретать данные товары, где располагаются "ниши" предлагаемого продукта на рынке?

Первым этапом в такой работе является оценивание потенциальной емкости рынков, т. е. общей стоимости товара, который покупатели определенных регионов смогут приобрести, например, за месяц или за год.

В первой главе был проанализирован рынок аналогичных продуктов, примерно схожих по характеристикам и их ценовых политик. Был сделан вывод, что данный продукт не уступает в ценовой политике,

а также полностью адаптирован для Республики Казахстан, вся документация периведена на русском языке, имеется техническая поддержка на русском также языке.

Второй этап - оценка потенциальной суммы продаж, т. е. части рынка, которую Вы можете надеяться завоевать, и соответственно максимальной суммы реализации, на которую Вы можете рассчитывать.

В круг покупателей могут войти как достаточно крупные, так и мелкие компании, потому как цена разработки алгоритма может легко войти в бюджет любой компании.

Стоит также указать следующие пункты, связанные с конкурентоспособностью:

- *Финансовый план*

Затраты не являются постоянной величиной. Они зависят от многих обстоятельств, в том числе от состояния рынка. По предварительному анализу максимальная сумма затрат приходится на накладные расходы. Оплата будет производиться единовременно за законченную работу.

- *Квалификация работников*

Наличие квалифицированных работников позволит увеличить эффективность разработок. В настоящее время Инженер совмещает должности аналитика, консультанта, программиста, тестировщика и менеджера по продажам.

- *Защита продукта*

Для защиты от несанкционированного доступа предусмотрено следующее:

- идентификация пользователя;
- проверка полномочий пользователя при работе с продуктом;
- разграничение доступа пользователей на уровне задач и данных.

- *Оценка риска и страхования*

Возможны следующие риски:

нарушение заказчиком установленных сроков приемки законченной работы;

нарушение исполнителями конечного срока по выполнению работы;

неустранение исполнителями в согласованные сроки замечаний по качеству продукта.

Для уменьшения риска на выполнение работы определяется система штрафов за нарушение обязательств.

- *Организационный план*

После завершения этапов создания и тестирования продукта следует этап приемки-сдачи продукта в опытную эксплуатацию. Цель опытной эксплуатации заключается в подготовке продукта и персонала заказчика к производственной эксплуатации. На этапе приемки-сдачи продукта в производственную эксплуатацию продукт вместе с документацией полностью переходит к заказчику.

В организационно экономической части проекта разработаны основные разделы бизнес-плана, составлен календарный график разработки. По плану на разработку проекта потребуется 46 дней. Даны расчеты затрат и договорной цены разработки устройства. Суммарная цена проекта составила 1645920 тенге. Проведена оценка целесообразности алгоритма, которая показала, что алгоритм экономически целесообразен, так как он обладает высокими показателями характеристик.

6. Безопасность жизнедеятельности

В данном разделе дипломной работы определяются оптимальные условия работы пользователя ПК. Конечными пользователями разрабатываемого в дипломной работе алгоритма являются инженеры-программисты, разрабатывающие ПО для мобильных платформ основываясь на данном алгоритме.

Условия труда программистов, которые работают с персональным компьютером, определяются:

- особенностью основных элементов рабочих мест (пространственные параметры рабочих мест и их элементов; размещение компонентов рабочих мест относительно пользователей с учетом видов деятельности);
- условиями окружающей среды (освещением в помещениях и на рабочих местах, микроклиматом, шумами, специфическими факторами, обусловленными особенностью средств отображения данных и пр.);
- характеристикой информационных взаимодействий пользователя и ПК.
- законодательно закреплен целый ряд норм, выполнение которых позволяет во многом обеспечить оптимальные условия, а также уменьшить влияние вредных факторов на здоровье работника.

В данном разделе проведен анализ условий работы сотрудника банка на ПК в соответствии с нормативными документами, разработаны технические и организационные решения по следующим вопросам:

- освещение рабочего места
- микроклимат помещения
- шум и вибрация на рабочем месте
- эргономические параметры устройств ввода/вывода ПК

Рабочее место пользователя находится в помещении, имеющем следующие характеристики:

- площадь – 35 м²;

- длина – 7 м;
- ширина – 5 м;
- высота – 3 м;
- количество окон – 2;
- площадь одного окна – 3 м;
- высота свеса светильников от основного потолка – 0 м;
- число рабочих мест – 6;
- побеленный потолок, стены окрашены в светлую краску.

На рис. 13 приведен план помещения, для которого разрабатываются мероприятия по охране труда.

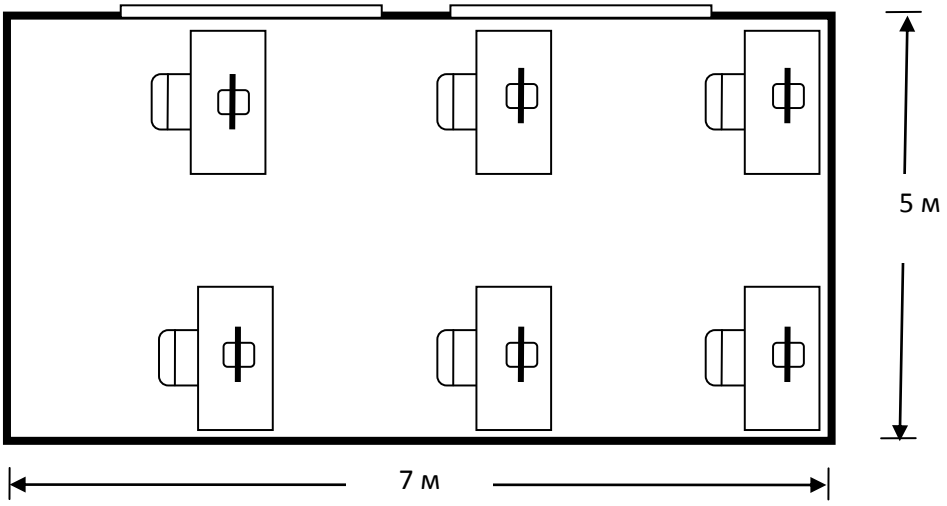

Рисунок 12-План помещения

Основные элементы рабочего места: стол, стул, ЖК-монитор, клавиатура, мышь, системный блок. Блок оборудован жестким диском, системой охлаждения и приводами.

Характеристики рассматриваемого рабочего места:

- длина рабочей области – 1,5 м;
- ширина рабочей области – 0,8 м;
- высота рабочей поверхности от пола – 0,7 м;
- высота сиденья над уровнем пола – 0,5 м;
- расстояние от глаз до экрана – 65 см;

- расстояние от глаз до клавиатуры – 45 см;
- расстояние от глаз до документов – 50 см;
- возможность регулирования экрана по высоте и наклону.

Персональные компьютеры применяются преимущественно при наборе текста (ввод данных) и в диалоговых режимах в процессе редактирования, верстки, правки текстов и оформления отчетной документации:

- операции по вводу информации характеризуют высокими скоростями переработки данных, высокими темпами работ, низкой потребностью в обмене данными и небольшой частотой принятия решений. Такая работа не требует больших умственных и зрительных усилий, но сопровождается локальной мышечной нагрузкой;
- диалоговые виды работ (редактирование, правка, верстка и др.) характеризуют средней скоростью ввода данных, непостоянной (неритмичной) необходимостью для обмена информацией с персональным компьютером, которые сопровождаются принятием решений.

Характер выполняемых работ в соответствии с СанПиНом 2.2.2 546-96 можно отнести к группе «Б» – творческая работа в режиме диалога с ЭВМ.

Основное рабочее положение – положение сидя.

Персональные компьютеры являются источниками вредных и опасных факторов производственной среды: электромагнитного поля и статического электричества. Достаточно часто условия работы при работе на персональном компьютере могут усугубиться повышенным уровнем шумов, неудовлетворительным состоянием микроклиматических условий и недостаточной освещенностью на фоне общего увеличения зрительного и нервно-эмоционального напряжения. Работа на персональном компьютере часто сопровождается небольшой двигательной активностью и монотонией.

Особенность работы на персональном компьютере состоит в постоянном и значительном напряжении функций зрительного

анализатора, которое обусловлено необходимостью разрешения объектов (символы, знаки и т.п.), если на мониторе присутствуют:

- строчная структура экрана;
- мелькание изображения;
- недостаточная освещенность поля монитора;
- недостаточная контрастность объектов разрешения;
- необходимость постоянной переадаптации зрительного аппарата к различному уровню освещенности монитора, оригинала и клавиатуры.

Освещение на рабочих местах пользователей персональных компьютеров следует организовывать таким образом, чтобы пользователи могли бы без напряжения зрения выполнять свои задачи. Утомляемость органа зрения зависит от следующих причин:

- недостаточное освещение;
- чрезмерная освещение;
- неправильное направление света.

Недостаточное освещение вызывает дополнительное напряжение органов зрения, ослабляет внимание, приводит к наступлению преждевременной усталости. Чрезмерно яркая освещенность способна вызвать ослепление, раздражение и резь в глазах. Неправильное направление света на рабочих местах создает резкую тень, блики, дезориентирует пользователя.

Параметры микроклимата изменяются в достаточно широком пределе, в то время как необходимое условие жизнедеятельности человека состоит в поддержании постоянной температуры своего тела из-за свойства терморегуляции, т.е. способности организма регулировать отдачу тепла в окружающее пространство.

Наличие шумов вызывает ухудшает условий труда, оказывает вредное воздействие на организм. При длительном воздействии шумов на людей уменьшается концентрация внимания, возникает нарушение физиологический функций, появляется усталость в связи с повышенной энергетической потребностью и нервно-психическим напряжением, ухудшается речевая коммутация, может снижаться острота слуха, увеличивается кровяное давление, ослабляется внимание. Сильные и продолжительные шумы могут становиться причинами

функционального изменения сердечнососудистой и нервной системы. Длительное действие интенсивных шумов (выше 90 дБ) на слух человека может привести к его частичной или полной потере.

Нервно-эмоциональное напряжение пользователей при работе на персональном компьютере может возникать в результате дефицита времени, больших объемов и плотности данных, особенности диалогового режима общения человека и персонального компьютера, (сбой, оперативное ожидание, психологическая особенность работы оператора персональном компьютере, связанная с эмоционально-волевой сферой), ответственность за безошибочность данных.

Темп работы пользователей на персональном компьютере при вводе информации (текст, данные и т.п.) определяет объем и характер производственных заданий и временя его выполнения. При выполнении операций ввода данных количество малых стереотипных движений кисти и пальцев рук за смену может превышать 50 тыс., что в соответствии с гигиенической классификацией условий труда относят к категории вредных и опасных работ.

При интенсивной работе на персональном компьютере могут развиваться различные заболевания кисти рук:

- синдром запястных каналов (ладонь и запястье немеют, онемение большого, указательного и среднего пальца) появляется в результате длительной интенсивной работы при условиях недостаточности рабочего места для рук;
- отеки и болезненность в области сухожилий пальцев, чаще всего большого;
- воспаление общего сухожилия мышц-разгибателей, которые расположены около локтя, из-за их чрезмерной нагрузки (обуславливается неправильным положением руки, обычно, если они располагаются выше, чем требуется).

Неправильные параметры визуальных эргономических параметров персонального компьютера могут привести к обострению хронических заболеваний глаза, проявлению наследственных предрасположенностей.

В соответствии с требованиями СанПиН 2.2.2/2.4.1340-03 помещение для эксплуатации персонального компьютера должно обладать естественным и искусственным освещением. Естественное и

искусственное освещение должны соответствовать требованию действующих нормативных документов. Окна в помещении, где эксплуатируют вычислительную технику, преимущественно должны ориентироваться на север или северо-восток, оконные проемы должны оборудоваться регулируемыми устройствами типа: жалюзи, занавеси, внешние козырьки и др.

Рабочее место следует располагать таким образом, чтобы монитор ориентировался боковой стороной к световому проему и естественный свет падал преимущественно слева. Искусственное освещение следует осуществлять системой общего равномерного освещения. Освещенность на поверхности рабочего стола должна составлять 250-550 лк. Освещение не должно создавать блики на поверхности монитора. Освещенность поверхности монитора не должна превышать 350 лк.

Яркость у светящейся поверхности (окно, светильник и др.), которая находится в поле зрения работника, должна превышать 250 кд/м2, яркость бликов на экране персонального компьютера не должна быть больше 40 кд/м2, яркость потолка должна быть меньше 200 кд/м2.

Яркость светильников для создания общего освещения в зоне углов излучения от 500 до 900 градусов с вертикалью в продольных и поперечных плоскостях не должна составлять более 200 кд/м2, защитные углы светильников должны быть не менее 400.

Неравномерное распределение яркости в поле зрения работающего за персональным компьютером должно быть ограничено, соотношение яркостей рабочих поверхностей не может превышать 3:1-5:1, а между рабочей поверхностью и поверхностью стен и оборудования 10:1.

Светильники для местного освещения не должны обладать просвечивающими отражателями с защитными углами менее 350. В качестве источника освещения при искусственном освещении следует применять преимущественно люминесцентные лампы типа ЛБ и компактные люминесцентные лампы (КЛЛ). Для светильников местного освещения допустимо использование ламп накаливания, в том числе галогенных ламп.

Для нормированного освещения следует использовать светильники у которых имеются зеркальные параболические решетки, снабженные электронным пускорегулирующим аппаратом (ЭПРА) или многоламповые светильники, снабженные электромагнитным пускорегулирующим аппаратом (ЭПРА), состоящим из равного числа опережающих и отстающих циклов. Применение светильника без

рассеивателя и экранирующей решетки недопустимо. Коэффициент пульсаций не должен быть больше 5%.

Светильник общего освещения должен располагаться сбоку от рабочего места, параллельно линии зрения пользователей при порядном расположении мониторов или локально над рабочим столом при периметральном расположении персональных компьютеров.

Для обеспечения нормируемого значения освещенности следует проводить чистку стекла в оконных рамах и светильников не менее двух раз в году и проводить своевременную замену неисправных ламп.

Основным принципом в нормировании микроклимата является обеспечение оптимальных условий для теплообмена тела пользователя с окружающей средой. Санитарные нормы СанПиН-2.2.2.542-96 устанавливают величину параметров микроклимата, которые создают комфортные условия для работы. Нормы должны быть установлены устанавливаются от поры года, категорий работы и характера производственных помещений. В табл. 7 приведем величины оптимальных значений параметров микроклимата для работы, производимой сидя и не требующей физического напряжения, при которой расход энергии около 150 ккал/ч.

Таблица 7. Характеристика помещений

Время года	Температура воздуха, 0С	Относительная влажность, %	Скорость движения воздуха в помещении, м/с
Холодное	22-25	40-70	0,2
Теплое	23-26	40-70	0,2

Уровень положительных и отрицательных аэроионов в воздухе помещения, в котором находятся персональные компьютеры, должен соответствовать нормам, которые приведены в табл. 8.

Таблица 8-Нормы содержания ионов

1	2	3
Уровни	Число ионов в 1 см3 воздуха	
	n+	n-

1	2	3

Продолжение таблицы 8

1	2	3
Минимально необходимые	400	600
Оптимальные	1500-3000	3000-5000
Максимально допустимые	50000	50000

Согласно нормативам, в ГОСТ 12.1.003-83 характеристиками постоянных шумов на рабочем месте является уровень звукового давления (в дБ) в октавных полосах со среднегеометрическими частотами 31,5, 64, 130, 250, 600, 1100, 2100, 4100, 8000Гц. Для рабочего места пользователя ПК уровень звукового давления для широкополосных постоянных и непостоянных (кроме импульсных) шумов не должен быть больше следующих значений: 85, 72, 61, 55, 47, 44, 43, 40, 39 (дБ) соответственно.

Для тональных и импульсных шумов допустимое значение должно быть на 5 дБ меньше. Уровень звука не должен быть больше 50 дБА, а в зале обработки данных с использованием вычислительных машинах - 65дБА.

Для решения вопроса о необходимости и целесообразности уменьшения шумов следует знать уровень шума на рабочих местах пользователей.

Заключение

Качество программных средств космического назначения следует оценивать исходя из определенных моделей качества. За последнее время было создано большое количество международных стандартов, которые регламентируют модели качества программного средства. Однако применение данных стандартов требует проведения их адаптации методом добавления или исключения некоторых положений стандартов в применении к принципиальным особенностям конкретного программного продукта и условиям их применения.

Построение моделей качества программных средств является одним из важнейших этапов оценки качества и дает возможность определения необходимости применения при оценке качества программных средств космического назначения той или иной характеристики и подхарактеристики. От полноты и адекватности используемых систем характеристик будет зависеть достоверность получаемых оценок. На выбор моделей качества оказывают влияние множество различных факторов.

Основные особенности программных средств космического назначения, которые следует учесть при создании моделей качества, состоят в:

- повышенных требованиях к надежности и функциональной безопасности;
- необходимости гарантированного придания программным средствам космического назначения заданных свойств качеств, безопасности и способности противостояния нарушению функционирования систем, сбоям и ошибкам разного рода;
- ПСКН критического применения должны обладать такими важными характеристиками как отказоустойчивость и восстанавливаемость, наличием встроенных функций самодиагностики и тестирования.
- для всесторонней оценки качества программных средств космического назначения возможно применения параллельно несколько разных моделей.

При выполнении дипломной работы цель достигнута. Изучены методы оценки качества программных средств космического назначения.

При выполнении работы выполнены следующие задачи:

1. Изучена предметная область;

2. Рассмотрены основные требования к оценке качества программных средств космического назначения;
3. Проведен анализ методов оценки показателей качества программных средств космического назначения;
4. Выбраны и обоснованы методы оценки надежности программных средств космического назначения;
5. Сделаны выводы;

Список литературы

1. Соммервилл И. Инженерия программного обеспечения / пер. с англ. Минько А. А., Момотюк А. А., Сингаевская Г. И. — М.: ВИЛЬЯМС, 2002. — 624 с., ил.
2. Тестирование программ / Липаев В. В. — М.: Радио и связь, 1986. — 437 с.
3. Брауде Э., Технология разработки программного обеспечения / пер. с англ. — Спб.: ПИТЕР, 2004. — 655 с., ил.
4. Орлов С., Технологии разработки программного обеспечения /Орлов С. А. — Спб.: ПИТЕР, 2002. — 464 с., ил.
5. Липаев В., Надежность программных средств. — М.: СИНТЕГ, 1998. — 358 с.
6. Вигерс К., Разработка требований к программному обеспечению / пер. с англ.. — М.: «Русская Редакция», 2004. — 576 с., ил.
7. Тюгашев А. А., Ильин И. А., Ермаков И. Е. Пути повышения надежности и качества программного обеспечения в космической отрасли / Управление большими системами. Выпуск 39. М.: ИПУ РАН, 2012. - С.288-299.
8. Липаев В.В. Выбор и оценивание характеристик качества программных средств. Методы и стандарты. Серия «Информационные технологии». - М.: СИНТЕГ, 2001.-228 с., 20 илл.
9. Липаев В.В. Обеспечение качества программных средств.— Москва: Синтег, 2001.-380с., 28 илл
10. Липаев Оценка качества программных средств, Сетевой журнал №3.2002.- (http://www.setevoi.ru/cgi-bin/text.pl/magazines/2002/3/52).
11. Липаев В.В. Качество программных средств. Методические рекомендации. Под общей ред. проф., д.т.н. А.А. Полякова. М: Янус-К, 2002.400 с., 47 илл.

12. Липаев В.В. Экономика производства программных продуктов. Издание второе - М.: СИНТЕГ, 2011. - 358 с.
13. ГОСТ Р ИСО/МЭК 9126-93. Информационная технология. Оценка программной продукции. Характеристики качества и руководства по их применению [текст]. – Введ. 1994–07–01. – М.: Издательство стандартов, 1994. – 15 с.: ил.
14. Александр Попов «Метрики качества программного обеспечения», http://www.pmprofy.ru/pm-books_article.asp?article_id=672&path=pm-books
15. Костогрызов А.И., Нистратов Г.А. "Стандартизация, математическое моделирование, рациональное управление и сертификация в области системной и программной инженерии: М.: Изд. «Вооружения и конверсия», 2004,395 с.
16. Безопасность критических инфраструктур: математические и инженерные методы оценки и обеспечения / под ред. Харченко В.С. – Харьков: Нац. аэрокосм. ун-т «ХАИ». – 2011. – 603с.
17. Исмаил Е.Е. Современные модели качества программных средств и их особенности // МНЖ «Поиск». – № 3 (1), 2015. – с. 272– 282.
18. ISO/IEC 25010:2011 Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models (Системная и программная инженерия. Требования к качеству и оценка системы и программного продукта (SQuaRE). Модели качества систем и программных средств.
19. Мишин В.М. Исследование систем управления: Учебник для вузов. – 2-изд. стереотип. – М.: ЮНИТИ-ДАНА, 2005. – 527 с.
20. Hyatt L.E., Rosenberg L.H. A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality // Proceedings of Product Assurance Symposium and Software Product Assurance Workshop. Noordwijk, 1996. P. 209-212.

21. ECSS-Q-80B-2003 Space Product Assurance: Software Product Assurance.-(http://www.lab.dit.upm.es/~insw/alumnos/recursos/ECSS-Q-80B_10October2003.pdf).
22. Безопасность критических инфраструктур: математические и инженерные методы оценки и обеспечения / под ред. Харченко В.С. – Харьков: Нац. аэрокосм. ун-т «ХАИ». – 2011. – 603с.
23. Исмаил Е.Е. Современные модели качества программных средств и их особенности // МНЖ «Поиск». – № 3 (1), 2015. – с. 272– 282.
24. ISO/IEC 25010:2011 Systems and software engineering. Systems and software Quality Requirements and Evaluation (SQuaRE). System and software quality models (Системная и программная инженерия. Требования к качеству и оценка системы и программного продукта (SQuaRE). Модели качества систем и программных средств.
25. Мишин В.М. Исследование систем управления: Учебник для вузов. – 2-изд. стереотип. – М.: ЮНИТИ-ДАНА, 2005. – 527 с.
26. Hyatt L.E., Rosenberg L.H. A Software Quality Model and Metrics for Identifying Project Risks and Assessing Software Quality // Proceedings of Product Assurance Symposium and Software Product Assurance Workshop. Noordwijk, 1996. P. 209-212.
27. ECSS-Q-80B-2003 Space Product Assurance: Software Product Assurance.-(http://www.lab.dit.upm.es/~insw/alumnos/recursos/ECSS-Q-80B_10October2003.pdf).