Федеральное государственное бюджетное образовательное учреждение
высшего образования «Московский авиационный институт
(национальный исследовательский университет)»

На правах рукописи

Мельничук Александр Владимирович

РАЗРАБОТКА ИНФОРМАЦИОННОЙ СИСТЕМЫ ДЛЯ РАСЧЕТА ВЗЛЕТНО-ПОСАДОЧНЫХ ХАРАКТЕРИСТИК ВОЗДУШНЫХ СУДОВ НА БАЗЕ ЭЛЕКТРОННОГО ПЛАНШЕТА ПИЛОТА

05.13.01 — системный анализ, управление и обработка информации
(авиационная и ракетно-космическая техника)

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель:
Доктор технических наук, доцент
Судаков Владимир Анатольевич

Москва 2020

Оглавление

ВВЕДЕНИЕ

Актуальность темы исследования

Несмотря на то, что взлет и посадка составляют меньшую часть полета, указанные этапы являются наиболее сложными и имеют критичное значение с точки зрения безопасности полетов. При этом, огромное значение для обеспечения безопасного выполнения взлета и посадки имеет расчет взлетно-посадочных характеристик (ВПХ). Основным назначением расчета ВПХ является определение:

- максимальной взлетной и посадочной массы воздушного судна (ВС);
- взлетных и посадочных скоростей (V_1 – скорость принятия решения, представляющая собой максимальную скорость ВС при разбеге на взлетно-посадочной полосе (ВПП), при которой в случае отказа двигателя взлёт может быть как безопасно прекращён, так и безопасно продолжен; V_R – скорость подъема передней опоры шасси; V_2 – безопасная скорость взлета; V_{REF} – скорость захода на посадку).

Значения ВПХ зависят от множества эксплуатационных условий: фактической взлетной массы ВС, температуры окружающего воздуха, давления на аэродроме, характеристик взлетно-посадочной полосы (заявленные длины, уклон, состояние ее поверхности), наличия препятствий по направлению взлета, скорости и направления ветра, а также от управляющих воздействий пилота путем регулирования таких параметров ВС, как тяга двигателя, положение механизации крыла, режим торможения. Также при расчете ВПХ могут учитываться ограничения, определенные наличием допустимых отложенных дефектов и отклонений конфигурации ВС (в соответствии с MEL – перечнем минимального исправного оборудования и CDL – перечнем допустимых повреждений и неисправностей), или определенные политикой эксплуатанта.

В настоящее время для определения ВПХ на отечественных ВС используются номограммы и/или таблицы зависимостей взлетно-посадочных характеристик, приведенные в Руководстве летной эксплуатации (РЛЭ) ВС. Выполнение расчета с их помощью вручную – это длительный процесс,

требующий повышенного внимания, а использование некорректных результатов расчета может привести к авиационному инциденту или авиакатастрофе.

В свою очередь, широкое распространение для расчета ВПХ на импортных ВС в последние годы получили системы EFB (Electronic Flight Bag – электронные полетные планшеты). EFB – это компьютерная информационная система для летного экипажа, состоящая из оборудования и прикладных программ и позволяющая ему использовать функции EFB по хранению, обновлению, отображению и обработке данных, применяемых при выполнении полета или обязанностей, связанных с полетом. Однако существующие системы обладают следующими недостатками:

- база знаний EFB для импортных ВС «жестко» прописана в программном коде, в связи с чем их модификация требует существенных затрат времени и средств на оплату труда программистов;
- существующие решения не позволяют учитывать при определении ВПХ правила, определяемые политикой компании;
- большинство существующих решений не позволяют обеспечить инвариантность по отношению к типам воздушных судов.

Таким образом, учитывая вышеуказанные недостатки существующих систем и то, что EFB для российских гражданских типов ВС не представлены на коммерческом рынке, разработка общих принципов создания программного инструмента автоматизированного определения ВПХ для пилотов, применимого для широкого спектра военных и гражданских воздушных судов российского производства и позволяющего за счет применения технологии продукционной экспертной системы обеспечить гибкость определения ВПХ путем применения правил, содержащихся в РЛЭ, MEL, CDL или определенных политикой эксплуатанта, является актуальной научно-технической задачей.

В таблице 1 представлены рассмотренные в настоящей работе задачи и методы/технологии, использовавшиеся для их решения.

Таблица 1 – Задачи и методы/технологии,
использовавшиеся в работе для их решения

Методы, технологии решения → / Задачи ↓	Методика создания специализированной прикладной информационной системы					Математические методы	
	Методы системного анализа	Технология экспертной системы	Онтологическое проектирование	Методы объектно-ориентированного программирования	Технологии проектирования баз данных	Многокритериальные методы анализа альтернатив	Метод ранжирования альтернатив на основе нечетких предпочтений ЛПР, заданных в нечетких областях
1. Автоматизация расчета ВПХ							
2. Выбор аппаратной платформы информационной системы для расчета ВПХ							
3. Реализация прототипа программного обеспечения EFB							

 - выполнено автором лично;

 - автор принимал участие в реализации;

 - впервые или существенно расширена область применения метода/технологии решения;

 - расширена область применения метода/технологии решения.

Область исследования

Область исследования соответствует пунктам паспорта специальности 05.13.01 «Системный анализ, управление и обработка информации»:

2. Формализация и постановка задач системного анализа, оптимизации, управления, принятия решений и обработки информации.
5. Разработка специального математического и алгоритмического обеспечения систем анализа, оптимизации, управления, принятия решений и обработки информации.
10. Методы и алгоритмы интеллектуальной поддержки при принятии управленческих решений в технических системах.
13. Методы получения, анализа и обработки экспертной информации.

Степень разработанности

Тема разработки и использования унифицированных систем для определения взлетно-посадочных характеристик воздушных судов остается слабо изученной как в отечественной, так и зарубежной литературе.

Автоматизация расчета взлетно-посадочных характеристик самолета основана на математической модели их зависимостей, представленных в официальном руководстве по летной эксплуатации. В работе К.А. Арепьева [1], а также М.Р. Алкиной [2, 3] подробно рассмотрены методы перевода представленных в РЛЭ данных к такому математическому виду, который может быть использован для автоматизации расчета.

В работе В.Н. Моисеева [4] рассмотрены методы и прикладные программные средства оцифровки номограмм для автоматизации инженерно-штурманских расчетов, а также представлены способы предварительной подготовки цифровых изображений номограмм и рекомендации по их аппроксимации.

Однако, в вышеуказанных работах уделено недостаточно внимания технологиям разработки специализированных систем определения ВПХ, предназначенных для использования членами летных экипажей при подготовке к полету и его выполнении.

Использование правил в автоматизированном расчете взлетно-посадочных характеристик подробно рассматривается в работе турецких исследователей и разработчиков (Metin Zontul, Uğur Batak, Orkun Polat), в которой предлагается [5]:

1. разработать программное обеспечение для EFB на базе ноутбука;

2. для реализации расчетов использовать готовые вычислительные модули, предварительно подготовленные и предоставленные компаниями Boeing и Airbus;

3. разработать для указанного ПО единую базу данных на базе MS SQL Server, включающую в себя как данные по аэропортам и воздушным судам, так и сами правила.

Однако предложенный авторами подход обладает следующими недостатками:

1. Авиационные власти РФ не выдают эксплуатационного одобрения на применение ноутбуков в качестве устройств EFB. Также эргономика ноутбуков не позволяет обеспечить безопасность их эксплуатации летными экипажами в кабине ВС.

2. Концепция использования готовых вычислительных модулей, разрабатываемых и предоставляемых производителями иностранных воздушных судов, не подходит к российским ВС в связи с отсутствием для них указанных программных компонентов.

3. Предложенная реализация правил на базе реляционной СУБД является громоздкой и трудно масштабируемой. Отсутствует механизм логического вывода, что не позволяет выполнять рассуждения и работать с конфликтными множествами правил.

В последние годы широкое распространение для расчета ВПХ иностранных ВС в автоматизированном режиме получили специализированные программные приложения (например, для ВС производства корпорации Boeing – «Boeing Onboard Performance Tool» [6], для ВС производства Airbus – «FlySmart+» [7]), реализованные на базе электронного планшета летчика (Electronic Flight Bag - EFB), в качестве которого используются планшетные компьютеры. Однако они обладают следующими недостатками:

- «жесткая» привязка к типам ВС;
- не позволяют использовать правила эксплуатанта при выполнении расчетов ВПХ.

Цели и задачи исследования

Целью является формирование методики создания информационных систем определения ВПХ, инвариантных по отношению к типам ВС.

Для достижения выбранной цели необходимо решить следующие задачи:

1. разработать информационную систему для расчета ВПХ с применением технологии продукционной экспертной системы, в том числе:

• разработать архитектуру информационной системы расчета взлетно-посадочных характеристик воздушных судов с применением технологии продукционной экспертной системы;

• разработать онтологическое представление программного обеспечения для расчета взлетно-посадочных характеристик;

• сформировать алгоритмы расчета ВПХ путем оцифровки номограмм зависимостей ВПХ, представленных в руководстве по летной эксплуатации воздушного судна;

2. создать метод и алгоритм выбора аппаратного обеспечения информационной системы расчета взлетно-посадочных характеристик воздушных судов, используемого в кабине лётного экипажа в формате планшетного компьютера, на основе нечетких суждений;

3. выполнить реализацию программного обеспечения информационной системы в виде клиент-серверного приложения для электронного полётного планшета.

Научная новизна

На основании выполненных исследований получены следующие новые научные результаты:

• предложена оригинальная методика создания информационных систем расчета взлетно-посадочных характеристик на базе электронного полётного планшета, включающая:

- архитектуру информационной системы расчета взлетно-посадочных характеристик воздушных судов с применением технологии продукционной экспертной системы;

- онтологию программного обеспечения для расчета ВПХ, позволившую сформировать структуру и атрибуты базы данных;

- алгоритмы расчетов зависимостей ВПХ, по номограммам представленным в РЛЭ воздушного судна;

• разработана методика выбора аппаратной платформы для EFB на основе нечетких суждений с применением нового метода нечетких областей предпочтений, которая позволила формализовать и упростить процедуру выбора планшетных компьютеров;

• доказана перспективность практического применения предложенного подхода к автоматизированному расчету взлетно-посадочных характеристик в производственной деятельности авиакомпаний.

Теоретическая и практическая значимость работы

Теоретическая значимость исследования обоснована тем, что:

• доказана принципиальная возможность обеспечения инвариантности по отношению к типам воздушных судов в едином программном приложении для EFB и обеспечения гибкости определения взлетно-посадочных характеристик за счет применения технологии продукционной экспертной системы, позволяющей использовать правила производителя ВС и эксплуатанта (авиакомпании).

• применительно к проблематике диссертации результативно использованы новая методика создания информационных систем определения ВПХ и метод многокритериальной оценки альтернатив на основе предпочтений,

заданных в нечетких областях, позволивший формализовать и упростить процедуру выбора планшетных компьютеров;

- изложены принципы построения архитектуры и требования к компонентам программного комплекса, реализующего автоматизированное определение взлетно-посадочных характеристик воздушных судов;
- раскрыты существенные недостатки имеющихся систем определения ВПХ воздушных судов и решена проблема выбора рациональной аппаратной платформы EFB;
- изучены специфические особенности процесса расчета взлетно-посадочных характеристик воздушных судов, проведен анализ структуры данных (категории данных, их взаимосвязь и источники) и их сопоставление с возможностями, предоставляемыми средой создания экспертных систем CLIPS;
- проведена модернизация существующих методов разработки программного обеспечения EFB (позволило упростить разработку программного обеспечения), так и автоматизированного определения ВПХ (позволило обеспечить инвариантность по отношению к типам ВС и большую по сравнению с существующими системами гибкость в процессе определения ВПХ за счет использования технологии продукционной экспертной системы).

Значение полученных соискателем результатов исследования для практики подтверждается тем, что:

- создан программный комплекс информационной системы, реализованной в виде клиент-серверного приложения для электронного планшета летчика (EFB);
- выполнена государственная регистрация программного обеспечения, что подтверждается свидетельством о регистрации №2019660978 (Приложение 1);
- разработанная информационная система апробирована в производственной деятельности Авиакомпании АО «Авиакомпания «РусДжет», что подтверждается актом от 23.09.2019 г. (Приложение 2);
- результаты диссертационной работы внедрены в учебный процесс федерального государственного бюджетного образовательного учреждения

высшего образования «Московский авиационный институт (национальный исследовательский университет)» на кафедре «Математическая кибернетика» (Приложение 3);

- определены перспективы практического использования разработанной информационной системы для расчета взлетно-посадочных характеристик, позволяющей значительно сократить время, затрачиваемое пилотами при подготовке к полету в части определения взлетно-посадочных характеристик, повысить эффективность летной эксплуатации воздушных судов и безопасность выполняемых полетов;
- представлены: методика выбора аппаратной платформы для электронного планшета пилота на основе нечетких суждений с применением нового метода нечетких областей предпочтений, которая позволила формализовать и упростить процедуру выбора планшетных компьютеров; методические рекомендации по использованию разработанной в ходе диссертационного исследования информационной системы летными экипажами, а также предложения по ее дальнейшему совершенствованию.

Методология и методы исследования

Методология и методы исследования основаны на методах системного анализа, онтологического проектирования, экспертных систем, объектно-ориентированного программирования и нечетких множеств.

Положения, выносимые на защиту

Положения, выносимые на защиту, включают в себя следующее:

1. Методика создания информационных систем расчета взлетно-посадочных характеристик на базе электронного полётного планшета, включающая:

1.1. архитектуру информационной системы расчета взлетно-посадочных характеристик воздушных судов с применением технологии продукционной экспертной системы;

1.2. онтологическое представление программного обеспечения для расчета взлетно-посадочных характеристик, позволившее сформировать структуру и атрибуты его базы данных;

1.3. алгоритмы расчетов зависимостей ВПХ, построенные по оцифрованным номограммам, представленным в РЛЭ воздушного судна.

2. Метод и алгоритм выбора аппаратного обеспечения информационной системы расчета взлетно-посадочных характеристик воздушных судов, используемого в кабине лётного экипажа в формате планшетного компьютера, на основе нечетких суждений.

3. Реализация программного обеспечения информационной системы в виде клиент-серверного приложения для электронного полётного планшета.

Степень достоверности и апробация результатов

Алгоритм функционирования разработанного программного комплекса основан на зависимостях взлетно-посадочных характеристиках ВС, представленных в виде комплекса номограмм в официальном одобренном авиационными властями руководстве по летной эксплуатации и построенных производителем воздушного судна. Достоверность результатов, полученных с применением разработанного программного комплекса, подтверждается актом об апробации в АО «Авиакомпания «РусДжет» от 23.09.2019 г (Приложение 2).

Основные положения диссертации докладывались и обсуждались на международных и всероссийских конференциях: XIII Всероссийское совещание по проблемам управления ИПУ РАН (Москва, 2019); 12-я международная конференция «Управление развитием крупномасштабных систем» ИПУ РАН (Москва, 2019); 11-я международной конференции «Управление развитием крупномасштабных систем» ИПУ РАН (Москва, 2018); Международная молодежная научная конференция «XXXIII Туполевские чтения (школа молодых ученых» (Казань, 2017); 16-я Международная конференция «Авиация и космонавтика - 2017» (Москва, 2017); XLII Международная молодёжная научная конференция «Гагаринские чтения» МАИ (Москва, 2016).

Личный вклад автора

В работах [28-30, 67-69, 85, 86] автору принадлежит разработка архитектуры информационной системы для расчета ВПХ с применением технологии продукционной экспертной системы, оцифровка номограмм зависимостей ВПХ и построение их математических моделей для автоматизации расчёта. Работа [40] выполнена автором полностью самостоятельно. В указанной работе приводятся результаты разработки алгоритма логического вывода и выбора инструментального средства для реализации технологии экспертной системы. В работе [57] автору принадлежит разработка онтологической модели предметной области систем EFB с концептуальным описанием программного приложения EFB для расчета взлетно-посадочных характеристик воздушных судов. В работе [87] автору принадлежит разработка унифицированной структуры базы данных информационной системы и программная реализация прототипа клиент-серверного приложения для автоматизированного определения ВПХ ВС на базе электронного полетного планшета. Также автор принимал участие в разработке алгоритма нечеткого ранжирования альтернатив применительно к задаче выбора модели электронного планшета, выполнил обоснование его программной реализации.

Структура и объем диссертации

Диссертация состоит из введения, четырех глав, заключения, списка обозначений и сокращений и списка литературы. Объем диссертации составляет 193 машинописных страниц. Текст диссертации содержит 38 рисунков и 5 таблиц. Список литературы содержит 91 наименование.

Содержание работы

Во **введении** дано обоснование актуальности темы диссертации, сформулированы цель и задачи работы, аргументированы ее научная новизна, теоретическая и практическая ценность, а также представлено сжатое изложение содержания глав диссертации.

В **первой главе** рассмотрено влияние взлетно-посадочных характеристик на безопасность полетов ВС, обоснована целесообразность разработки программного обеспечения для определения ВПХ. Описано влияние основных факторов на параметры взлета и посадки самолета.

Представлен анализ особенностей существующих систем EFB, описаны их классификация и разделение на программные и аппаратные компоненты, приводятся нормативные требования к системам EFB для отечественных авиакомпаний.

Во **второй главе** показана разработанная архитектура информационной системы для определения взлетно-посадочных характеристик ВС, подробно рассмотрены ее компоненты. Разделение системы на блоки необходимо для обеспечения ее адаптивности и расширяемости, позволяя учитывать специфику различных типов воздушных судов.

Рассмотрены основные понятия и каноническая структура экспертной системы, описан способ представления знаний с помощью продукционных правил. Приведены примеры продукционных правил для определения ВПХ, на их примере показан процесс получения новых фактов в ходе логического вывода.

Представлена разработанная математическая формализация алгоритма прямого логического вывода. В конце первой главы предложено использование алгоритма Rete для ускорения логического вывода в разрабатываемой информационной системе. Обоснован выбор инструментального средства CLIPS для реализации технологии продукционной экспертной системы.

Обоснован выбор программного обеспечения для создания онтологии. Приводится описание разработанной онтологии предметной области EFB.

Приведена разработанная на базе онтологии EFB унифицированная модель структуры базы данных информационной системы.

Рассмотрен использующийся в работе подход к созданию математической модели ВПХ конкретного типа ВС на основании номограмм, представленных в РЛЭ. Описаны этапы оцифровки номограмм: предварительная подготовка

цифровой копии изображения, оцифровка кривых, создание математической модели зависимостей параметров. Показаны примеры полученных функций.

Третья глава посвящена задаче выбора аппаратной платформы для разрабатываемой инфомационной системы определения ВПХ. Для решения поставленной задачи предложен новый метод определения нечетких суждений ЛПР на основе экспертных суждений и разбиении пространства критериев на нечеткие области. Представлен математический аппарат, реализующий предложенный подход, формата представления нечетких предпочтений. Продемонстрирован пример результатов ранжирования аппаратных платформ EFB.

Четвертая глава посвящена программной реализации информационной системы для определения ВПХ в виде программного приложения для электронного полетного планшета EFB. Представлено обоснование выбора и описание средств разработки программного приложения. Описан интерфейс разработанного приложения и порядок работы в режимах расчета параметров взлета и расчета параметров посадки. Показан пример использования приложения для ВС Ту-204-100В и Sukhoi SuperJet 100. Представлены разработанные рекомендации по использованию приложения членами летных экипажей при подготовке и выполнении полета.

В **заключении** приведены выводы по работе.

ГЛАВА 1. Анализ зависимостей взлетно-посадочных характеристик воздушных судов, существующих систем EFB и разработка продукционных правил

1.1. Анализ зависимостей параметров взлета и посадки воздушных судов

1.1.1. Понятие взлетно-посадочных характеристик и их влияние на безопасность полетов воздушного судна

Согласно статистике Boeing, с 2009 по 2018 годы около 40% авиационных инцидентов и катастроф приходятся на этапы взлета и посадки [8]. И несмотря на то, что указанные этапы составляют меньшую часть полета, они являются наиболее сложными и имеют критическое значение с точки зрения безопасности полетов.

При этом, огромное значение для обеспечения безопасного выполнения взлета и посадки имеет расчет взлетно-посадочных характеристик (ВПХ) [9].

Основным назначением расчета ВПХ является определение:

- максимально допустимой взлетной массы ВС;
- характерных скоростей на взлёте и посадке (V_1, V_R, V_2, V_{REF});
- максимально допустимой посадочной массы ВС.

V_1 (скорость принятия решения) – скорость при разбеге ВС на ВПП, при достижении которой в случае отказа двигателя взлёт может быть как безопасно прерван, так и безопасно продолжен. В случае отказа двигателя до достижения самолетом скорости V_1 – взлёт прекращается, если отказ двигателя произошел на скорости выше V_1 – то взлёт продолжается. При ее расчете учитываются: фактический вес ВС, положение механизации крыла, давление и температура окружающего воздуха. Значение данной скорости возрастает при снижении давления, увеличении температуры или веса ВС.

V_2 – безопасная скорость взлета для конкретных условий взлёта, наименьшее значение скорости, на которой самолет управляем при наборе высоты после взлета с отказавшим двигателем. Данная скорость должна быть достигнута до высоты 10,7м. (35 футов) над поверхностью ВПП.

V_R (скорость подъема передней опоры шасси) – значение скорости, при которой пилот начинает поднимать переднюю стойку шасси при взлете ВС.

V_{REF} (посадочная скорость) – скорость самолёта при посадке на высоте с выпущенной в посадочное положение механизацией и выпущенными шасси.

В соответствии с требованиями норм летной годности, расчеты ВПХ должны выполняться с учетом отказа двигателя.

Значения ВПХ зависят от множества эксплуатационных условий [10-12]: фактической взлетной массы воздушного судна, температуры окружающего воздуха, давления на аэродроме, характеристик взлетно-посадочной полосы (заявленные длины, уклон, состояние ее поверхности), наличия препятствий по направлению взлета, скорости и направления ветра. Значительное влияние на ВПХ оказывают управляющие воздействия пилота путем регулирования таких параметров ВС, как тяга двигателя, положение механизации крыла, режим торможения. Также при расчете ВПХ могут учитываться ограничения, вызванные наличием допустимых отложенных дефектов и отклонений конфигурации ВС (в соответствии с MEL – перечнем минимального исправного оборудования и CDL – перечнем допустимых повреждений и неисправностей), или определенные политикой эксплуатанта.

MEL (Minimum Equipment List) – представляет собой перечень минимального наличия исправных систем, приборов и оборудования, при котором разрешено эксплуатировать самолет.

CDL (Configuration Deviations List) – перечень допустимых повреждений и неисправностей и связанных с ними ограничений в полете.

Для воздушных судов российского производства расчет ВПХ производится пилотами с использованием специализированных таблиц анализа условий выполнения взлета и посадки (именуемые также Runway Analysis) или номограмм зависимостей ВПХ в соответствии с руководством по летной эксплуатации (РЛЭ) воздушного судна.

Таблицы анализа условий выполнения взлета и посадки содержат информацию об ограничениях летно-технических характеристик определенного

воздушного судна для конкретной ВПП с учетом взлетной тяги, температуры окружающего воздуха, составляющей ветра, состояния поверхности ВПП, конфигурации самолета и режима работы систем, влияющих на тягу силовых установок.

Номограммы представляют собой графическое представление зависимостей ВПХ от значений влияющих на них критериев. Их построение осуществляется производителем с помощью математического моделирования и результатов проведения комплекса сертификационных летных испытаний [13]. При этом, в связи с невозможностью охвата всего объема эксплуатационных условий при проведении летных испытаний и, прежде всего, в связи с их дороговизной и сложностью, в последнее время основная масса исследований в области летной эксплуатации ВС на этапах движения по ВПП осуществляется с помощью математического моделирования. Ввиду влияния на взлетно-посадочные характеристики самолета множества факторов стохастической природы (резкая смена значений коэффициента сцепления на ВПП в связи с наличием луж и сугробов, порывистый боковой ветер, уклоны ВПП И т.д.) требуется использование стохастической математической модели [14]. Стохастические системы широко используются для описания процессов управления движением летательных аппаратов [15-17].

Учитывая, что выполнение расчетов ВПХ с использованием номограмм – сложный и длительный процесс, требующий повышенного внимания, он имеет следующие недостатки:

1) Возникновение риска совершения ошибки при расчетах, обусловленным человеческим фактором, что может привести к таким авиационным инцидентам, как выкатывание воздушного судна за пределы взлетно-посадочной полосы (ВПП), касанию самолётным хвостом поверхности ВПП (tailstrike) [18], сваливанию и т.д.

2) Затруднение выбора оптимальных параметров конфигурации воздушного судна, режима тяги двигателя на взлете, выбора режима торможения при выполнении посадки.

Особую критичность расчет взлетно-посадочных характеристик приобретает при выполнении взлета и посадки на высокогорных аэродромах, при высоких температурах воздуха на аэродроме, попутном ветре и при наличии на взлетно-посадочной полосе загрязнения, влаги, льда или снега.

1.1.2. Факторы, влияющие на взлетные характеристики самолёта

Этапом взлёта называется участок полёта, включающий в себя разбег и отрыв самолёта с последующим набором высоты.

Сегмент собственно взлёта начинается с момента страгивания самолёта с места на линии старта, разбега по взлётной полосе, отрыва от земли и набора высоты.

Разбег представляет собой движение ВС по ВПП с увеличением скорости от нуля до скорости отрыва.

Длина разбега зависит от скорости отрыва и ускорения на разбеге, использование закрылков оказывает наибольшее влияние на ее уменьшение. Однако, существенное влияние на взлётные характеристики оказывают также следующие факторы:

1. Режим работы двигателей: при увеличении значения тяги двигателей происходит увеличение силы ускорения и уменьшение длина разбега.

2. Величина взлётного веса самолёта: при его увеличении, для выполнения взлета увеличатся также потребные скорость и длина разбега. Таким образом, увеличение взлётного веса ВС негативно влияет на взлётные характеристики ВС.

3. Положение механизации крыла: увеличение угла выпуска закрылков приводит к увеличению коэффициента подъёмной силы. Таким образом, потребная подъёмная сила образуется при меньшем значении скорости ВС, самолёт раньше отрывается от земли, уменьшается взлётная дистанция. Но чем больше выпуск

механизации крыла, тем выше и значение силы сопротивления, увеличивающейся быстрее подъёмной силы, что приводит к уменьшению градиента набора высоты.

При малых углах отклонения закрылков в основном увеличивается подъёмная сила без существенного роста сопротивления и достигается высокое аэродинамическое качество самолёта. Малые углы чаще применяются при взлёте в условиях высоких температур, когда, несмотря на увеличение длины разбега, удаётся получить более высокий (нормируемый) градиент набора высоты.

4. Направление и скорость ветра: наличие ветра изменит длину разбега и величину взлётной дистанции: встречная составляющая ветра уменьшит их, а попутная – увеличит. Наличие встречного или попутного ветра может изменить величину максимально допустимого взлётного веса самолёта, не повлияет на скорости V_1 и V_2. Боковая составляющая ветра не оказывает влияния на взлётные характеристики ВС.

5. Состояние поверхности взлётной полосы: снижение коэффициента силы трения на ВПП приводит к уменьшению длины разбега. При этом, при наличии на ВПП слякоти и воды, значительно увеличится длина разбега ВС, а также увеличится длина пробега ВС на прерванном взлёте, что приводит к необходимости уменьшения взлетного веса ВС и скорости принятия решения.

6. Превышение аэродрома над уровнем моря: необходимо учитывать поправку на превышение аэродрома, поскольку, чем больше превышение аэродрома, тем меньше плотность воздуха, т.е. меньше его давление. Следовательно, длина разбега увеличится, а взлётные характеристики ухудшатся, что может привести к необходимости уменьшения взлётного веса самолёта.

7. Давление воздуха: чем оно ниже, тем меньше значение плотности воздуха. Чем ниже значение плотности воздуха, тем выше должна быть скорость ВС, и, как следствие, увеличится длина взлётной дистанции.

8. Температура окружающего воздуха: с ее увеличением, снижается плотность воздуха, для компенсации которой истинная скорость должна быть увеличена. Следовательно, взлётная дистанция увеличится. Возрастание значения температуры воздуха и уменьшение давления в совокупности приводят к

увеличению взлётной дистанции и уменьшению градиента набора высоты. При этом, может возникнуть необходимость снижения фактического веса самолёта.

9. Угол уклона взлётной полосы. Представлен в процентном отношении и может быть положительным (на гору) или отрицательным (с горы). Угол уклона полосы оказывает влияние на ВПХ ВС аналогично влиянию составляющей ветра, так как возникает горизонтальная составляющая веса самолёта, направленная по ходу или против движения самолёта. При уклоне ВПП вверх, ухудшается возможность ускорения самолёта, увеличивается взлётная дистанция, укорачивается дистанция прерванного взлёта. Поэтому, в зависимости от имеющихся для данного взлёта ограничений, уклон полосы вверх может как увеличить максимально допустимы взлётный вес, так и уменьшить его.

Продолженный взлёт выполняется как нормальный до момента отказа двигателя, после чего взлёт продолжается и завершается с отказавшим двигателем.

Прерванный взлёт выполняется как нормальный до момента отказа двигателя, после чего выполняется его прекращение путем торможением ВС до полной его остановки.

Дистанция прерванного взлёта включает в себя расстояния, пройденные ВС: на разбеге с работающими двигателями, за время принятия решения о прекращении взлёта и расстояния при выполнении торможения. При ее расчете предполагается, что отказ двигателя произошёл на скорости V_1.

Вышеперечисленные в настоящем подразделе факторы являются исходными данными информационной системы при определении взлётных характеристик ВС.

1.1.3. Определение возможности выполнения взлёта на уменьшенном режиме работы двигателей

В практике полётов часто бывает, что фактический взлётный вес самолёта меньше максимально допустимого для данных условий. В этом случае для выполнения взлета возможно использование уменьшенного режимы работы двигателей (т.е. ниже взлётного режим), что позволит продлить ресурс работы

двигателей (поскольку будут сокращены как инженерно-технические, так и эксплуатационные расходы на топливо).

Выполнение взлёта самолёта на режиме работы двигателей меньше взлётного подразделяется на два вида: взлёт с уменьшенной тягой и взлёт с ограниченной тягой.

При выполнении взлёта с уменьшенной тягой предполагается уменьшение тяги двигателей не более, чем на 25% от взлётной. Уменьшенная тяга при таком взлёте не является ограничением тяги и предполагает в любой момент в процессе выполнения взлёта, при необходимости, увеличить ее значение до взлётной. Использование уменьшенной тяги разрешено на сухой и влажной ВПП, но запрещается при наличии на ВПП загрязнений и в случае неработоспособности антиюзового устройства.

Взлёт с ограниченной тягой – это взлёт с тягой двигателей меньше потенциальной максимальной взлётной тяги. Для выполнения такого взлёта в РЛЭ данного типа ВС должны быть установлены отдельные и независимые ограничения с указанными лётно-техническими характеристиками. Выполнение взлёта с ограниченной тягой допускается производить и с загрязнённой ВПП (т.е. при наличии на ней воды, слякоти, снега или льда).

Возможность выполнения взлёта на пониженной тяге предусматривается производителем в руководстве летной эксплуатации (а также, в случае наличия технических ограничений, и в MEL) воздушного судна. Поскольку взлет на пониженной тяге значительно влияет на повышение экономической эффективности эксплуатации ВС, важно его предусмотрение и в разрабатываемой информационной системе.

1.1.4. Факторы, влияющие на посадочные характеристики самолёта

Посадка ВС является заключительным этапом полёта и состоит из двух участков: снижения по глиссаде (или по посадочной прямой) и собственно посадки, начинающейся с высоты 50 футов (15 м). На втором участке осуществляется выравнивание самолёта, касание ВПП и пробега по ней до полной остановки ВС.

После выпуска механизации самолёта в посадочное положение резко возрастает сопротивление, поэтому, для того чтобы удержать самолёт на глиссаде, необходима значительная тяга двигателей.

На выдерживании пилот выполняет постепенное уменьшение скорости до посадочной.

Выполнение пробега ВС осуществляется с целью торможения до его остановки или замедления до скорости руления.

При расчёте длины пробега самолёта необходимо учитывать:

- влияние посадочной скорости;
- влияние положения механизации крыла;
- влияние посадочного веса;
- применение тормозов;
- применение посадочных спойлеров;
- направление и скорость ветра у земли;
- температуру и давление воздуха;
- состояние посадочной полосы.

Посадочный вес самолёта ограничивается длиной посадочной полосы и требованиями к градиенту набора высоты при посадке в случае необходимости ухода на второй круг.

При наличии на ВПП слоя воды, слякоти, снега или льда, условия торможения ухудшаются пропорционально ухудшению сцепления колёс самолёта с поверхностью полосы.

Условия торможения – это заявленное состояние покрытия ВПП, позволяющее пилоту оценить ожидаемые качество или степень торможения. Условия торможения характеризуются следующими определениями: хорошее, среднее, плохое и отсутствует, также используются их производные формы: между хорошим и средним, между средним и плохим.

Коэффициент сцепления – это отношение максимально возможной силы трения и вертикальной нагрузки, действующей на колесо, к фактической для данной ВПП. Когда условия торможения на ВПП из-за дождя, снега или

образования льда начинают ухудшаться, администрация аэропорта использует специальное оборудование для определения коэффициента сцепления.

Информация о состоянии ВПП публикуется администрацией аэропорта.

1.1.5. Влияние дефектов MEL и CDL на взлетно-посадочные характеристики самолета

При определении ВПХ должны также учитываться требования соответствующей эксплуатационной документации ВС: руководства по летной эксплуатации, перечня минимального исправного оборудования (MEL), перечня допустимых повреждений и неисправностей (CDL). Также эксплуатант может вводить собственные требования, устанавливающие дополнительные ограничения при определении ВПХ.

MEL и CDL представляют собой документы, состоящие из разделов, каждый пункт которых относится к конкретному отклонению/неисправности ВС и содержит описание условий эксплуатации ВС при возникновении отклонения/неисправности, процедуры для летного экипажа и инженерно-технического состава. Условия эксплуатации ВС, представленные в пунктах MEL/CDL могут быть представлены в виде импликаций вида «если А, то В», где А – наличие определенного отклонения/неисправности ВС и соответствие некоторых эксплуатационных факторов определенным значениям, В – определенные эксплуатационные ограничения, которые могут выражаться в ограничении максимальной взлетной массы ВС, ограничения использования определенных систем ВС, ограничения взлетных режимов двигателя, ограничения режимов торможения и т.д.

Аналогичным образом могут быть представлены и правила, определяемые политикой эксплуатанта. В данном случае А – выполнение некоторых эксплуатационных условий, соответствие внешних эксплуатационных факторов определенным значениям, В – описание соответствующих ограничений, влияющих на ВПХ воздушного судна.

Совокупность вышеуказанных правил будет представлять собой продукционную модель базы знаний разрабатываемой информационной системы для определения взлетно-посадочных характеристик ВС.

1.2. Анализ существующих систем EFB

EFB (Electronic Flight Bag – электронный полетный планшет) - это компьютерная информационная система для летного экипажа, состоящая из оборудования и прикладных программ и позволяющая ему использовать функции EFB по хранению, обновлению, отображению и обработке данных, применяемых при выполнении полета или обязанностей, связанных с полетом [19].

Системы EFB позволяют работать с бортовой документацией воздушного судна (руководства по лётной эксплуатации, руководства по производству полетов, сборников аэронавигационной информации и других, предусмотренных нормативным документом ФАП-128 п.5.68. [20]) в электронном формате, осуществлять комплекс инженерно-штурманских расчетов, могут использоваться в качестве средства коммуникации с наземными службами авиакомпании, и т.д.

Работа с электронной документацией обладает множеством преимуществ перед применением бумажных носителем, в числе которых – качественное улучшение эргономики в кабине воздушного судна, существенного сокращения времени на поиск необходимой информации за счет использования таких функций, как ссылок, закладок и поиска, снижению веса, что в свою очередь приводит к снижению расхода топлива и т.д.

В настоящее время, в Российской Федерации требования к системам EFB отечественных коммерческих эксплуатантов содержатся в нормативных документах ФАП-128, ФАП-246 [21] и ICAO Doc 10020.

Для внедрения системы EFB в свою производственную деятельность, авиакомпания должны получить эксплуатационное разрешение от авиационных властей [22-24]. Получение данного эксплуатационного разрешения является многоступенчатым процессом, включающим в себя подготовку авиакомпаний плана внедрения и подготовки пакета доказательной документации, подача заявки

в авиационные власти для получения разрешения на тестовое использование системы EFB при производстве полетов, подготовка и подача итогового эксплуатационного отчета и, наконец, получение от авиационных властей официального разрешения на использование системы EFB, вносимого в эксплуатационную спецификацию воздушного судна в соответствии с нормативным документом ФАП-246.

Производители авиационной техники и информационных технологий представляют различные системы, которые могут быть использованы в качестве EFB. Поэтому при выборе системы EFB необходимо определиться с целями и задачами, для достижения и решения которых планируется использование данной системы.

Оборудование EFB может быть переносным или встроенным, а выбор решения включает оценку по таким параметрам, как аппаратная платформа, программное обеспечение, крепежное устройство, система электропитания (рисунок 1.1).

1.2.1. Переносные EFB

Под переносными EFB подразумевается переносная аппаратная платформа EFB (переносной планшет EFB), используемая в кабине экипажа, не являющаяся частью сертифицированной конфигурации воздушного судна. Переносные EFB можно использовать как на борту ВС, так и за его пределами.

На переносные планшеты EFB устанавливаются ПО типов A и B. Кроме того, допускается установка различных вспомогательных (не относящихся к EFB) приложений.

Вес, размеры и размещение переносных планшетов EFB не должны оказывать влияния на безопасность полёта.

Электропитание на переносные планшеты EFB может поступать от сертифицированного источника питания ВС.

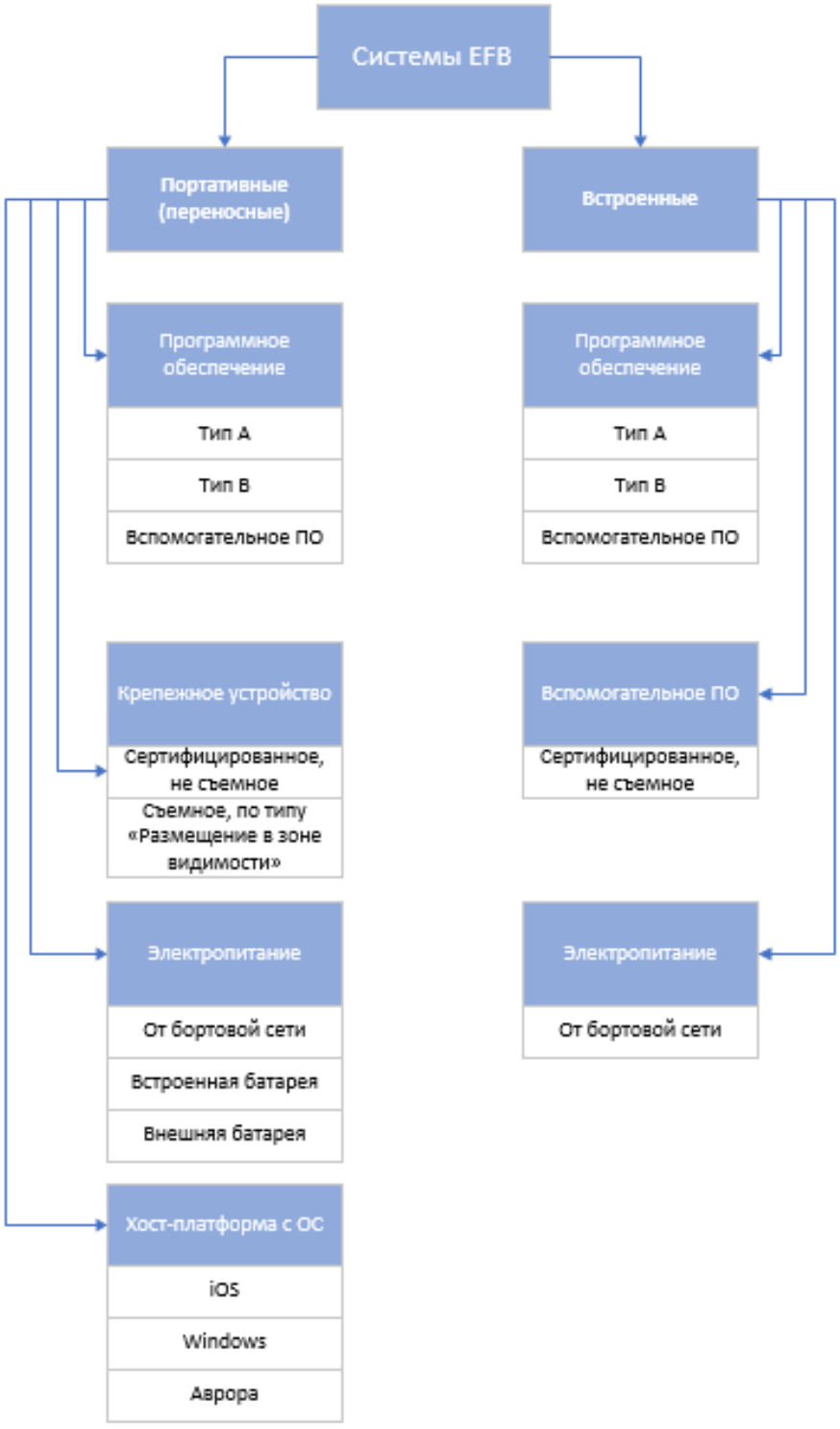


Рисунок 1.1 – Классификация и основные компоненты систем EFB

Использование переносных EFB разрешено на всех этапах полёта, если они установлены в крепежном устройстве (сертифицированном или класса «размещение в зоне видимости»). В иных случаях Переносной EFB должен быть убран на критичных этапах полёта.

Переносные EFB допускается подключать к системам ВС по беспроводному или беспроводному каналу связи только если необходимые средства связи и устройства защиты интерфейса передачи данных предусмотрены конструкцией ВС. Передача и приём данных с/на EFB не должны влиять на работу сертифицированных систем ВС, что должно быть подтверждено соответствующими проверками и тестированием. Отказы системы EFB не должны негативно влиять на работу сертифицированных систем ВС.

В настоящей работе предлагается разработка информационной системы на базе аппаратной платформы переносных EFB, поскольку они обладают меньшей стоимостью по сравнению со встроенными EFB, а также позволяют летному экипажу выполнять предварительное определение ВПХ на этапе подготовки к полету до прибытия на ВС.

1.2.2. Встроенные EFB

Встроенные EFB считаются частью воздушного судна и потому требуют подтверждения на соответствие нормам лётной годности. Платформа EFB включает операционную систему (ОС).

Поскольку в настоящей работе предлагается разработка информационной системы на базе аппаратной платформы переносных EFB, встроенные EFB далее рассматриваться не будут.

1.2.3. Программное обеспечение системы EFB

Функциональность системы EFB зависит от установленного ПО. Классификация ПО основана на его влиянии на безопасность и предназначена для чёткого разделения приложений ПО и нужна для оценки каждого из приложений ПО.

- ПО типа А – это ПО EFB, чей отказ (неисправность) никак не влияет на безопасность; не требуют одобрения и может устанавливаться как на переносные, так и на встроенные EFB.

- ПО типа B – это ПО EFB, неисправность либо неправильное использование которого ограничены условиями несущественного отказа и может устанавливаться как на переносные, так и на встроенные EFB. По типа B требует проведения эксплуатационной оценки и не требует сертификата на соответствие нормам лётной годности.

  К программному обеспечению типа B относятся такие приложения для работы с судовой документацией ВС (MEL, CDL, РЛЭ и т.д.), рабочим планом полета, интерактивными аэронавигационными картами и схемами, приложения для расчета взлетно-посадочных характеристик ВС, центровки ВС и т.п.
- Вспомогательное ПО – это приложения, не относящиеся к EFB и выполняющие функции, которые напрямую не связаны с деятельностью экипажа на борту ВС.

Разрабатываемая информационная система, исходя из своего предназначения, относится к ПО типа B. В данном случае для ее легального использования в авиакомпании не потребуется сертификации на соответствие нормам лётной годности, но понадобится проведение эксплуатационной оценки авиакомпанией.

1.2.4. Крепление EFB «Размещение в зоне видимости»

Крепежное устройство, которое фиксируется на теле члена экипажа (например, наколенное крепление) или на/к детали воздушного судна (например, с помощью присосок). Позволяет удерживать одобренные легкие устройства (планшеты EFB весом не более 1 кг.) в зоне видимости пилота с рабочего места. Не требует сертификации.

1.2.5. Источник питания

Для электропитания системы EFB может использоваться как внутренний источник питания (аккумуляторная батарея), так и бортовая сеть.

Система питания EFB должна соответствовать нормам лётной годности.

Обязательным является проведение проверки электрической нагрузки сети с подключённой EFB в целях проверки оказания отрицательного влияния на другие системы ВС и подтверждения того, что потребление питания не выходит за установленные пределы, а также, что электрическая сеть ВС защищена от отказов и неисправностей системы EFB. Должна быть обеспечена возможность отключения в любой момент источника питания EFB.

1.2.6. Использование приложений EFB для определения ВПХ

В последние годы широкое распространение для расчета ВПХ иностранных ВС в автоматизированном режиме получили специализированные программные приложения, реализованные на базе системы EFB (например, для ВС производства корпорации Boeing – «Boeing Onboard Performance Tool», его интерфейс представлен на рисунке 1.2; для ВС производства Airbus – «FlySmart+», интерфейс представлен на рисунке 1.3).

Рисунок 1.2 – Интерфейс Boeing Onboard Performance Tool

Рисунок 1.3 – Интерфейс FlySmart+

Использование вышеуказанных программных приложений позволяет ускорить процесс расчета ВПХ, а их вычислительные мощности позволяют осуществлять расчеты ВПХ для всех возможных положений механизации крыла и режимов тяги при взлете и выбрать рациональные их значения, что позволит повысить экономическую эффективность эксплуатации ВС (выраженную в экономии ресурсов двигателей, снижении расхода топлива, увеличению коммерческой загрузки), обеспечивая при этом высокий уровень безопасности при взлете и посадке [25-27].

Тем не менее, иностранные программные решения обладают такими недостатками, как невозможность использования правил эксплуатанта в определении ВПХ, а также «жесткая» привязка к типам ВС.

Для автоматизированного расчета ВПХ, турецкими исследователями и разработчиками (Metin Zontul, Uğur Batak, Orkun Polat) было разработано основанное на правилах программное средство для EFB на базе ноутбука (интерфейс представлен на рисунке 1.4). Реализованный подход обладает рядом существенных недостатков, в числе которых – отсутствие механизма логического

вывода, что лишает предложенную систему возможности осуществлять рассуждения и работать с конфликтными множествами и неэргономичность аппаратного средства, что крайне затрудняет использование в кабине пилотов.

Рисунок 1.4 – Интерфейс ПО для расчета ВПХ, созданного турецкими исследователями и разработчиками (Metin Zontul, Uğur Batak, Orkun Polat)

Выводы по главе 1

В резюме по первой главе сделаны следующие выводы:

- Расчет ВПХ – это сложный трудоемкий процесс, ручное проведение которого может приводить к ошибкам и избыточной нагрузке на экипаж. Поэтому целесообразна разработка автоматизированных систем расчета ВПХ.
- Существующие подходы к созданию автоматизированных систем расчета ВПХ ориентированы преимущественно на конкретные типы иностранных летательных аппаратов и не позволяют оперативно переориентироваться с одних воздушных судов на другие. Также в настоящее время на коммерческом рынке для

большинства отечественных самолетов отсутствуют готовые аналогичные решения.

• Правила, представленные в эксплуатационной документации ВС (MEL, CDL) и применяющиеся при определении ВПХ, могут быть представлены в виде продукций, работа с которыми реализуется посредством осуществления прямого логического вывода. Для применения таких импликаций в настоящей работе предложено использование технологии продукционной экспертной системы, что позволит обеспечить большую по сравнению с существующими системами гибкость в процессе определения ВПХ.

ГЛАВА 2. Разработка информационной системы для определения взлетно-посадочных характеристик

2.1. Архитектура информационной системы

Для реализации рассматриваемой в работе системы, автором разработана архитектура, представленная на рисунке 2.1.

Разделение системы на блоки необходимо для обеспечения ее адаптивности и расширяемости, позволяя учитывать специфику различных типов воздушных судов [28-30].

Как видно из представленной схемы, программный комплекс системы расчета взлетно-посадочных характеристик воздушных судов построен в соответствии с клиент-серверной архитектурой, где в качестве клиента выступает электронный планшет летчика (устройство EFB), подключенный к сети интернет посредством беспроводной технологии интернет-соединения (Wi-Fi или с использованием сотовой оператора мобильной связи - 3G, LTE и т.п.), а в качестве сервера – серверная ЭВМ с установленным на ней серверными компонентами.

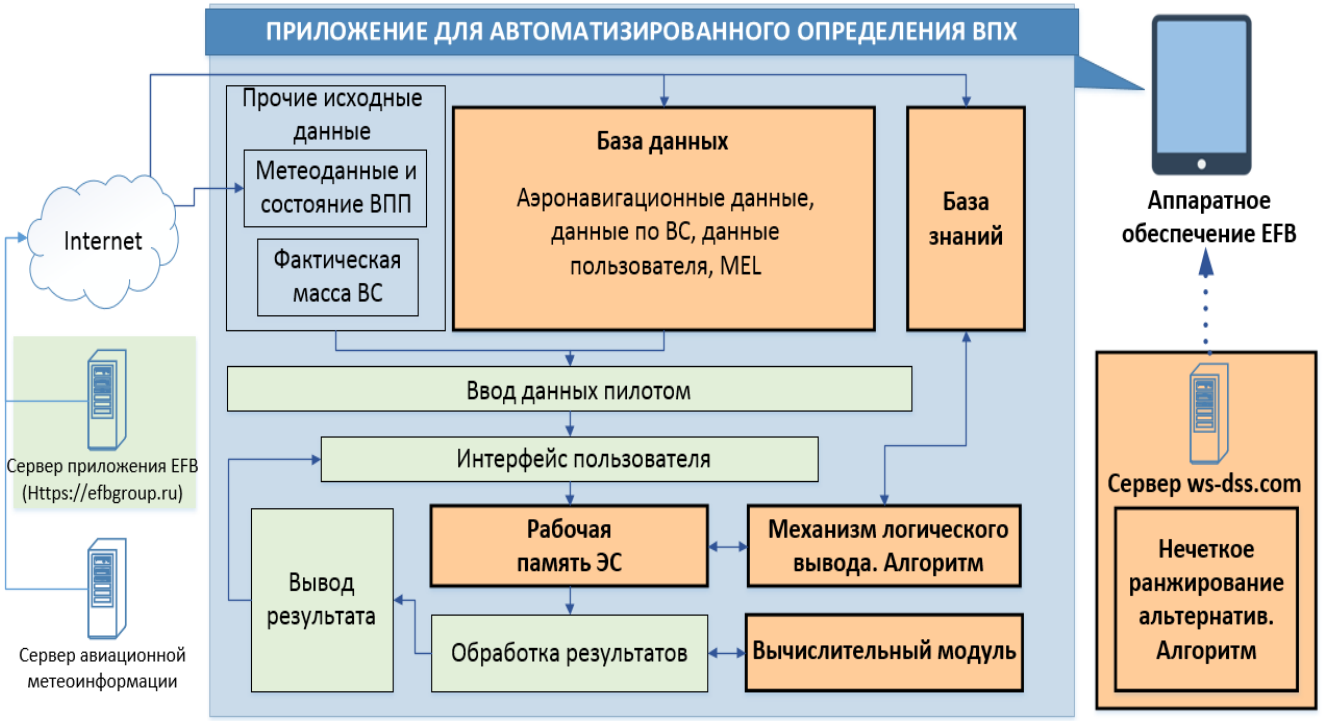


Рисунок 2.1 – Архитектура информационной системы

В составе программного комплекса на стороне клиента можно выделить следующие основные компоненты:

- база данных, включающая в себя аэронавигационное данные [31], данные по ВС, данные пользователя и данные перечня минимально исправного оборудования (MEL);

- элементы, реализующие технологию продукционной экспертной системы (база знаний, рабочая память ЭС, механизм логического вывода),

- модуль обработки результатов;

- вычислительный модуль;

- интерфейс пользователя, используемый для ввода данных пользователем и вывода полученных результатов.

База данных подвержена необходимости регулярных обновлений. Ее актуальность, целостность, точность и качество имеют критическое значение в процессе расчета характеристик взлета и посадки. Подробнее унифицированная модель указанной базы данных рассмотрена в параграфе 2.3 «Онтология предметной области EFB».

Наполнение расчетного модуля производится либо предварительно рассчитанными специфическими для конкретного типа ВС таблицами анализа зависимостей ВПХ, либо электронными данными (оцифрованными номограммами) из руководства по летной эксплуатации (метод оцифровки подробно рассмотрен в параграфе 2.4. «Оцифровка номограмм зависимостей ВПХ, представленных в РЛЭ воздушного судна»).

База знаний предназначена для хранения продукционных правил, используемых при определении ВПХ (подробнее описание элементов, реализующих технологию продукционной экспертной системы и примеры продукционных правил приведены в параграфе 2.2 «Разработка продукционных правил для реализации технологии продукционной экспертной системы и логический вывод», использование правил продемонстрировано в параграфе 4.3. «Пример решения прикладной задачи определения ВПХ с помощью разработанного программного приложения»). Правила описывают влияющие на

взлетно-посадочные характеристики ограничения, установленные производителем ВС и эксплуатантом/авиационными властями. Помимо вышеуказанных ограничений, влияние на расчет взлетно-посадочных характеристик могут оказать отложенные дефекты и отклонения конфигурации по MEL/CDL. Указанные документы представляют собой категоризированные перечни систем, приборов и оборудования ВС, и состоящие из разделов и пунктов, для которых могут быть созданы правила, в соответствии с которыми информационной системой будет выполняться необходимая корректировка параметров. Полный перечень правил базы знаний определяется эксплуатантом ВС и может, при необходимости, изменяться. Формирование базы знаний осуществляется экспертами предметной области. Сформированная база знаний публикуется на сервере. Передача базы знаний с сервера на электронный планшет EFB осуществляется по беспроводному интернет-каналу.

Рабочая память обеспечивает хранение исходных и промежуточных фактов. Данный блок задействуется на этапе осуществления логического вывода. На первом шаге в рабочую память размещаются факты – исходные данные (фактическая масса ВС, данные по аэропорту и ВПП, текущие эксплуатационные условия, открытые пункты MEL). В процессе логического состояние рабочей памяти изменяется путем добавления в нее новых фактов и/или удаления имеющихся.

Механизм логического вывода – компонент, обеспечивающий сопоставление правил базы знаний с фактами в рабочей памяти и разрешение конфликтов (определение порядка выполнения активированных правил).

Интерфейс пользователя ориентирован на оперирование программным обеспечением посредством сенсорного экрана устройства и позволяет осуществлять вывод полученных результатов. Для ввода данных, навигации в приложении и использования различных функций приложения (выбор режимов расчета, выполнение расчета, вывод на печать) используются интерактивные графические объекты (функциональные кнопки, экранная клавиатура). Подробнее

интерфейс разработанного программного обеспечения описан в параграфе 4.2. «Интерфейс и порядок работы с программным приложением EFB».

Реализованный автором работы сервер приложения EFB обеспечивает хранение актуальной базы знаний и базы данных, которые по запросу загружаются на устройство EFB пилота. Автором также разработано эффективное представление базы знаний и базы данных на сервере в формате JSON. Загруженные с сервера база данных и база знаний хранятся на устройстве в режиме офлайн.

Пример 2.1. Ниже представлен фрагмент JSON-файла, предназначенного для хранения данных аэродромов и ВПП:


```
[
  {
     "id":1,
     "adrwy":"UUWW01",
     "ad":"UUWW",
     "name":"Vnukovo",
     "iata":"VKO",
     "rwy":"01",
     "hdg":13,
     "elev":192.90,
     "runway_slope":0.01,
     "tora_available_run_length":3060.00,
     "toda_available_takeoff_distance":3210.00,
     "asda_accelerate_stop_distance":3060.00,
     "lda_available_landing_distance":3060.00
  },
  {
     "id":2,
     "adrwy":"UUWW01-2",
     "ad":"UUWW",
     "name":"Vnukovo",
     "iata":"VKO",
     "rwy":"01-2",
```

```
    "hdg":13,
    "elev":192.90,
    "runway_slope":0.01,
    "tora_available_run_length":2452.00,
    "toda_available_takeoff_distance":2602.00,
    "asda_accelerate_stop_distance":2452.00,
    "lda_available_landing_distance":3060.00
}]
```


2.2. Разработка продукционных правил для реализации технологии продукционной экспертной системы и логический вывод

Изучение искусственного интеллекта (ИИ), состоит из нескольких направлений, одним из которых является исследования и разработка экспертных систем [32-34].

Экспертные системы представляют собой программное средство, использующее экспертные знания для обеспечения высокоэффективного решения задач в некоторой предметной области.

Принцип работы экспертной системы заключается в передаче в нее пользователем некоторых исходных фактов и выводе экспертной системой в качестве результата новых фактов, полученных на основе экспертных знаний. Основными компонентами экспертных систем являются база знаний и машина логического вывода [35]. База знаний содержит в себе экспертные знания, на основании которых машина логического вывода формирует заключения.

Формирование заключений машиной логического вывода осуществляется путем сопоставления исходных данных рабочей памяти (фактов) и знаний из базы знаний.

Рабочая память служит для обеспечения хранения исходных и промежуточных фактов.

В настоящее время широкое распространение получили экспертные системы продукционного типа, то есть основанные на правилах вида ЕСЛИ-ТО,

указывающих, какие заключения должны быть сделаны в той или иной ситуации [36, 37]. Их близость к логическим импликациям позволяет легко реализовать логический вывод, а сама продукционная эвристика близка образу мышления эксперта.

Условие (левая часть правила) называется антецедентом, правая часть, содержащая множество действий при выполнении правила, называется консеквентом.

Продукционные экспертные системы подразделяются на системы прямого логического вывода и системы обратного логического вывода. Первые начинают свою работу с начального множества фактов и осуществляют логический вывод с использованием правил для вывода новых фактов или выполнения определенных действий. Экспертные системы с обратным логическим выводом начинают работу с некоторой гипотезы, которую необходимо доказать пользователю, и осуществляют поиск правил, позволяющих доказать ее истинность.

Множество фактов, хранящихся в рабочей памяти экспертной системы, определяют текущее состояние рабочей памяти.

Правило, условия которого в антецеденте выполнены, называется активированным, и размещается в рабочий список экспертной системы. В рабочем списке может быть размещено одновременно несколько активированных правил.

В случае размещения в рабочем списке одновременно нескольких правил, имеющих одинаковое значение приоритета, машина логического вывода должна принимать решение о том, какое из правил необходимо выполнить в первую очередь. В настоящей работе для решения конфликтов используется стратегия «вглубь», при которой первыми выполняются правила, имеющие наивысший приоритет, при этом вновь активируемые правила выполняются ранее правил с таким же приоритетом.

На данный момент разработано достаточно большое количество экспертных систем, решающих широкий круг задач в разных предметных областях, в том числе, в аэрокосмической отрасли.

В частности, в КБ ОАО «Туполев» были разработаны [38, 39]:

- экспертная система в помощь авиаспециалистам при проведении ими технического обслуживания систем и оборудования самолета Ту-204 с получением интеллектуальной информационной поддержки и выдачей рекомендаций по их устранению;

- экспертная система анализа неисправности на самолете Ту-204 с получением интеллектуальной информационной поддержки при определении анализа причин возникновения неисправностей в оборудовании самолета и выдачей рекомендаций по их устранению за счет использования аккумулированных знаний экспертов предприятия и нормативно-справочной документации.

Для описания базы знаний вышеуказанных экспертных систем предлагалась продукционная модель представления знаний, обладающей возможностью дальнейшего уточнения и расширения для решения определенных задач, а для описания предметной области и обеспечения логического вывода использовалась оболочка экспертных систем CLIPS.

Для применения таких импликаций в настоящей работе предложено использование технологии продукционной экспертной системы, включающей в себя базу знаний для хранения правил определения ВПХ, машину логического вывода и рабочую память для хранения фактов.

На рисунках 2.2 и 2.3 представлены примеры продукционных правил разработанной информационной системы и процесс получения новых фактов в процессе логического вывода [40].

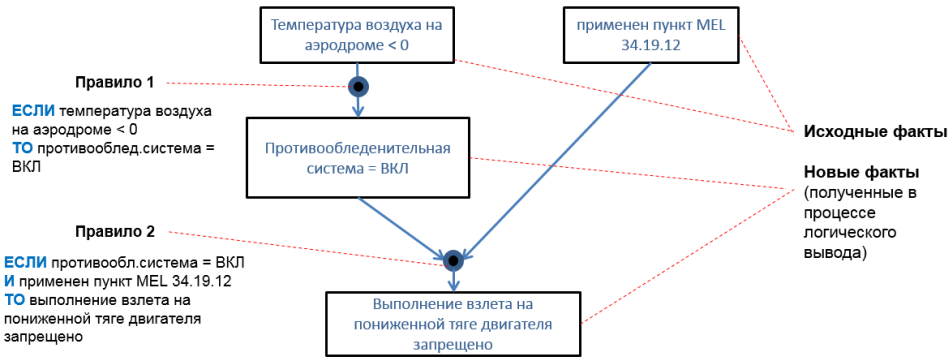


Рисунок 2.2 – Пример продукционных правил ЭС и их работы

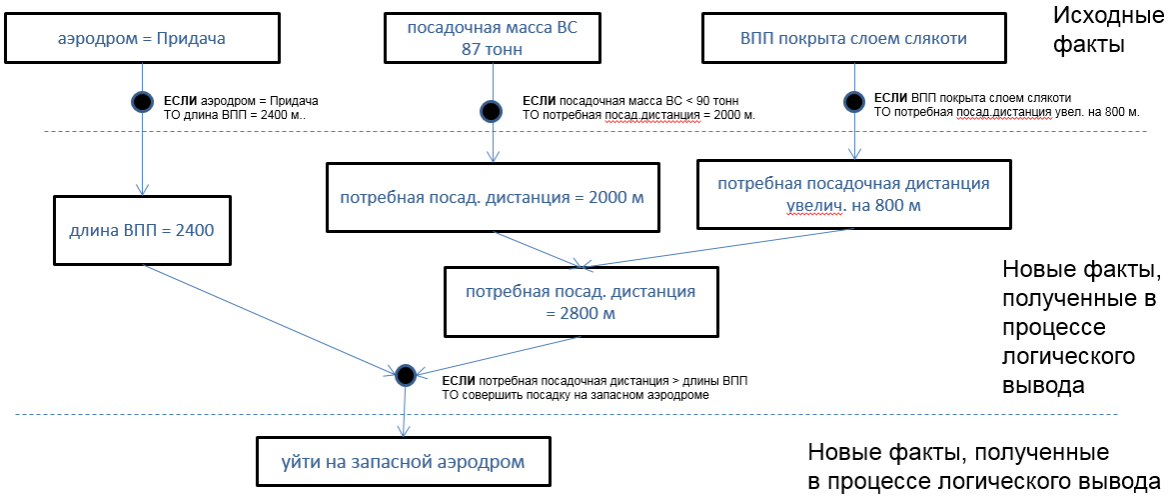


Рисунок 2.3 – Пример продукционных правил ЭС и их работы

В процессе логического вывода входными данными являются начальный набор фактов и правила базы знаний ЭС, выходными данными являются множество новых фактов и состояние рабочей памяти, перешедшей в новое состояние.

Для описания алгоритма логического вывода введем следующие обозначения:

WM_0 - начальное известное множество фактов;

$f = (v, a)$ – факт (например, атрибут x с номером v принял значение a $(x_v = a)$;

v – номер атрибута;

a – значение атрибута;

l – номер правила;

p_l – приоритет правила l;

i – номер итерации выполнения алгоритма (шаг алгоритма);

C – конфликтное множество правил;

CS_i - конфликтное множество правил на шаге i;

l^* – номер активированного правила, имеющего максимальный приоритет;

C – конфликтное множество правил;

CS_i - конфликтное множество правил на шаге i.

Правило задает отображение множества заданных фактов X_l на множество добавляемых A_l и удаляемых D_l фактов: r_l: $X_l \Rightarrow (A_l, D_l)$, где X_l - множество фактов антецедентов, A_l - множество добавляемых фактов, D_l - множество удаляемых фактов.

WM_i - рабочая память.

Рабочая память содержит начальные факты и факты, полученные путем применения правил на i-м шаге алгоритма.

Тогда алгоритм логического вывода примет следующий вид:

Алгоритм 2.1 – Прямой логический вывод продукционной экспертной системы

1. $i = 0$
2. $i = i + 1$
3. $CS_i = \{ l: (X_l \subseteq WM_{i-1}) \& \left(\forall k \left((0 < k < i) \rightarrow l \notin CS_k\right)\right)\}$
4. Если $CS_i = \emptyset$, то конец алгоритма.
5. $A = \emptyset$; $D = \emptyset$; $C = CS_i$
6. Пока $C \neq \emptyset$ выполнять цикл:

$$l^* = \arg \max_{l \in C} p_l$$

$$A = A \cup \left[A_{l^*} \setminus \left\{ f_j: \exists f_k \left((f_k \in A) \& \left(pr_1 f_j = pr_1 f_k\right)\right)\right\}\right]$$

$$D = D \cup \left[D_{l^*} \setminus \left\{ f_j: \exists f_k \left((f_k \in D) \& \left(pr_1 f_j = pr_1 f_k\right)\right)\right\}\right]$$

$$C = C \setminus \{l^*\}$$

7. $WM_i = WM_{i-1} \cup A \setminus D$
8. Переход к шагу 2.

Графическое описание данного алгоритма в виде UML-схемы представлено на рисунке 2.4.

Скорость работы продукционной экспертной системы, особенно при работе с большими количествами правил базы знаний и фактов в рабочей памяти, зависит от механизма логического вывода решения.

В работе для обеспечения ускорения вывода решения за счет минимизации сопоставлений используется алгоритм Rete [41]. При своем быстродействии, алгоритм Rete обладает высокими потребностями в оперативной памяти. Однако, поскольку современные аппаратные планшетные компьютеры, используемые в качестве платформ EFB, укомплектованы модулями оперативной памяти большого объема (2 Гб и более), недостаток алгоритма Rete становится крайне незначительным.

В данном алгоритме каждое правило рассматривается как один или несколько образцов, с каждым из которых связывается список всех активных сопоставляющихся с ним элементов рабочей памяти.

Модификация данного списка выполняется на каждой итерации перехода рабочей памяти в новое состояние.

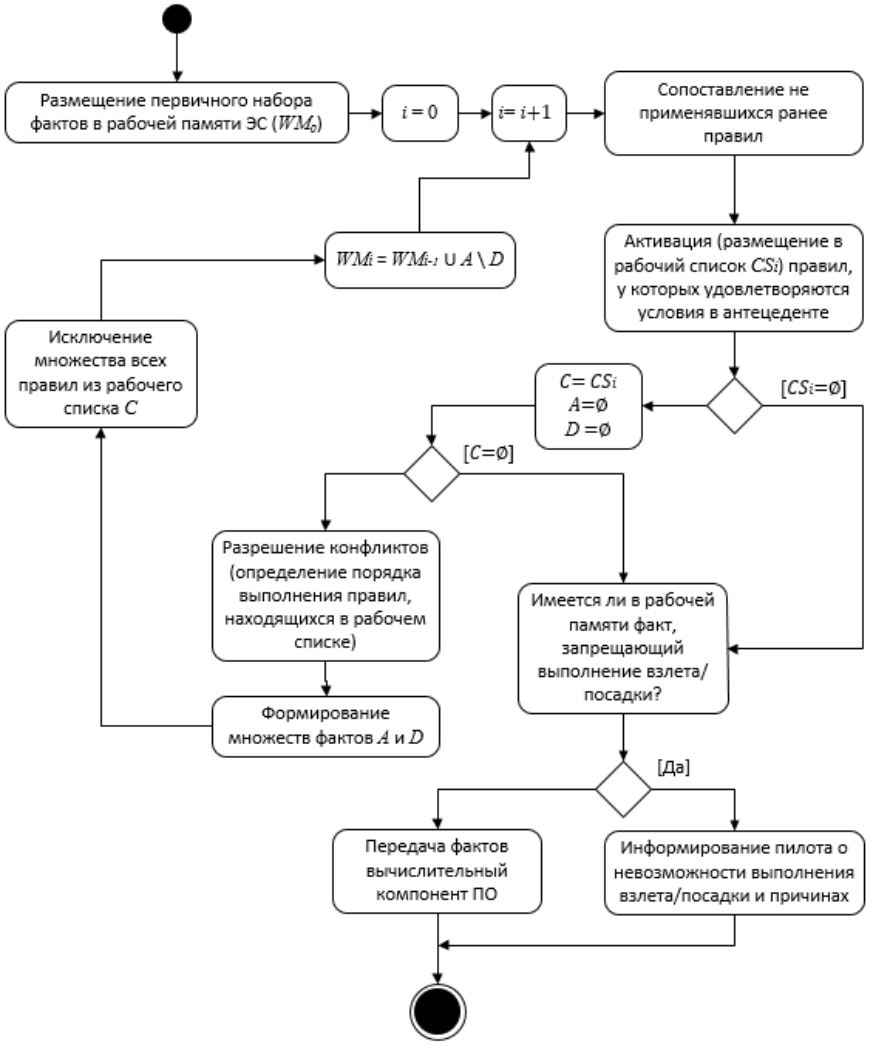


Рисунок 2.4 – Алгоритм логического вывода

При добавлении нового факта в рабочую память, машина логического вывода выявляет все правила, сопоставляющиеся с ним, и добавляет его к спискам соответствующих образов. В случае удаления факта из рабочей памяти, машина логического вывода удаляет его из всех списков образцов, которые с ним сопоставлялись. Таким образом, при использовании алгоритма Rete, в отличии от «наивной реализации» логического вывода, машина вывода не сопоставляет всю рабочую память со всеми правилами.

На вход алгоритма Rete подаются продукционные правила базы знаний. В качестве результата формируется специальный ациклический граф, узлам которого соответствуют части условий правил.

Базовая топология сети Rete представлена на рисунке 2.5. Данная диаграмма дает логическую интерпретацию сети Rete в общем виде и демонстрирует взаимосвязи между узлами сети.

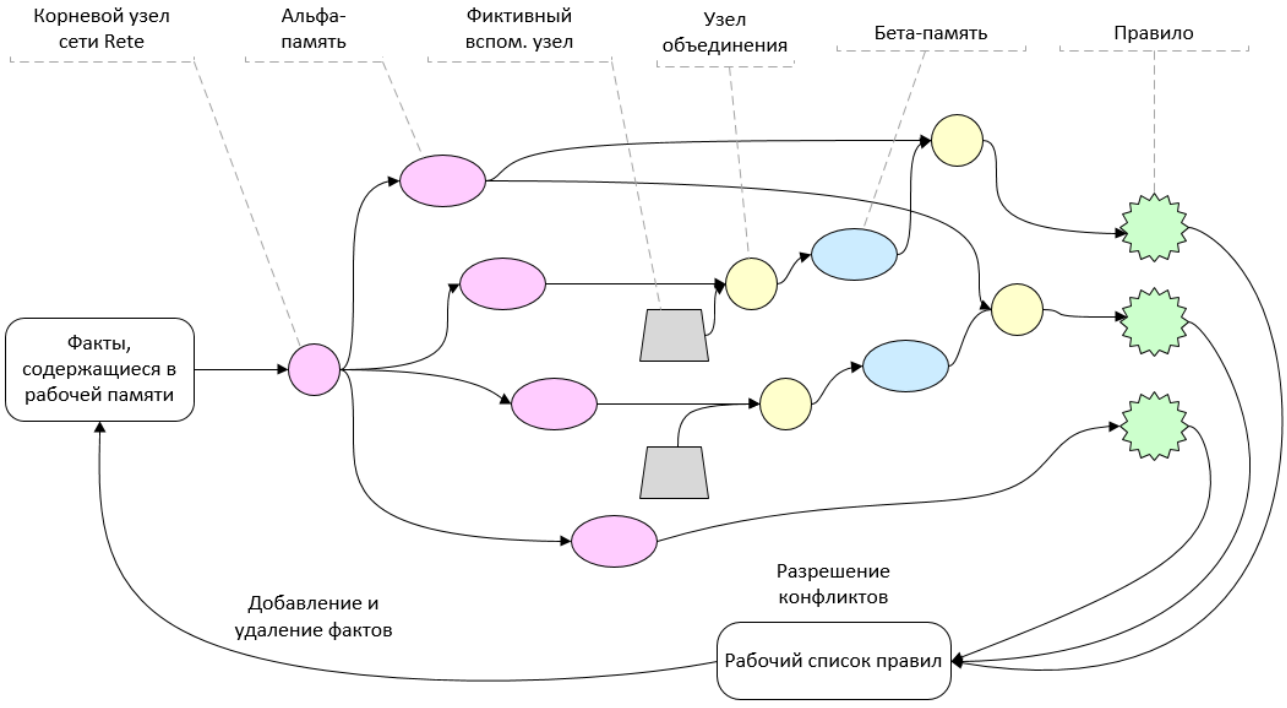


Рисунок 2.5 – Базовая топология сети Rete

Как видно из диаграммы, сеть Rete состоит из альфа- (представлена слева) и бета- (представлена справа) сетей. Альфа-сеть отвечает за отбор элементов рабочей

памяти. В случае успешного сопоставления элемента рабочей памяти с условиями правил, выполняется его передача на следующий узел сети. Бета-сеть осуществляет объединение элементов рабочей памяти и записывает результат в бета-память.

Таким образом, при добавлении нового факта, осуществляется его прогонка по сети, при этом отмечаются узлы, условиям которых данный факт соответствует. При выполнении полного условия правила, когда система достигает листа графа, правило активируется (добавляется в конфликтное множество правил).

Для реализации технологии продукционной экспертной системы и применения алгоритма Rete необходимо использовать подходящее инструментальное средство. На сегодняшний день существует множество инструментальных средств создания экспертных систем, наиболее распространенными среди которых являются CLIPS [42], PROLOG [44], LISP [44], Drools [45] и Jess [46].

CLIPS (C Language Integrated Production System) — программное инструментальное средство создания экспертных систем, разработанное в 1984 году в NASA. Синтаксис и название предложены Чарльзом Форги (Charles Forgy). CLIPS является продукционной системой. Реализация вывода использует алгоритм Rete.

Пролог (англ. Prolog) — программная система логического программирования, сосредоточенная вокруг небольшого набора основных механизмов, включая сопоставление с образцом, древовидного представления структур данных и автоматического перебора с возвратами.

LISP (LISt Processing language) — семейство языков программирования, программы и данные в которых представляются системами линейных списков символов. Лисп был создан для проведения работ в области искусственного интеллекта. Применяется и в качестве средства обычного промышленного программирования, от встроенных скриптов до веб-приложений массового использования.

Drools – это программная система с открытым кодом для разработки правил, созданная на языке Java и использующая при выполнении правил алгоритм Рете,

позволяющая описывать правила декларативным образом, используя простой для изучения и понимания язык, не связанный с XML [47].

Jess (Java Expert System Shell) – это оболочка для разработки экспертных систем, написанная полностью на языке Java компании Sun Microsystems. Сначала Jess являлся модификацией языка CLIPS, но позднее сформировался в отдельное инструментальное средство.

Для выбора инструментального средства был проведен сравнительный анализ, результаты которого приведены в таблице 2.1:

Таблица 2.1 – Выбор инструментального средства для реализации ЭС

Критерий	CLIPS	PROLOG	LISP	Drools	Jess
Интеграция с мобильными операционными системами, используемых в аппаратных платформах EFB	V	V	V	V	V
Свободно-распространяемое ПО с открытым исходным кодом	V	V	V/-	V	-
Функции процедурного программирования	V	V	V	V	V
Встроенная машина логического вывода	V	V	-	V	V
Поддержка прямого логического вывода	V	-	-	V	V
Поддержка алгоритма Rete	V	-	-	V	V

Из таблицы видно, что CLIPS и Drools обладают равными преимуществами.

Для реализации технологии продукционной экспертной системы в рамках диссертации был выбран CLIPS, поскольку, наряду с вышеуказанными преимуществами, изначально предназначался для разработки экспертных систем для аэрокосмической сферы, в которой безопасность имеет критически важное значение.

Его изначальное предназначение и применение в таких организациях, как NASA [48, 49], конструкторское бюро «Туполев» и других, позволяет сделать вывод о высокой надежности программной реализации системных библиотек данного инструментального средства.

2.3. Онтология предметной области EFB

Онтология – это формальное явное описание терминов предметной области и отношений между ними [50], смысловая модель предметной области [51].

В центре большинства онтологий находятся классы. Классы описывают понятия предметной области. Класс может иметь подклассы, которые представляют более конкретные понятия, чем надкласс.

Атрибуты описывают свойства классов и экземпляров.

Процесс разработки онтологии включает в себя [52, 53]:

- определение классов в онтологии;
- расположение классов в таксономическую иерархию;
- определение атрибутов и описание допускаемых значений этих атрибутов;
- заполнение значений атрибутов.

Формально модель онтологии может быть представлена в виде следующего кортежа:

$$S_{EFB} = <X, A, R>,$$

где $X = \{x_i \mid i = 1, \ldots, N\}$ – конечное множество терминов предметной области, представляющих собой понятия, объекты, классы, образующие онтологию S_{EFB} (N – количество понятий онтологии S_{EFB});

$A = \{a_j \mid j = 1, \ldots, M\}$ – конечное множество атрибутов понятий (M – количество атрибутов, описывающих понятия из X);

$R = \{r_k \mid k = 1, \ldots, L\}$ – конечное множество отношений между понятиями (L – количество отношений между понятиями x_i онтологии S_{EFB}).

Существует несколько возможных подходов для разработки иерархии классов [54]:

- Процесс нисходящей разработки начинается с определения самых общих понятий предметной области с последующей конкретизацией понятий.
- Процесс восходящей разработки начинается с определения самых конкретных классов, листьев иерархии, с последующей группировкой этих классов в более общие понятия.

– Процесс комбинированной разработки – это сочетание нисходящего и восходящего подходов: сначала мы определяем более заметные понятия, а затем соответствующим образом обобщаем и ограничиваем их. Онтология должна обеспечить стандартизацию терминологии и унифицированную информационную основу для всех участников, вовлеченных в процессы разработки и внедрения систем EFB.

Если разработчик склонен к рассмотрению предметной области сверху вниз, то ему, возможно, больше подойдет нисходящий метод. При этом, зачастую при разработке онтологий самым простым является комбинированный метод, поскольку понятия, расположенные «посередине», поскольку они более явны и наглядны в предметной области. Автор настоящей диссертации при разработке онтологии предметной области руководствовался комбинированным методом.

Однако, какой бы метод ни был избран, обычно мы начинаем с определения классов. Для этого, на первом шаге выбираются термины, которые описывают основные объекты предметной области.

Атрибутами в онтологии могут стать несколько типов свойств объектов [55, 56]:

- «внутренние» свойства;
- «внешние» свойства;
- части, если объект имеет структуру; они могут быть как физическими, так и абстрактными «частями»;
- отношения с другими индивидными концептами; это отношения между отдельными членами класса и другими элементами.

Информационная система работает с большим количеством разнородных фактов, для хранения которых требуется создание соответствующей базы данных.

В рамках работы для формализации представления структуры базы данных, а также для разработки общей онтологии предметной области систем EFB с описанием концептуальной модели программного приложения EFB для расчета взлетно-посадочных характеристик воздушных судов, была построена онтология предметной области EFB [57].

Онтология предметной области создавалась с применением программного обеспечения Protégé версии 5.5.0 (рисунок 2.6), поскольку данный редактор сопровождается обширной и детальной документацией, поддерживается значительным сообществом, состоящим из разработчиков и учёных, правительственных и корпоративных пользователей, использующих его для решения задач, связанных со знаниями в таких разнообразных областях, как биомедицина, сбор знаний и корпоративное моделирование, а также доступен для свободного скачивания с официального сайта вместе с плагинами и онтологиями [58].

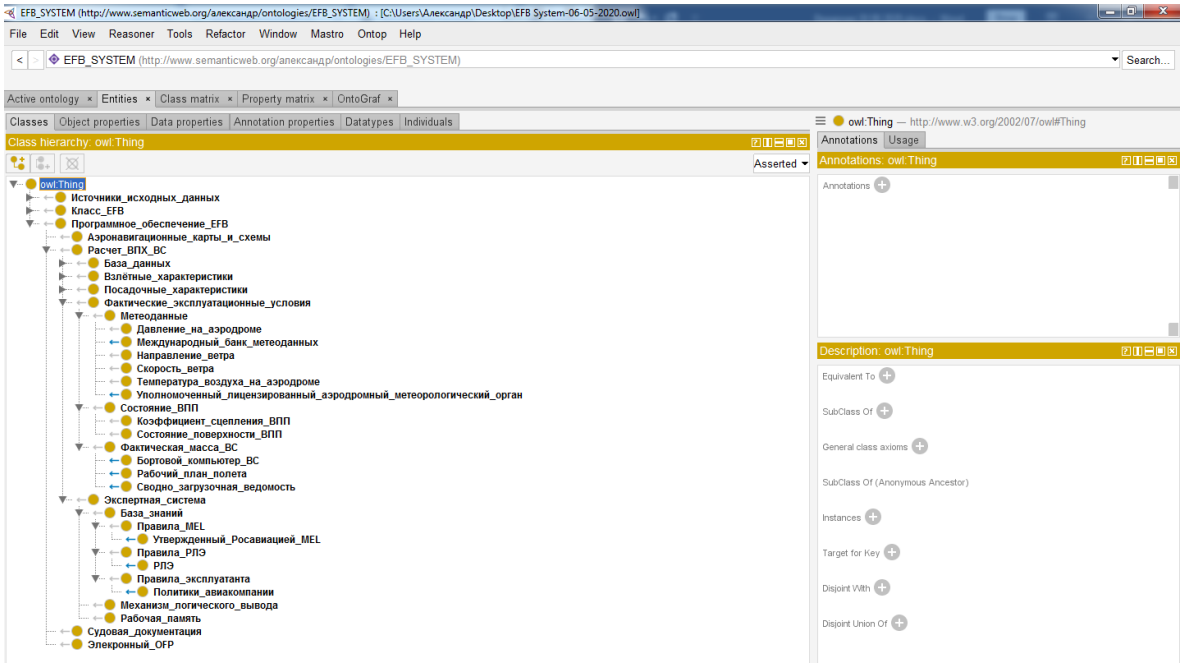


Рисунок 2.6 – Фрагмент онтологии предметной области EFB, реализованный в Protégé 5.5.0

В качестве базовых классов онтологии предметной области EFB введем следующие понятия:

– класс EFB, определяющий аппаратную реализацию EFB;
– программное обеспечение EFB, определяющее функции, реализуемые системой EFB;

– источники исходных данных – источники данных и информации, используемой для реализации функций EFB.

Согласно документу ICAO Doc 10020, оборудование EFB делится на два класса: портативное и установленное (т.е. являющееся частью конфигурации ВС). При этом, аппаратным обеспечением EFB портативного класса является электронное устройство, представляющее собой планшетный компьютер.

Информация и исходные данные, используемые в EFB, должны поступать только из официальных источников. Для понятия «Источники исходных данных» введены классы, определяющие организации, работников и бортовые системы, являющиеся официальными поставщиками и источниками информации и данных, используемых летными экипажами в EFB при подготовке к полету и при его выполнении:

– «Уполномоченный лицензированный аэродромный метеорологический орган»;
– «Международный банк метеоданных»;
– «Филиал ЦАИ ФГУП Госкорпорация по ОрВД», включающий подкласс «База аэронавигационных данных АРНАД»;
– «Лицензированный полетный диспетчер», включающий дочерний объект «Рабочий план полета»;
– «Наземная служба аэродрома», включающий дочерний объект «Сводно-загрузочная ведомость»;
– «Бортовой компьютер ВС»;
– «Производитель ВС», включающий дочерний объект «Руководство по летной эксплуатации воздушного судна»;
– «Авиакомпания», включающий дочерние объекты «Утвержденный Росавиацией MEL» и «Политики авиакомпании».

В качестве классов понятия «Программное обеспечение EFB» выступают следующие функции EFB, реализуемые конкретным программным приложением:

– судовая документация;

- электронный рабочий план полета (OFP);
- аэронавигационные карты и схемы;
- расчет взлетно-посадочных характеристик воздушного судна.

Для описания концептуальной модели программного приложения EFB, реализующего функции расчета взлетно-посадочных характеристик воздушных судов, были введены следующие сущности:

- база данных;
- взлетные характеристики;
- посадочные характеристики;
- фактические эксплуатационные условия;
- экспертная система.

Введем следующие классы и атрибуты сущности «база данных»:

1. Атрибуты класса «MEL»:
    a. Номер пункта MEL.
    b. Описание пункта MEL.
2. Класс «аэронавигационная информация» включает в себя:
    a. подкласс «Аэродром» с атрибутами «ICAO-код аэродрома», «IATA-код аэродрома», «Название аэродрома», «Название города», «Превышение аэродрома»;
    b. подкласс «Взлетно-посадочная полоса», включающий в себя, в свою очередь, атрибуты конкретной ВПП – заявленные дистанции («TORA» – располагаемая дистанция разбега, «TODA» – располагаемая дистанция взлета, «ASDA» – располагаемая дистанция прерванного взлета, «LDA» – располагаемая посадочная дистанция), курс и уклон и идентификатор ВПП.
    c. подкласс «Координаты препятствий», включающий в себя атрибуты удаления препятствия от торца взлетно-посадочной полосы («Distance»), значения боковых удалений («Offset»).

d. подкласс «Препятствия», включающий в себя атрибуты «Высота препятствия», «Идентификатор препятствия», «Название препятствия» и «Тип препятствия».

3. Атрибуты класса «Воздушное судно»:
    a. Бортовой номер ВС.
    b. Тип ВС.
    c. Положения закрылков.
    d. Режимы работы двигателя.
    e. Режимы торможения.
4. Атрибуты класса «Пользователь»:
    a. Имя.
    b. Фамилия.
    c. Логин.
    d. Пароль.
    e. Роль пользователя.

Введем атрибуты для описания сущности «Взлетные характеристики»:

- скорость принятия решения;
- скорость подъема передней опоры шасси;
- безопасная скорость взлета;
- максимально допустимая взлетная масса.

Введем атрибуты для описания сущности «Посадочные характеристики»:

- скорость захода на посадку;
- максимально допустимая посадочная масса.

Для описания сущности «Фактические эксплуатационные условия» введен атрибут «Фактическая масса ВС» и классы «Авиационные метеорологические данные» и «Состояние ВПП».

Класс «Авиационные метеорологические данные» включает в себя следующие атрибуты:

- давление на аэродроме;

– направление ветра;
– скорость ветра;
– температура воздуха на аэродроме.

Класс «Состояние ВПП» включает в себя следующие атрибуты:

– коэффициент сцепления ВПП.
– состояние поверхности ВПП.

Для описания сущности «Экспертная система» введем классы «База знаний», «Механизм логического вывода» и «Рабочая память».

Класс «База знаний» включает в себя подклассы:

– «Правила MEL»;
– «Правила руководства по летной эксплуатации ВС» и
– «Правила эксплуатанта».

Всего разработанная онтология предметной области EFB включает в себя 44 класса и 34 атрибута.

На рисунке 2.7 представлены фрагмент онтологического представления и явное указание соответствия одного из классов и атрибутов онтологии таблице базы данных и ее атрибутам соответственно.

Рисунок 2.7 – Фрагмент онтологии предметной области EFB

На схеме (рисунок 2.8) показан фрагмент унифицированной модели базы данных ЭС, структура которой сформирована на базе разработанной онтологии.

Данная схема составлена согласно стандарту IDEF1X, который учитывает такие требования, как простота изучения и возможность автоматизации.

Как видно из представленной на рисунке 2.4 схемы, классы онтологии послужили основой для следующих таблиц базы данных:

- класс онтологии «Аэродром» > таблица БД «Airport»;
- класс онтологии «Взлетно-посадочная полоса» > таблица БД «Runway»;
- класс онтологии «координаты препятствий» > таблица БД «Obstacle_runway_coordinates»;
- класс онтологии «препятствия» > таблица БД «Obstacle»;
- класс онтологии «воздушное судно» > таблица БД «Aircraft»;
- класс онтологии «MEL» > таблица БД «MEL_Item»;
- класс онтологии «пользователь» > таблица БД «App_User».

База данных ЭС позволяет хранить разнородные факты, а также обеспечивает разграничение доступа для пользователей системы на уровне ролей, которые им назначены в зависимости от требуемого типа ВС (доступ к определенным ВС) и географии полетов (доступ к аэронавигационным данным определенных аэродромов).

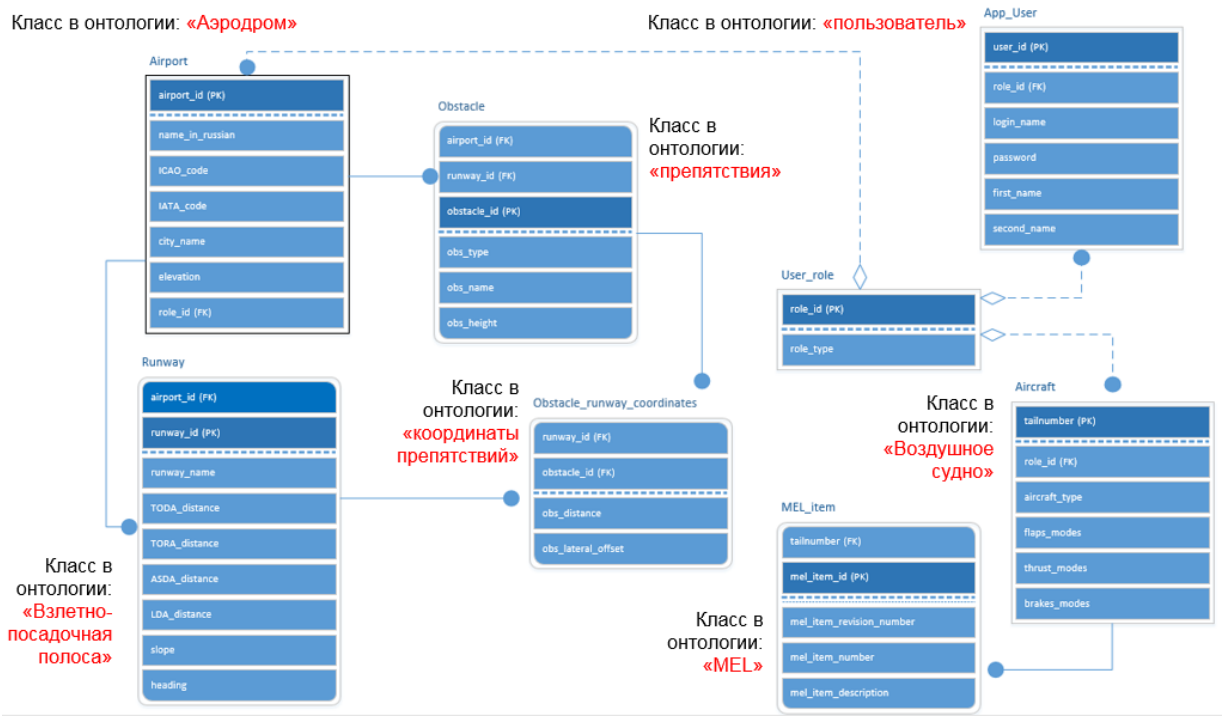


Рисунок 2.8 – Фрагмент унифицированной модели БД

Таблица «Airport» содержит следующие атрибуты:

- airport_id – уникальный идентификатор аэропорта, задается при создании базы данных;
- name_in_russian – предназначен для хранения названия аэродрома на русском языке, ICAO_code – предназначен для хранения значения четырехбуквенного ИКАО-кода аэродрома; IATA_code – предназначен для хранения значения трехбуквенного ИАТА-кода аэродрома; city_name – для хранения названия города, в котором находится аэродром; elevation – для хранения значения превышения аэродрома. Источником данных для указанных атрибутов может быть использована база аэронавигационных данных «АРНАД» [59];
- role_id – роль пользователя, позволяет определить каким категориям пользователей доступен данный аэродром.

Таблица «Runway» включает в себя атрибуты:

- airport_id – уникальный идентификатор аэропорта;
- runway_id – уникальный идентификатор ВПП;
- runway_name – наименование ВПП; TODA_distance – значение располагаемой дистанции взлета ВПП; TORA_distance – значение располагаемой дистанции разбега ВПП; ASDA_distance – значение располагаемой дистанции прерванного взлета дистанции ВПП; LDA – значение располагаемой посадочной дистанции distance ВПП; slope – значение уклона ВПП; heading – курс ВПП. Источником данных для указанных атрибутов может быть использована база аэронавигационных данных «АРНАД».

Таблица «Obstacle» включает в себя атрибуты:

- airport_id – уникальный идентификатор аэропорта;
- runway_id – уникальный идентификатор ВПП;
- obstacle_id – уникальный идентификатор аэродромного препятствия;
- obs_type – тип аэродромного препятствия; obs_name – наименование аэродромного препятствия; obs_height – значение высоты аэродромного

препятствия. Источником данных для указанных атрибутов может быть использована база аэронавигационных данных «АРНАД».

Таблица «Obstacle_runway_coordinates» содержит следующие атрибуты:

- runway_id – уникальный идентификатор ВПП;
- obstacle_id – уникальный идентификатор аэродромного препятствия;
- obs_distance – значение удаления препятствия от торца взлетно-посадочной полосы в метрах; obs_lateral_offset – значение бокового удаления аэродромного препятствия в метрах. Источником данных для указанных атрибутов также может быть использована база аэронавигационных данных «АРНАД».

Таблица «Aircraft» содержит следующие атрибуты:

- tailnumber – бортовой номер ВС;
- role_id – идентификатор роли пользователя, для определения того, каким пользователям какие ВС доступны в приложении;
- aircraft_type – тип ВС;
- flaps_modes – значения возможных положений механизации крыла на взлете/посадке, предусмотренные РЛЭ конкретного типа ВС;
- thrust_modes – возможные значения тяги двигателя на взлете, предусмотренные РЛЭ ВС;
- brakes_modes – возможные значения режимов торможения, предусмотренные РЛЭ ВС.

Таблица «MEL_Item» включает в себя атрибуты:

- tailnumber – бортовой номер ВС;
- mel_item_id – уникальный идентификатор MEL;
- mel_item_revision_number – номер ревизии MEL;
- mel_item_number – номер пункта MEL;
- mel_item_description – описание пункта MEL.

Таблица «App_User» включает в себя атрибуты:

- user_id – уникальный идентификатор пользователя приложения;

– role_id – уникальный идентификатор роли пользователя;
– login_name – имя пользователя, задается при создании базы данных;
– password – пароль, задается при создании базы данных;
– first_name – имя пользователя приложения;
– second_name – фамилия пользователя приложения.

Таблица «User_Role» включает в себя атрибуты:

– role_id – уникальный идентификатор роли пользователя приложения;
– role_type – тип роли пользователя (например, «Пилот ВС Ту-204», «Пилот ВС МС-21», «Администратор» и т.д.), определяющий доступ к аэронавигационной базе данных и типам ВС.

2.4. Оцифровка номограмм зависимостей ВПХ, представленных в РЛЭ воздушного судна

В соответствии с ICAO Doc 10020, в тех случаях, когда программное обеспечение и данные изготовителя отсутствуют, бумажные номограммы РЛЭ или FCOM могут быть оцифрованы третьими сторонами, разрабатывающими данные для их собственных продуктов.

Таким образом, вычислительный модуль может быть основан на оцифрованном материале, утвержденном в соответствии с требованиями летной годности руководства по летной эксплуатации. Примеры номограмм, используемых для определения ВПХ, представлены на рисунке 2.9:

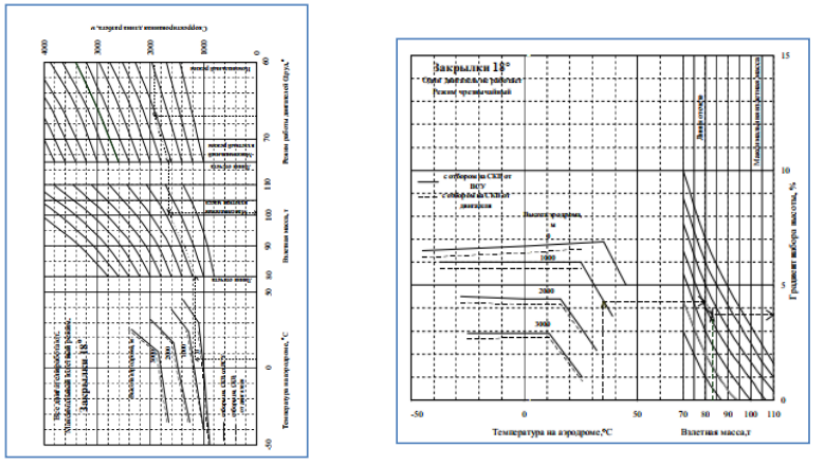


Рисунок 2.9 – Номограммы зависимостей ВПХ ВС Ту-204

Оцифровка номограмм выполнялась способами, рассмотренными в [1, 2, 4, 60]:

1) Представление зависимости выходного параметра от исходных параметров в виде регрессионной модели;
2) Разделение номограммы на отдельные участки, для каждой из которых строится своя отдельная зависимость (данный способ является модификацией вышеуказанного). Разбиение номограммы на участки позволит учесть особенности каждого из участка номограммы.

Оцифровка номограмм зависимостей ВПХ, представленных в РЛЭ воздушного судна Ту-204-100В выполнена с применением комплекса ПО Wolfram Mathematica [61] и GetData Graph Digitize [62].

Wolfram Mathematica является системой компьютерной алгебры, позволяющей решать широкий круг вычислительных задач. Обладает такими преимуществами, как возможность подключения к интерфейсам других программ (в частности, Microsoft Excel), имеет электронную поддержку пользователей. В настоящей работе данный программный комплекс используется для формирования функциональных зависимостей ВПХ на основе оцифрованных данных номограмм.

Программное обеспечение GetData Graph Digitize предназначено для оцифровки графических данных. Программа имеет простой и удобный интерфейс пользователя, поддерживает множество форматов изображений (TIFF, JPEG, BMP, PCX), позволяет менять порядок точек в оцифрованных кривых, осуществлять экспорт полученных данных в форматах TXT, Excel (XLS), XML, DFX, EPS. Ее использование рассматривалось во множестве работ, посвященных построению математических моделей на основе графических данных для автоматизации инженерно-штурманских расчетов [1, 4].

Оцифровка номограмм зависимостей ВПХ состояла из нескольких этапов.

2.4.1. Цифровая копия изображения номограммы

Изображение с номограммой сохранялось с минимальными потерями качества, с разрешением в пределах 300-600 dpi. В случае необходимости, для

достижения четкого и однородного изображения кривых соответствующей номограммы, выполнялось сканирование в градациях серого.

С помощью редакторов изображений, при необходимости, выполнялось устранение деформаций (устранение перекоса, нелинейных деформаций на растре и т.п.), повышаются резкость изображения.

2.4.2. Координаты точек кривых номограммы

Получение координат точек кривых комплекса номограмм выполнялось с применением ПО GetData Graph Digitizer [62]. На первом этапе изображение открывалось в вышеуказанной программе в формате jpeg и устанавливалась система координат. Для установки системы координат использовалась «Команды» > «Установить систему координат». Далее, при нажатии левой кнопку мыши, выбиралась точка начала координат и в появившемся окне вводилось значение начала координат (Xmin). Далее, по аналогии с Xmin, выполнялась установка значений Xmax, Ymin и Ymax. Для удобной установки следует воспользоваться функцией масштабирования («Вид» > «Лупа»). После этого отобразятся координатные оси и появится окно «Параметры Системы координат». В нем, при необходимости, допускается переназначение опорных точек и установка логарифмического масштаба оси.

Также для визуального контроля качества установки системы координат можно воспользоваться функцией отображения сетки с установленным шагом «Вид» > «Показывать сетку». Если система координат была установлена корректно, то линии сетки отобразятся параллельно осям графика на изображении.

На втором этапе выполнялась установка точки на кривых номограммы (как представлено на рисунке 2.10).

Для этого использовался режим установки точек («Команды» > «Режим установки точек»). После активации данного режима, устанавливались точки на кривых нажатием левой кнопки мыши.

Чтобы отобразить перечень координат выбранных точек, следует перейти в меню «Вид» > «Окно информации». Чтобы удалить точку, использовать специальный ластик в меню «Команды» > «Ластик точек данных».

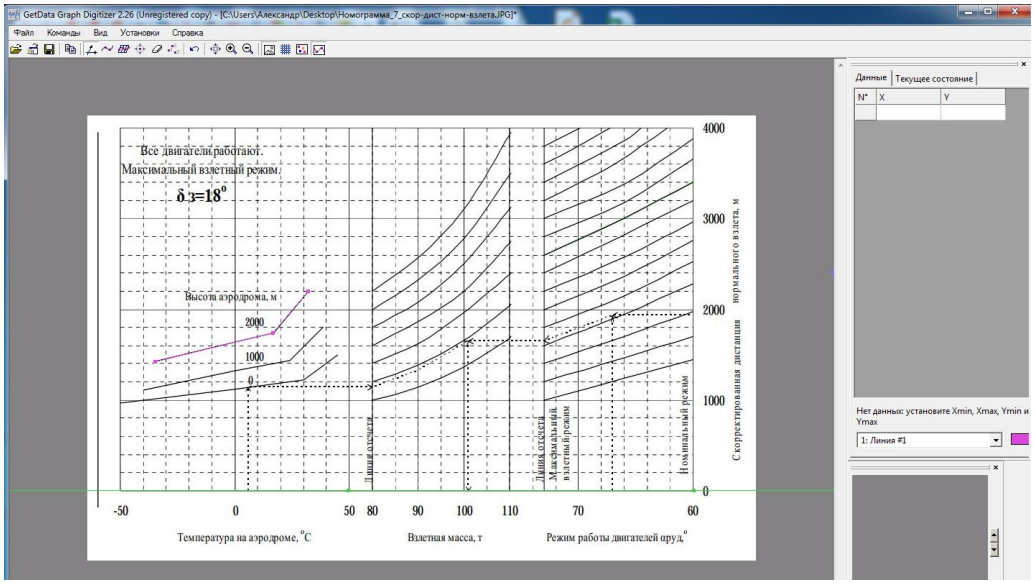


Рисунок 2.10 – Оцифровка линий номограммы определения взлетной массы ВС Ту-204, ограниченной скорректированной располагаемой дистанцией разбега при нормальном взлете с закрылками 18°

Наибольшее количество точек необходимо устанавливать в местах изломов кривых, на линейных участках кривой допускается использование небольшого их количества.

Чтобы оцифровать следующую кривую номограммы, необходимо добавить новую линию с помощью функции «Команды» > «Добавить линию». После чего можно будет выставить точки на второй кривой аналогично рассмотренному выше способу.

На третьем шаге оцифровки полученные данные по каждой линии экспортировались с помощью команды «Файл» > «Экспорт данных». После вызова данной команды, в открывшемся окне выбирался путь сохранения и указывалось имя сохраняемого файла. Экспорт данных выполнялся в txt-файле.

2.4.3. Математическая модель зависимостей ВПХ

Для создания математической модели зависимостей взлетно-посадочных характеристик на основании подготовленных оцифрованных данных в настоящей работе использовалось программное обеспечение Wolfram Mathematica.

Сначала выполняется копирование координат точек из полученного текстового файла в пустую ячейку Excel.

Следующим шагом создавался новый документ Wolfram Mathematica, в который «перетягивался» файл Excel. Таким образом формируется список списков с координатами точек, которому присваивается переменная data. После это импортированные данные выводились при помощи функции ListPlot[].

Для аппроксимации точек полиномом 5й степени, использовалась функция LinearModelFit[], приводящая в итоге к получению объекта класса FittedModel[]. Ему присваивалась переменная fit.

На следующем шаге вычислялся коэффициент детерминации R^2 – чем он ближе к значению 1, тем большую долю вариации полученное уравнение объясняет.

Для проведения регрессионного анализа, указывалась в качестве аргумента функция fit «ANOVATable».

Для отображения полученного уравнения в явном виде, к переменной fit применялась функция Normal[].

Рассмотренным в данном разделе способом были построены математические модели зависимостей ВПХ по оцифрованным номограммам (аппроксимации оцифрованных номограмм с включением элементов логики), представленных в Руководстве по летной эксплуатации ВС [63-66].

Пример 2.2. Взлетная масса ВС Ту-204, ограниченная нормируемым градиентом набора высоты, представляет собой следующую функцию:

$$m_{\text{взл}} = f(T,\ H_{\text{аэр}}),$$

где T – температура воздуха на аэродроме, $H_{\text{аэр}}$ – высота аэродрома.

В зависимости от значений, которые принимают параметры T и $H_{\text{аэр}}$, взлетная масса $m_{\text{взл}}$, ограниченная набором высоты, может быть определена 12 разными способами. Например,

- при $15 \le T \le 20$ и $1800 \le H_{\text{аэр}} \le 2000$:

$$m_{\text{взл}} = (4 - 0{,}2T)\left((118061 - 8{,}485H_{\text{аэр}}) - (119430 - 10{,}596H_{\text{аэр}})\right) + (119430 - 10{,}596H_{\text{аэр}})$$

- при $T \le 10$ и $2000 \le H_{\text{аэр}}$:

$$m_{\text{взл}} = 120750 - 9{,}875H_{\text{аэр}}$$

- при $T \le 15$ и $1539 \le H_{\text{аэр}} \le 2000$:

$$m_{\text{взл}} = 118061 - 8.48485H_{\text{аэр}}$$

Скорости V_R и V_2 рассматриваемого типа ВС на взлете с закрылками 18^{o} представляют собой следующие зависимости:

$$V_R = 1{,}2001\frac{m_{\text{взл.факт}}}{1000} + 118{,}9931$$

$$V_2 = V_R + 20$$

Выводы по главе 2

В резюме по второй главе сделаны следующие выводы:

- Разработанная архитектура ЭС позволяет за счет модульной реализации обеспечить ее адаптивность и расширяемость, позволяя учитывать специфику различных типов воздушных судов без необходимости внесения изменений в код программного приложения.
- Выполнена реализация правил, представленных в эксплуатационной документации ВС и используемых для определения взлетно-посадочных характеристик, в виде продукций. Для работы с ними реализовано использование технологии продукционной экспертной системы, что позволит обеспечить большую по сравнению с существующими системами гибкость в процессе определения ВПХ.
- Для реализации технологии продукционной экспертной системы было выбрано инструментальное средство CLIPS. Оно обладает такими преимуществами, как возможность интеграции с мобильными операционными

системами, используемыми в аппаратных платформах EFB, открытый исходный код и свободное распространение, возможность применения функционала процедурного программирования, встроенной машины логического вывода, поддержка прямого логического вывода и алгоритма Rete, а также изначальное предназначение и применение в организациях аэрокосмической отрасли, где безопасность имеет критически важное значение, позволяют сделать вывод о высокой надежности программной реализации системных библиотек данного инструментального средства, и обусловили его выбор для создания ЭС в рамках данной диссертации.

- Разработанное онтологическое представление предметной области EFB позволило сформировать структуру базы данных, обеспечивающей хранение большого количества разнородных фактов, используемых в работе информационной системы.
- Оцифровка номограмм, представленных в РЛЭ воздушного судна, с помощью комплекса специализированного ПО Wolfram Mathematica и GetData Graph Digitizer, позволила построить математическую модель зависимостей ВПХ, используемую для автоматизации расчета ВПХ.

ГЛАВА 3. Выбор аппаратной платформы для информационной системы

3.1. Алгоритм ранжирования альтернатив на основе нечетких предпочтений ЛПР, заданных в нечетких областях при выборе аппаратной платформы EFB

При работе экспертной системы решаются задачи выбора наилучших альтернатив на основе многокритериальной информации, причем шкалы критериев позволяют учесть нечеткую вербальную информацию о предпочтениях лица, принимающего решения.

При выборе модели планшета ЛПР необходимо учесть множество факторов, определенных спецификой парка воздушных судов авиакомпании, ее политики в части использования системы EFB.

Это приводит к тому, что перед эксплуатантом на этапе разработки плана внедрения EFB возникает непростая многокритериальная задача выбора подходящей для него модели планшетного компьютера, разнообразие которых непрерывно растет с развитием рынка и нормативной базы, регламентирующей их одобрение и применение в авиации [67-69].

Согласно классификации ICAO, системы EFB в зависимости от степени интеграции в кабине ВС разделяют на встроенные и портативные.

В качестве портативных EFB используются планшетные компьютеры, и они получили широкое распространение в авиации в качестве устройств EFB, поскольку обладают невысокой стоимостью, для них активно разрабатывается и поддерживается специализированное программное обеспечение, реализующее функции EFB, их использование одобрено авиационными властями.

Кроме необходимости учета множества практических и нормативных факторов, нерациональный выбор модели планшета EFB может привести к существенным перерасходам (для крупной авиакомпании требуется осуществление закупки планшетов на десятки миллионов рублей).

Конкретные модели планшетов обладают определенными характеристиками с конкретными значениями, т.е. исходные критерии – четкие. Однако, поскольку суждения ЛПР имеют нечеткий характер, для выбора аппаратной платформы информационной системы используется теория нечетких множеств.

Разработка математических методов для решения задач многокритериального выбора в интеллектуальных системах имеют огромное значение в аэрокосмической сфере как на этапах проектирования и разработки, так и дальнейшей эксплуатации авиационной и ракетно-космической техники [70-73].

В настоящей работе для выбора аппаратной платформы системы EFB предлагается новый метод определения нечетких суждений ЛПР, объединяющий подходы, применяемые при нечетком автоматическом управлении [74-76] на основе экспертных суждений и идею разбиения пространства критериев на нечеткие области, предлагающуюся в комплексе с методом поддержки принятия решений.

Предлагается разбить области определения всех критериев на нечеткие интервалы. При этом, допускается пересечение таких интервалов с учетом ограничений на максимальную и минимальную степень принадлежности любого значения критерия. Далее для некоторых комбинаций нечетких значений критериев ЛПР высказывает свои суждения в нечеткой шкале предпочтений. Полученная модель системы ценностей ЛПР проверяется на предмет покрытия всех точек критериального пространства с уровнем предпочтений не ниже заданного. Далее полученная модель может использоваться для оценки произвольного количества альтернатив в автоматизированном режиме. На вход модели могут подаваться как четкие, так и нечеткие значения критериев, и далее метод позволяет строить функции принадлежности альтернатив всем областям предпочтений. На заключительном этапе предлагается проводить дефаззификацию нечетких рангов альтернатив.

При формировании критериев и предпочтений использовались следующие соображения:

- эргономика использования в кабине пилотов оценивалась экспертно. Как правило, пространство кабины пилотов ВС очень ограничено, и работа с громоздкими устройствами является проблематичной.
- обеспечение комфортного восприятия информации зависит от размера экрана. Если экран менее 9 дюймов, то возможны затруднения при считывании информации. Средний экран (9-11 дюймов) и большой экран (более 12 дюймов) обеспечивают комфортное восприятие информации.
- учитывалась потребность в 3G/LTE-интернете на устройстве и потребность в встроенном модуле GPS/ГЛОНАСС.
- объем памяти, зависящий от того, будет ли с помощью электронного планшета осуществляться фото- и видеосъемка и планируется ли хранение указанных материалов, имеется ли необходимость в двух- или трехкратном резервирование библиотеки документов.

В рамках задачи даны:

Критерии и нечеткие шкалы согласно таблице 3.1;

20 альтернатив (моделей электронных планшетов).

Для каждого из критериев задана нечеткая шкала – множество возможных значений критерия разбито на отдельные нечеткие градации:

$S_i = \left\{t_{i_1}, t_{i_2}, \ \ldots, t_{i_{qi}}\right\}$ - шкала i-го критерия, где i – номер критерия (i=1..n),

q_i – число градаций в шкале S_i, j – номер градации критерия (j=1.. q_i), t_{ij} – нечеткая градация шкалы критерия.

Таблица 3.1 – Исходные критерии и нечеткие шкалы

Критерий	Шкала
Память	"Недостаточная", "Достаточная"
Интернет 3G/LTE	"Да","Нет"
GPS/ГЛОНАСС	"Да","Нет"
Диагональ экрана	"Маленький экран", "Средний экран", "Большой экран"

Год выпуска	"Менее 3 лет", "Более 3 лет"
Эргономика в кабине	"Плохая эргономика", "Хорошая эргономика"
Текущая стоимость	"Низкая", "Средняя", "Высокая"

Исходные данные для ранжирования в формате JSON представлены в Приложении 5.

Нечеткие градации задаются с помощью функции принадлежности:

$\mu_{ij}(x)$ – функция принадлежности четких значений i-го критерия j-й градации, где x – это значение i-го критерия.

В заданных шкалах можно разбить критериальное пространство на области, определяемые как комбинации значений градации. Множество нечетких областей предпочтений M_k, которое образуется как декартовое произведение S_i, будем называть полным множеством нечетких альтернатив:

$$A = \{t_{11}, t_{12}, \ \dots, \ t_{1q_1}\} \times \{t_{21}, t_{22}, \ \dots, \ t_{2q_2}\} \times \dots \times \{t_{n1}, t_{n2}, \ \dots, \ t_{nq_n}\}.$$

Мощность вышеуказанного множества:

$$Q = |A| = \prod_{i=1}^{n} q_i.$$

В идеальном случае следует задать уровень предпочтений для всех элементов множества A. Опыт решения аналогичных задач с четкими градациями показал высокую трудоемкость данной процедуры.

В нечеткой постановке градации могут покрывать достаточно большие области критериального пространства, поэтому нечеткий уровень предпочтений будем задавать только для некоторых из этих областей.

Введем следующие обозначения:

k – номер области предпочтений;

K – число областей, на которых ЛПР задал свои предпочтения;

$M_k = (j_1, j_2, \dots, j_n)$ – представляет собой кортеж номеров градаций, участвующих в определении k-й области (предполагаем, что должны быть заданы нечеткие области по всем критериям);

p_k – нечеткий уровень предпочтений k-й области;

$\rho_k(y)$ – нечеткая функция принадлежности для k-й области;

y – четкое значение предпочтительности альтернативы;

$\tilde{y}$ – четкий ранг альтернативы, результат работы системы.

Примеры разбиения шкал критериев «Размер экрана» и «Стоимость» электронного планшета EFB на нечеткие градации представлены на рисунках 3.1 и 3.2 соответственно.

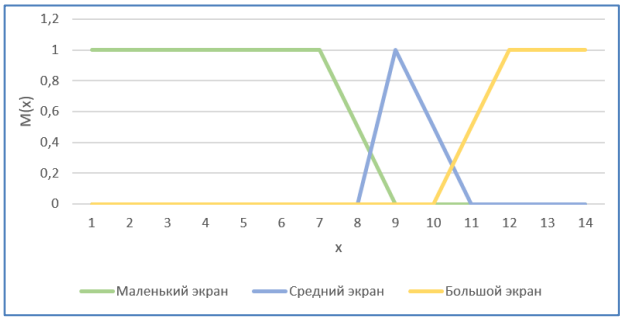


Рисунок 3.1 – Разбиение шкалы критерия «Размер экрана»

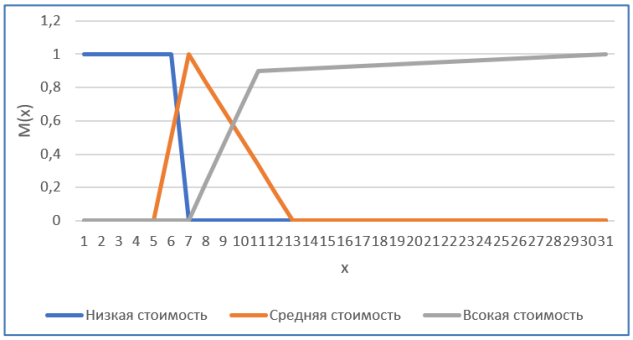


Рисунок 3.2 – Разбиение шкалы критерия «Стоимость»

Алгоритм 3.1 – Определение ранга альтернативы для значений критериев в нечетких областях пространства шкал критериев

1. $p_k(y, x_1, x_2, \ldots, x_n) = \min\left(\min_i\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right], \rho_k(y)\right)$
2. $p(y, x_1, x_2, \ldots, x_n) = \max_k p_k(y, x_1, x_2, \ldots, x_n)$
3. $\tilde{y}(x_1, x_2, \ldots, x_n) = \frac{\int y p(y, x_1, x_2, \ldots, x_n) dy}{\int p(y, x_1, x_2, \ldots, x_n) dy}$,
4. Ранжирование альтернатив с полученными значениями четких рангов $\tilde{y}$.

На первом шаге работы алгоритма выполняется определение предпочтительности альтернативы с применением нечеткой импликации вида: ЕСЛИ нечеткие значения критериев равны нечетким градациям области M_k, ТО предпочтительность альтернативы равна заданному для области нечеткому уровню предпочтений p_k.

$$p_k(y, x_1, x_2, \ldots, x_n) = \min\left(\min_i\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right],\ \rho_k(y)\right).$$

На втором шаге выполняется определение итоговой нечеткой предпочтительности альтернативы путем объединения всех нечетких значений предпочтений по всем областям:

$$p(y, x_1, x_2, \ldots, x_n) = \max_k p_k(y, x_1, x_2, \ldots, x_n).$$

После построения итоговой функции принадлежности для некоторых альтернатив может получиться так, что $p(y, x_1, x_2, \ldots, x_n) = 0$ или близко к значению 0. Это говорит о том, что существуют такие точки критериального пространства, которые не охвачены областями предпочтений. В данном случае необходимо осуществить изменение шкал и/или добавить новые области (увеличить k).

На третьем шаге производится дефаззификация предпочтений методом центра тяжести:

$$\tilde{y}(x_1, x_2, \ldots, x_n) = \frac{\int y p(y,\ x_1, x_2, \ldots, x_n) dy}{\int p(y,\ x_1, x_2, \ldots, x_n) dy},$$

где $\tilde{y}$ - четкий ранг альтернативы, результат работы системы.

Для решения задачи определения уровня предпочтительности альтернативы для заданных значений критериев $(x_1, x_2, \ldots, x_n)$ при участии автора настоящей диссертационной работы был разработан алгоритм 3.1, представленный на следующей странице.

Вход:

- `x` – массив значений критериев для оцениваемой альтернативы;
- `M` – массив областей предпочтений, первый индекс `k` номер области, а второй номер критерия;

- `mu(i, j, x)` – функция принадлежности значения `x` для `i`-го критерия и `j`-й градации;
- `Y =[1, 2, 3, …, h]` – массив уровней предпочтительности;
- `ro(k,y)` - функция принадлежности значения `y` к `k`-й области.

Выход:

`yd` – дефаззифицированное значение предпочтительности альтернативы.

Приведем фрагмент программы содержащий алгоритм нахождения значений $\mathrm{m}=\min_{i=\overline{1,n}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$.

```
1. k←1
2. while k ≤ K
   2.1.     m ←0; i ←1
   2.2.     while i<=n
      2.2.1.      j ←M[k][i]
      2.2.2.      p ← mu(i, j, x[i])
      2.2.3.      if m > p or i==1:  m ← p
      2.2.4.      i ← i+1
   2.3.     y ←1;
   2.4.     while y ≤ h
      2.4.1.      p[k][y] ← ro(k,y)
      2.4.2.      if  m ≤ p[k][y]:  p[k][y] ← m
      2.4.3.      if pmax[y] <  p[k][y] or k ==1:  pmax[y] ← p[k][y]
      2.4.4.      h ←h+1
   2.5.     k ←k + 1
3. y ←1; s1←0; s2 ←0
4. while y ≤ h
   4.1.     s1 ← s1 + y*pmax[y]
   4.2.     s2 ← s2 + pmax[y]
   4.3.     h ←h+1
5. yd ←s1/s2
```

Обоснование.

Покажем, что шаги 2.1-2.2.4 вычисляют $\mathrm{m}=\min_{i=\overline{1,n}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$.

Для доказательства будем использовать метод математической индукции.

База:

Пусть `i` = 1.

Тогда `j` = $M[k][1] = \mathrm{pr}_1 M_k$,

`p` = `mu(1, j, x[1])` = $\mu_{1\mathrm{j}}(x_1) = \mu_{1\mathrm{pr}_1 M_k}(x_1) = \min\limits_{i=\overline{1,1}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$

Так как `i` = 1 по второму условию `i==1` в строке 2.2.3 алгоритма `m` = `p` = $=\min\limits_{i=\overline{1,1}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$ база доказана.

Обозначим `m` = m_n.

Шаг индукции:

Пусть для $i = \overline{1, n-1}$: Шаги 2.1-2.3 вычисляют $m_{n-1} = \min\limits_{i=\overline{1,n-1}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$

Докажем, что для $i = \overline{1, n}$: Шаги 2.1-2.3 вычисляют m_n=$\min\limits_{i=\overline{1,n}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$

$m_n = \min\limits_{i=\overline{1,n}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$=min($\min\limits_{i=\overline{1,n-1}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right], \mu_{n\mathrm{pr}_n M_n}(x_n)$), подставив индуктивное предположение получим эквивалентную формулу, которую нужно доказать:

$$m_n = \min(m_{n-1}, \mu_{n\mathrm{pr}_n M_n}(x_n))$$

Далее проведем доказательство разбором случаев:

Если $m_{n-1} > \mu_{n\mathrm{pr}_n M_n}(x_n)$ то $m_n = \mu_{n\mathrm{pr}_n M_n}(x_n)$, строка 2.2.3 гарантирует нам это.

А если $m_{n-1} \le \mu_{n\mathrm{pr}_n M_n}(x_n)$ то $m_n = m_{n-1}$, строка так же 2.2.3 гарантирует нам это.

Следовательно, `m` = $\min\left(m', \mu_{n\mathrm{pr}_n M_n}(x_n)\right) = \min\limits_{i=\overline{1,n}}\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$ что и требовалось доказать. ■

Аналогично можно показать, что:

`pmax[y]`= $\max\limits_k p_k(y, x_1, x_2, \ldots, x_n)$ для заданных во входных данных фиксированных значений критериев $X = [x_1, x_2, \ldots, x_n]$ по строкам 2.3-2.2.4. В формулу $p_k(y, x_1, x_2, \ldots, x_n) = \min\left(\min\limits_i\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right], \rho_k(y)\right)$ подставляется посчитанное значение m на k-м шаге: `m[k]` = $\min\limits_i\left[\mu_{i\mathrm{pr}_i M_k}(x_i)\right]$ и далее итеративно

вычисляется `pmax[y]` = $\max_k \min(m[k], \rho_k(y))$ условие в строке 2.4.3 гарантирует нахождение максимума.

А $\tilde{y}(x_1, x_2, \ldots, x_n)=\frac{\int yp(y,\ x_1, x_2, \ldots, x_n)dy}{\int p(y,\ x_1, x_2, \ldots, x_n)dy}$ находится путем вычисления суммы:

$$\text{yd} = \frac{\sum_y y \cdot pmax[y]}{\sum_y pmax[y]}$$

Данный алгоритм использует операции на нечетких множествах, предложенные в работах Тэрано Т., Асаи К., Сугэно М. [77]. Многокритериальная свертка с нечеткой оценками альтернатив активно исследована в работах профессора Борисова А.Н. из Рижского университета [78]. Подходы к принятию решений в нечеткой информационной среде в настоящее время активно развиваются польскими учеными, например профессором Лешеком Рутковским [79]. Идеи данного алгоритма базируются на идеях разбиении пространства критериев на области предпочтений, предложенные научным руководителем соискателя в работах [80, 81].

Примером альтернативного подхода к решению данной задачи является методика моделирования на основе графа факторов и фреймов предложенная в работах академика Б.Н. Четверушкина и В.А. Судакова [82], однако в силу нечеткости суждений лиц, принимающих решения, в данной задаче, выбор был сделан в пользу алгоритма 3.1.

Пример 3.1. Определение четкого значения ранга на примере одной из заданных альтернатив.

Дано:

I. <u>критерии и шкалы, разбитые на нечеткие градации (термы):</u>

1) Память: "Недостаточная", "Достаточная";
2) Интернет 3G/LTE: "Имеется", "Отсутствует";
3) GPS/ГЛОНАСС: "Имеется ", "Отсутствует";
4) Диагональ экрана: "Маленький экран", "Средний экран", "Большой экран";
5) Год выпуска: "Менее 3 лет", "Более 3 лет";

6) Эргономика в кабине: "Плохая эргономика", "Хорошая эргономика";
7) Текущая стоимость: "Низкая", "Средняя", "Высокая".

II. Функции принадлежности градаций критериев:

Назначение функций принадлежности выполняется прямым методом, при котором ЛПР задает для каждой градации значение функций принадлежности на всей области определения.

Функции принадлежности градаций задаются на графике в виде многоугольника, для чего ЛПР задает существенные для него значения критериев (точки на шкале).

Таким способом заданы следующие функции принадлежности:

1) Для критерия «Память»:
   a. градация «Недостаточная»:
      $\mu_{11}(0) = 0, \mu_{11}(32) = 1, \mu_{11}(128) = 0$
   b. градация «Достаточная»:
      $\mu_{12}(16) = 0, \mu_{12}(128) = 1, \mu_{12}(256) = 1$
2) Для критерия «Интернет 3G/LTE»:
   a. градация «Имеется»:
      $\mu_{21}(\text{"Да"}) = 1, \mu_{21}(\text{"Нет"}) = 0$
   b. градация «Отсутствует»:
      $\mu_{22}(\text{"Да"}) = 0, \mu_{22}(\text{"Нет"}) = 1$
3) Для критерия «GPS/ГЛОНАСС»:
   a. градация «Имеется»:
      $\mu_{31}(\text{"Да"}) = 0, \mu_{31}(\text{"Нет"}) = 1$
   b. градация «Отсутствует»:
      $\mu_{32}(\text{"Да"}) = 0, \mu_{32}(\text{"Нет"}) = 1$
4) Для критерия «Диагональ экрана»:
   a. градация «Маленький экран»:
      $\mu_{41}(0) = 1, \mu_{41}(7) = 1, \mu_{41}(9) = 0$

b. градация «Средний экран»:

$\mu_{42}(8) = 0, \mu_{42}(9) = 1, \mu_{42}(11) = 0$

c. градация «Большой экран»:

$\mu_{43}(9) = 0, \mu_{43}(12) = 1, \mu_{43}(13) = 1$

5) Для критерия «Год выпуска»:

a. градация "Менее 3 лет":

$\mu_{51}(13) = 0, \mu_{51}(14) = 1, \mu_{51}(17) = 1$

b. градация "Более 3 лет":

$\mu_{52}(10) = 1, \mu_{52}(13) = 1, \mu_{52}(14) = 0$

6) Для критерия «Эргономика в кабине ВС»:

a. градация "Плохая эргономика":

$\mu_{61}(1) = 1, \mu_{61}(2) = 0, \mu_{61}(3) = 0$

b. градация "Хорошая эргономика":

$\mu_{62}(1) = 0, \mu_{62}(2) = 0.7, \mu_{62}(3) = 1$

7) Для критерия «Текущая стоимость»:

a. градация "Низкая стоимость":

$\mu_{71}(0) = 1, \mu_{71}(25000) = 1, \mu_{71}(30000) = 0$

b. градация "Средняя стоимость":

$\mu_{72}(20000) = 0, \mu_{72}(30000) = 1, \mu_{72}(60000) = 0$

c. градация "Высокая стоимость":

$\mu_{73}(30000) = 0, \mu_{73}(50000) = 0.9, \mu_{73}(150000) = 1$

III. Области критериального пространства и предпочтений и функции принадлежности агрегирующего критерия:

ЛПР задает области критериального пространства и свои предпочтения на них. В рамках примера зададим три области и предпочтения (представлено в формате JSON):

1) Область предпочтений M_1 = <1, {1}, 2, {1}, 3, {1}, 4, {1, 2, 3}, 5, {1, 2}, 6, {2}, 7 {1, 2, 3}>:

$\rho_1(0) = 0, \rho_1(1) = 1, \rho_1(2) = 1, \rho_1(4) = 0$

2) Область предпочтений M_2 = <1, {2}, 2, {1}, 3, {1}, 4, {2, 3}, 5, {1}, 6, {2}, 7 {2, 3}>:

$\rho_2(2) = 0, \rho_2(3) = 1$

3) Область предпочтений M_3 = <1, {2}, 2, {1}, 3, {1}, 4, {1, 2, 3}, 5, {1}, 6, {2}, 7 {1, 2}>:

$\rho_3(3) = 0, \rho_3(5) = 1$

IV. Альтернатива и четкие значения ее критериев:

1) "Память" = 128,
2) "Интернет 3G/LTE" = "Да",
3) "GPS/ГЛОНАСС" = "Да",
4) "Диагональ экрана" = 9.7,
5) "Год выпуска" = 2014,
6) "Эргономика в кабине" = 2,
7) "Текущая стоимость" = 46990

Задача: найти четкий ранг альтернативы.

Решение:

I. Расчет принадлежности значений критериев нечетким градациям областей предпочтений

1) Для критерия «Память»:

$\mu_{11}(128) = 0$

$\mu_{12}(128) = 1$

2) Для критерия «Интернет 3G/LTE»:

$\mu_{21}(\text{"Да"}) = 1$

$\mu_{22}(\text{"Да"}) = 0$

3) Для критерия «GPS/ГЛОНАСС»:

$\mu_{31}(\text{"Да"}) = 1$

$\mu_{32}(\text{"Да"}) = 0$

4) Для критерия «Диагональ экрана»:

$\mu_{41}(9.7) = 0$

$\mu_{42}(9.7) = 0{,}65$

$\mu_{43}(9.7) = 0{,}23$

5) Для критерия «Год выпуска»:

$\mu_{51}(14) = 1$

$\mu_{52}(14) = 0$

6) Для критерия «Эргономика в кабине ВС»:

$\mu_{61}(2) = 0$

$\mu_{62}(2) = 0{,}7$

7) Для критерия «Текущая стоимость»:

$\mu_{71}(46990) = 0$

$\mu_{72}(46990) = 0{,}767$

$\mu_{73}(46990) = 0{,}765$

II. Определение нечетких уровней предпочтительности альтернативы

Определим нечеткий уровень предпочтительности альтернативы для заданных значений критериев с помощью функции принадлежности:

$$p_k(\text{у}, 128, \text{Да}, \text{Да}, 9.7, 14, 2, 46990) = \min\left(\min_i\left[\mu_{i\text{pr}_i M_k}(x_i)\right],\ \rho_k(y)\right).$$

$\min_i\left[\mu_{i\text{pr}_i M_1}(x_i)\right] = 0$

$\min_i\left[\mu_{i\text{pr}_i M_2}(x_i)\right] = 0{,}65$

$\min_i\left[\mu_{i\text{pr}_i M_3}(x_i)\right] = 0{,}65$

Графики функций принадлежности нечетких уровней предпочтений p_2 и p_3 примут вид, как представлено на рисунке 3.3:

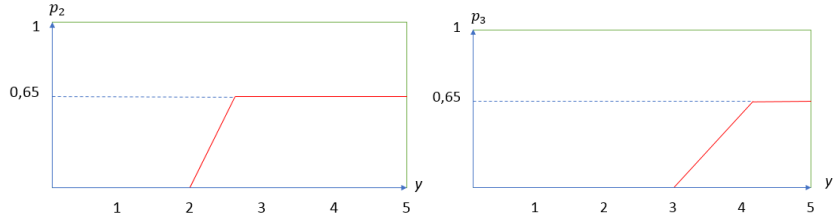


Рисунок 3.3 – Графики функций принадлежности нечетких уровней предпочтений p_2 (слева) и p_3 (справа)

III. Определение итоговой нечеткой предпочтительности альтернативы

Далее выполняется объединение всех нечетких значений предпочтений по всем областям, что даст итоговую нечеткую предпочтительность альтернативы:

$$p(y, 128, \text{Да}, \text{Да}, 9.7, 14, 2, 46990) = \max_k p_k(y, 128, \text{Да}, \text{Да}, 9.7, 14, 2, 46990).$$

График функции принадлежности агрегированной нечеткой предпочтительности альтернативы примет вид, как представлено на рисунке 3.4:

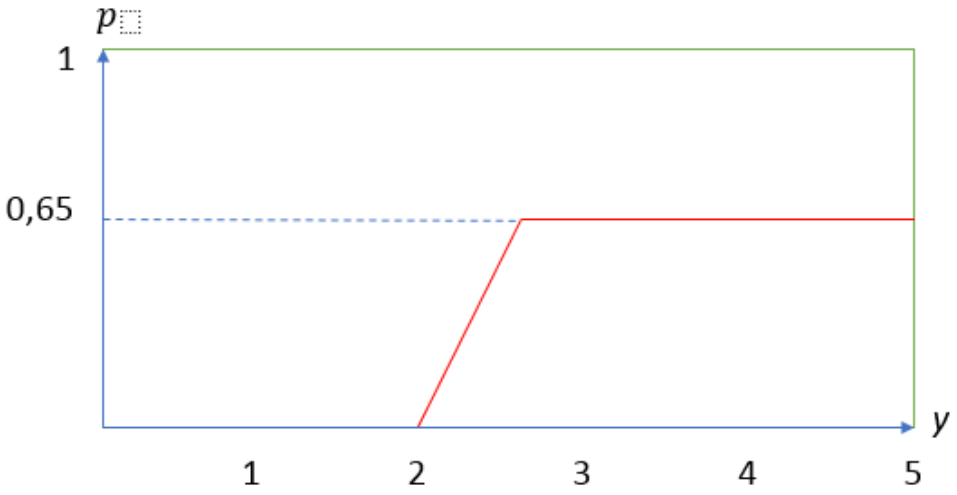


Рисунок 3.4 – График функции принадлежности итоговой нечеткой предпочтительности альтернативы

IV. Дефаззификация

Для определения четкого ранга альтернативы выполняется дефаззификация итоговой нечеткой предпочтительности альтернативы методом центра тяжести:

$$\tilde{y}(128, \text{Да}, \text{Да}, 9.7, 14, 2, 46990) = \frac{\int y p(y, 128, \text{Да}, \text{Да}, 9.7, 14, 2, 46990) dy}{\int p(y, 128, \text{Да}, \text{Да}, 9.7, 14, 2, 46990) dy}.$$

Таким образом, четкий ранг $\tilde{y}$ рассмотренной альтернативы ≈ 3,65.

Диаграмма активности нечеткого ранжирования в общем виде представлена на рисунке 3.5.

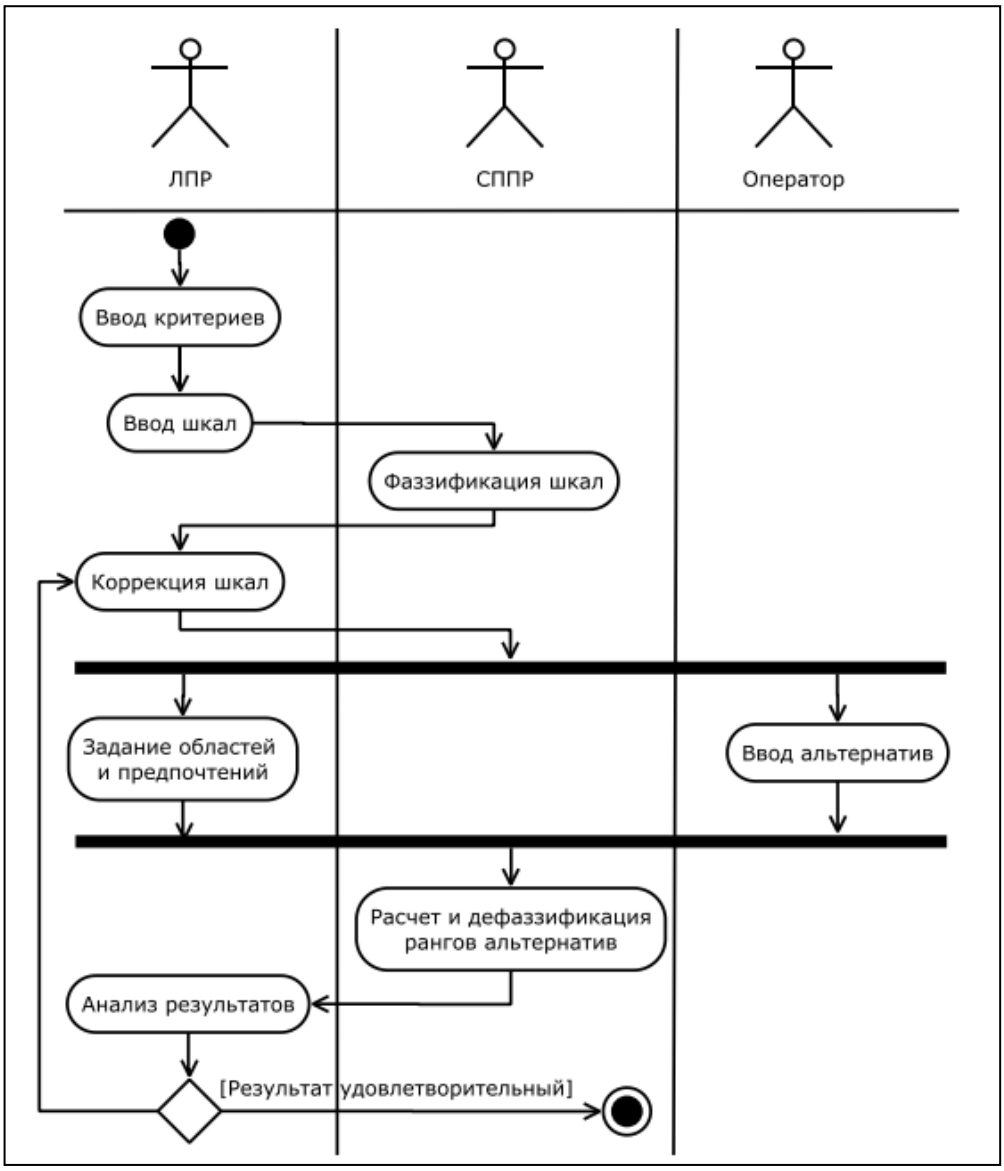


Рисунок 3.5 – Диаграмма активности при нечетком ранжировании

3.2. Результаты ранжирования альтернатив (аппаратных платформ разрабатываемой информационной системы)

В таблице 3.2 приведены исходные данные и результаты ранжирования аппаратных платформ с итоговым четким рангом альтернативы $\tilde{y}$.

Предложенный метод позволит эксплуатантам (авиакомпаниям) на этапе разработки плана внедрения EFB решить многокритериальную задачу выбора подходящей модели планшетного компьютера путем ранжирования альтернатив на основе нечетких предпочтений ЛПР, заданных в нечетких областях.

Из таблицы 3.2 видно, что в результате вывода получилось 3 наилучших неразличимых альтернативы, из которых ЛПР выберет одну.

Таблица 3.2 – Исходные данные и результаты ранжирования

Модель устройства	Память	Интернет 3G/LTE	GPS/ ГЛОНАСС	Диагональ экрана	Год выпуска	Эргономика в кабине	Текущая стоимость	Итоговая оценка
iPad Air 2 128 Гб Wi-Fi + Cellular	128	Да	Да	9.7	2014	2	46990	3,65
iPad Pro 9'7 128 Гб Wi-Fi + Cellular	128	Да	Да	9.7	2016	2	61990	3,65
iPad Pro 9'7 256 Гб Wi-Fi + Cellular	256	Да	Да	9.7	2016	2	68990	3,65
iPad Air 2 32 Гб Wi-Fi + Cellular	32	Да	Да	9.7	2014	2	39990	1,89
iPad Pro 9'7 32 Гб Wi-Fi + Cellular	32	Да	Да	9.7	2016	2	54990	1,89
iPad Mini 2 32 Гб Wi-Fi Only	32	Нет	Нет	7.9	2013	3	19990	1,74
iPad Mini 2 32 Гб Wi-Fi + Cellular	32	Да	Да	7.9	2013	3	29990	1,74
iPad Mini 4 32 Гб Wi-Fi Only	32	Нет	Нет	7.9	2015	3	29990	1,74
iPad Mini 4 32 Гб Wi-Fi + Cellular	32	Да	Да	7.9	2015	3	39990	1,74
iPad Air 2 32 Гб Wi-Fi Only	32	Нет	Нет	9.7	2014	2	29990	1,70
iPad Pro 9'7 32 Гб Wi-Fi Only	32	Нет	Нет	9.7	2016	2	44990	1,70
iPad Pro 12'9 32 Гб Wi-Fi Only	32	Нет	Нет	12.9	2015	1	58990	0,33
iPad Pro 12'9 128 Гб Wi-Fi Only	128	Нет	Нет	12.9	2015	1	65990	0,33
iPad Pro 12'9 256 Гб Wi-Fi Only	256	Нет	Нет	12.9	2015	1	72990	0,33
iPad Pro 12'9 128 Гб Wi-Fi + Cellular	128	Да	Да	12.9	2015	1	75990	0,33
iPad Pro 12'9 256 Гб Wi-Fi + Cellular	256	Да	Да	12.9	2015	1	82990	0,33
iPad Mini 4 128 Гб Wi-Fi Only	128	Нет	Нет	7.9	2015	3	36990	0,00
iPad Mini 4 128 Гб Wi-Fi +Cellular	128	Да	Да	7.9	2015	3	46990	0,00
iPad Air 2 128 Гб Wi-Fi Only	128	Нет	Нет	9.7	2014	2	36990	0,00
iPad Pro 9'7 128 Гб Wi-Fi Only	128	Нет	Нет	9.7	2016	2	51990	0,00
iPad Pro 9'7 256 Гб Wi-Fi Only	256	Нет	Нет	9.7	2016	2	58990	0,00

Результаты расчета ранга альтернатив в формате JSON представлены в Приложении 6.

Выводы по главе 3

В резюме по третьей главе сделаны следующие выводы:

Рассмотренная задача выбора модели планшетного компьютера является нетривиальной многокритериальной задачей, требующей применения современных подходов теории принятия решений.

Предложенный метод поддержки принятия решений позволяет проводить ранжирование альтернатив на основе нечетких предпочтений ЛПР, заданных в нечетких областях, с учетом нечетких суждений экспертов.

Разработанный подход позволил успешно решить задачу выбора электронного планшета для использования в качестве EFB.

ГЛАВА 4. Программная реализация

4.1. Программная реализация системы для расчета ВПХ

В настоящее время активно развивается отечественная нормативная база, регламентирующая одобрение и применение планшетных компьютеров в авиации в качестве EFB, при этом стремительно растет разнообразие моделей планшетных компьютеров, разрешенных к использованию в качестве EFB. Эксплуатанты ВС уже сейчас на этапе внедрения EFB сталкиваются с непростой задачей выбора наиболее рациональной модели устройства для их конкретного случая.

С учетом существующего многообразия электронных планшетов, и, прежде всего, для обеспечения работы программного обеспечения на отечественной элементной базе для минимизации зависимости от зарубежных производителей и снижения санкционных рисков, конечной целью является создание универсальной системы расчета ВПХ, т.е. без привязки к одной определенной аппаратной платформе. Достижение указанной цели возможно как за счет адаптации программно-алгоритмического прототипа разрабатываемой системы под существующие платформы, что является простой технической задачей, так и за счет разработки с использованием универсальных систем программирования, транслирующих код для соответствующих виртуальных машин, обеспечивающим возможность использования системы на всем многообразии существующих устройств, без необходимости индивидуальной доработки под каждое из них. К таким универсальным решениям сейчас относятся системы программирования React Native [83] и Flutter [84]. Несмотря на то, что данные системы разрабатываются преимущественно зарубежными разработчиками, они являются свободными и распространяются с открытыми исходными текстами, что снижает риски прекращения поддержки, позволяет самостоятельно дорабатывать системные библиотеки и минимизирует вероятность вредоносных «закладок» в коде.

В целях упрощения тестирования разрабатываемого прототипа и обеспечения возможности быстрой адаптации к другим платформам за счет

простоты синтаксиса его разработка проведена в среде программирования Xcode на языке Swift [85-87]. Исходный код программы представлен в Приложении 4.

Xcode позволяет разрабатывать приложения в соответствии с парадигмой «модель-контроллер-представление» (Model-View-Controller – MVC), реализующей способ разделения кода для приложений с графическим интерфейсом пользователя [88].

Шаблон MVC разделяется по функциональным возможностям на три категории:

- модель, состоящая из классов, в которых хранятся данные приложения;

- представление, создающее окна, элементы управления и другие элементы, которые пользователь видит и с которыми взаимодействует;

- контроллер, связывающий модель и представление, реализующий логику приложения, в соответствии с которой оно обрабатывает данные, введенные пользователем.

Парадигма MVC обеспечивает максимальное повторное использование кода и позволяет создавать как можно более независимые один от другого объекты, реализующие каждый из вышеуказанных типов кода.

Swift является open-source (т.е. с открытым исходным кодом) мультипарадигмальным языком программирования, созданным компанией Apple. Предназначен, прежде всего, для разработки программного обеспечения для операционных систем MacOS, iOS, iPadOS.

В качестве системы управления базой данных была выбрана платформа Realm [89]. Realm является кросс-платформенной СУБД, совместимой с различными высокоуровневыми языками программирования: Swift, Objective-C, Java, Kotlin, C# и JavaScript. Realm распространяется бесплатно, с открытым исходным кодом. Скорость работы Realm значительно превосходит такие СУБД, как Core Data, SQLite, ORMLite и Greendao [90].

При проектировании разрабатываемой системы (в целях снижения рисков ошибок при вводе данных пилотами) необходимо учитывать принципы, изложенные в нормативном документе ICAO Doc 10020:

- приложения для расчета ВПХ должны использовать данные, взятые из РЛЭ или других источников, отвечающих требованиям авиационных властей;
- проверка приложения должна проводиться при работающем приложении на типовых операционной системе и аппаратном устройстве
- входные и выходные данные должны четко различаться, а вся информация, необходимая для выполнения конкретной задачи, должна быть легко доступна пользователю;
- все данные, необходимые для выполнения расчета взлетно-посадочных характеристик, должны запрашиваться и отображаться с использованием корректных и однозначных наименований, а единицы измерения должны соответствовать единицам измерения, используемым для тех же данных в других источниках в кабине пилотов;
- обозначение вводимых пользователем данных должно четко отличаться от обозначения значений, заданных по умолчанию, а также значений, полученных от других источников (например, бортовых систем, или интернет-ресурсов);
- регистрационный номер воздушного судна, для которого производится расчет, должен четко отображаться для летных экипажей в том случае, когда имеются различия данных между регистрационными номерами;
- система должна принимать введенные данные для расчета только в том случае, если они не выходят за пределы сертифицированных эксплуатационных условий воздушного судна, а также укладываются в диапазон эксплуатационных условий, утвержденный эксплуатантом;
- все допущения, принятые для расчета взлетно-посадочных характеристик (например, состояние работы систем ВС, полная или пониженная тяга) должны быть четко отображены;
- пользователь должен иметь возможность легко исправить входные данные (например, для учета последних изменений);
- любые выбранные отложенные дефекты/отклонения MEL/CDL должны быть четко видимы и идентифицируемы;

- изменения пилотом данных о взлетно-посадочной полосе и/или препятствиях должны быть четко отображены и идентифицируемы.

Функциональным назначением разрабатываемой системы являются определение максимально допустимой взлетной и посадочной масс воздушного судна, рациональных параметров взлета и посадки.

Система будет использоваться для решения следующих задач:

- определение максимально допустимой взлетной массы воздушного судна для фактических условий взлета;
- определение характерных скоростей для фактической взлетной массы и фактических условий взлета;
- определение рациональных параметров взлетной и посадочной конфигурации воздушного судна, при которых обеспечивается наивысшая экономическая эффективность его эксплуатации (режим тяги двигателей, положение механизации крыла самолета, режим торможения), и обеспечивающих необходимый уровень безопасности для выполнения взлета и посадки;
- определение максимальной посадочной массы;
- определение характерных скоростей на посадке для фактической массы самолета и условий посадки.

Опираясь на вышеуказанные основные функции разрабатываемой системы, а также на ее архитектуру, представленную в главе 2, можно сформировать ее функциональную структуру. Функциональная структура системы отражена на схеме, приведенной на рисунке 4.1.

Реализация прототипа выполнена на примере ВС Ту-204-100 и RRJ-95B.

Продукции в ЭС представлены на языке CLIPS (C Language Integrated Production System) - гибком и мощном языке представления знаний, созданным в NASA в 1985 году для разработки экспертных систем.

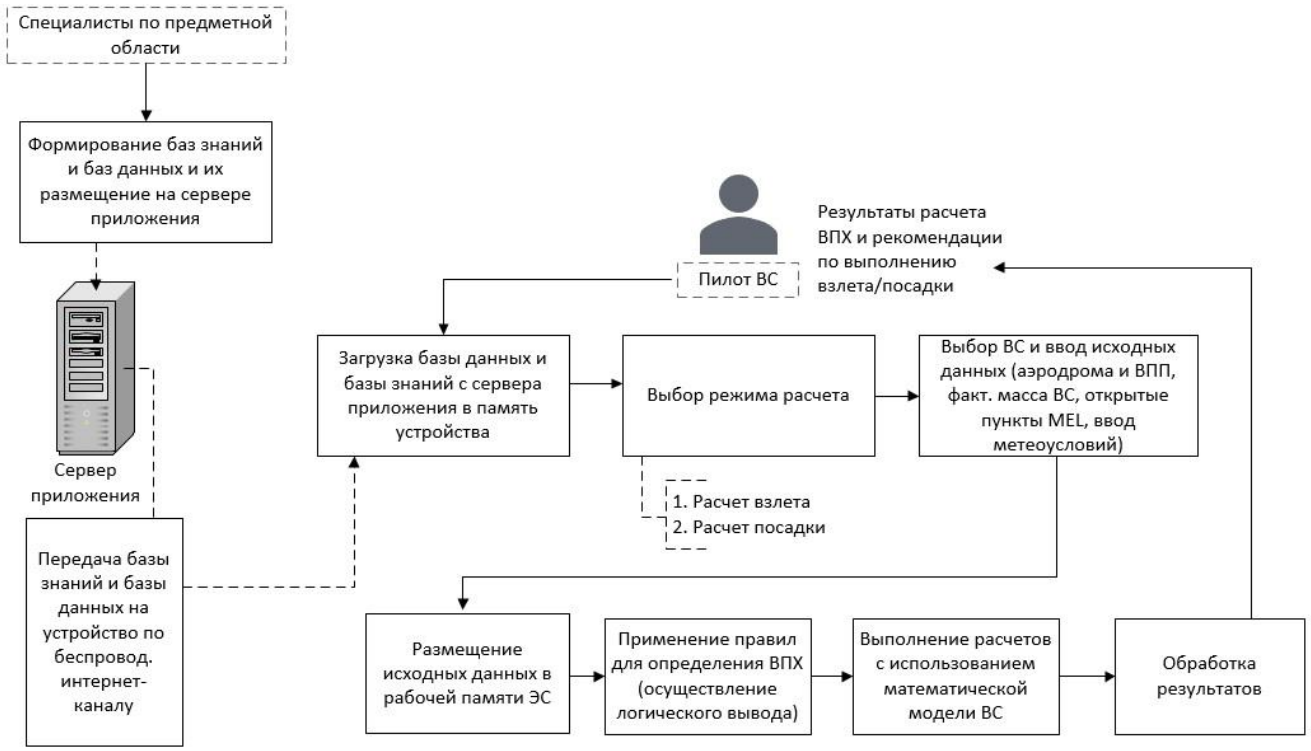


Рисунок 4.1 – Функциональная структура системы для расчета ВПХ

CLIPS обладает рядом следующих преимуществ:

- является свободно распространяемым программным продуктом с открытым исходным кодом (что минимизирует вероятность наличия вредоносного кода и позволяет, при необходимости, осуществлять самостоятельную доработку библиотеки)
- обеспечивает инвариантность по отношению к аппаратным и программным платформам за счет предусмотренной интеграции посредством соответствующих фреймворков;
- широко распространен во всем мире, как в государственных организациях, так и коммерческих частных компаниях; в частности – в аэрокосмической отрасли (в том числе использовался для разработки ЭС в КБ ОАО «Туполев»);
- являясь общественным достоянием, до сих пор обновляется и поддерживается своим изначальным автором, Гэри Райли (Gary Riley).

Для интеграции инструментального средства создания экспертных систем CLIPS в программное iOS-приложение использовался специализированный фреймворк.

Примеры продукционных правил базы знаний разрабатываемой ЭС, реализованных на CLIPS представлены на рисунке 4.2:

```
(defrule test-rule-1
    (test (= ?dest_airport "Pridacha"))
    =>
    (bind ?land_dist_available 2400)
)

(defrule test-rule-2
    (test (< $land_mass 90000))
    =>
    (bind ?land_dist_req 2000)
)

(defrule test-rule-3
    (sost_pov_vpp_land is slyakot)
    =>
    (+ ?land_dist_req 800)
)

(defrule test-rule-4
    (test (> ?land_dist_req ?land_dist_available))
    =>
    (assert (uiti na zapasnoi aerodrom)
)
```

```
(defrule test-rule-5
    (test (< $oat 0))
    =>
    (assert (anti_ice_system is ON)
)

(defrule test-rule-6
    (anti_ice_system is ON)
    (MEL_item_341910 is applied)
    =>
    (assert (vzlet na ponizhennoi tyage zapreschen)
)
```

Рисунок 4.2 – Примеры продукционных правил ЭС, реализованных на CLIPS

Реализован механизм обновления базы данных и базы знаний «по воздуху». Обновление осуществляется путем загрузки соответствующих JSON-файлов на электронной планшет с сервера efbgroup.ru по беспроводному интернет-каналу (рисунок 4.3).

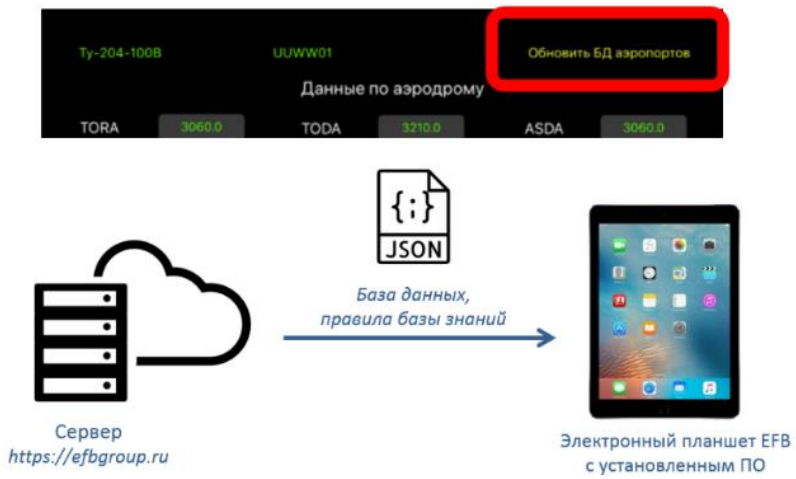


Рисунок 4.3 – Реализация загрузки базы данных и правил ЭС в EFB

Исходный код разработанного автором программного обеспечения для определения взлетно-посадочных характеристик содержит 2350 строк, размер файлов с исходным кодом – 116 КВ. Размер скомпилированного файла программы (с подключенной СУБД, графическими элементами и дополнительными библиотеками) составляет ≈ 39,3 МВ.

4.2. Интерфейс и порядок работы с программным приложением EFB

Работа приложения реализована в двух режимах – режим «РАСЧЕТ ВЗЛЕТА» и «РАСЧЕТ ПОСАДКИ».

Перед первым использованием приложения необходимо загрузить соответствующие базы данных. Для этого, нажать кнопку «Обновить БД аэропортов» в верхней части экрана и дождаться появления окна «Базы данных обновлены».

4.2.1. Режим «Расчет взлета»

Рассмотрим режим «РАСЧЕТ ВЗЛЁТА». Данный режим предназначен для расчета параметров взлета: максимальной взлетной массы и скоростей на взлете, режима работы двигателя, положения механизации крыла для заданных эксплуатационных условий.

Сначала необходимо выбрать воздушное судно, для которого будет выполняться расчет взлетных характеристик. Для этого требуется нажать кнопку «Выбрать ВС» в верхнем левом углу экрана и выбрать из выпадающего списка необходимое ВС (рисунок 4.4):

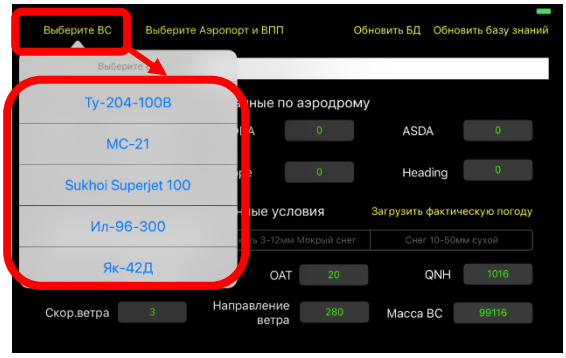


Рисунок 4.4 – Выбор воздушного судна в приложении EFB

После выбора ВС на экране отобразится интерфейс ввода исходных данных: панель для отображения и ввода характеристик ВПП и аэродрома взлета, ниже – панель указания состояния поверхности ВПП, атмосферного давления, температуры наружного воздуха, скорости ветра и его направления, фактической взлетной массы.

На втором шаге пользователь должен выбрать из списка аэродром, номер взлетно-посадочной полосы и идентификатор перекрестка, с которого будет осуществляться взлет (рисунок 4.5):

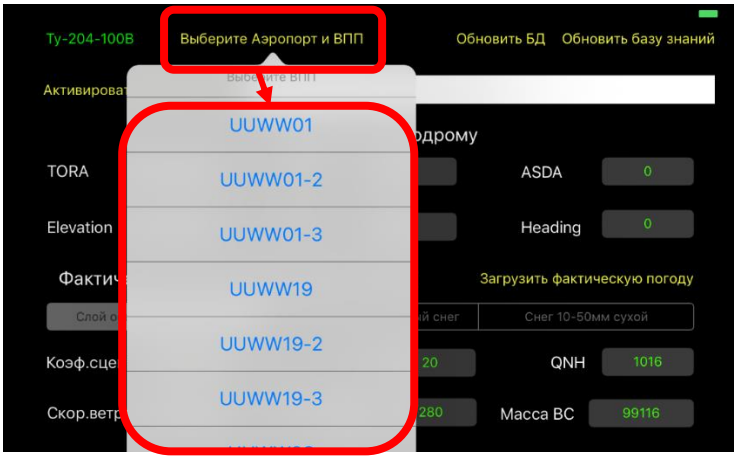


Рисунок 4.5 – Выбор аэродрома, ВПП и перекрестка ВПП для выполнения взлета

После их выбора, автоматически заполнятся поля, предназначенные для характеристик выбранных аэродрома и ВПП (рисунок 4.6):

- TORA (располагаемая дистанция разбега самолета, указанная в метрах);
- TODA (располагаемая дистанция продолженного взлета, указанная в метрах);
- ASDA (располагаемая дистанция прерванного взлета, указанная в метрах);
- Elevation (значение превышение аэродрома – самой высокой точки ВПП над уровнем моря, указывается в метрах);
- Slope (уклон ВПП в процентах);
- Heading (курс выбранной ВПП).

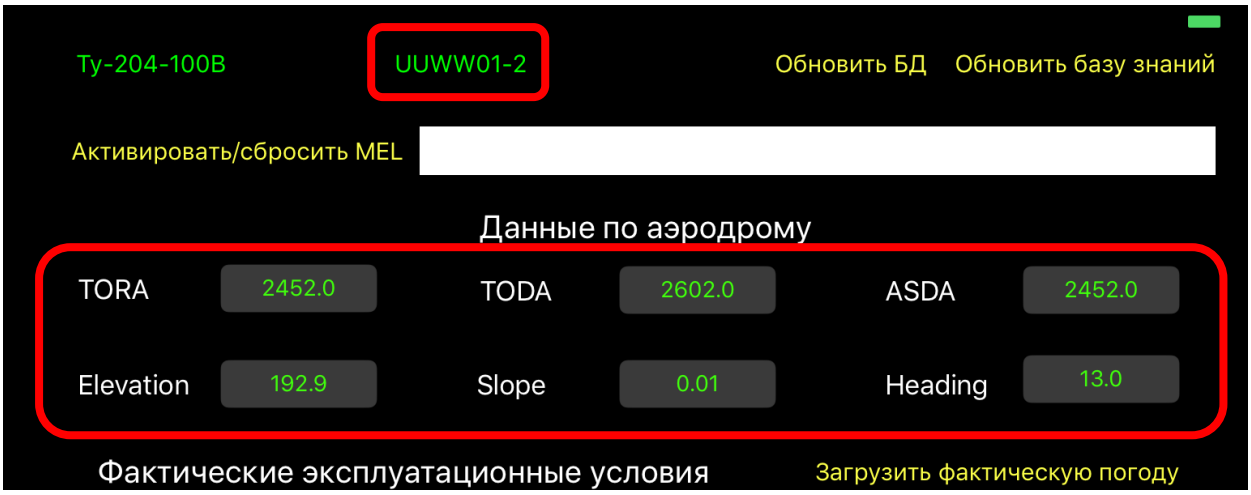


Рисунок 4.6 – Данные по аэродрому заполнились автоматически

При необходимости, допускается редактирование значений в вышеуказанных полях. Для этого необходимо нажать на соответствующее поля и с помощью появившейся экранной клавиатуры ввести новое значение.

После этого, при наличии открытых MEL, нажать кнопку «Активировать/сбросить MEL» и выбрать из выпадающего списка все открытые на момент выполнения расчета пункты MEL (рисунок 4.7):

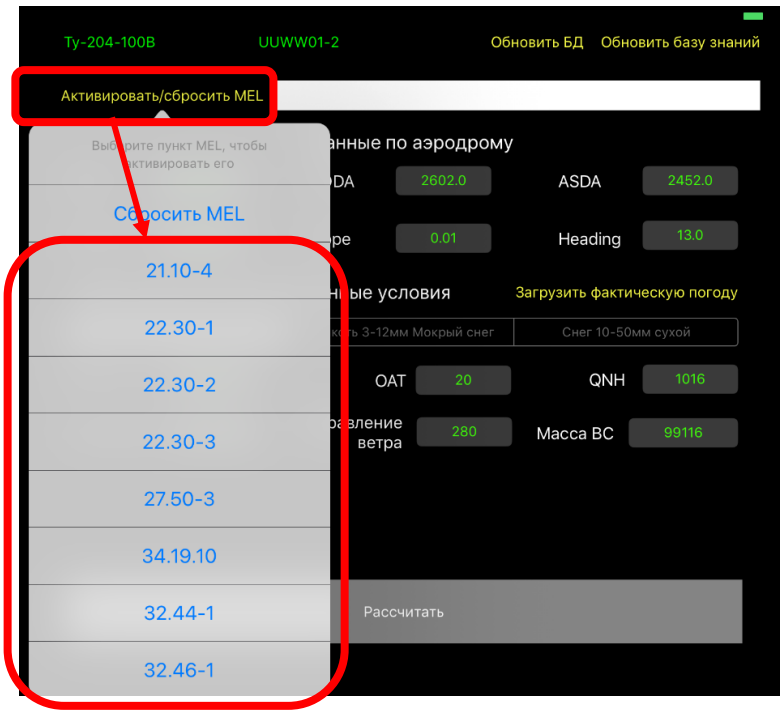


Рисунок 4.7 – Выбор открытых MEL воздушного судна в приложении EFB

Чтобы «сбросить» выбранные пункты MEL, необходимо нажать «Активировать/сбросить MEL» и выбрать «Сбросить MEL».

Далее, пользователь должен выбрать соответствующее состояние поверхности ВПП (рисунок 4.8):

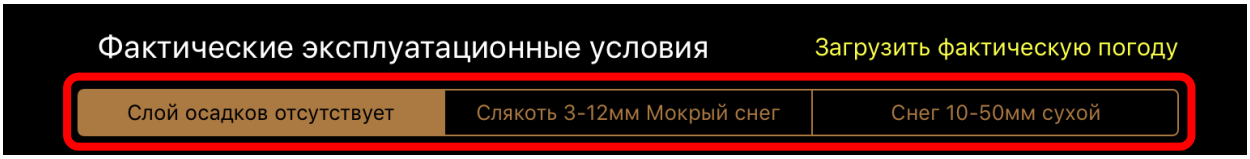


Рисунок 4.8 – Панель выбора состояния поверхности ВПП

После этого, требуется заполнить поля для значений фактических эксплуатационных условий (рисунок 4.9):

- Коэф.сцеп – коэффициент сцепления взлетно-посадочной полосы;

- OAT – значение температуры воздуха на аэродроме, указывается в градусах Цельсия;

- QNH – значение барометрической высоты аэродрома, заданная по атмосферному давлению QNH в гектопаскалях.

- Скор.ветра – для значения скорости ветра, в метрах/секунду;

- Направление ветра – вводится значение направления ветра;

- Масса ВС – значение фактической массы воздушного судна, указанное в килограммах.

Рисунок 4.9 – Ввод значений фактических эксплуатационных условий

После заполнения вышеуказанных параметров, необходимо нажать кнопку «Рассчитать» (рисунок 4.10):

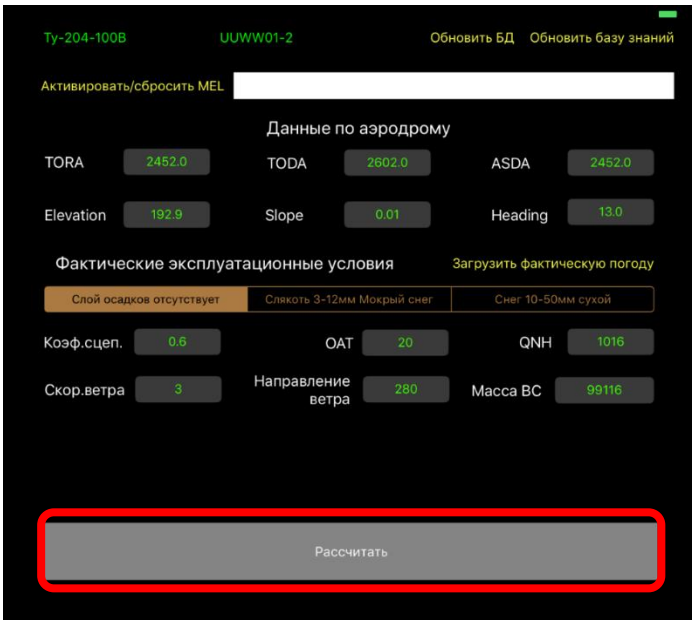


Рисунок 4.10 – Запуск выполнения расчета с помощью кнопки «Рассчитать»

После выполнения расчета появится интерфейс, включающий в себя: исходные данные, использованные для расчета, результаты расчета параметров взлета и визуализированное представление ВПП, с указанием ее номера, положением на ней воздушного судна перед началом разбега, располагаемой дистанции разбега самолета, и разложение ветра на продольную и поперечную составляющие с указанием направлений указанных составляющих и значений соответствующих скоростей в метрах/секунду.

Чтобы вернуться не предыдущий экран для корректировки исходных данных и повторного выполнения расчета, необходимо нажать «Вернуться назад» (рисунок 4.11):

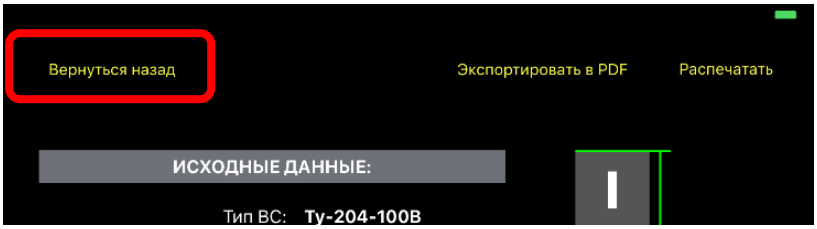


Рисунок 4.11 – Кнопка возврата на экран ввода исходных данных

Чтобы экспортировать результаты расчета в виде PDF-файла, для его сохранения на электронном планшете EFB или отправки по электронной почте на требуемый email-адрес, нажать кнопку «Экспортировать в PDF» (рисунок 4.12):

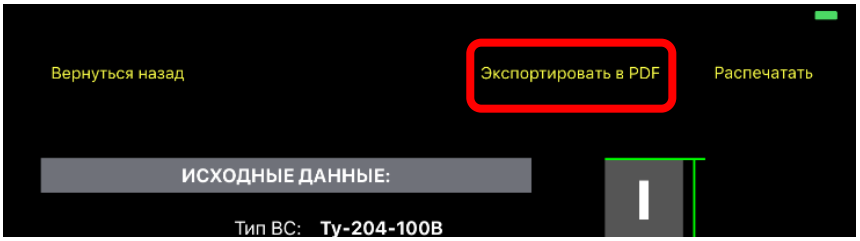


Рисунок 4.12 – Кнопка экспорта результатов расчета в файл PDF

Чтобы полученные результаты распечатать на принтере, нажать кнопку «Распечатать» (рисунок 4.13):

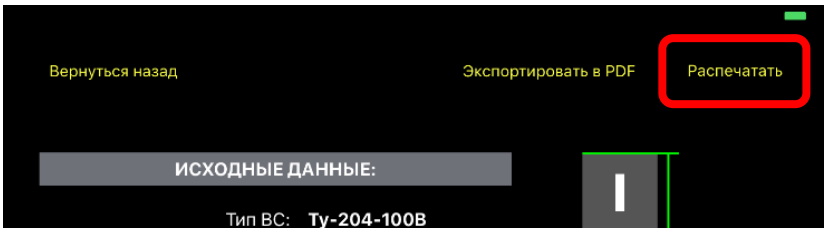


Рисунок 4.13 – Вывод результатов расчета на печать

4.2.2. Режим «Расчет посадки»

Рассмотрим режим «РАСЧЕТ ПОСАДКИ».

Данный режим предназначен для расчета параметров посадки воздушного судна: максимальной посадочной массы, скорости захода на посадку, режима торможения и других, предусмотренных РЛЭ воздушного судна.

Для перевода программного приложения в данный режим, необходимо нажать кнопку «РАСЧЕТ ПОСАДКИ» в нижней части экрана.

Сначала необходимо выбрать воздушное судно, для которого будет выполняться расчет взлетных характеристик. Для этого требуется нажать кнопку «Выбрать ВС» в верхнем левом углу экрана и выбрать из выпадающего списка необходимое ВС.

После выбора ВС на экране отобразится кнопка для выполнения расчета, перед запуском которого потребуется ввести необходимые данные с использованием интерфейса ввода исходных данных: характеристик ВПП и аэродрома посадки, состояния поверхности ВПП, атмосферного давления, температуры наружного воздуха, скорости ветра и его направления, фактической посадочной массы.

На втором шаге пользователь должен выбрать из списка аэродром и взлетно-посадочную полосу для выполнения посадки. После их выбора, автоматически заполнятся поля, предназначенные для характеристик выбранных аэродрома и ВПП (рисунок 4.14):

- LDA (располагаемая посадочная дистанция, указанная в метрах);
- Elevation (значение превышение аэродрома – самой высокой точки ВПП над уровнем моря, указывается в метрах);
- Slope (уклон ВПП в процентах);
- Heading (курс выбранной ВПП).

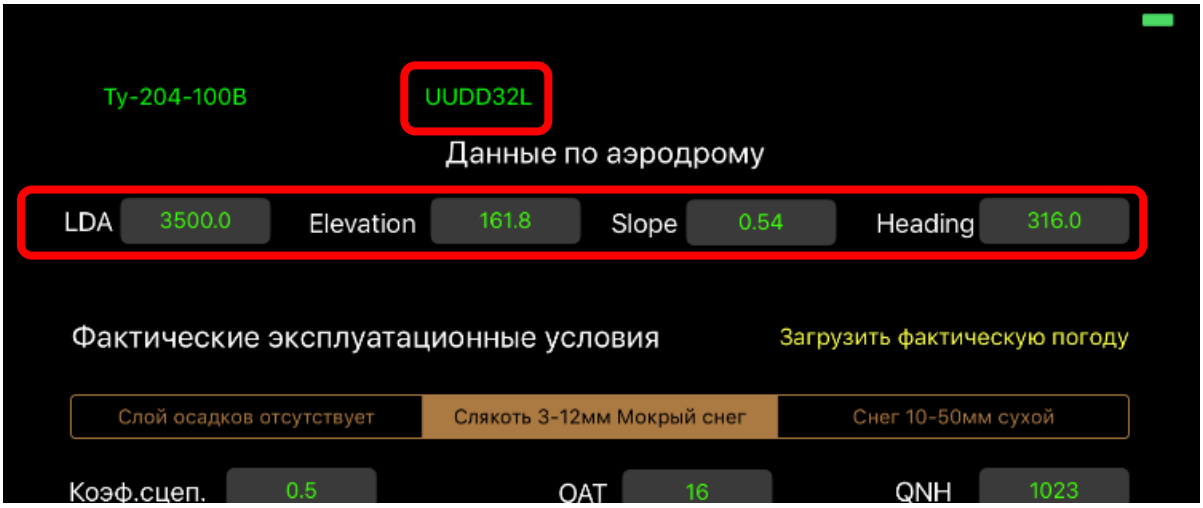


Рисунок 4.14 – Выбор аэродрома и ВПП для выполнения посадки и автоматическое заполнение полей заявленных характеристик выбранной ВПП

При необходимости, допускается редактирование значений в вышеуказанных полях. Для этого необходимо нажать на соответствующее поля и с помощью появившейся экранной клавиатуры ввести новое значение.

Далее, пользователь должен выбрать соответствующее состояние поверхности ВПП.

После этого, требуется заполнить поля для значений фактических эксплуатационных условий:

- Коэф.сцеп – коэффициент сцепления взлетно-посадочной полосы;

- OAT – значение температуры воздуха на аэродроме, указывается в градусах Цельсия;

- QNH – значение барометрической высоты аэродрома, заданная по атмосферному давлению QNH в гектопаскалях.

- Скор.ветра – для значения скорости ветра, в метрах/секунду;

- Направление ветра – вводится значение направления ветра;

- Масса ВС – значение посадочной массы воздушного судна, указанное в килограммах.

После заполнения вышеуказанных параметров, необходимо нажать кнопку «Рассчитать».

В результате выполнения расчета в нижней половине экрана программой будет осуществлен вывод значений максимальной посадочной массы, безопасной скорости захода на посадку, потребной длины ВПП. В зависимости от типа ВС, могут быть выведены дополнительные данные, предусмотренные в РЛЭ (такие, как режим торможения, градиенты ухода на второй круг и т.д.).

4.3. Пример решения прикладной задачи определения ВПХ с помощью разработанного программного приложения

В данном разделе представлен пример решения прикладной задачи определения ВПХ с помощью разработанного программного приложения.

Дано:

- исходные данные:

- тип ВС: Ту-204-100В;
- фактическая масса ВС: 99116 кг.;
- аэропорт: Внуково;

- номер ВПП: 01, перекресток: 02;
- состояние поверхности ВПП: сухая, без слоя осадков;
- коэффициент сцепления ВПП: 0,6;
- температура воздуха на аэродроме: 20ºС;
- барометрическая высота аэродрома QNH: 1016 гПа;
- скорость ветра: 3 м/с;
- направление ветра: 280;
- применен пункт MEL 34.19.10;

- база знаний ЭС (в целях упрощения демонстрационного примера, рассмотрим 10 продукционных правил)**:**

1. RULE_1_MEL_22.30-1: ЕСЛИ «Применен MEL_22.30-1» ТО «Взлет с пониженной тягой не выполняется»;
2. RULE_2_MEL_22.30-2: ЕСЛИ «Применен MEL_22.30-2» ТО «Взлет с пониженной тягой не выполняется»;
3. RULE_3_MEL_22.30-2: ЕСЛИ «ПРИМЕНЕН MEL_22.30-3» ТО «Взлет с пониженной тягой и понижение тяги в наборе высоты не выполняется»;
4. RULE_4_MEL_27.50-3: ЕСЛИ «Применен MEL_27.50-3» И «Коэффициент сцепления > 0,45» ТО «Взлёт запрещён»;
5. RULE_5_MEL_34.19.10: ЕСЛИ «Применен MEL_34.19.10» И «Аэропорт вылета = Домодедово» ТО «Вылет запрещен»;
6. RULE_6_MEL_34.19.10: ЕСЛИ «Применен MEL_34.19.10» ТО «Максимальная взлётная масса самолета определяется как при нормальной эксплуатации и уменьшается на 12 тонн»;
7. RULE_7_MEL_56.10-1: ЕСЛИ «Применен MEL_56.10-1» И «Аэропорт вылета = Домодедово» ТО «Взлёт запрещён»;
8. RULE_8_OPERATOR_POLICY_1: ЕСЛИ «Состояние поверхности ВПП – мокрый снег» И «Фактическая масса ВС > 95000» ТО «Увеличить потребную взлётную дистанцию на 500 метров»;

9. RULE_9_OPERATOR_POLICY_2: ЕСЛИ «Температура воздуха на аэродроме > 35°C» И «барометрическая высота аэродрома < 950 гПа» ТО «Максимальная взлётная масса самолета определяется как при нормальной эксплуатации и уменьшается на 10 тонн» И «Взлет на пониженной тяге запрещен» И «Потребную дистанцию взлёта увеличить на 500 метров»;
10. RULE_10_OPERATOR_POLICY_3: ЕСЛИ «Направление ветра = курс ВПП» И «Скорость ветра > 10 м/с» ТО «Увеличить потребную взлетную дистанцию на 700 метров».

Требуется найти: взлетные скорости и режим работы двигателя на взлёте.

Для выполнения расчета необходимо перевести приложение в режим «Расчет взлёта», выбрать требуемый тип ВС, аэропорт, ВПП и ввести вышеуказанные данные по фактическим эксплуатационным условиям, как представлено на рисунке 4.15:

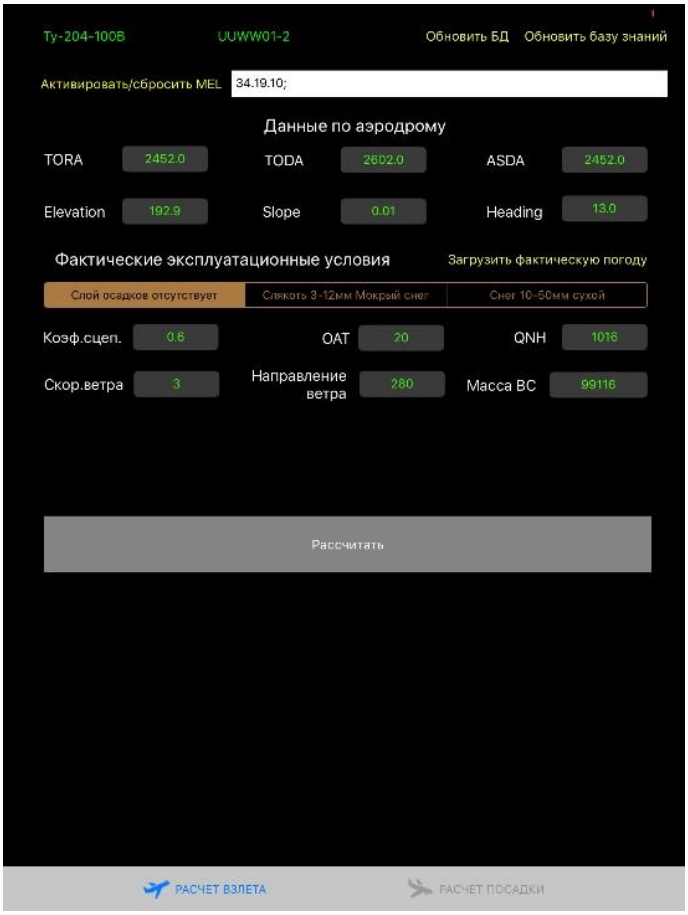


Рисунок 4.15 – Ввод исходных данных
для расчета взлетных характеристик ВС Ту-204-100В

После завершения ввода исходных данных нажать «Рассчитать».

На первом шаге программа разместит исходные данные в рабочей памяти информационной системы.

На следующем шаге информационная система выполнит сопоставление фактов из рабочей памяти с правилами базы знаний.

В результате сопоставления правила, антецеденты которых примут значение «ИСТИНА», активируются (помещаются в рабочий список). В рассматриваемом примере активируется только одно правило: RULE_6_MEL_34.19.10.

Далее правило выполняется, и в рабочую память добавляется факт «уменьшить максимальную взлётную массу самолета на 12 тонн» (в виде вспомогательной переменной, значение которой равно 12000 кг – величина, на которую на следующем этапе будет уменьшено значение максимальной взлетной массы).

На следующем шаге производится комплекс расчетов с использованием математической модели ВС.

Полученный результат показывает, что выполнение взлета при заданных условиях возможно со следующими значениями параметров взлета:

- скорость принятия решения V_1 = 237,9 км/ч;
- скорость подъема передней опоры шасси V_R = 237,9 км/ч;
- скорость на взлёте V_2 = 257,9 км/ч;
- скорость начала уборки механизации на взлете V_3 = 342,4 км/ч;
- скорость при полетной конфигурации V_4 = 371,7 км/ч;
- режим работы двигателя на взлете: 66,0; n_2 = 95%.

Далее выполняется обработка полученных результатов (округление значений, передача данных в модуль вывода результатов, подготовка визуализации положения ВС на ВПП и составляющих ветра) и, непосредственно, вывод результатов на экран – значений взлетных скоростей и допустимого режима работы двигателя для выполнения взлета (рисунок 4.16):

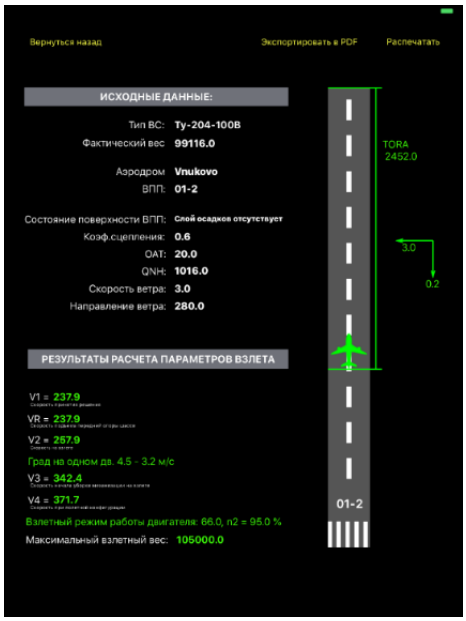


Рисунок 4.16 – Результаты расчета взлетных характеристик ВС Ту-204-100В

На приведенных далее изображениях также представлены примеры практического использования приложения для определения взлетно-посадочных характеристик ВС:

- на рисунке 4.17 изображен пример расчета посадочных характеристик самолета Ту-204-100В;

- на рисунках 4.18 и 4.19 – исходные данные и результаты расчета взлетных характеристик для Sukhoi Superjet 100.

Рисунок 4.17 – Пример исходных данных и результатов расчета посадочных характеристик ВС Ту-204-100В

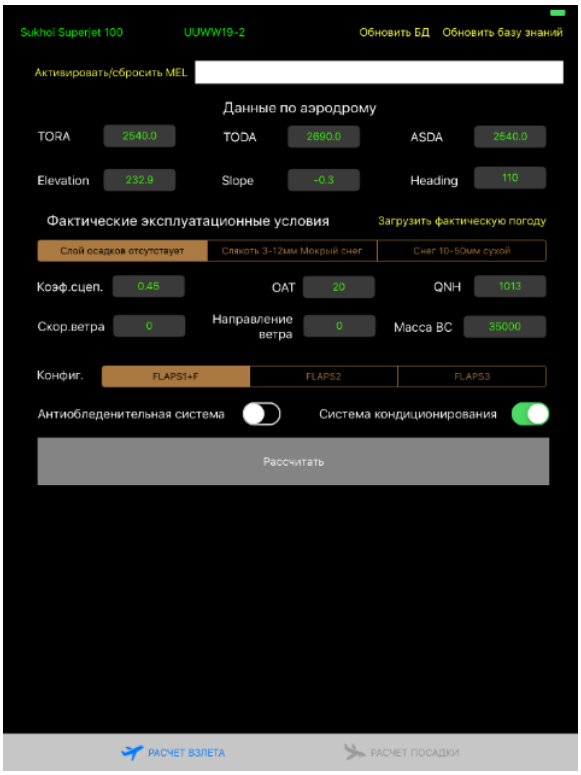


Рисунок 4.18 – Пример исходных данных для расчета взлетных характеристик ВС Sukhoi Superjet 100

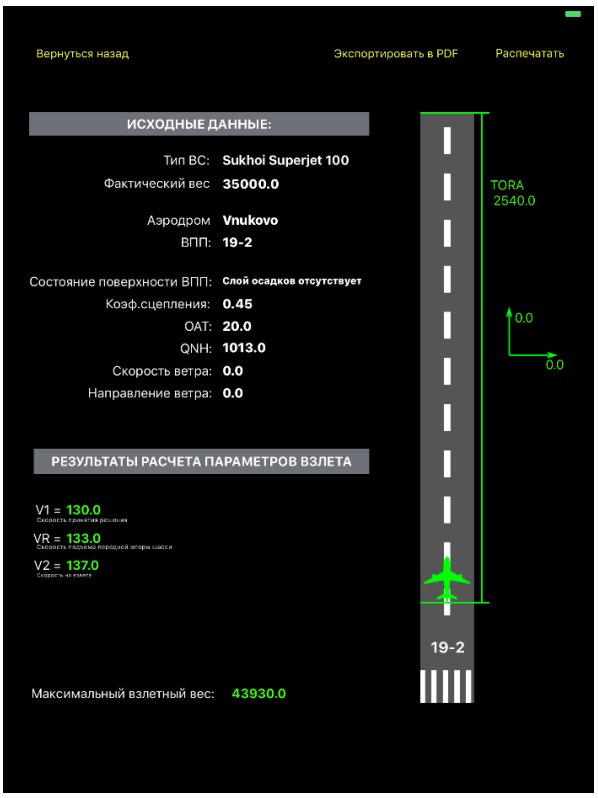


Рисунок 4.19 – Пример результатов расчета взлетных характеристик ВС Sukhoi Superjet 100

4.4. Рекомендации по использованию программного приложения летным экипажем

Поскольку приложение предназначено, прежде всего, для работы членов летных экипажей воздушных судов, предлагаются рекомендации по его использованию с учетом специфики стандартных процедур подготовки к полету и его выполнения.

Приложение следует использовать на этапах предполетной и предпосадочной подготовки.

В ходе предполетной подготовки летной подготовки (в помещении для проведения брифинга) летный экипаж получает от соответствующих служб следующую информацию:

- рабочий план полета (OFP);

- о техническом статусе воздушного судна с учетом MEL/CDL;

- метеорологические сводки;

- информацию NOTAM (извещение, рассылаемое средствами электросвязи и содержащее информацию о введении в действие, состоянии или изменении любого аэронавигационного оборудования, обслуживания и правил, или информацию об опасности, имеющую важное значение для персонала, связанного с выполнением полетов [91], которая может включать в себя сведения о неработающих рулёжных дорожках и взлётно-посадочных полосах, информацию о снеге, слякоти, льде или стоячей воде на ВПП рулежных дорожках аэродромов;

При изучении и анализе вышеуказанной информации, командир воздушного судна принимает решение о необходимости корректировки потребного запаса топлива на борту воздушного судна.

Далее, с использованием вышеуказанных сведений, члены летного экипажа выполняют с помощью предложенного программного приложения предварительный расчет как взлетных, так и посадочных характеристик ВС (для оценки возможности выполнения посадки с прогнозируемыми условиями и посадочной массой ВС).

В случае выявления превышения фактической массы воздушного судна допустимой для взлета при фактических условиях, командир воздушного судна имеет право принять решение переносе времени вылета.

В целях снижения риска ошибки при вводе исходных данных, расчет взлетно-посадочных характеристик ВС командир воздушного судна и второй пилот должны выполнять самостоятельно и независимо друг от друга, каждый на предназначенным для него электронном планшете EFB, и осуществить перекрестный контроль полученных результатов. При расхождении результатов расчетов оба члена экипажа должны выполнить повторный расчет и перекрестный контроль взлетно-посадочных характеристик.

Предполетная подготовка продолжается также после прибытия летного экипажа на борт ВС. После получения сводно-загрузочной ведомости с указанием финального значения фактической массы ВС, пилоты производят с помощью приложения окончательный расчет взлетно-посадочных характеристик.

До начала снижения, после получения от диспетчера организации воздушного движения или по специализированному каналу вещания погоды о фактических метеоусловиях на аэродроме посадки, уточнения информации о ВПП посадки, летный экипаж под руководством командира ВС проводит предпосадочную подготовку, в ходе которой с помощью предложенного программного приложения рассчитывает посадочные характеристики ВС для оценки возможности выполнения посадки на указанном аэродроме и принятии решения о посадочных параметрах ВС (положение механизации, режим торможения) при выполнении посадки.

4.5. Результаты пробных испытаний разработанного ПО для EFB

Проведены пробные испытания разработанного программного обеспечения EFB при подготовке летных экипажей к планируемому полету. Проведенные эксперименты показали, что разработанное ПО позволило в среднем сократить общее время предполетной подготовки на 18% (в штатных условиях). Полученные результаты представлены в таблице 4.1.

Таблица 4.1 – Результаты пробных испытаний разработанного программного приложения при подготовке летных экипажей к планируемому полету

		Пилот 1	Пилот 2	Пилот 3	Пилот 4
Общее время предполетной подготовки членов летного экипажа	С применением ПО, минут	65	71	68	64
	Без применения ПО, минут	53	58	57	51
Сокращение времени предполетной подготовки, %		18,5	18,3	16,2	20,3

Выводы по главе 4

В резюме по четвертой главе сделаны следующие выводы:

- Разработан прототип информационной системы в виде программного приложения для электронного полетного планшета.
- Для обеспечения работы программного обеспечения на отечественной элементной базе для минимизации зависимости от зарубежных производителей и снижения санкционных рисков предложены методические принципы создания универсальной системы расчета ВПХ путем адаптации программно-алгоритмического прототипа разрабатываемой системы под существующие платформы, либо за счет использования универсальных систем программирования, транслирующих код для соответствующих виртуальных машин.
- Применение разработанного программного обеспечения позволило сократить время подготовки летных экипажей к планируемому полету на 18%.

ЗАКЛЮЧЕНИЕ

Определение ВПХ является нетривиальным и трудоемким процессом, выполнение которого «вручную» приводит к избыточной нагрузке на летный экипаж и может привести к ошибкам. В связи с этим является целесообразной разработка автоматизированных систем расчета ВПХ.

Разработка системы автоматизированного расчета ВПХ осуществлялась комплексно и включала в себя как создание программного обеспечения, так и метода выбора рациональной аппаратной платформы.

Применяемые при определении ВПХ правила имеют вид «ЕСЛИ (условие), ТО (действие)», а их совокупность может быть представлена в качестве продукционной модели знаний. Для работы с продукциями предложено использование технологии продукционной экспертной системы с применением механизма прямого логического вывода. Полный перечень указанных правил определяется эксплуатантом ВС и может, при необходимости, изменяться.

Предложен подход к реализации механизма логического вывода с использованием алгоритма Rete, позволяющего минимизировать количество сопоставлений и, тем самым, повысить скорость работы информационной системы.

Для реализации технологии продукционной экспертной системы было выбрано программное инструментальное средство CLIPS, поскольку оно обладает рядом таких преимуществ, как поддержка мобильных платформ, свободное распространение и открытый исходный код, встроенная машина прямого логического вывода с реализаций алгоритма Rete.

В целях обеспечения адаптивности и расширяемости информационной системы предложена модульная архитектура. Такой поход позволил без необходимости внесения изменений в исходный код ЭС обеспечить инвариантность по отношению к типам воздушных судов.

Разработанная онтология предметной области EFB позволила сформировать структуру базы данных, обеспечивающей хранение разнородных фактов, используемых в работе ЭС.

Рассмотрен подход к оцифровке номограмм, представленных в РЛЭ, с использованием комплекса специализированного программного обеспечения Wolfram Mathematica и GetData Graph Digitizer. Данный подход позволил на основе оцифрованных данных сформировать математическую модель зависимостей ВПХ воздушного судна Ту-204 и реализовать их автоматизированный расчет.

Для решения задачи выбора модели планшетного компьютера разработан новый подход, позволяющий выполнять ранжирование альтернатив на основе заданных в нечетких областях нечетких предпочтений ЛПР. Разработанный подход позволил успешно решить задачу выбора электронного планшета для использования в качестве EFB.

Прототип информационной системы для определения ВПХ реализован в виде программного приложения для электронного полетного планшета EFB, получено соответствующее свидетельство о государственной регистрации программы для ЭВМ (Приложение 1). Разработка прототипа выполнена в среде разработки Xcode на языке Swift. В качестве СУБД информационной системы выбрано программное инструментальное средство Realm. Для обеспечения работы программного обеспечения на отечественной элементной базе для минимизации зависимости от зарубежных производителей и снижения санкционных рисков предложены принципы как адаптации программно-алгоритмического прототипа разрабатываемой системы под существующие платформы, так и использования универсальных систем программирования, транслирующих код для соответствующих виртуальных машин.

Основным итогом диссертационной работы является разработка и апробация информационной системы для автоматизированного расчета взлетно-посадочных характеристик воздушных судов в режиме реального времени, реализованной на базе электронного планшета летчика, а также разработка математического метода и программного обеспечения, предназначенного для рационального выбора аппаратной составляющей разработанной системы, что выразилось в следующих научных и практических результатах:

1. Создана методика создания информационных систем определения взлетно-посадочных характеристик, включающая:

   1.1. архитектуру информационной системы расчета взлетно-посадочных характеристик воздушных судов с применением технологии продукционной экспертной системы,

   1.2. онтологическое представление программного обеспечения для расчета взлетно-посадочных характеристик, позволившее сформировать структуру и атрибуты его базы данных,

   1.3. алгоритмы расчетов зависимостей ВПХ, построенные по оцифрованным номограммам, представленным в РЛЭ воздушного судна.

   расчетов зависимостей ВПХ, по номограммам представленным в РЛЭ воздушного судна.
2. Разработаны метод и алгоритм выбора аппаратного обеспечения информационной системы расчета взлетно-посадочных характеристик воздушных судов, используемого в кабине лётного экипажа в формате планшетного компьютера, на основе нечетких суждений;
3. Выполнена реализация программного обеспечения информационной системы в виде клиент-серверного приложения для электронного полётного планшета. На примере реализации методики показана перспективность практического применения данного подхода к автоматизированному расчету взлетно-посадочных характеристик в производственной деятельности авиакомпаний, что засвидетельствовано актом об апробации в производственной деятельности Авиакомпании АО «Авиакомпания «РусДжет» от 23.09.2019 года.

Разработанная информационная система для расчета взлетно-посадочных характеристик позволит значительно сократить время, затрачиваемое пилотами при подготовке к полету в части определения взлетно-посадочных характеристик, повысить эффективность летной эксплуатации воздушных судов и безопасность выполняемых полетов.

В связи с применением в разработанном программном обеспечении базы данных, содержащей информацию об аэродромах и параметрах взлетно-посадочных полос, потребуется обеспечить регулярное поддержание актуальности указанной базы данных на основе аэронавигационной информации из официальных источников (например, сборников аэронавигационной информации Российской Федерации, издаваемых филиалом «ЦАИ» ФГУП «Госкорпорация по ОрВД»). Также, для минимизации человеческого фактора в процессе ручного ввода исходных данных и в целях совершенствования реализованной информационной системы, дальнейшая разработка темы предполагает интеграцию с системами планирования полетов, используемых в авиакомпаниях, а также со специализированными источниками авиационных метеосводок.

СПИСОК СОКРАЩЕНИЙ И УСЛОВНЫХ ОБОЗНАЧЕНИЙ

ASDA - Accelerate-Stop Distance Available

CDL - Configuration Deviation List

EFB - Electronic Flight Bag

LDA - Landing Distance Available

MEL - Minimum Equipment List

MVC - Model-View-Controller

NOTAM - Notice to airmen

OAT - Outside Air Temperature

OFP - Operational Flight Plan

QNH - Q-code Nautical Height

TODA - Takeoff Distance Available

TORA - Takeoff Run Available

ВПП - взлетно-посадочная полоса

ВПХ - взлетно-посадочные характеристики

ВС - воздушное судно

ОрВД - организация воздушного движения

ПО - программное обеспечение

РЛЭ - руководство по летной эксплуатации

СУБД - система управления базами данных

ЭС - экспертная система

СПИСОК ЛИТЕРАТУРЫ

1. Арепьев К.А. Моделирование типовых характеристик воздушных судов на базе данных РЛЭ и лётных испытаний / Арепьев К.А. // Сборник научных трудов ГосНИИ ГА. – 2010. – №311. – С. 139-144.
2. Алкина М.Р. Автоматизация расчета взлетных летно-технических характеристик / Алкина М.Р., Калинина И.В. // Материалы докладов VII конференции молодых ученых «Навигация и управление движением». – 2005. – СПб.: ГЦН РФ ЦНИИ «Электроприбор». – С. 193-198.
3. Алкина М.Р. методы получения унифицированных алгоритмов расчета параметров взлетно-посадочных характеристик / Алкина М.Р., Зайцева Н.А. // Материалы докладов XIV конференции молодых ученых «Навигация и управление движением». – 2012. – СПб.: ГЦН РФ ЦНИИ «Электроприбор». – С. 150-156.
4. Моисеев В.Н. Особенности оцифровки номограмм для автоматизации инженерно-штурманских расчетов / Моисеев В.Н. // Перспективы развития информационных технологий. – 2016. – №33. – С. 20-26.
5. Zontul M. Rule Based Aircraft Performance System / Metin Zontul // International Journal of Soft Computing and Engineering (IJSCE). – 1993. – Volume-3, Issue-4. – P.61-66.
6. Раздел Onboard Performance Tool на официальном сайте Boeing [электронный ресурс] – Режим доступа: https://www.boeingservices.com/flight-operations/navigation-solutions/onboard-performance-tool/
7. Раздел FlySmart+ на официальном сайте NavBlue [электронный ресурс] – Режим доступа: https://www.navblue.aero/product/flysmart-plus/
8. Statistical Summary of Commercial Jet Airplane Accidents [Электронный ресурс]. – Boeing, 2019. – 28 с. – Режим доступа: https://www.boeing.com/resources/boeingdotcom/company/about_bca/pdf/statsum.pdf

9. Аэронавигационное обеспечение полетов: Методические указания по изучению дисциплины и выполнению контрольной работы / СПб ГУ ГА, С.-Петербург, 2007. – 34 с.
10. Шаров В.Д., Некрасов В.Г., Гордеев О.Ю. Характеристики состояния поверхности ВПП и их оценка при выполнении полетов : доклад на конференции Федерального агентства воздушного транспорта «Безопасность на ВПП» [Электронный ресурс]. – Режим доступа: http://favt.ru/public/materials//8/7/1/7/5/8717510eacc47fbe440b8d09fa5a6941.pdf
11. Ефремов А.В. Динамика полета : учебник для студентов высших учебных заведений / А.В. Ефремов, В.Ф. Захарченко, В.Н. Овчаренко - М.: Машиностроение, 2011. – 776 с.
12. Чепурных И.В. Динамика полета самолетов : учеб. Пособие / Чепурных И.В. – Комсомольск-на-Амуре : ФГБОУ ВПО «КнАГТУ», 2014. – 112 с.
13. Кощеев А.Б. Аэродинамика самолетов семейства Ту-204/214 : учебное пособие / Кощеев А.Б., Платонов А.А., Хабров А.В. – М.: ОАО «Туполев», издательство «Полигон-Пресс», 2009. – 304 с.
14. Бехтина Н.Б. Некоторые задачи математического моделирования движения по взлетно-посадочным полосам тяжелых транспортных самолетов / Н.Б. Бехтина, М.С. Кубланов, А.П. Степушин // Научный вестник МГТУ ГА. – 2010. – № 151. – С. 99-104.
15. Хрусталев М.М. Идентификаторы пониженной размерности в задаче стабилизации беспилотного летательного аппарата в неспокойной атмосфере / Хрусталев М.М., Халина А.С. // Труды МАИ. – 2018. – №102. – С. 22.
16. Хрусталев М.М. Метод Галѐркина в задачах оптимизации квазилинейных динамических стохастических систем с информационными ограничениями / Хрусталев М.М., Румянцев Д.С., Царьков К.А. // Труды МАИ. – 2013. – №66. – С. 20.

17. Хрусталев М.М. Простой алгоритм стабилизации ориентации спутника с гибким элементом / Хрусталев М.М., Халина А.С. // Труды МАИ. – 2012. – №55. – С. 12.
18. Проблемы безопасности на ВПП по результатам расследований авиационных происшествий : Доклад Межгосударственного авиационного комитета [Электронный ресурс]. – Режим доступа: http://favt.ru/public/materials//a/1/9/3/7/a1937b4da8cb6269ecac632813871a63.pdf
19. ICAO Document 10020. Руководство по электронным полетным планшетам (EFB). Издание 2, 2018. — 508 с.
20. Приказ Минтранса России от 31.07.2009 N 128 (ред. от 22.04.2020) "Об утверждении Федеральных авиационных правил "Подготовка и выполнение полетов в гражданской авиации Российской Федерации" (Зарегистрировано в Минюсте России 31.08.2009 N 14645) [Электронный ресурс]. – Режим доступа: http://www.consultant.ru/document/cons_doc_LAW_91259/
21. Приказ Минтранса России от 13.08.2015 N 246 (ред. от 14.01.2020) "Об утверждении Федеральных авиационных правил "Требования к юридическим лицам, индивидуальным предпринимателям, осуществляющим коммерческие воздушные перевозки. Форма и порядок выдачи документа, подтверждающего соответствие юридических лиц, индивидуальных предпринимателей, осуществляющих коммерческие воздушные перевозки, требованиям федеральных авиационных правил" (Зарегистрировано в Минюсте России 07.10.2015 N 39163) [Электронный ресурс]. – Режим доступа: http://www.consultant.ru/document/cons_doc_LAW_187361/
22. Процесс получения эксплуатантами коммерческой и деловой авиации РФ эксплуатационного одобрения Росавиации на использование в кабине воздушного судна электронной системы бортовой документации (EFB) : методическое пособие [Электронный ресурс]. – Некоммерческая организация "Российская ассоциация эксплуатантов воздушного

транспорта", 2015. – 37 с. – Режим доступа: http://www.ato.ru/files/attached_materials/metodicheskoe_posobie_efb_versiya_hhhh_ot_06_07_2015.pdf

23. Airworthiness and operational consideration for Electronic Flight Bags (EFBs) : AMC 20-25 [Электронный ресурс]. – European Aviation Safety Agency, 2014. – 49 с. – Режим доступа: https://www.easa.europa.eu/sites/default/files/dfu/2014-001-R-Annex%20II%20-%20AMC%2020-25.pdf
24. Authorization for Use of Electronic Flight Bags : Advisory Circular No: 120-76D [Электронный ресурс]. – Federal Aviation Administration, 2017. – 35 с. – Режим доступа: https://www.faa.gov/documentLibrary/media/Advisory_Circular/AC_120-76D.pdf
25. Мельничук А.В., Марценюк Е.А. Предпосылки создания ЭС для выбора электронного полетного планшета электронной информационной системы EFB для летного экипажа воздушного судна // Международная молодежная научная конференция «XXXIII Туполевские чтения (школа молодых ученых»: Материалы конференции. Сборник докладов, в 4 т. – Казань: Изд-во Академии наук РТ, 2017. – Т.2. – С.781-784.
26. Мельничук А.В., Марценюк Е.А. Предпосылки создания ЭС для определения требуемых характеристик процесса взлета/посадки ВС в зависимости от погодных условий и конкретных параметров взлетно-посадочной полосы // 16-я Международная конференция «Авиация и космонавтика - 2017»: Сборник тезисов докладов, Москва: Типография «Люксор», 2017. – С.174-175.
27. Мельничук А.В., Судаков В.А. Предпосылки создания системы автоматизированного расчета взлетно-посадочных характеристик воздушного судна // Гагаринские чтения – 2016: XLII Международная молодёжная научная конференция: Сборник тезисов докладов: В 4 т. М.: Московский авиационный институт (Национальный исследовательский университет), 2016. – С.428-429.

28. Мельничук А.В., Нестеров В.А., Судаков В.А., Сыпало К.И. Разработка приложения для определения рациональных характеристик процессов взлета и посадки воздушных судов с применением экспертной системы // Ежеквартальный научный журнал «Электронные информационные системы». – М.: АО «НТЦ ЭЛИНС», 2019. №1 (20). С.63-72.
29. А.В. Мельничук, В.А.Нестеров, В.А.Судаков, К.И.Сыпало. Разработка экспертной системы электронного планшета летчика (EFB) для определения рациональных характеристик процессов взлета и посадки воздушных судов // 11-я международной конференции «Управление развитием крупномасштабных систем» (MLSD’2018): Труды конференции, в 3 т. – Москва: ИПУ РАН, 2018. – Т.2. – С.310-316.
30. А.В. Мельничук, В.А.Нестеров, В.А.Судаков, К.И.Сыпало. Реализация экспертной системы в программном приложении электронного планшета летчика для определения рациональных характеристик процессов взлета и посадки воздушных судов // 11-я международной конференции «Управление развитием крупномасштабных систем» (MLSD’2018): Материалы конференции, в 2 т. – Москва: ИПУ РАН, 2018. – Т.2. – С.147-149.
31. Сборник аэронавигационной информации Российской Федерации. Книга 1. Международные аэродромы Российской Федерации. [Электронный ресурс]. – Федеральное агентство воздушного транспорта. – Режим доступа: http://www.caiga.ru/common/AirInter/validaip/html/rus.htm
32. Новиков Ф. А. Искусственный интеллект: представление знаний и методы поиска решений: Учеб. пособие. / Новиков Ф. А. – СПб.: Изд-во Политехн. ун-та, 2010. – 240 с.
33. Джексон П. Введение в экспертные системы. 3-е издание. Пер. с англ. — М.: Изд. дом «Вильямс», 2001. - 624 с.
34. Джексон П. Введение в экспертные системы / Джексон П. – М.: Изд. Дом «Вильямс», 2001.
35. Гаврилова Т.А. Базы знаний интеллектуальных систем. / Гаврилова Т.А., Хорошевский В.Ф. – Издательский дом «Питер», 2001. – 384 с.

36. Частиков А.П. Разработка экспертных систем. Среда CLIPS / Частиков А.П., Гаврилова Т.А., Белов Д.Л. – 2003. – Спб: БХВ-Петербург. – 608 с.
37. Поспелов Д.А. Продукционные модели / Поспелов Д.А. // Искусственный интеллект. Кн.2. – М.: «Радио и связь», 1990.
38. Перфильев О.В. Экспертная система интеллектуальной поддержки авиаспециалистов при техническом обслуживании систем и оборудования самолета / Перфильев О.В. // Известия Самарского научного центра Российской академии наук. – 2014. – том 16, №1(5). – С. 1545-1549.
39. Перфильев О.В. Концепция экспертной системы анализа причин неисправностей самолёта Ту-204 и его модификаций / Перфильев О.В., Липатова С.В. // Известия Самарского научного центра Российской академии наук. – 2013. – том 15, №4(4). – С. 892-896.
40. Мельничук А.В. Разработка продукционной экспертной системы для определения взлетно-посадочных характеристик воздушного судна // Научно-технический вестник Поволжья. 2020. № 10.
41. Форги Ч., Rete: быстрый алгоритм для многошаблонных/многообъектных задач сопоставления с образцом, Искусственный интеллект, 19, с. 17-37, 1982.
42. Официальный сайт CLIPS [электронный ресурс] – Режим доступа: http://www.clipsrules.net/
43. Международный стандарт ISO для языка Пролог [электронный ресурс] – Режим доступа: https://web.archive.org/web/20040811091714/http://www.sju.edu/~jhodgson/wg17/wg17web.html
44. Russian Lisp Users Group [электронный ресурс] – Режим доступа: http://lisp.ru/
45. Официальный сайт Drools [электронный ресурс] – Режим доступа: https://www.drools.org/
46. Официальный сайт Jess [электронный ресурс] – Режим доступа: https://jess.sandia.gov/

47. Реализация бизнес-логики при помощи процессора правил Drools [Электронный ресурс]. – Режим доступа: https://www.ibm.com/developerworks/ru/library/j-drools/index.html
48. Loftin R.B. An intelligent training system for space shuttle flight controllers / Loftin R.B., Wang L., Baffes P., Hua G. // Telematics and Informatics. – 1988. – Volume 5, Issue 3. – P. 151-161.
49. Silberberg D. The NASA Personnel Security Processing Expert System / Silberberg D., Thomas R. // IAAI'96: Proceedings of the eighth annual conference on Innovative applications of artificial intelligence – 1996. – P. 1527–1535.
50. Gruber T. A. Translation approach to portable ontology specification / Gruber T. A. // Knowledge Acquistion. – 1993. – Vol. 5. – P.199-220.
51. Смирнов С.В. Онтологии как смысловые модели / Смирнов С.В. // Онтология проектирования. – 2013. – №2(8). – С.12-19
52. Луцкий М.Г. Разработка онтологии безопасности авиации / Луцкий М.Г. // Инженерия программного обеспечения. – 2010. – №4. – С. 56-62.
53. Боргест Н.М. Онтология проектирования самолета / Боргест Н.М. // Искусственный интеллект. – 2011. – №4. – С. 260-165.
54. Noy N.F. Ontology Development 101: A Guide to Creating Your First Ontolog [Электронный ресурс] / N.F. Noy, D.L. McGuinnes // Stanford Knowledge Systems Laboratory – 2001. – Режим доступа: https://protege.stanford.edu/publications/ontology_development/ontology101.pdf
55. Набатов А.Н. Применение онтологического подхода к процессу проектирования информационной системы / Набатов А.Н., Веденяпин И.Э., Мухтаров А.Р. // Труды МАИ. – 2018. – №102. – С. 26.
56. Грегер С.Э. Построение онтологии архитектуры информационной системы / Грегер С.Э., Поршнев С.В. // Фундаментальные исследования. – 2013. – №10. – С. 2405-2409.
57. Мельничук А.В., Судаков В.А. Применение онтологического подхода к процессу разработки и внедрения систем Electronic Flight Bag // Моделирование и анализ данных. 2020. Том 10. № 1. С. 157–165.

58. Официальный сайт сообщества Protégé [электронный ресурс] – Режим доступа: https://protege.stanford.edu/
59. Раздел аэронавигационной базы данных «АРНАД» на официальном сайте филиала ЦАИ ФГУП Госкорпорация по ОрВД [электронный ресурс] – Режим доступа: https://www.caicaishop.ru/catalog/aeronautical-database/
60. Круковец А.С. Разработка метода интерполяции значений номограммы [Электронный ресурс] / Круковец А.С., Горелкин Г.А. // Современные научные исследования и инновации. – 2015. – №5-2(49). – С. 64-71. – Режим доступа: http://web.snauka.ru/issues/2015/05/53846/
61. Официальный сайт Wolfram Mathematica [электронный ресурс] – Режим доступа: https://www.wolfram.com/mathematica/
62. Официальный сайт GetData [Электронный ресурс]. – Режим доступа: http://getdata-graph-digitizer.com/ru/download.php/
63. Руководство по лётной эксплуатации. Самолет Ту-204-100В. – ПАО «Туполев», 2017. – 2456 с.
64. Главный перечень минимального состава оборудования. Самолет Ту-204-100В. – ОАО «Туполев», 2008. – 261 с.
65. Летное руководство. RRJ-95. – ЗАО «Гражданские самолеты сухого», 2013. – 776 с.
66. Главный перечень минимального состава оборудования самолета RRJ-95. – ЗАО «Гражданские самолеты сухого», 2019. – 582 с.
67. Мельничук А.В., Нестеров В.А., Судаков В.А., Сыпало К.И. Разработка программного приложения планшетного компьютера для определения параметров взлета и посадки воздушных судов // XIII Всероссийское совещание по проблемам управления ВСПУ-2019: Труды. – Москва: ИПУ РАН, 2019. – С. 940-945.
68. Мельничук А.В., Нестеров В.А., Судаков В.А., Сыпало К.И. Реализация программного приложения для определения взлетно-посадочных характеристик российских воздушных судов с использованием принципов экспертной системы // 12-я международная конференция «Управление

развитием крупномасштабных систем» (MLSD’2019): Труды конференции. – Москва: ИПУ РАН, 2019. – С. 738-745.

69. Мельничук А.В., Нестеров В.А., Судаков В.А., Сыпало К.И. Разработка системы определения параметров взлета и посадки воздушных судов на базе электронного полетного планшета // 12-я международная конференция «Управление развитием крупномасштабных систем» (MLSD’2019): Материалы конференции, научное электронное издание – Москва: ИПУ РАН, 2019. – С.756-759.
70. Смерчинская С.О., Яшина Н.П. Интеллектуальная система поддержки принятия решений / Смерчинская С.О., Яшина Н.П. // Информатизация инженерного образования. Труды Международной научно-практической конференции. – 2016. – С. 214-217.
71. Смерчинская С.О., Яшина Н.П. Агрегирование предпочтений с учетом важности критериев / Смерчинская С.О., Яшина Н.П. // Труды МАИ. – 2015. – №84. – С. 31.
72. Мельничук А.В., Сивакова Т.В., Судаков В.А. Решение задач оптимизации с использованием мультиагентных моделей / Мельничук А.В., Сивакова Т.В., Судаков В.А. // Препринты ИПМ им. М.В.Келдыша. – 2019. – № 100. – 16 с.
73. Редько А.О., Смерчинская С.О., Яшина Н.П. Агрегирование предпочтений при переменной важности критериев / Редько А.О., Смерчинская С.О., Яшина Н.П. // Труды МАИ. – 2016. – №85. – С. 18.
74. Заде Л. От обработки чисел к обработке слов – от манипулирования измерениями к манипулированию восприятием. Международный журнал прикладной математики и компьютерной науки, с. 307-324, т. 12, №3, 2002.
75. Заде Л.А. Роль мягких вычислений и нечеткой логики в понимании, конструировании и развитии информационных/интеллектуальных систем. / Пер. с англ. // Новости искусственного интеллекта. – 2001. – № 2–3. – С. 7–11.
76. Заде Л.А. Основы нового подхода к анализу сложных систем и процессов принятия решений.- В кн.: Математика сегодня. - М.: Знание, 1974, с. 5-49.

77. Тэрано Т., Асаи К., Сугэно М. Прикладные нечеткие системы: Пер. с япон. / Тэрано Т., Асаи К., Сугэно М. – М.: Мир, 1993. – 368 с.
78. Борисов А.Н., Крумберг О.А., Федоров И.П. Принятие решений на основе нечетких моделей : Примеры использования / Борисов А.Н., Крумберг О.А., Федоров И.П. – Рига: Зинатне, 1990.– 184 с.
79. Рутковский Л. Методы и технологии искусственного интеллекта / Пер. с польск. И.Д. Рудинского. – М.: Горячая линия–Телеком, 2010. – 520 с.
80. Осипов В.П., Судаков В.А. Многокритериальный анализ решений при нечетких областях предпочтений // Препринты ИПМ им. М.В. Келдыша. 2017. № 6. С. 1-16.
81. Осипов В.П., Сивакова Т.В., Посадский А.И., Судаков В.А. Комбинированная методика нечеткого ранжирования альтернатив на основе функций предпочтений // Электротехнические и информационные комплексы и системы. 2019. Т. 15. № 1. С. 87-93.
82. Четверушкин Б.Н., Судаков В.А. Факторное моделирование для инновационно-активных предприятий // Математическое моделирование. 2020. Т. 32. № 3. С. 115-126.
83. Официальный сайт React Native [электронный ресурс] – Режим доступа: https://reactnative.dev/
84. Официальный сайт Flutter [электронный ресурс] – Режим доступа: https://flutter.dev/
85. A. Melnichuk, V. Nesterov, V. Sudakov and S. Kirill, "Development of Electronic Flight Bag Software Based on Expert System for Computing of Optimal Aircraft Performance," 2019 Twelfth International Conference "Management of large-scale system development" (MLSD), Moscow, Russia, 2019, pp. 1-4.
86. A. V. Melnichuk et al 2020 IOP Conf. Ser.: Mater. Sci. Eng. 714 012019 https://doi.org/10.1088/1757-899X/714/1/012019.
87. Мельничук А.В., Судаков В.А. Компьютерная поддержка решений пилота на этапах взлета и посадки // Моделирование и анализ данных. 2019. Том 09. № 4. С. 112–120.

88. Маскри М. Swift 3: разработка приложений в среде Xcode для iPhone и iPad с использованием iOS SDK / Маскри М., Топли К., Марк Д. – 2017. – Спб: ООО «Альфа-книга». – 896 с.
89. Официальный сайт Realm [электронный ресурс] – Режим доступа: https://realm.io/
90. Национальная библиотека им. Н. Э. Баумана [электронный ресурс] – Режим доступа: https://ru.bmstu.wiki/Realm#cite_note-1
91. ICAO. Правила аэронавигационного обслуживания. Организация воздушного движения. — 16. — 2016. — С. стр. 44 (1-22). — 508 с.

Приложение 1. Свидетельство о государственной регистрации программы для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ

СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2019661964

«Программное приложение для расчета взлётно-посадочных характеристик воздушных судов»

Правообладатель: ***Мельничук Александр Владимирович (RU)***

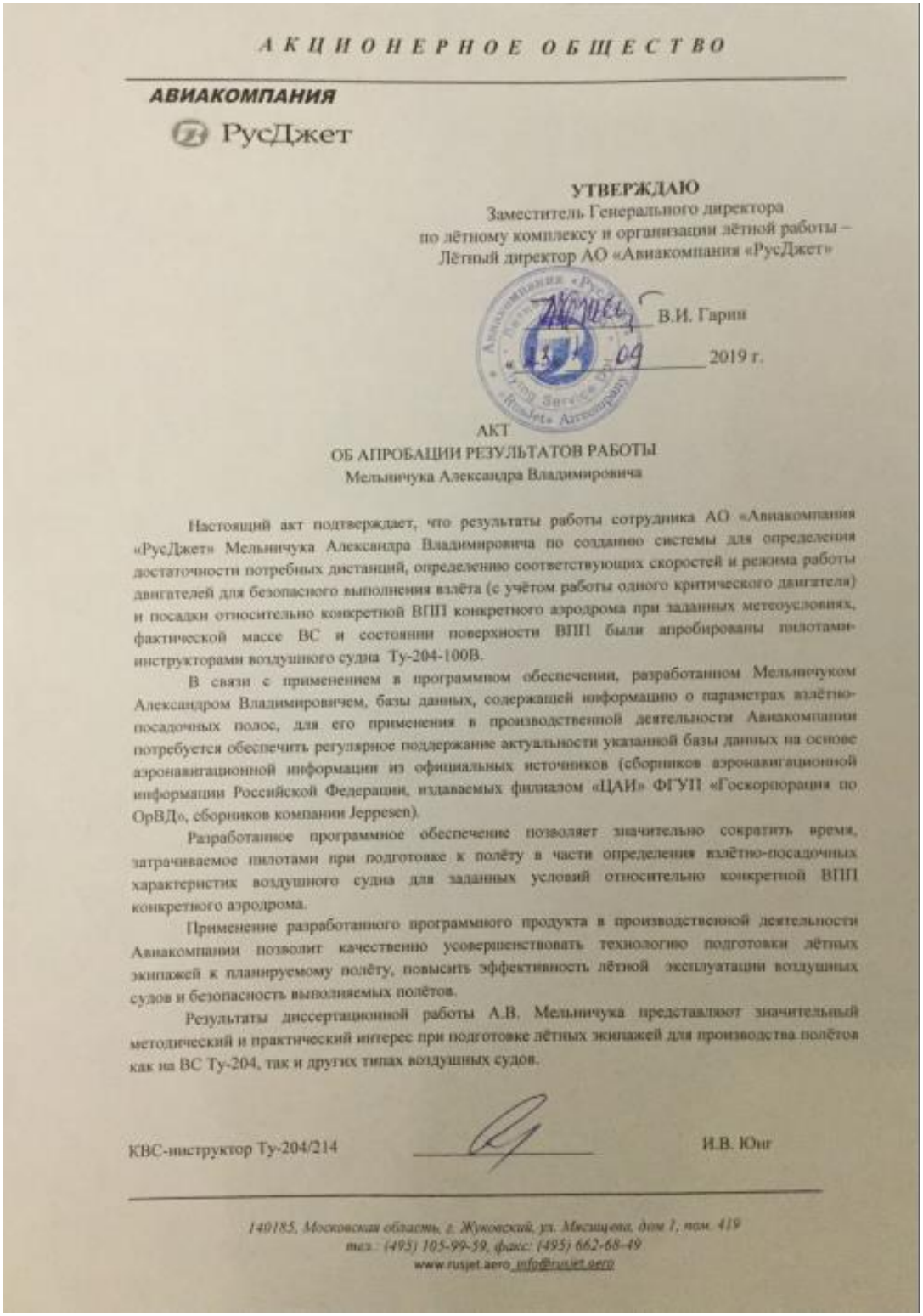
Автор: ***Мельничук Александр Владимирович (RU)***

Заявка № **2019660978**
Дата поступления **05 сентября 2019 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ ***12 сентября 2019 г.***

Руководитель Федеральной службы по интеллектуальной собственности

Г.П. Ивлиев

Приложение 2. Акт об апробации результатов работы



АКЦИОНЕРНОЕ ОБЩЕСТВО

АВИАКОМПАНИЯ

РусДжет

УТВЕРЖДАЮ
Заместитель Генерального директора
по лётному комплексу и организации лётной работы –
Лётный директор АО «Авиакомпания «РусДжет»

В.И. Гарин

«15» 04 2019 г.

АКТ
ОБ АПРОБАЦИИ РЕЗУЛЬТАТОВ РАБОТЫ
Мельничука Александра Владимировича

Настоящий акт подтверждает, что результаты работы сотрудника АО «Авиакомпания «РусДжет» Мельничука Александра Владимировича по созданию системы для определения достаточности потребных дистанций, определению соответствующих скоростей и режима работы двигателей для безопасного выполнения взлёта (с учётом работы одного критического двигателя) и посадки относительно конкретной ВПП конкретного аэродрома при заданных метеоусловиях, фактической массе ВС и состоянии поверхности ВПП были апробированы пилотами-инструкторами воздушного судна Ту-204-100В.

В связи с применением в программном обеспечении, разработанном Мельничуком Александром Владимировичем, базы данных, содержащей информацию о параметрах взлётно-посадочных полос, для его применения в производственной деятельности Авиакомпании потребуется обеспечить регулярное поддержание актуальности указанной базы данных на основе аэронавигационной информации из официальных источников (сборников аэронавигационной информации Российской Федерации, издаваемых филиалом «ЦАИ» ФГУП «Госкорпорация по ОрВД», сборников компании Jeppesen).

Разработанное программное обеспечение позволяет значительно сократить время, затрачиваемое пилотами при подготовке к полёту в части определения взлётно-посадочных характеристик воздушного судна для заданных условий относительно конкретной ВПП конкретного аэродрома.

Применение разработанного программного продукта в производственной деятельности Авиакомпании позволит качественно усовершенствовать технологию подготовки лётных экипажей к планируемому полёту, повысить эффективность лётной эксплуатации воздушных судов и безопасность выполняемых полётов.

Результаты диссертационной работы А.В. Мельничука представляют значительный методический и практический интерес при подготовке лётных экипажей для производства полётов как на ВС Ту-204, так и других типах воздушных судов.

КВС-инструктор Ту-204/214 И.В. Юнг

140185, Московская область, г. Жуковский, ул. Мясищева, дом 1, пом. 419
тел.: (495) 105-99-59, факс: (495) 662-68-49
www.rusjet.aero info@rusjet.aero

Приложение 3. Акт о внедрении результатов работы в учебный процесс

«УТВЕРЖДАЮ»

Проректор по учебной работе федерального государственного бюджетного образовательного учреждения высшего образования «Московский авиационный институт (национальный исследовательский университет)» (МАИ)

_____ Д.А. Конорез

_____ 2020 г.

АКТ

о внедрении результатов диссертационной работы

Мельничука Александра Владимировича

на тему «Разработка информационной системы для расчета взлетно-посадочных характеристик воздушных судов на базе электронного планшета пилота»

в учебный процесс МАИ

Комиссия кафедры «Математическая кибернетика» в составе: заведующий кафедрой, д.ф.-м.н., профессор А.В. Пантелеев (председатель комиссии), д.т.н., профессор В.А. Судаков (член комиссии), к.ф.-м.н., доцент К.А. Рыбаков (член комиссии), к.ф.-м.н. А.С. Алексейчук (член комиссии), составила настоящий акт о внедрении нижеперечисленных результатов диссертационной работы Мельничука Александра Владимировича на тему «Разработка информационной системы для расчета взлетно-посадочных характеристик воздушных судов на базе электронного планшета пилота», представленной на соискание ученой степени кандидата технических наук по специальности 05.13.01 – «Системный анализ, управление и обработка информации» (авиационная и ракетно-космическая техника):

1. Методика создания информационных систем расчета взлетно-посадочных характеристик на базе электронного полётного планшета, включающая:
   1.1. архитектуру информационной системы расчета взлетно-посадочных характеристик воздушных судов с применением технологии продукционной экспертной системы,
   1.2. онтологию информационного обеспечения для расчета взлетно-посадочных характеристик, позволившую сформировать структуру и атрибуты базы данных,
   1.3. алгоритмы расчетов зависимостей ВПХ по номограммам, представленным в РЛЭ воздушного судна.
2. Методы и алгоритмы выбора аппаратного обеспечения информационной системы расчета взлетно-посадочных характеристик воздушных судов, используемого в кабине лётного экипажа в формате планшетного компьютера.
3. Программное обеспечение информационной системы в виде клиент-серверного приложения для электронного полётного планшета.

в учебный процесс федерального государственного бюджетного образовательного учреждения высшего образования «Московский авиационный институт (национальный исследовательский университет)» при проведении лекционных и практических занятий на кафедре «Математическая кибернетика» по курсу «Анализ данных и системы поддержки принятия решения».

Председатель комиссии: д.ф.-м.н., проф. А.В. Пантелеев

Члены комиссии: д.т.н., доц. В.А. Судаков

к.ф.-м.н., доц. К.А. Рыбаков

к.ф.-м.н. А.С. Алексейчук

СОГЛАСОВАНО

Директор Дирекции Института № 8 «Информационные технологии и прикладная математика» к.ф.-м.н., доцент С.С. Крылов

Приложение 4. Исходный текст основных модулей программы

FirstViewController.swift

```
import UIKit
import Realm
import RealmSwift

var brit = Bool()
var inpapname = String()
var inprwname = String()
var aircraftname = String()
var headwind = Double()
var crosswind = Double()
var crosswind_max = Double()
var qfe = Double()
var fin_m_vzl_max = Double()
var vzl_rezh = Double()
var n2 = Double()
var v1 = Double()
var v_pst = Double()
var v_2 = Double()
var grad_blok21 = Double()
var v3 = Double()
var v4 = Double()
var rwconditiontext = String()
var v_y = Double()
var tora_full_length = Double()
var tora_available_run_length = Double()

class FirstViewController: UIViewController {

    @IBOutlet weak var toraTextField: UITextField!
    @IBOutlet weak var todaTextField: UITextField!
    @IBOutlet weak var asdaTextField: UITextField!
    @IBOutlet weak var elevTextField: UITextField!
    @IBOutlet weak var slopeTextField: UITextField!
    @IBOutlet weak var hdgTextField: UITextField!
    @IBOutlet weak var rwconditionSegmented: UISegmentedControl!
    @IBOutlet weak var kscepTextField: UITextField!
    @IBOutlet weak var oatTextField: UITextField!
    @IBOutlet weak var qnhTextField: UITextField!
    @IBOutlet weak var windspeedTextField: UITextField!
```

```
    @IBOutlet weak var winddirTextField: UITextField!
    @IBOutlet weak var mTextField: UITextField! // масса
    @IBOutlet weak var calculationsresult: UILabel!
    @IBOutlet weak var toraLabel: UILabel!
    @IBOutlet weak var todaLabel: UILabel!
    @IBOutlet weak var calcButton: UIButton!

    @IBAction func dnldrwdataTapped(_ sender: UIButton) {
            let serializer = JSONSerializer()
            serializer.serialize(input: "Airports")

        print (Realm.Configuration.defaultConfiguration.fileURL)

        let dbalert = UIAlertController(title: "Справка", message: "База
данных аэродромов обновлена", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        dbalert.addAction(okbutton)
        self.present(dbalert, animated: true, completion: nil)
    }

    @IBAction func rwListTapped(_ sender: UIButton) {

        let uirealm = try! Realm()
        var runways = uirealm.objects(Airport.self)

        var elementscount = runways.count

        var dedicrw = Airport()

        let rwlist = UIAlertController(title: "Выберите ВПП", message:nil,
preferredStyle: .actionSheet)

        for dedicrw in runways {
            var rwname: String = dedicrw.adrwy
            var rwItem = UIAlertAction(title: rwname, style: .default,
handler: {action in
                inpapname = dedicrw.name
                inprwname = dedicrw.rwy
                tora_full_length = dedicrw.tora_full_length
                tora_available_run_length =
dedicrw.tora_available_run_length
```

```
                sender.setTitle(rwname, for: .normal)
                sender.setTitleColor(.green, for: .normal)
                self.toraTextField.text =
String(dedicrw.tora_available_run_length)
                self.todaTextField.text =
String(dedicrw.toda_available_takeoff_distance)
                self.asdaTextField.text =
String(dedicrw.asda_accelerate_stop_distance)
                self.elevTextField.text = String(dedicrw.elev)
                self.slopeTextField.text = String(dedicrw.runway_slope)
                self.hdgTextField.text = String(dedicrw.hdg)
                brit = dedicrw.brit
                })

            rwlist.addAction(rwItem)
        }

        if let ppc = rwlist.popoverPresentationController {
            ppc.sourceView = sender
            ppc.sourceRect = sender.bounds
        }

        present(rwlist, animated: true, completion: nil)
    }

    @IBAction func acselTapped(_ sender: UIButton) {

        let aircraftlist = UIAlertController(title: "Выберите ВС",
message:nil, preferredStyle: .actionSheet)

        let aircraftname1: String = "Ту-204-100В"
        let aircraftname2: String = "МС-21"
        let aircraftname3: String = "Sukhoi Superjet 100"
        let aircraftname4: String = "Ил-96-300"
        let aircraftname5: String = "Як-42Д"


        let aircraftItem1 = UIAlertAction(title: aircraftname1, style:
.default, handler: {action in
            sender.setTitle(aircraftname1, for: .normal)
            sender.setTitleColor(.green, for: .normal)
```

```
            self.calcButton.isHidden = false
            aircraftname = aircraftname1
        })

        let aircraftItem2 = UIAlertAction(title: aircraftname2, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "OK", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname2, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calcButton.isHidden = true
            self.calculationsresult.text = ""
            aircraftname = aircraftname2
        })

        let aircraftItem3 = UIAlertAction(title: aircraftname3, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "OK", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname3, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calcButton.isHidden = true
            self.calculationsresult.text = ""
            aircraftname = aircraftname3
        })

        let aircraftItem4 = UIAlertAction(title: aircraftname4, style:
.default, handler: {action in
```

```
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "ОК", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname4, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calcButton.isHidden = true
            self.calculationsresult.text = ""
            aircraftname = aircraftname4
        })

        let aircraftItem5 = UIAlertAction(title: aircraftname5, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "ОК", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname5, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calcButton.isHidden = true
            self.calculationsresult.text = ""
            aircraftname = aircraftname5
        })

        aircraftlist.addAction(aircraftItem1)
        aircraftlist.addAction(aircraftItem2)
        aircraftlist.addAction(aircraftItem3)
        aircraftlist.addAction(aircraftItem4)
        aircraftlist.addAction(aircraftItem5)

        if let ppc = aircraftlist.popoverPresentationController {
            ppc.sourceView = sender
            ppc.sourceRect = sender.bounds
        }
```

```
        present(aircraftlist, animated: true, completion: nil)

    }

    @IBAction func metarTapped(_ sender: UIButton) {

        let metaralert = UIAlertController(title: "Предупреждение", message:
"Не подключен источник данных. Введите инфомацию о погоде вручную",
preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        metaralert.addAction(okbutton)
        self.present(metaralert, animated: true, completion: nil)
    }

    @IBAction func calcTapped(_ sender: UIButton) {

        var tora: Double = Double(toraTextField.text!)!
        var toda: Double = Double(todaTextField.text!)!
        var asda: Double = Double(asdaTextField.text!)!
        var elev: Double = Double(elevTextField.text!)!
        var slope: Double = Double(slopeTextField.text!)!
        var hdg: Double = Double(hdgTextField.text!)! // heading of the
runway
        print ("Проверка brit:", brit)

        let m_max: Double = 105000

        //ввод данных
        var rwCondition: Int = 1//состояние ВПП (1 - сух, 2 - сляк, 3 -
снег)
        let kBR: Double = Double(kscepTextField.text!)! //коэффициент
сцепления
        let m_vzl: Double = Double(mTextField.text!)! //взлетная масса f18
на loadsheet
        let t: Double = Double(oatTextField.text!)!
        let qnh: Double = Double(qnhTextField.text!)!
        let m_vlz_fakt: Double = m_vzl
        let wind_dir: Double = Double(winddirTextField.text!)! //направление
ветра
```

```
        let wind_speed: Double = Double(windspeedTextField.text!)!
//скорость ветра
        if rwconditionSegmented.selectedSegmentIndex == 0 {
            rwCondition = 1
            rwconditiontext = "Слой осадков отсутствует"
        } else if rwconditionSegmented.selectedSegmentIndex == 1 {
            rwCondition = 2
            rwconditiontext = "Слякоть / 3-12 мм Мокрый снег"
        } else if rwconditionSegmented.selectedSegmentIndex == 2 {
            rwCondition = 3
            rwconditiontext = "Снег 10-50 мм сухой"
        }

        var hPa: Double

        var sub_angle: Double
        var sub_angle_rad: Double
        // - давление на аэродроме
        hPa = -4.6708 * pow(10,(-7)) * pow(elev, 2) + 0.0363 * elev + 0.1842
        if brit == true {
            qfe = qnh - hPa
        } else {
            qfe = qnh - hPa * 3.2808
        }
        // ветер
        let pi = 3.14159
        sub_angle = wind_dir - hdg
        sub_angle_rad = sub_angle *  pi / 180
        crosswind = sin(sub_angle_rad) * wind_speed
        headwind = cos(sub_angle_rad) * wind_speed

        //БЛОК 1 Скорректированная располагаемая дистанция продолженного
взлета (ЧР)
        var tora_z: Double
        var tora_b4: Double
        var tora_d2: Double
        var min: Double
        var min_e4: Double
        var min_f4: Double
        var min_vspom: Double
        tora_z = tora - 50
        if rwCondition == 2 || rwCondition == 3 {
```

```
            tora_b4 = tora_z * 0.83333333333
        } else {
            tora_b4 = tora_z
        }
        if tora_b4 <= 1800 {
            tora_d2 = 1.125 * tora_b4 + 25
        } else if (tora_b4 > 1800) && (tora_b4 <= 3400) {
            tora_d2 = 1.0625 * tora_b4 + 137.5
        } else {
            tora_d2 = 1.03 * tora_b4 + 248
        }
        // - по уклону (начало)
        if toda < tora_d2 {
            min = toda
        } else {
            min = tora_d2
        }
        if slope > 0 {
            min_e4 = ((0.841667 * min + 53.3333) - min) * (0.5 * slope) +
min
        } else {
            min_vspom = (1.16667 * min - 66.6667) - min
            min_e4 = min_vspom * ( -0.5 * slope) + min
        }

        if headwind < 0 {
            min_f4 = (( 0.895833 * min_e4 - 83.3333) - min_e4) * ( -0.2 *
headwind) + min_e4
        } else {
            min_f4 = ((1.225 * min_e4 + 100) - min_e4) * (0.05 * headwind) +
min_e4
        }

        //БЛОК 2 Располагаемая дистанция прерванного взлета (РДПВ) с учетом
слоя осадков на ВПП
        var  asda_bez_osad: Double
        var asda_slak: Double
        var asda_snow: Double
        var asda_final_d3: Double = 0
        if rwCondition == 1 {
            asda_final_d3 = asda - 50 //без осадков
        } else if rwCondition == 2 {
```

```
            asda_final_d3 = (asda - 50)*0.584333 //Слякоть 3-12мм Мокрый
снег
        } else {
            asda_final_d3 = (asda - 50)*0.622333 //Снег 10-50мм сухой
        }

        //БЛОК 3 Скорректированная располагаемая дистанция прерванного
взлета (РДПВ)
        var  kRWCondition: Double
        var asda_final_a4: Double
        var vspom: Double
        var vspom2: Double
        var asda_korr_b4: Double = 0
        if rwCondition == 2 || rwCondition == 3 {
           kRWCondition = 0.57
        }
        if rwCondition == 1 && kBR > 0.57 {
            kRWCondition = 0.57
        } else {
            kRWCondition = kBR
        }
        asda_final_a4 = asda_final_d3 - (1 - ( -0.00000228938 *
asda_final_d3 + 0.628663)) * asda_final_d3 * ( -3.7036 * kRWCondition +
2.1111)
        if headwind < 0 {
            asda_korr_b4 = asda_final_a4 - (( 1 - (0.0000151515 *
asda_final_a4 + 0.639394)) * ((-headwind/5)) * asda_final_a4)
        } else {
            vspom = ( -0.0000127093 * asda_final_a4 + 1.36408) - 1;
            vspom2 = vspom * (headwind / 20) + 1;
            asda_korr_b4 = vspom2 * asda_final_a4
        }

        //БЛОК 4 Приведенная взлетная масса и относительная скорость
принятия решения V1/V п.ст в зависимости от скорректированных располагаемых
дистанций
        var v1_vr: Double
        var pre_m_vzl_b4: Double
        var priv_m_vzl_b5: Double
        if ((0.00022222 * asda_korr_b4 + 0.6067) - ( -0.000000026626 *
pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553)) * (0.00000012302
```

```
* pow(min_f4, 2) - 0.0010766 * min_f4 + 2.3381) + ( -0.000000026626 *
pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553) > 1 {
            v1_vr = 1
        } else {
            v1_vr = ((0.00022222 * asda_korr_b4 + 0.6067) - ( -
0.000000026626 * pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553))
* (0.00000012302 * pow(min_f4, 2) - 0.0010766 * min_f4 + 2.3381) + ( -
0.000000026626 * pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553)
        }
        pre_m_vzl_b4 = (((0.0124 * min_f4 + 92.1754 ) - (0.000000000839986 *
pow(min_f4, 3) - 0.00000820431 * pow(min_f4, 2) + 0.0386714 * min_f4 +
46.8137)) * (0.00058823 * asda_korr_b4 - 0.8823) + (0.000000000839986 *
pow(min_f4, 3) - 0.00000820431 * pow(min_f4, 2) + 0.0386714 * min_f4 +
46.8137)) * 1000
        if pre_m_vzl_b4 > 126000 {
            priv_m_vzl_b5 = 126000
        } else {
            priv_m_vzl_b5 = pre_m_vzl_b4
        }

        // БЛОК 5 Скорости на взлете с закрылками 18 (предкрылки 19)
        v_pst = 1.2001 * (m_vzl / 1000)  + 118.9931 // Vп.ст.
        v_2 = v_pst + 20


        // БЛОК 6 гПа ---> мм.рт.ст
        var mm_rt_st: Double
        var height_c2: Double
        mm_rt_st = qfe * 0.75
        height_c2 = (0.0078727 * pow(mm_rt_st, 2) - 22.9056 * mm_rt_st +
12861.4077)


        // БЛОК 7 Взлетная масса самолета в зависимости от приведенной массы
и условий на аэродроме
        var m_b8: Double
        var m_b8_vspom: Double
        var m_b8_vspom2: Double
        var m_vzl_b10: Double
        if t<0 {
            m_b8_vspom2 = -0.0000001 * pow(height_c2, 3) + 0.00155 *
pow(height_c2, 2) + 6.95 * height_c2 + 86200
```

```
            m_b8_vspom = m_b8_vspom2 - (0.000000283333 * pow(height_c2, 3) -
0.0008 * pow(height_c2, 2) + 10.7167 * height_c2 + 73400)
            m_b8 = m_b8_vspom * (0.02 * t + 1) + (0.000000283333 *
pow(height_c2, 3) - 0.0008 * pow(height_c2, 2) + 10.7167 * height_c2 +
73400)
        } else if (t > (31 - 0.007 * height_c2)) && (height_c2 >= 2000) &&
(height_c2 <= 3000) {
            m_b8 = ((18.5 * height_c2 + 104500) - (17.5 * height_c2 +
67500)) * (0.025 * t - 0.25) + (17.5 * height_c2 + 67500)
        } else if t>(31 - 0.007 * height_c2) && height_c2 >= 1000 &&
height_c2 <= 2000 {
            m_b8 = ((23 * height_c2 + 95500) - (15 * height_c2 + 79000)) *
(0.030303 * t - 0.515152) + (15 * height_c2 + 79000)
        } else if t > (36 - 0.012 * height_c2) && height_c2 >= 0 &&
height_c2 <= 1000 {
            m_b8 = ((11.7 * height_c2 + 107000) - (17 * height_c2 + 82000))
* (0.0384615 * t - 0.923077) + (17 * height_c2 + 82000)
        } else {
            m_b8 = ((-0.0000009 * pow(height_c2, 3) + 0.0053 *
pow(height_c2, 2) + 2.6 * height_c2 + 96800)-(-0.0000001 * pow(height_c2, 3)
+ 0.00155 * pow(height_c2, 2) + 6.95 * height_c2 + 86200)) * 0.02 * t + (-
0.0000001 * pow(height_c2, 3) + 0.00155 * pow(height_c2, 2) + 6.95 *
height_c2 + 86200)
        }
        if (priv_m_vzl_b5 - m_b8) > 0 {
            m_vzl_b10 = 110000 - ((0.00000375 * pow(m_b8, 2) + 0.425 * m_b8
+ 42000) - priv_m_vzl_b5) / ((0.00000375 * pow(m_b8, 2) + 0.425 * m_b8 +
42000) - m_b8) * 20000
        } else {
            m_vzl_b10 = ((0.000000118519 * pow(m_b8, 2) + 0.802519 * m_b8 -
2857.04) - priv_m_vzl_b5) / ((0.000000118519 * pow(m_b8, 2) + 0.802519 *
m_b8 - 2857.04) - m_b8) * 20000 + 70000
        }


         // БЛОК 8 Взлетная масса самолета в зависимости от МАХ приведенной
массы 114 т. и условий на аэродроме
        var m_b8_blok_8: Double
        var m_b9_blok_8: Double = 114000
        var m_b10_blok8: Double
        if t < 0 {
```

```
            m_b8_blok_8 = ((-0.0000001 * pow(height_c2, 3) + 0.00155 *
pow(height_c2, 2) + 6.95 * height_c2 + 86200) - (0.000000283333 *
pow(height_c2, 3) - 0.0008 * pow(height_c2, 2) + 10.7167 * height_c2 +
73400)) * (0.02 * t + 1) + (0.000000283333 * pow(height_c2, 3) - 0.0008 *
pow(height_c2, 2) + 10.7167 * height_c2 + 73400)
        } else if t > (31 - 0.007 * height_c2) && height_c2 >= 2000 &&
height_c2<=3000 {
            m_b8_blok_8 = ((18.5 * height_c2 + 104500) - (17.5 * height_c2 +
67500)) * (0.025 * t - 0.25) + (17.5 * height_c2 + 67500)
        } else if t > (31 - 0.007 * height_c2) && height_c2 >= 1000 &&
height_c2 <= 2000 {
            m_b8_blok_8 = ((23 * height_c2 + 95500) - (15 * height_c2 +
79000)) * (0.030303 * t - 0.515152) + (15 * height_c2 + 79000)
        } else if t > (36 - 0.012 * height_c2) && height_c2 >= 0 &&
height_c2 <= 1000 {
            m_b8_blok_8 = ((11.7 * height_c2 + 107000) - (17 * height_c2 +
82000)) * (0.0384615 * t - 0.923077) + (17 * height_c2 + 82000)
        } else {
            m_b8_blok_8 = ((-0.0000009 * pow(height_c2, 3) + 0.0053 *
pow(height_c2, 2) + 2.6 * height_c2 + 96800)-(-0.0000001 * pow(height_c2, 3)
+ 0.00155 * pow(height_c2, 2) + 6.95 * height_c2 + 86200)) * 0.02 * t + (-
0.0000001 * pow(height_c2, 3) + 0.00155 * pow(height_c2, 2) + 6.95 *
height_c2 + 86200)
        }
        if (m_b9_blok_8 - m_b8_blok_8) > 0 {
            m_b10_blok8 = 110000 - ((0.00000375 * pow(m_b8_blok_8, 2) +
0.425 * m_b8_blok_8 + 42000) - m_b9_blok_8) / ((0.00000375 *
pow(m_b8_blok_8, 2) + 0.425 * m_b8_blok_8 + 42000) - m_b8_blok_8) * 20000
        } else {
            m_b10_blok8 = ((0.000000118519 * pow(m_b8_blok_8, 2) + 0.802519
* m_b8_blok_8 - 2857.04) - m_b9_blok_8) / ((0.000000118519 *
pow(m_b8_blok_8, 2) + 0.802519 * m_b8_blok_8 - 2857.04) - m_b8_blok_8) *
20000 + 70000
        }

        // БЛОК 9 Взлетная масса, ограниченная градиентом набора высоты с
одним неработающим двигателем
        var m_b9_blok9: Double
        if t <= 10 && height_c2 >= 2000 {
            m_b9_blok9 = (120750 - 9.875 * height_c2)
        } else if t <= 15 && height_c2 >= 1539 && height_c2 <= 2000 {
            m_b9_blok9 = (118061 - 8.48485 * height_c2)
```

```
        } else if (3200 - 60 * t) > height_c2 && t <= 35 && t >= 30 {
            m_b9_blok9 = ((116364 - 12.7273 * height_c2) - (113000 - 13.75 *
height_c2)) * (7 - 0.2 * t) + (113000 - 13.75 * height_c2)
        } else if (3200 - 60 * t) > height_c2 && t <= 30 && t >= 25 {
            m_b9_blok9 = ((124462 - 15.3846 * height_c2) - (116364 - 12.7273
* height_c2)) * (6 - 0.2 * t) + (116364 - 12.7273 * height_c2)
        } else if (2200 - 20 * t) > height_c2 && t <= 25 && t >= 20 {
            m_b9_blok9 = ((119430 - 10.596 * height_c2) - (124462 - 15.3846
* height_c2)) * (5 - 0.2 * t) + (124462 - 15.3846 * height_c2)
        } else if (2600 - 40 * t) > height_c2 && t <= 20 && t >= 15 {
            m_b9_blok9 = ((118061 - 8.48485 * height_c2) - (119430 - 10.596
* height_c2)) * (4 - 0.2 * t) + (119430 - 10.596 * height_c2)
        } else if t >= 10 && t <= 15 {
            m_b9_blok9 = ((120750 - 9.875 * height_c2) - (126000 - 12.5 *
height_c2)) * (3 - 0.2 * t) + (126000 - 12.5 * height_c2)
        } else if t >= 15 && t <= 20 {
            m_b9_blok9 = ((126000 - 12.5 * height_c2) - (123429 - 12.8571 *
height_c2)) * (4 - 0.2 * t) + (123429 - 12.8571 * height_c2)
        } else if t >= 20 && t <= 25 {
            m_b9_blok9 = ((123429 - 12.8571 * height_c2) - (120726 - 13.242
* height_c2)) * (5 - 0.2 * t) + (120726 - 13.242 * height_c2)
        } else if t >= 25 && t <= 30 {
            m_b9_blok9 = ((120726 - 13.242 * height_c2) - (118457 - 14.3571
* height_c2)) * (6 - 0.2 * t) + (118457 - 14.3571 * height_c2)
        } else if t >= 30 && t <= 35 {
            m_b9_blok9 = ((118457 - 14.3571 * height_c2) - (113000 - 13.75 *
height_c2)) * (7 - 0.2 * t) + (113000 - 13.75 * height_c2)
        } else {
            m_b9_blok9 = ((113000 - 13.75 * height_c2) - (107000 - 12.7778 *
height_c2)) * (8 - 0.2 * t) + (107000 - 12.7778 * height_c2)
        }

        // БЛОК 10 Скорректированная располагаемая дистанция нормального
взлета. (РДВ)(L1)
        var a4_blok10: Double
        var toda_b4_blok10: Double
        var toda_a2: Double
        toda_a2 = toda - 50
        if slope > 0 {
            a4_blok10 = (((0.0000118829 * toda_a2 + 0.1046) * slope / (-2))
+ 1) * toda_a2
        } else {
```

```
            a4_blok10 = (((0.000014323 * toda_a2 + 0.1052) * slope / (-2)) +
1) * toda_a2
        }
        if headwind < 0 {
            toda_b4_blok10 = (1 - (0.000013021 * a4_blok10 + 0.8229)) *
(headwind / 5) * a4_blok10 + a4_blok10
        } else {
            toda_b4_blok10 = ((-0.0000065099 * a4_blok10 + 0.276) * headwind
/ 20 + 1) * a4_blok10
        }

        // БЛОК 11 Взлетная масса, ограниченная взлетной дистанцией при
нормальном взлете (РДВ)
        var s_blok_10: Double //подразумевается блок 11
        var izlom_t: Double
        var alfa_rud_b13_blok11: Double = 73
        var toda_b14_blok_11: Double
        var s_b16_blok11: Double // S'
        var k_vert_b17_blok11: Double
        var k_1000_b18_blok11: Double
        var b19_blok11: Double
        var k_2000_b20_blok11: Double
        var b21_blok11: Double
        var k_okon_b22_blok11: Double
        var b23_blok11: Double
        var b24_blok11: Double
        var m_vzl_b15_blok11: Double
        izlom_t = (-0.000000000166667 * pow(height_c2, 3) + 0.0000005 *
pow(height_c2, 2) + 30 - 0.00633333 * height_c2)
        if height_c2 <= 3000 && height_c2 > 2000 && t > izlom_t {
            s_blok_10 = ((0.7 * height_c2 + 1350) - (0.45 * height_c2 +
350)) * (0.02 * t) + (0.45 * height_c2 + 350)
        } else if height_c2 <= 3000 && height_c2 > 2000 && t <= izlom_t && t
> 0 {
            s_blok_10 = ((0.4 * height_c2 + 1100) - (0.43 * height_c2 +
760)) * (0.02 * t) + (0.43 * height_c2 + 760)
        } else if height_c2 <= 3000 && height_c2 > 2000 && t <= 0 {
            s_blok_10 = ((0.5 * height_c2 + 900) - (0.36 * height_c2 + 620))
* (0.01 * t + 0.5) + (0.36 * height_c2 + 620)
        } else if height_c2 <= 2000 && height_c2 > 1000 && t > izlom_t {
            s_blok_10 = ((0.62 * height_c2 + 1510) - (0.45 * height_c2 +
350)) * (0.02 * t) + (0.45 * height_c2 + 350)
```

```
        } else if height_c2 <= 2000 && height_c2 > 1000 && t <= izlom_t {
            s_blok_10 = ((0.34 * height_c2 + 1220) - (0.28 * height_c2 +
780)) * (0.01 * t + 0.5) + (0.28 * height_c2 + 780)
        } else if height_c2 <= 1000 && t > izlom_t {
            s_blok_10 = ((0.53 * height_c2 + 1600) - (0.15 * height_c2 +
650)) * (0.02 * t) + (0.15 * height_c2 + 650)
        } else {
            s_blok_10 = ((0.27 * height_c2 + 1290) - (0.1 * height_c2 +
960)) * (0.01 * t + 0.5) + (0.1 * height_c2 + 960)
        }
        toda_b14_blok_11 = (toda_b4_blok10 - (0.820513 * toda_b4_blok10 -
189.744)) * (0.0769231 * alfa_rud_b13_blok11 - 4.61538) + (0.820513 *
toda_b4_blok10 - 189.744)
        s_b16_blok11 = (1.75 * s_blok_10 - 50)
        if (toda_b14_blok_11 - s_blok_10) / (s_b16_blok11 - s_blok_10) > 1 {
            k_vert_b17_blok11 = 1
        } else {
            k_vert_b17_blok11 = (toda_b14_blok_11 - s_blok_10) /
(s_b16_blok11 - s_blok_10)
        }
        if (-0.291651 * pow(k_vert_b17_blok11, 2) + 1.29165 *
k_vert_b17_blok11) < 0 {
            k_1000_b18_blok11 = 0
        } else if (-0.291651 * pow(k_vert_b17_blok11, 2) + 1.29165 *
k_vert_b17_blok11) > 1 {
            k_1000_b18_blok11 = 1
        } else {
            k_1000_b18_blok11 = (-0.291651 * pow(k_vert_b17_blok11, 2) +
1.29165 * k_vert_b17_blok11)
        }
        b19_blok11 = k_1000_b18_blok11 * 30000 + 80000
        if (-0.530222 * pow(k_vert_b17_blok11, 2) + 1.53022 *
k_vert_b17_blok11) < 0 {
            k_2000_b20_blok11 = 0
        } else if (-0.530222 * pow(k_vert_b17_blok11, 2) + 1.53022 *
k_vert_b17_blok11) > 1 {
            k_2000_b20_blok11 = 1
        } else {
            k_2000_b20_blok11 = (-0.530222 * pow(k_vert_b17_blok11, 2) +
1.53022 * k_vert_b17_blok11)
        }
        b21_blok11 = k_2000_b20_blok11 * 30000 + 80000
```

```
        if (k_2000_b20_blok11 - k_1000_b18_blok11) * (0.000833333 *
s_blok_10 - 0.833333) + k_1000_b18_blok11 > 1 {
            k_okon_b22_blok11 = 1
        } else {
            k_okon_b22_blok11 = (k_2000_b20_blok11 - k_1000_b18_blok11) *
(0.000833333 * s_blok_10 - 0.833333) + k_1000_b18_blok11
        }
        b23_blok11 = k_okon_b22_blok11 * 30000 + 80000
        b24_blok11 = (0.000263889 * pow(s_blok_10, 2) + 0.997222 * s_blok_10
+ 428.889)
        m_vzl_b15_blok11 = k_okon_b22_blok11 * 30000 + 80000

        // БЛОК 12 Скорректированная располагаемая длина разбега нормального
взлета (РДР)
        var a4_blok12: Double // скорректированная по уклону
        var b4_blok12: Double // скорректированная по ветру
        if slope <= 0 {
            a4_blok12 = ((1.17292 * tora_z - 116.667) - tora_z) * (-0.5 *
slope) + tora_z
        } else {
            a4_blok12 = (tora_z - (0.829167 * tora_z + 43.3333)) * (1 - 0.5
* slope) + (0.829167 * tora_z + 43.3333)
        }
        if slope >= 0 {
            b4_blok12 = ((1.16364 * a4_blok12 + 105.455) - a4_blok12) *
(0.05 * headwind) + a4_blok12
        } else {
            b4_blok12 = (a4_blok12 - (0.875 * a4_blok12 - 150)) * (0.2 *
headwind + 1) + (0.875 * a4_blok12 - 150)
        }

        // БЛОК 13 Взлетная масса, ограниченная длиной разбега при
нормальном взлете(РДР)
        var izlom_t_blok13: Double
        var alfa_rud_b14_blok13: Double = 73
        var tora_b15_blok13: Double //РДР'
        var m_vzl_b16_blok13: Double
        var b17_blok13: Double // S'
        var k_vert_b18_blok13: Double
        var k_1000_b19_blok13: Double
        var s_blok12: Double
        var b20_blok13: Double
```

```
        var k_2200_b21_blok13: Double
        var b22_blok13: Double
        var k_okon_b23_blok13: Double
        izlom_t_blok13 = 30 - 0.00633333 * height_c2
        if height_c2 >= 2000 && t >= izlom_t_blok13 {
            s_blok12 = ((0.73 * height_c2 + 1030) - (0.39 * height_c2 +
290)) * (0.02 * t) + (0.39 * height_c2 + 290)
        } else if height_c2 < 2000 && height_c2 >= 1000 && t >=
izlom_t_blok13 {
            s_blok12 = ((0.63 * height_c2 + 1230) - (0.34 * height_c2 +
390)) * (0.02 * t) + (0.34 * height_c2 + 390)
        } else if height_c2 < 1000 && t >= izlom_t_blok13 {
            s_blok12 = ((0.39 * height_c2 + 1470) - (0.22 * height_c2 +
510)) * (0.02 * t) + (0.22 * height_c2 + 510)
        } else if height_c2 < 1000 && t < izlom_t_blok13 {
            s_blok12 = ((0.27 * height_c2 + 1120) - (0.08 * height_c2 +
880)) * (0.01 * t + 0.5) + (0.08 * height_c2 + 880)
        } else if height_c2 >= 1000 && height_c2 < 2000 && t >= 0 && t <
izlom_t_blok13 {
            s_blok12 = ((0.3 * height_c2 + 1100) - (0.3 * height_c2 + 875))
* (0.02 * t) + (0.3 * height_c2 + 875)
        } else if height_c2 >= 2000 && height_c2 < 3000 && t >= 0 && t <
izlom_t_blok13 {
            s_blok12 = ((0.325 * height_c2 + 1050) - (0.325 * height_c2 +
825)) * (0.02 * t) + (0.325 * height_c2 + 825)
        } else if height_c2 >= 1000 && height_c2 < 2000 && t < 0 {
            s_blok12 = ((0.3 * height_c2 + 875) - (0.19 * height_c2 + 770))
* (0.02 * t + 1) + (0.19 * height_c2 + 770)
        } else /*if height_c2 >= 2000 && t < 0 */ {
            s_blok12 = ((0.325 * height_c2 + 825) - (0.425 * height_c2 +
300)) * (0.02 * t + 1) + (0.425 * height_c2 + 300)
        }

        tora_b15_blok13 = (b4_blok12 - (0.820513 * b4_blok12 - 189.744)) *
(0.0769231 * alfa_rud_b14_blok13 - 4.61538) + (0.820513 * b4_blok12 -
189.744)

        b17_blok13 = (1.75 * s_blok12 - 50)
        if (tora_b15_blok13 - s_blok12) / (b17_blok13 - s_blok12) > 1 {
           k_vert_b18_blok13 = 1
        } else {
```

```
            k_vert_b18_blok13 = (tora_b15_blok13 - s_blok12) / (b17_blok13 -
s_blok12)
        }
        k_1000_b19_blok13 = (-0.291651 * pow(k_vert_b18_blok13, 2) + 1.29165
* k_vert_b18_blok13)
        b20_blok13 = k_1000_b19_blok13 * 30000 + 80000
        k_2200_b21_blok13 = (-0.530222 * pow(k_vert_b18_blok13, 2) + 1.53022
* k_vert_b18_blok13)
        b22_blok13 = k_2200_b21_blok13 * 30000 + 80000
        k_okon_b23_blok13 = (k_2200_b21_blok13 - k_1000_b19_blok13) *
(0.000833333 * s_blok12 - 0.833333) + k_1000_b19_blok13
        m_vzl_b16_blok13 = k_okon_b23_blok13 * 30000 + 80000

        // БЛОК 14 Приведенная масса самолета в зависимости от взлетной
массы и условий на аэродроме. ОБРАТНЫЙ РАСЧЕТ ФАКТ.
        var b8_blok14: Double
        var vspom_blok14: Double
        var vspom2_blok14: Double
        var m_priv_b10_blok14: Double
        if t<0 {
            b8_blok14 =  ((-0.0000001 * pow(height_c2, 3) + 0.00155 *
pow(height_c2, 2) + 6.95 * height_c2 + 86200) - (0.000000283333 *
pow(height_c2, 3) - 0.0008 * pow(height_c2, 2) + 10.7167 * height_c2 +
73400)) * (0.02 * t + 1) + (0.000000283333 * pow(height_c2, 3) - 0.0008 *
pow(height_c2, 2) + 10.7167 * height_c2 + 73400)
        } else if t > (31 - 0.007 * height_c2) && height_c2 >= 2000 &&
height_c2 <= 3000 {
            vspom_blok14 = (18.5 * height_c2 + 104500) - (17.5 * height_c2 +
67500)
            vspom2_blok14 = 17.5 * height_c2 + 67500
            b8_blok14 = vspom_blok14 * (0.025 * t - 0.25) + vspom2_blok14
        } else if t > (31 - 0.007 * height_c2) && height_c2 >= 1000 &&
height_c2 <= 2000 {
            b8_blok14 = ((23 * height_c2 + 95500) - (15 * height_c2 +
79000)) * (0.030303 * t - 0.515152) + (15 * height_c2 + 79000)
        } else if t > (36 - 0.012 * height_c2) && height_c2 >= 0 &&
height_c2 <= 1000 {
            b8_blok14 = ((11.7 * height_c2 + 107000) - (17 * height_c2 +
82000)) * (0.0384615 * t - 0.923077) + (17 * height_c2 + 82000)
        } else {
            b8_blok14 = ((-0.0000009 * pow(height_c2, 3) + 0.0053 *
pow(height_c2, 2) + 2.6 * height_c2 + 96800) - (-0.0000001 * pow(height_c2,
```

```
3) + 0.00155 * pow(height_c2, 2) + 6.95 * height_c2 + 86200)) * 0.02 * t +
(-0.0000001 * pow(height_c2, 3) + 0.00155 * pow(height_c2, 2) + 6.95 *
height_c2 + 86200)
        }
        if m_vlz_fakt > 90000 {
            m_priv_b10_blok14 = ((0.00000375 * pow(b8_blok14, 2) + 0.425 *
b8_blok14 + 42000) - b8_blok14) * (0.00005 * m_vlz_fakt - 4.5) + b8_blok14
        } else {
            m_priv_b10_blok14 = ((0.000000118519 * pow(b8_blok14, 2) +
0.802519 * b8_blok14 - 2857.04) - b8_blok14) * (4.5 - 0.00005 * m_vlz_fakt)
+ b8_blok14
        }


        // БЛОК 15 Обратный расчет. ПРОДОЛЖЕННЫЙ  ВЗЛЕТ
        var m_priv_b4_blok15: Double
        var b1_blok15: Double
        var v1_vr_blok15: Double
        m_priv_b4_blok15 = m_priv_b10_blok14 / 1000
        if ((80.7143 * m_priv_b4_blok15 - 7440) - (-0.00168398 *
pow(m_priv_b4_blok15, 4) + 0.701585 * pow(m_priv_b4_blok15, 3) - 108.508 *
pow(m_priv_b4_blok15, 2) + 7452.96 * m_priv_b4_blok15 - 191285)) *
(0.000588235 * asda_korr_b4 - 0.882353) + (-0.00168398 *
pow(m_priv_b4_blok15, 4) + 0.701585 * pow(m_priv_b4_blok15, 3) - 108.508 *
pow(m_priv_b4_blok15, 2) + 7452.96 * m_priv_b4_blok15 - 191285) < 1500 {
           b1_blok15 = 1500
        } else {
            b1_blok15 = ((80.7143 * m_priv_b4_blok15 - 7440) - (-0.00168398
* pow(m_priv_b4_blok15, 4) + 0.701585 * pow(m_priv_b4_blok15, 3) - 108.508 *
pow(m_priv_b4_blok15, 2) + 7452.96 * m_priv_b4_blok15 - 191285)) *
(0.000588235 * asda_korr_b4 - 0.882353) + (-0.00168398 *
pow(m_priv_b4_blok15, 4) + 0.701585 * pow(m_priv_b4_blok15, 3) - 108.508 *
pow(m_priv_b4_blok15, 2) + 7452.96 * m_priv_b4_blok15 - 191285)
        }
        if ((0.00022222 * asda_korr_b4 + 0.6067) - (-0.000000026626 *
pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553)) * (0.00000012302
* pow(b1_blok15, 2) - 0.0010766 * b1_blok15 + 2.3381) + (-0.000000026626 *
pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553) > 1 {
           v1_vr_blok15 = 1
        } else {
           v1_vr_blok15 = ((0.00022222 * asda_korr_b4 + 0.6067) - (-
0.000000026626 * pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553))
```

```
* (0.00000012302 * pow(b1_blok15, 2) - 0.0010766 * b1_blok15 + 2.3381) + (-
0.000000026626 * pow(asda_korr_b4, 2) + 0.00026641 * asda_korr_b4 + 0.3553)
        }

        // БЛОК 16 V1
        if v_pst * v1_vr_blok15 < 185 {
            v1 = 185
        } else {
            v1 = v_pst * v1_vr_blok15
        }


        // БЛОК 17 Скорость начала уборки механизации на взлете V3 (закр.18;
предкр.19) и скорость при полетной конфигурации V4
        var m_vzl_blok17: Double
        m_vzl_blok17 = m_vzl / 1000
        v3 = 1.8 * m_vzl_blok17 + 164.004
        v4 = 1.92 * m_vzl_blok17 + 181.3978

        // БЛОК 18 Приведенная масса самолета в зависимости от взлетной
массы и условий на аэродроме. ОБРАТНЫЙ РАСЧЕТ  МАКС.
        var b3_blok18: Double
        var b4_blok18: Double
        var b5_blok18: Double
        var b6_blok18: Double
        var b7_blok18: Double
        var m_blok18: Double

        var m_priv_blok18: Double
        var masses = [Double] ()
        b3_blok18 = ((-0.0000001 * pow(height_c2, 3) + 0.00155 *
pow(height_c2, 2) + 6.95 * height_c2 + 86200) - (0.000000283333 *
pow(height_c2, 3) - 0.0008 * pow(height_c2, 2) + 10.7167 * height_c2 +
73400)) * (0.02 * t + 1) + (0.000000283333 * pow(height_c2, 3) - 0.0008 *
pow(height_c2, 2) + 10.7167 * height_c2 + 73400)
        b4_blok18 = ((-0.0000009 * pow(height_c2, 3) + 0.0053 *
pow(height_c2, 2) + 2.6 * height_c2 + 96800) - (-0.0000001 * pow(height_c2,
3) + 0.00155 * pow(height_c2, 2) + 6.95 * height_c2 + 86200)) * 0.02 * t +
(-0.0000001 * pow(height_c2, 3) + 0.00155 * pow(height_c2, 2) + 6.95 *
height_c2 + 86200)
        b5_blok18 = ((18.5 * height_c2 + 104500) - (17.5 * height_c2 +
67500)) * (0.025 * t - 0.25) + (17.5 * height_c2 + 67500)
```

```
        b6_blok18 = ((23 * height_c2 + 95500) - (15 * height_c2 + 79000)) *
(0.030303 * t - 0.515152) + (15 * height_c2 + 79000)
        b7_blok18 = ((11.7 * height_c2 + 107000) - (17 * height_c2 + 82000))
* (0.0384615 * t - 0.923077) + (17 * height_c2 + 82000)
        if t<0 {
            m_blok18 = b3_blok18
        } else if t > (31 - 0.007 * height_c2) && height_c2 >= 2000 &&
height_c2 <= 3000 {
            m_blok18 = b5_blok18
        } else if t > (31 - 0.007 * height_c2) && height_c2 >= 1000 &&
height_c2 <= 2000 {
            m_blok18 = b6_blok18
        } else if t > (36 - 0.012 * height_c2) && height_c2 >= 0 &&
height_c2 <= 1000 {
            m_blok18 = b7_blok18
        } else {
            m_blok18 = b4_blok18
        }
        masses = [m_vzl_b10, m_b10_blok8, m_b9_blok9, m_vzl_b15_blok11,
m_vzl_b16_blok13, m_max]
        fin_m_vzl_max = masses.min()!

        if fin_m_vzl_max > 90000 {
            m_priv_blok18 = ((0.00000375 * pow(m_blok18, 2) + 0.425 *
m_blok18 + 42000) - m_blok18) * (0.00005 * fin_m_vzl_max - 4.5) + m_blok18
        } else {
            m_priv_blok18 = ((0.000000118519 * pow(m_blok18, 2) + 0.802519 *
m_blok18 - 2857.04) - m_blok18) * (4.5 - 0.00005 * fin_m_vzl_max) + m_blok18
        }


        // БЛОК 19 Взлетный режим работы двигателей в зависимости от
приведенной массы
        var m_b1_blok19: Double
        var ogr_grad_2_4: Double
        var b4_blok19: Double
        var uchit_2_4: Double
        var m_shrt_blok19: Double
        var k_vert_blok19: Double
        var k_goriz_blok19: Double
        var b10_blok19: Double
        var b11_blok19: Double
```

```
        if m_priv_blok18 >= 114000 {
            m_b1_blok19 = 114000
        } else {
            m_b1_blok19 = m_priv_blok18
        }
        if m_b1_blok19 <= 100000 {
            m_shrt_blok19 = m_b1_blok19 - 14000
        } else {
            m_shrt_blok19 = 1.2375 * m_b1_blok19 - 37750
        }
        k_vert_blok19 = ((m_b1_blok19 - m_priv_b10_blok14) / (m_b1_blok19 -
m_shrt_blok19))
        k_goriz_blok19 = (-0.0336392 * pow(k_vert_blok19, 2) + 1.03364 *
k_vert_blok19 + 0.00000000000000110502)
        b10_blok19 = (73 - 13 * k_vert_blok19)
        b11_blok19 = (73-13 * k_goriz_blok19)
        b4_blok19 = b10_blok19 - (b10_blok19 - b11_blok19) * 10
        ogr_grad_2_4 = (0.000722222 * m_priv_b10_blok14 - 9.26111)
        if b4_blok19 < ogr_grad_2_4 {
            uchit_2_4 = ogr_grad_2_4
        } else {
            uchit_2_4 = b4_blok19
        }
        if height_c2 > 2000 || kBR < 0.45 || rwCondition == 2 || rwCondition
== 3 {
            vzl_rezh = 73
        } else if  m_priv_b10_blok14 < 95000 || (m_b1_blok19 -
m_priv_b10_blok14) > 21700 || uchit_2_4 <= 60 {
            vzl_rezh = 60
        } else if uchit_2_4 <= 66 && uchit_2_4 > 60 {
            vzl_rezh = 66
        } else if uchit_2_4 > 66 {
            vzl_rezh = 73
        } else {
            vzl_rezh = 999999 // т.е. ошибка
        }


        // БЛОК 20 Распологаемый для взлета режим работы двигателей ПС-90А
(n2,%) в зависимости от Альфа РУД и температуры наружного воздуха

        var p_aer_blok20: Double
```

```
        var b3_blok20: Double
        var b4_blok20: Double
        var b5_blok20: Double
        var b6_blok20: Double
        var b7_blok20: Double
        var b8_blok20: Double
        var b9_blok20: Double
        var b10_blok20: Double
        var b11_blok20: Double
        var b12_blok20: Double
        var b13_blok20: Double
        var b14_blok20: Double
        var b15_blok20: Double
        var b16_blok20: Double
        var b17_blok20: Double
        var c3_blok20: Double
        var c4_blok20: Double
        var c5_blok20: Double
        var c6_blok20: Double
        var c7_blok20: Double
        var c8_blok20: Double
        var c9_blok20: Double
        var c10_blok20: Double
        var c11_blok20: Double
        var c12_blok20: Double
        var c13_blok20: Double
        var c14_blok20: Double
        var c15_blok20: Double
        var c16_blok20: Double
        var c17_blok20: Double
        var d3_blok20: Double
        var d8_blok20: Double
        var d13_blok20: Double
        var error: String
        if mm_rt_st > 760 {
            p_aer_blok20 = 760
        } else {
            p_aer_blok20 = mm_rt_st
        }
        b3_blok20 = -1.6456 * pow(10,(-4)) * pow(t, 2) + 0.1707 * t +
92.0308
```

```
        b4_blok20 = -3.2967 * pow(10, (-5)) * pow(t, 2) + 0.1663 * t +
92.3621
        b5_blok20 = -5.4945 * pow(10, (-6)) * pow(t, 2) + 0.1677 * t +
92.7569
        b6_blok20 = 2.1978 * pow(10, (-5)) * pow(t, 2) + 0.1678 * t +
93.0407
        b7_blok20 = -4.3956 * pow(10, (-5)) * pow(t, 2) + 0.167 * t +
93.3231
        b8_blok20 = -1.4846 * pow(10, (-4)) * pow(t, 2) + 0.1644 * t +
91.7824
        b9_blok20 = -9.1733 * pow(10, (-5)) * pow(t, 2) + 0.1628 * t +
91.7939
        b10_blok20 = 0.1649 * t + 92.1084
        b11_blok20 = 2.0776 * pow(10, (-10)) * pow(t, 2) + 0.165 * t +
92.3978
        b12_blok20 = 0.1654 * t + 92.6577
        b13_blok20 = -1.3305 * pow(10, (-4)) * pow(t, 2) + 0.1615 * t +
90.5267
        b14_blok20 = -7.7732 * pow(10, (-5)) * pow(t, 2) + 0.1604 * t +
90.5299
        b15_blok20 = 2.618 * pow(10, (-5)) * pow(t, 2) + 0.1625 * t +
90.8497
        b16_blok20 = -1.3736 * pow(10, (-5)) * pow(t, 2) + 0.1628 * t +
91.1496
        b17_blok20 = 2.5974 * pow(10, (-5)) * t + 0.1642 * t + 91.4248
        if b3_blok20 > 97.3 {
            c3_blok20 = 97.3
        } else {
            c3_blok20 = b3_blok20
        }
        if b4_blok20 > 97.2 {
            c4_blok20 = 97.2
        } else {
            c4_blok20 = b4_blok20
        }
        if b5_blok20 > 96.8 {
            c5_blok20 = 96.8
        } else {
            c5_blok20 = b5_blok20
        }
        if b6_blok20 > 96.5 {
            c6_blok20 = 96.5
```

```
} else {
    c6_blok20 = b6_blok20
}
if b7_blok20 > 96.1 {
    c7_blok20 = 96.1
} else {
    c7_blok20 = b7_blok20
}
if b8_blok20 > 96.6 {
    c8_blok20 = 96.6
} else {
    c8_blok20 = b8_blok20
}
if b9_blok20 > 96.5 {
    c9_blok20 = 96.5
} else {
    c9_blok20 = b9_blok20
}
if b10_blok20 > 96.2 {
    c10_blok20 = 96.2
} else {
    c10_blok20 = b10_blok20
}
if b11_blok20 > 95.9 {
    c11_blok20 = 95.9
} else {
    c11_blok20 = b11_blok20
}
if b12_blok20 > 95.6 {
    c12_blok20 = 95.6
} else {
    c12_blok20 = b12_blok20
}
if b13_blok20 > 95.2 {
    c13_blok20 = 95.2
} else {
    c13_blok20 = b13_blok20
}
if b14_blok20 > 95.2 {
    c14_blok20 = 95.2
} else {
    c14_blok20 = b14_blok20
```

```
        }
        if b15_blok20 > 94.9 {
            c15_blok20 = 94.9
        } else {
            c15_blok20 = b15_blok20
        }
        if b16_blok20 > 94.6 {
            c16_blok20 = 94.6
        } else {
            c16_blok20 = b16_blok20
        }
        if b17_blok20 > 94.3 {
            c17_blok20 = 94.3
        } else {
            c17_blok20 = b17_blok20
        }
        if p_aer_blok20 > 760 {
            d3_blok20 = 84
        } else if p_aer_blok20 <= 760 && p_aer_blok20 >= 730 {
            d3_blok20 = c3_blok20 + (c4_blok20 - c3_blok20) * (-0.0333 *
p_aer_blok20 + 25.331)
        } else if p_aer_blok20 < 730 && p_aer_blok20 >= 674.2 {
            d3_blok20 = c4_blok20 + (c5_blok20 - c4_blok20) * (-0.0179 *
p_aer_blok20 + 13.0824)
        } else if p_aer_blok20 < 674.2 && p_aer_blok20 >= 634.25 {
            d3_blok20 = c5_blok20 + (c6_blok20 - c5_blok20) * (-0.025 *
p_aer_blok20 + 16.876)
        } else if p_aer_blok20 < 634.25 && p_aer_blok20 >= 596.31 {
            d3_blok20 = c6_blok20 + (c7_blok20 - c6_blok20) * (-0.0264 *
p_aer_blok20 + 16.7171)
        } else {
            d3_blok20 = 999999 // т.е. ошибка
        }
        if p_aer_blok20 > 760 {
            d8_blok20 = 84
        } else if p_aer_blok20 <= 760 && p_aer_blok20 >= 730 {
            d8_blok20 = c8_blok20 + (c9_blok20 - c8_blok20) * (-0.0333 *
p_aer_blok20 + 25.331)
        } else if p_aer_blok20 < 730 && p_aer_blok20 >= 674.2 {
            d8_blok20 = c9_blok20 + (c10_blok20 - c9_blok20) * (-0.0179 *
p_aer_blok20 + 13.0824)
        } else if p_aer_blok20 < 674.2 && p_aer_blok20 >= 634.25 {
```

```
            d8_blok20 = c10_blok20 + (c11_blok20 - c10_blok20) * (-0.025 *
p_aer_blok20 + 16.876)
        } else if p_aer_blok20 < 634.25 && p_aer_blok20 >= 596.31 {
            d8_blok20 = c11_blok20 + (c12_blok20 - c11_blok20) * (-0.0264 *
p_aer_blok20 + 16.7171)
        } else {
            d8_blok20 = 999999 // т.е. ошибка
        }
        if p_aer_blok20 > 760 {
            d13_blok20 = 83
        } else if p_aer_blok20 <= 760 && p_aer_blok20 >= 730 {
            d13_blok20 = c13_blok20 + (c14_blok20 - c13_blok20) * (-0.0333 *
p_aer_blok20 + 25.331)
        } else if p_aer_blok20 < 730 && p_aer_blok20 >= 674.2 {
            d13_blok20 = c14_blok20 + (c15_blok20 - c14_blok20) * (-0.0179 *
p_aer_blok20 + 13.0824)
        } else if p_aer_blok20 < 674.2 && p_aer_blok20 >= 634.25 {
            d13_blok20 = c15_blok20 + (c16_blok20 - c15_blok20) * (-0.025 *
p_aer_blok20 + 16.876)
        } else if p_aer_blok20 < 634.25 && p_aer_blok20 >= 596.31 {
            d13_blok20 = c16_blok20 + (c17_blok20 - c16_blok20) * (-0.0264 *
p_aer_blok20 + 16.7171)
        } else {
            d13_blok20 = 999999 // т.е. ошибка
        }
        if vzl_rezh == 60 {
            n2 = d13_blok20
        } else if vzl_rezh == 66 {
            n2 = d8_blok20
        } else {
            n2 = d3_blok20
        }

        // БЛОК 21 Градиенты набора высоты на взлете при скорости V2
        var b4_blok21: Double
        var b5_blok21: Double
        var b6_blok21: Double
        var b7_blok21: Double
        var b8_blok21: Double
        var b9_blok21: Double
        var b10_blok21: Double
        var line_otsch: Double
```

```
        var left_from_line: Double
        var right_from_line: Double
        b4_blok21 = ((0.00589 * t + 8.183) - 7.5) * (1 - 0.001 * height_c2)
+ 7.5
        b5_blok21 = 30 - 0.0062 * height_c2
        b6_blok21 = ((12.13 - 0.12568 * t) - (11.432 - 0.16578 * t)) * (1 -
0.001 * height_c2) + (11.432 - 0.16578 * t)
        b7_blok21 = (7.5 - 4.3) * (1.5 - 0.0005 * height_c2) + 4.3
        b8_blok21 = 30.76 - 0.0069598 * height_c2
        b9_blok21 = ((11.432 - 0.16578 * t) - (8.3767 - 0.14569 * t)) * (2 -
0.001 * height_c2) + (8.3767 - 0.14569 * t)
        b10_blok21 = ((8.3767 - 0.14569 * t) - (5.5017 - 0.12163 * t)) * (3
- 0.001 * height_c2) + (5.5017 - 0.12163 * t)
        if height_c2 <= 1000 && t <= b5_blok21 {
            line_otsch = b4_blok21
        } else if height_c2 <= 1000 && t > b5_blok21 {
            line_otsch = b6_blok21
        } else if height_c2 > 1000 && t <= b8_blok21 {
            line_otsch = b7_blok21
        } else if height_c2 > 1000 && height_c2 <= 2000 && t > b8_blok21 {
            line_otsch = b9_blok21
        } else if height_c2 > 2000 && t > b8_blok21 {
            line_otsch = b10_blok21
        } else {
            line_otsch = 999999

        left_from_line = ((1.8463 * pow(10, (-9)) * pow(m_vlz_fakt, 2) -
5.6687 * pow(10, (-4)) * m_vlz_fakt + 40.5325) - (5.866 * pow(10, (-9)) *
pow(m_vlz_fakt, 2) - 1.1099 * pow(10, (-3)) * m_vlz_fakt + 54.248)) * (0.25
* line_otsch - 0.75) + (5.866 * pow(10, (-9)) * pow(m_vlz_fakt, 2) - 1.1099
* pow(10, (-3)) * m_vlz_fakt + 54.248)
        right_from_line = ((1.976 * pow(10, (-9)) * pow(m_vlz_fakt, 2) -
5.554 * pow(10,(-4)) * m_vlz_fakt + 40.76) - (1.6238 * pow(10, (-9)) *
pow(m_vlz_fakt, 2) - 4.5031 * pow(10,(-4)) * m_vlz_fakt + 30.605)) *
(0.24999 * line_otsch - 1.249) + (1.6238 * pow(10, (-9)) * pow(m_vlz_fakt,
2) - 4.5031 * pow(10, (-4)) * m_vlz_fakt + 30.605)
        if m_vlz_fakt < 80000 {
            grad_blok21 = left_from_line
        } else {
            grad_blok21 = right_from_line
        }
```

```
        // БЛОК 22 Градиент ---> Vy
        var v_blok22: Double
        v_blok22 = v_2 / 3.6
        v_y = v_blok22 * grad_blok21 / 100

        // БЛОК 23 РИС 2.2.0
        if rwCondition == 1 {
            crosswind_max = 33.3333 * kBR - 5
        } else {
            crosswind_max = 5
        }

        //Округление выходных значений
        crosswind_max = (crosswind_max*10).rounded()/10
        crosswind = (crosswind*10).rounded() / 10
        headwind = (headwind*10).rounded() / 10
        qfe = (qfe*10).rounded() / 10
        mm_rt_st = (mm_rt_st*10).rounded() / 10
        height_c2 = (height_c2*10).rounded() / 10
        fin_m_vzl_max = (fin_m_vzl_max*10).rounded() / 10
        vzl_rezh = (vzl_rezh*10).rounded() / 10
        n2 = (n2*10).rounded() / 10
        v1 = (v1*10).rounded() / 10
        v_pst = (v_pst*10).rounded() / 10
        v_2 = (v_2*10).rounded() / 10
        grad_blok21 = (grad_blok21*10).rounded() / 10
        v_y = (v_y*10).rounded() / 10
        v3 = (v3*10).rounded() / 10
        v4 = (v4*10).rounded() / 10

        // ВЫВОД РЕЗУЛЬТАТОВ
        calculationsresult.text = " Crosswind MAX = \(Double(crosswind_max))
, Crosswind ACTUAL = \(Double(crosswind)) \n HEADWIND = \(Double(headwind))
\n \n QFE = \(Double(qfe)) / \(Double(mm_rt_st)) мм.рт.ст.    Elev =
\(Double(height_c2)) \n \n MAX TAKEOFF MASS = \(Double(fin_m_vzl_max)) \n \n
Взлетный режим работы двигателя: \(Double(vzl_rezh)), n2 = \(Double(n2)) %
\n \n Скорость принятия решения V1 = \(Double(v1)) \n Скорость подъема
передней стойки шасси VR = \(Double(v_pst)) \n Скорость на взлете V2 =
\(Double(v_2)) Град на одном дв. \(Double(grad_blok21)) - \(Double(v_y)) м/с
\n Скорость начала уборки механизации на взлете V3 = \(Double(v3)) \n
Скорость при полетной конфигурации V4 = \(Double(v4))"
    }
```

```
    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a
nib.

        let toratap = UITapGestureRecognizer(target: self, action:
#selector(toradoubleTapped))
        toratap.numberOfTapsRequired = 2
        toraLabel.addGestureRecognizer(toratap)
    }

    @objc func toradoubleTapped() {
        let helpalert = UIAlertController(title: "Справка", message: "TORA
(Take-off Run Available) - располагаемая длина разбега", preferredStyle:
.alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func todaTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "TODA
(Take-off Distance Available) - располагаемая взлетная дистанция",
preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }


    @IBAction func asdaTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "ASDA
(Accelerate Stop Distance Available) - располагаемая дистанция прерванного
взлета", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }
```

```
    @IBAction func elevTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message:
"Elevation - превышение аэродрома — высота самой высокой точки ВПП
относительно уровня моря", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func slopeTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "Slope
- средний уклон между двумя концами или точками на ВПП (разность высот между
двумя указанными точками, деленная на расстояние между ними). Выражается в
процентах, перед которыми ставится знак «плюс», если уклон восходящий, или
«минус», если уклон нисходящий", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func hdgTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message:
"Heading - курс оси ВПП согласно магнитному курсу", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }


    @IBAction func kscepTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message:
"Коэффициент сцепления ВПП", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }
```

```
    @IBAction func oatTap(_ sender:
        UITapGestureRecognizer) {
            let helpalert = UIAlertController(title: "Справка", message:
"OAT - Outside Air Temperature - температура наружного воздуха",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "OK", style: .cancel,
handler: nil)
            helpalert.addAction(okbutton)
            self.present(helpalert, animated: true, completion: nil)
    }

     @IBAction func qnhTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "QNH -
атмосферное давление в районе аэродрома, приведенное к среднему уровню моря
(MSL). Указывается в гектопаскалях. Сообщается по ATIS и органами УВД по
запросу экипажа", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }


    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
    //метка для возвращения из окна результатов
    @IBAction func backTapped(_ sender: UIStoryboardSegue) {
        //self.navigationController?.popViewController(animated: true)
    }
    //передача данных в контроллер представления результатов
    override func prepare(for segue: UIStoryboardSegue, sender: Any?) {
        var toresultvc = segue.destination as! TOResultViewController
        toresultvc.inpapname = inpapname
        toresultvc.inprwname = inprwname
        toresultvc.aircraftname = aircraftname
        toresultvc.m_vzl = Double(mTextField.text!)!
        toresultvc.kBR = Double(kscepTextField.text!)!
        toresultvc.rwconditiontext = rwconditiontext
        toresultvc.t = Double(oatTextField.text!)!
        toresultvc.qnh = Double(qnhTextField.text!)!
```

```
        toresultvc.wind_dir = Double(winddirTextField.text!)!
        toresultvc.wind_speed = Double(windspeedTextField.text!)!
        toresultvc.crosswind_max = crosswind_max
        toresultvc.crosswind = crosswind
        toresultvc.headwind = headwind
        toresultvc.qfe = qfe
        toresultvc.fin_m_vzl_max = fin_m_vzl_max
        toresultvc.vzl_rezh = vzl_rezh
        toresultvc.n2 = n2
        toresultvc.v1 = v1
        toresultvc.v_pst = v_pst
        toresultvc.v_2 = v_2
        toresultvc.grad_blok21 = grad_blok21
        toresultvc.v3 = v3
        toresultvc.v4 = v4
        toresultvc.v_y = v_y
        toresultvc.tora_full_length = tora_full_length
        toresultvc.tora_available_run_length = tora_available_run_length
    }
}
}
```

SecondViewController.swift

```
import UIKit
import RealmSwift

class SecondViewController: UIViewController {
    var brit_l = Bool()
    @IBOutlet weak var ldaTextField: UITextField!
    @IBOutlet weak var elevlandTextField: UITextField!
    @IBOutlet weak var slopelandTextField: UITextField!
    @IBOutlet weak var hdglandTextField: UITextField!
    @IBOutlet weak var rwcondlandSegmented: UISegmentedControl!
    @IBOutlet weak var ksceplandTextField: UITextField!
    @IBOutlet weak var oatlandTextField: UITextField!
    @IBOutlet weak var qnhTextField: UITextField!
    @IBOutlet weak var windspeedlandTextField: UITextField!
    @IBOutlet weak var winddirlandTextField: UITextField!
    @IBOutlet weak var masslandTextField: UITextField!
    @IBOutlet weak var calclandButton: UIButton!
    @IBOutlet weak var labelTest: UILabel!
```

```
    //ВЫПАДАЮЩИЙ СПИСОК

    @IBAction func acsellandTapped(_ sender: UIButton) {
        let aircraftlist = UIAlertController(title: "Выберите ВС",
message:nil, preferredStyle: .actionSheet)
        let aircraftname1: String = "Ту-204-100В"
        let aircraftname2: String = "МС-21"
        let aircraftname3: String = "Sukhoi Superjet 100"
        let aircraftname4: String = "Ил-96-300"
        let aircraftname5: String = "Як-42Д"

        let aircraftItem1 = UIAlertAction(title: aircraftname1, style:
.default, handler: {action in
            sender.setTitle(aircraftname1, for: .normal)
            sender.setTitleColor(.green, for: .normal)
            self.calclandButton.isHidden = false
        })

        let aircraftItem2 = UIAlertAction(title: aircraftname2, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "ОК", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname2, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calclandButton.isHidden = true
            self.labelTest.text = ""
        })

        let aircraftItem3 = UIAlertAction(title: aircraftname3, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "ОК", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
```

```
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname3, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calclandButton.isHidden = true
            self.labelTest.text = ""
        })

        let aircraftItem4 = UIAlertAction(title: aircraftname4, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "OK", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname4, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calclandButton.isHidden = true
            self.labelTest.text = ""
        })

        let aircraftItem5 = UIAlertAction(title: aircraftname5, style:
.default, handler: {action in
            let aircraftalert = UIAlertController(title: "Загрузите данные
по ВС", message: "Загрузите данные по ВС для выполнения расчетов",
preferredStyle: .alert)
            let okbutton = UIAlertAction(title: "OK", style: .cancel,
handler: nil)
            aircraftalert.addAction(okbutton)
            self.present(aircraftalert, animated: true, completion: nil)

            sender.setTitle(aircraftname5, for: .normal)
            sender.setTitleColor(.red, for: .normal)
            self.calclandButton.isHidden = true
            self.labelTest.text = ""
        })

        aircraftlist.addAction(aircraftItem1)
        aircraftlist.addAction(aircraftItem2)
```

```
        aircraftlist.addAction(aircraftItem3)
        aircraftlist.addAction(aircraftItem4)
        aircraftlist.addAction(aircraftItem5)

        if let ppc = aircraftlist.popoverPresentationController {
            ppc.sourceView = sender
            ppc.sourceRect = sender.bounds
        }
        present(aircraftlist, animated: true, completion: nil)
    }

    //ВЫЗОВ СПИСКА АЭРОДРОМОВ
    @IBAction func rwlandlistTapped(_ sender: UIButton) {
        let uirealm = try! Realm()
        var runways = uirealm.objects(Airport.self)
        var elementscount = runways.count
        var dedicrw = Airport()

        let rwlist = UIAlertController(title: "Выберите ВПП", message:nil,
preferredStyle: .actionSheet)

        for dedicrw in runways {
            var rwname: String = dedicrw.adrwy
            var rwItem = UIAlertAction(title: rwname, style: .default,
handler: {action in
                inpapname = dedicrw.name
                inprwname = dedicrw.rwy
                tora_full_length = dedicrw.tora_full_length
                tora_available_run_length =
dedicrw.tora_available_run_length

                sender.setTitle(rwname, for: .normal)
                sender.setTitleColor(.green, for: .normal)
                self.ldaTextField.text =
String(dedicrw.lda_available_landing_distance)
                self.elevlandTextField.text = String(dedicrw.elev)
                self.slopelandTextField.text = String(dedicrw.runway_slope)
                self.hdglandTextField.text = String(dedicrw.hdg)
                self.brit_l = dedicrw.brit
            })

            rwlist.addAction(rwItem)
```

```
        }
        if let ppc = rwlist.popoverPresentationController {
            ppc.sourceView = sender
            ppc.sourceRect = sender.bounds
        }
        present(rwlist, animated: true, completion: nil)
    }

    //ЗАГРУЗКА ПОГОДЫ (METAR)
    @IBAction func metarTapped(_ sender: UIButton) {
        let metaralert = UIAlertController(title: "Предупреждение", message:
"Не подключен источник данных. Введите инфомацию о погоде вручную",
preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        metaralert.addAction(okbutton)
        self.present(metaralert, animated: true, completion: nil)
    }

    //РАСЧЕТ ПАРАМЕТРОВ ПОСАДКИ
    @IBAction func landingTapped(_ sender: UIButton) {

        //данные по аэродрому
        var lda: Double = Double(ldaTextField.text!)!
        var slope_l: Double = Double(slopelandTextField.text!)!
        var hdg_l: Double = Double(hdglandTextField.text!)!
        var elev_l: Double = Double(elevlandTextField.text!)!

        //данные по ВС
        var m_land_max: Double = 88000

        //задаваемые переменные
        var t_land: Double = Double(oatlandTextField.text!)!
        var qnh_land: Double = Double(qnhTextField.text!)!
        var k_br_l: Double = Double(ksceplandTextField.text!)!
        var land_weight: Double = Double(masslandTextField.text!)!
        var wind_dir_l: Double = Double(winddirlandTextField.text!)!
        var wind_speed_l: Double = Double(windspeedlandTextField.text!)!
        var rw_cond_l: Double = 0
        if rwcondlandSegmented.selectedSegmentIndex == 0 {
            rw_cond_l = 1
        } else if rwcondlandSegmented.selectedSegmentIndex == 1 {
```

```
            rw_cond_l = 2
        } else if rwcondlandSegmented.selectedSegmentIndex == 2 {
            rw_cond_l = 3
        }
        var kren: Double = 0

        //БЛОК раскладка ветра
        var headwind_l, crosswind_l, sub_angle_l, sub_angle_rad_l: Double
        let pi = 3.14159
        sub_angle_l = wind_dir_l - hdg_l
        sub_angle_rad_l = sub_angle_l *  pi / 180
        crosswind_l = sin(sub_angle_rad_l) * wind_speed_l
        headwind_l = cos(sub_angle_rad_l) * wind_speed_l

        // БЛОК давление на аэродроме
        var hPa_l, qfe_land: Double
        hPa_l = -4.6708 * pow(10,(-7)) * pow(elev_l, 2) + 0.0363 * elev_l +
0.1842
        if brit_l == true {
            qfe_land = qnh_land - hPa_l
        } else {
            qfe_land = qnh_land - hPa_l * 3.2808
        }

        //БЛОК 1 (гПа ---> мм.рт.ст-1)
        var mm_rt_st_land: Double
        var vysota_land: Double
        mm_rt_st_land = qfe_land * 0.75
        vysota_land = 0.0078727 * pow(mm_rt_st_land, 2) - 22.9056 *
mm_rt_st_land + 12861.4077

        //БЛОК 2 (Посадочная масса, ограниченная нормируемым градиентом
набора высоты с одним отказавшим двигателем)
        var b3_blok2_l: Double
        var b4_blok2_l: Double
        var b5_blok2_l: Double
        var b6_blok2_l: Double
        var b7_blok2_l: Double
        var b8_blok2_l: Double
        var b9_blok2_l: Double
        var m_grad_l: Double
```

```
        b3_blok2_l = ((98839 - 13.7328 * vysota_land) - (93129.3 - 13.6054 *
vysota_land)) * (9 - 0.2 * t_land) + (93129.3 - 13.6054 * vysota_land)
        b4_blok2_l = ((105200 - 14.6667 * vysota_land) - (98839 - 13.7328 *
vysota_land)) * (8 - 0.2 * t_land) + (98839 - 13.7328 * vysota_land)
        b5_blok2_l = ((109742 - 14.1935 * vysota_land) - (105200 - 14.6667 *
vysota_land)) * (7 - 0.2 * t_land) + (105200 - 14.6667 * vysota_land)
        b6_blok2_l = ((112654 - 13.3253 * vysota_land) - (109742 - 14.1935 *
vysota_land)) * (6 - 0.2 * t_land) + (109742 - 14.1935 * vysota_land)
        b7_blok2_l = ((117515 - 13.4146 * vysota_land) - (112654 - 13.3253 *
vysota_land)) * (5 - 0.2 * t_land) + (112654 - 13.3253 * vysota_land)
        b8_blok2_l = ((120479 - 13.1422 * vysota_land) - (117515 - 13.4146 *
vysota_land)) * (4 - 0.2 * t_land) + (117515 - 13.4146 * vysota_land)
        b9_blok2_l = ((111695 - 8.66483 * vysota_land) - (120479 - 13.1422 *
vysota_land)) * (3 - 0.2 * t_land) + (120479 - 13.1422 * vysota_land)
        if t_land >= 40 {
            m_grad_l = b3_blok2_l
        } else if t_land < 40 && t_land >= 35 {
            m_grad_l = b4_blok2_l
        } else if t_land < 35 && t_land >= 30 {
            m_grad_l = b5_blok2_l
        } else if t_land < 30 && t_land >= 25 {
            m_grad_l = b6_blok2_l
        } else if t_land < 25 && t_land >= 20 {
            m_grad_l = b7_blok2_l
        } else if t_land < 20 && t_land >= 15 {
            m_grad_l = b8_blok2_l
        } else if t_land < 15 && t_land >= 10 {
            m_grad_l = b9_blok2_l
        } else {
            m_grad_l = 88000
        }

         //БЛОК 3 (LDA)
        var lda_blok3_l: Double
        if brit_l {
            lda_blok3_l = lda / 3.2808
        } else {
            lda_blok3_l = lda
        }

        //БЛОК 4 (Располагаемая длина ВПП с учетом ее состояния на посадке)
        var b3_blok4_l: Bool
```

```
        var b4_blok4_l: Double
        var dlina_vpp_blok4: Double
        var k_br_l_blok4: Double

        if rw_cond_l == 2 || rw_cond_l == 3 {
            b3_blok4_l = true
        } else {
            b3_blok4_l = false
        }
        if b3_blok4_l == true {
            b4_blok4_l = lda_blok3_l * 0.73333333333
        } else {
            b4_blok4_l = lda_blok3_l
        }
        if k_br_l >= 0.57 {
            k_br_l_blok4 = 0.57
        } else {
            k_br_l_blok4 = k_br_l
        }
        if b3_blok4_l {
            dlina_vpp_blok4 = b4_blok4_l
        } else {
            dlina_vpp_blok4 = ((0.6833 * b4_blok4_l - 4.9995) - b4_blok4_l)
* (2.11111 - 3.7037 * k_br_l_blok4) + b4_blok4_l
        }

        //БЛОК 5 (Скорректированная располагаемая посадочная дистанция)
        var a4_blok5_l: Double
        var b4_blok5_l: Double
        var rpd_ap_osn: Double
        var rpd_ap_zap: Double
        if dlina_vpp_blok4 > 2600 && slope_l < 0 {
            a4_blok5_l = ((0.9 * dlina_vpp_blok4 + 100) - dlina_vpp_blok4) *
(-0.5 * slope_l) + dlina_vpp_blok4
        } else if dlina_vpp_blok4 <= 2800 {
            a4_blok5_l = ((1.0729 * dlina_vpp_blok4 - 29.2381) -
dlina_vpp_blok4) * (0.5 * slope_l) + dlina_vpp_blok4
        } else if dlina_vpp_blok4 > 2800 {
            a4_blok5_l = ((dlina_vpp_blok4 + 200) - dlina_vpp_blok4) * (0.5
* slope_l) + dlina_vpp_blok4
        } else {
            a4_blok5_l = 777777 //ошибка
```

```
        }
        if a4_blok5_l <= 3000 && headwind_l < 0 {
            b4_blok5_l = ((a4_blok5_l - 400) - a4_blok5_l) * (-0.2 *
headwind_l) + a4_blok5_l
        } else if headwind_l > 0 {
            b4_blok5_l = ((1.0556 * a4_blok5_l + 440.0142) - a4_blok5_l) *
(0.05 * headwind_l) + a4_blok5_l
        } else {
            b4_blok5_l = ((0.77 * a4_blok5_l + 290.0116) - a4_blok5_l) * (-
0.2 * headwind_l) + a4_blok5_l
        }
        rpd_ap_osn = b4_blok5_l
        rpd_ap_zap = 1.15 * b4_blok5_l + 30.0023

        //БЛОК 6 (Потребная посадочная дистанция от ФАКТИЧЕСКОЙ посадочной
массы)
        var d2_blok6_l, d3_blok6_l, d4_blok6_l, d5_blok6_l, d6_blok6_l,
d7_blok6_l, d8_blok6_l, d9_blok6_l, d10_blok6_l, d11_blok6_l, d12_blok6_l,
d13_blok6_l, d14_blok6_l, d15_blok6_l, d16_blok6_l, d17_blok6_l,
d18_blok6_l, d19_blok6_l, d20_blok6_l, d21_blok6_l, d22_blok6_l: Double
        var b3_blok6_l, b4_blok6_l, b5_blok6_l, b6_blok6_l, b7_blok6_l,
b8_blok6_l, b10_blok6_l, potr_pos_dist_ot_m, b13_blok6_l, b14_blok6_l,
b15_blok6_l, b16_blok6_l, b18_blok6_l, b19_blok6_l, b20_blok6_l,
b21_blok6_l: Double

        d2_blok6_l = 0.01 * pow(t_land, 2) + 3.5 * t_land + 1850
        d3_blok6_l = 0.0125 * pow(t_land, 2) + 3.9 * t_land + 1996
        d4_blok6_l = 0.0075 * pow(t_land, 2) + 4.05 * t_land + 2200
        d5_blok6_l = 3.8 * t_land + 2400
        d6_blok6_l = 0.0371111 * pow(t_land, 2) + 4.50778 * t_land + 1850
        d7_blok6_l = 0.0764897 * pow(t_land, 2) + 3.82021 * t_land + 1996
        d8_blok6_l = 0.0488975 * pow(t_land, 2) + 4.72205 * t_land + 2200
        d9_blok6_l = 6.45161 * t_land + 2400
        d10_blok6_l = 0.001 * vysota_land
        d11_blok6_l = 0.001 * vysota_land - 1
        d12_blok6_l = 0.001 * vysota_land - 2
        d13_blok6_l = 0.00000017 * pow(land_weight, 2) - 0.0158 *
land_weight + 1976
        d14_blok6_l = 0.00000023 * pow(land_weight, 2) - 0.0222 *
land_weight + 2304
        d15_blok6_l = 0.000000255 * pow(land_weight, 2) - 0.02305 *
land_weight + 2412
```

```
        d16_blok6_l = 0.00000039 * pow(land_weight, 2) - 0.0385 *
land_weight + 2984
        d17_blok6_l = 0.00000028 * pow(land_weight, 2) - 0.02 * land_weight
+ 2408
        d18_blok6_l = 0.000000319444 * pow(land_weight, 2) - 0.0336944 *
land_weight + 2451.11
        d19_blok6_l = 0.000000444444 * pow(land_weight, 2) - 0.0546111 *
land_weight + 3524.44
        d20_blok6_l = 0.000000972222 * pow(land_weight, 2) - 0.133556 *
land_weight + 6662.22
        d21_blok6_l = 0.000000777778 * pow(land_weight, 2) - 0.0871111 *
land_weight + 4391.11
        d22_blok6_l = 0.00000152 * pow(land_weight, 2) - 0.202 * land_weight
+ 9032
        b3_blok6_l = (d3_blok6_l - d2_blok6_l) * d10_blok6_l + d2_blok6_l
        b4_blok6_l = (d4_blok6_l - d3_blok6_l) * d11_blok6_l + d3_blok6_l
        b5_blok6_l = (d5_blok6_l - d4_blok6_l) * d12_blok6_l + d4_blok6_l
        b6_blok6_l = (d7_blok6_l - d6_blok6_l) * d10_blok6_l + d6_blok6_l
        b7_blok6_l = (d8_blok6_l - d7_blok6_l) * d11_blok6_l + d7_blok6_l
        b8_blok6_l = (d9_blok6_l - d8_blok6_l) * d12_blok6_l + d8_blok6_l

        if t_land <= 0 && vysota_land <= 1000 {
            b10_blok6_l = b3_blok6_l
        } else if t_land <= 0 && vysota_land > 1000 && vysota_land < 2000 {
            b10_blok6_l = b4_blok6_l
        } else if t_land <= 0 && vysota_land >= 2000 {
            b10_blok6_l = b5_blok6_l
        } else if t_land > 0 && vysota_land <= 1000 {
            b10_blok6_l = b6_blok6_l
        } else if t_land > 0 && vysota_land > 1000 && vysota_land < 2000 {
            b10_blok6_l = b7_blok6_l
        } else {
            b10_blok6_l = b8_blok6_l
        }

        b13_blok6_l = (d14_blok6_l - d13_blok6_l) * (0.005 * b10_blok6_l -
9) + d13_blok6_l
        b14_blok6_l = (d15_blok6_l - d14_blok6_l) * (0.005 * b10_blok6_l -
10) + d14_blok6_l
        b15_blok6_l = (d16_blok6_l - d15_blok6_l) * (0.005 * b10_blok6_l -
11) + d15_blok6_l
```

```
        b16_blok6_l = (d17_blok6_l - d16_blok6_l) * (0.005 * b10_blok6_l -
12) + d16_blok6_l
        b18_blok6_l = (d19_blok6_l - d18_blok6_l) * (0.005 * b10_blok6_l -
9) + d18_blok6_l
        b19_blok6_l = (d20_blok6_l - d19_blok6_l) * (0.005 * b10_blok6_l -
10) + d19_blok6_l
        b20_blok6_l = (d21_blok6_l - d20_blok6_l) * (0.005 * b10_blok6_l -
11) + d20_blok6_l
        b21_blok6_l = (d22_blok6_l - d20_blok6_l) * (0.0025 * b10_blok6_l -
5.5) + d20_blok6_l

        if land_weight <= 80000 && b10_blok6_l <= 2000 {
            potr_pos_dist_ot_m = b13_blok6_l
        } else if land_weight <= 80000 && b10_blok6_l <= 2200 && b10_blok6_l
> 2000 {
            potr_pos_dist_ot_m = b14_blok6_l
        } else if land_weight <= 80000 && b10_blok6_l <= 2400 && b10_blok6_l
> 2200 {
            potr_pos_dist_ot_m = b15_blok6_l
        } else if land_weight <= 80000 && b10_blok6_l > 2400 {
            potr_pos_dist_ot_m = b16_blok6_l
        } else if land_weight > 80000 && b10_blok6_l <= 2000 {
            potr_pos_dist_ot_m = b18_blok6_l
        } else if land_weight > 80000 && b10_blok6_l <= 2200 && b10_blok6_l
> 2000 {
            potr_pos_dist_ot_m = b19_blok6_l
        } else if land_weight > 80000 && b10_blok6_l <= 2400 && b10_blok6_l
> 2200 {
            potr_pos_dist_ot_m = b20_blok6_l
        } else {
            potr_pos_dist_ot_m = b21_blok6_l
        }

        //БЛОК 7 (Потребная посадочная дистанция. ОБРАТНЫЙ РАСЧЕТ)
        var potreb_pos_dist_blok7_l, b2_blok7_l: Double

        if potr_pos_dist_ot_m <= 2760 && headwind_l < 0 {
            b2_blok7_l = potr_pos_dist_ot_m - (potr_pos_dist_ot_m -
(potr_pos_dist_ot_m + 400)) * (-0.2 * headwind_l)
        } else if headwind_l > 0 {
```

```
            b2_blok7_l = potr_pos_dist_ot_m - (potr_pos_dist_ot_m -
(potr_pos_dist_ot_m - 440.0142) / 1.0556) * (0.05 * headwind_l) * (-0.00278
* headwind_l + 1.0556)
        } else {
            b2_blok7_l = potr_pos_dist_ot_m - (potr_pos_dist_ot_m -
(potr_pos_dist_ot_m - 290.0116) / 0.77) * (-0.2 * headwind_l) * (-0.046 *
headwind_l + 0.77) * (0.0027457 * pow(headwind_l, 2) + 0.0137 * headwind_l +
1)
        }

        if b2_blok7_l > 2600 && slope_l < 0 {
            potreb_pos_dist_blok7_l = b2_blok7_l - (b2_blok7_l - (b2_blok7_l
- 100) / 0.9) * (-0.5 * slope_l) * (-0.05 * slope_l + 0.9)
        } else if b2_blok7_l <= 2900 {
            potreb_pos_dist_blok7_l = b2_blok7_l - (b2_blok7_l - (b2_blok7_l
+ 29.2381) / 1.0729) * (0.5 * slope_l) * (-0.0364 * slope_l + 1.0729)
        } else {
            potreb_pos_dist_blok7_l = b2_blok7_l - (b2_blok7_l - (b2_blok7_l
- 200)) * (0.5 * slope_l)
        }

        //БЛОК 8 (Потребная длина ВПП с учетом ее состояния на посадке.
ОБРАТНЫЙ РАСЧЕТ)
        var podreb_dl_vpp_blok8_l, b4_blok8_l: Double
        b4_blok8_l = potreb_pos_dist_blok7_l - (potreb_pos_dist_blok7_l -
(potreb_pos_dist_blok7_l + 4.5559) / 0.6833) * (2.11111 - 3.7037 *
k_br_l_blok4) * (-1.172 * k_br_l_blok4 + 1.3516) * (1.9349 *
pow(k_br_l_blok4, 2) - 1.6834 * k_br_l_blok4 + 1.3309)
        if b3_blok4_l {
            podreb_dl_vpp_blok8_l = potreb_pos_dist_blok7_l / 0.73333333333
        } else {
            podreb_dl_vpp_blok8_l = b4_blok8_l
        }

        //БЛОК 9 (Скорости на посадке)
        var v_l_18, v_l_26, v_l_37: Double

        v_l_18 = (0.0014063 * land_weight + 130.6231)
        if (0.0015172 * land_weight + 114.4147) < 215 {
            v_l_26 = 215
        } else {
            v_l_26 = 0.0015172 * land_weight + 114.4147
```

```
        }

        if (0.0014583 * land_weight + 110.8375) < 215 {
            v_l_37 = 215
        } else {
            v_l_37 = 0.0014583 * land_weight + 110.8375
        }

        //БЛОК 10 (Посадочная масса, ограниченная посадочной дистанцией)
        var b3_blok10_l, b4_blok10_l, b5_blok10_l, b6_blok10_l, b7_blok10_l,
b8_blok10_l, b10_blok10_l, m_ogr_pos_dist_blok10, b13_blok10_l,
b14_blok10_l, b15_blok10_l, b16_blok10_l, b18_blok10_l, b19_blok10_l,
b20_blok10_l, b21_blok10_l: Double
        var d2_blok10_l, d3_blok10_l, d4_blok10_l, d5_blok10_l, d6_blok10_l,
d7_blok10_l, d8_blok10_l, d9_blok10_l, d10_blok10_l, d11_blok10_l,
d12_blok10_l: Double

        d2_blok10_l = 0.01 * pow(t_land, 2) + 3.5 * t_land + 1850
        d3_blok10_l = 0.0125 * pow(t_land, 2) + 3.9 * t_land + 1996
        d4_blok10_l = 0.0075 * pow(t_land, 2) + 4.05 * t_land + 2200
        d5_blok10_l = 3.8 * t_land + 2400
        d6_blok10_l = 0.0371111 * pow(t_land, 2) + 4.50778 * t_land + 1850
        d7_blok10_l = 0.0764897 * pow(t_land, 2) + 3.82021 * t_land + 1996
        d8_blok10_l = 0.0488975 * pow(t_land, 2) + 4.72205 * t_land + 2200
        d9_blok10_l = 6.45161 * t_land + 2400
        d10_blok10_l = 0.001 * vysota_land
        d11_blok10_l = 0.001 * vysota_land - 1
        d12_blok10_l = 0.001 * vysota_land - 2

        b3_blok10_l = (d3_blok10_l - d2_blok10_l) * d10_blok10_l +
d2_blok10_l
        b4_blok10_l = (d4_blok10_l - d3_blok10_l) * d11_blok10_l +
d3_blok10_l
        b5_blok10_l = (d5_blok10_l - d4_blok10_l) * d12_blok10_l +
d4_blok10_l
        b6_blok10_l = (d7_blok10_l - d6_blok10_l) * d10_blok10_l +
d6_blok10_l
        b7_blok10_l = (d8_blok10_l - d7_blok10_l) * d11_blok10_l +
d7_blok10_l
        b8_blok10_l = (d9_blok10_l - d8_blok10_l) * d12_blok10_l +
d8_blok10_l
```

```
        if t_land <= 0 && vysota_land <= 1000 {
            b10_blok10_l = b3_blok10_l
        } else if t_land <= 0 && vysota_land > 1000 && vysota_land < 2000 {
            b10_blok10_l = b4_blok10_l
        } else if t_land <= 0 && vysota_land >= 2000 {
            b10_blok10_l = b5_blok10_l
        } else if t_land > 0 && vysota_land <= 1000 {
            b10_blok10_l = b6_blok10_l
        } else if t_land > 0 && vysota_land > 1000 && vysota_land < 2000 {
            b10_blok10_l = b7_blok10_l
        } else {
            b10_blok10_l = b8_blok10_l
        }

        b13_blok10_l = (1 / (3 * b10_blok10_l - 3700)) * 20000 * (sqrt(-59 *
pow(b10_blok10_l, 2) + 75 * b10_blok10_l * rpd_ap_osn + 57700 * b10_blok10_l
- 92500 * rpd_ap_osn + 18922500) + 8 * b10_blok10_l - 10450)
        b14_blok10_l = (1000 * (sqrt(-4031 * pow(b10_blok10_l, 2) + 8000 *
b10_blok10_l * rpd_ap_osn - 7237600 * b10_blok10_l - 1280000 * rpd_ap_osn +
4569760000) + 17 * b10_blok10_l + 54800)) / (b10_blok10_l - 160)
        b15_blok10_l = (5000 * (sqrt(-28071 * pow(b10_blok10_l, 2) + 43200 *
b10_blok10_l * rpd_ap_osn + 29618400 * b10_blok10_l - 78720000 * rpd_ap_osn
+ 39840160000) + 309 * b10_blok10_l - 587600)) / (3 * (9 * b10_blok10_l -
16400))
        b16_blok10_l = (5000 * (-sqrt(8881 * pow(b10_blok10_l, 2) - 8800 *
b10_blok10_l * rpd_ap_osn - 26888400 * b10_blok10_l + 27360000 * rpd_ap_osn
+ 686440000) + 185 * b10_blok10_l - 521000)) / (11 * b10_blok10_l - 34200)
        b18_blok10_l = (3.25261 * pow(10, (-15)) * (sqrt(-5.85786 * pow(10,
43) * pow(b10_blok10_l, 2) + b10_blok10_l * (5.90768 * pow(10, 43) *
rpd_ap_osn + 7.0578 * pow(10, 46)) - 7.61435 * pow(10, 46) * rpd_ap_osn +
1.55718 * pow(10, 49)) + 1.60769 * pow(10, 22) * b10_blok10_l - 2.37588 *
pow(10, 25))) / (625 * b10_blok10_l - 805556)
        b19_blok10_l = (-2.32589 * pow(10, (-15)) * sqrt(-6.50195 * pow(10,
37) * pow(b10_blok10_l, 2) + 7.00488 * pow(10, 37) * b10_blok10_l *
rpd_ap_osn + 1.14207 * pow(10, 41) * b10_blok10_l - 1.283 * pow(10, 41) *
rpd_ap_osn + 9.87566 * pow(10, 42)) - 74789.9 * b10_blok10_l + 1.39232 *
pow(10, 8)) / (1831.58 - b10_blok10_l)
        b20_blok10_l = (1.1393 * pow(10, (-14)) * sqrt(1.99038 * pow(10, 37)
* pow(b10_blok10_l, 2) - 7.92432 * pow(10, 36) * b10_blok10_l * rpd_ap_osn -
7.11957 * pow(10, 40) * b10_blok10_l + 2.53578 * pow(10, 40) * rpd_ap_osn +
4.38489 * pow(10, 43)) - 119430 * b10_blok10_l + 3.31432 * pow(10, 8)) /
(3200 - b10_blok10_l)
```

```
        b21_blok10_l = (-4.12198 * pow(10, (-14)) * sqrt(-2.49042 * pow(10,
35) * pow(b10_blok10_l, 2) + 4.29777 * pow(10, 35) * b10_blok10_l *
rpd_ap_osn + 1.08526 * pow(10, 37) * b10_blok10_l - 6.40394 * pow(10, 38) *
rpd_ap_osn + 5.48219 * pow(10, 41)) - 62474.2 * b10_blok10_l + 8.86805 *
pow(10, 7)) / (1490.06 - b10_blok10_l)

        if rpd_ap_osn <= b10_blok10_l && b10_blok10_l <= 2000 {
            m_ogr_pos_dist_blok10 = b13_blok10_l
        } else if rpd_ap_osn <= b10_blok10_l && b10_blok10_l > 2000 &&
b10_blok10_l<=2200 {
            m_ogr_pos_dist_blok10 = b14_blok10_l
        } else if rpd_ap_osn <= b10_blok10_l && b10_blok10_l > 2200 &&
b10_blok10_l <= 2400 {
            m_ogr_pos_dist_blok10 = b15_blok10_l
        } else if rpd_ap_osn <= b10_blok10_l && b10_blok10_l > 2400 {
            m_ogr_pos_dist_blok10 = b16_blok10_l
        } else if rpd_ap_osn > b10_blok10_l && b10_blok10_l <= 2000 {
            m_ogr_pos_dist_blok10 = b18_blok10_l
        } else if rpd_ap_osn > b10_blok10_l && b10_blok10_l > 2000 &&
b10_blok10_l <= 2200 {
            m_ogr_pos_dist_blok10 = b19_blok10_l
        } else if rpd_ap_osn > b10_blok10_l && b10_blok10_l > 2200 &&
b10_blok10_l <= 2400 {
            m_ogr_pos_dist_blok10 = b20_blok10_l
        } else {
            m_ogr_pos_dist_blok10 = b21_blok10_l
        }

        //БЛОК 11 (Градиенты набора высоты при уходе на 2-ой круг с одним
отказавшим двигателем (с креном))
        var b4_blok11_l, b5_blok11_l, b6_blok11_l, b7_blok11_l, b8_blok11_l,
b9_blok11_l, b10_blok11_l, b11_blok11_l, b12_blok11_l, b13_blok11_l,
b14_blok11_l, b15_blok11_l, b16_blok11_l, b17_blok11_l,
grad_nab_vys_blok11_l, grad_nab_vys_10_blok11_l, grad_nab_vys_20_blok11_l,
grad_nab_vys_30_blok11_l, sub_grad_nab_vys_10_blok11_l,
sub_grad_nab_vys_20_blok11_l, sub_grad_nab_vys_30_blok11_l,
sub2_grad_nab_vys_10_blok11_l, sub2_grad_nab_vys_20_blok11_l,
sub2_grad_nab_vys_30_blok11_l: Double

        b4_blok11_l = 0.5 * (1 - 0.001 * vysota_land) + 5
        b5_blok11_l = 1.3 * (2 - 0.001 * vysota_land) + 3.7
```

```
        b6_blok11_l = (5.5 - (5.6666 - 0.0333 * t_land)) * (1 - 0.001 *
vysota_land) + (5.6666 - 0.0333 * t_land)
        b7_blok11_l = ((5.6666 - 0.0333 * t_land) - 3.7) * (2 - 0.001 *
vysota_land) + 3.7
        b8_blok11_l = ((9.5 - 0.1333 * t_land) - (9.168 - 0.168 * t_land)) *
(1 - 0.001 * vysota_land) + (9.168 - 0.168 * t_land)
        b9_blok11_l = ((9.168 - 0.168 * t_land) - (5.9232 - 0.13077 *
t_land)) * (2 - 0.001 * vysota_land) + (5.9232 - 0.13077 * t_land)
        b10_blok11_l = 30 - 0.01 * vysota_land
        b11_blok11_l = 30 - 0.004 * vysota_land
        b12_blok11_l = 23 - 0.003 * vysota_land
        b13_blok11_l = 35 - 0.009 * vysota_land

        if vysota_land <= 0 && t_land <= b10_blok11_l {
            b14_blok11_l = b4_blok11_l
        } else if vysota_land <= 0 && t_land > b10_blok11_l {
            b14_blok11_l = b8_blok11_l
        } else if vysota_land > 0 && vysota_land <= 1000 && t_land <=
b10_blok11_l {
            b14_blok11_l = b4_blok11_l
        } else if vysota_land > 0 && vysota_land <= 1000 && t_land >
b10_blok11_l && t_land <= b11_blok11_l {
            b14_blok11_l = b6_blok11_l
        } else if vysota_land > 0 && vysota_land <= 1000 && t_land >
b11_blok11_l {
            b14_blok11_l = b8_blok11_l
        } else if vysota_land > 1000 && vysota_land <= 2000 && t_land <=
b12_blok11_l {
            b14_blok11_l = b5_blok11_l
        } else if vysota_land > 1000 && vysota_land <= 2000 && t_land >
b12_blok11_l && t_land <= b13_blok11_l {
            b14_blok11_l = b7_blok11_l
        } else if vysota_land > 1000 && t_land > b13_blok11_l {
            b14_blok11_l = b9_blok11_l
        } else if vysota_land > 2000 && t_land <= b13_blok11_l {
            b14_blok11_l = b5_blok11_l
        } else {
            b14_blok11_l = 99999999
        }

        b15_blok11_l = ((4.4728 * pow(10,(-10)) * pow(land_weight, 2) -
0.0002979 * land_weight + 26.96) - (-1.3355 * pow(10,(-10)) *
```

```
pow(land_weight, 2) - 0.00016168 * land_weight + 15.779)) * (0.25 *
b14_blok11_l - 0.5) + (-1.3355 * pow(10, (-10)) * pow(land_weight, 2) -
0.00016168 * land_weight + 15.779)
        b16_blok11_l = ((1.5567 * pow(10, (-9)) * pow(land_weight, 2) -
0.000435 * land_weight + 30.821) - (1.8964 * pow(10, (-9)) *
pow(land_weight, 2) - 0.0004781 * land_weight + 28.116)) * (0.25 *
b14_blok11_l - 0.5) + (1.8964 * pow(10, (-9)) * pow(land_weight, 2) -
0.0004781 * land_weight + 28.116)

        if land_weight < 80000 {
            b17_blok11_l = b15_blok11_l
        } else {
            b17_blok11_l = b16_blok11_l
        }

        grad_nab_vys_blok11_l = ((-0.0016 * pow(kren, 2) - 0.0638 * kren +
7.992) - (-0.0011796 * pow(kren, 2) - 0.080414 * kren + 3.9805)) * (0.25 *
b17_blok11_l - 1) + (-0.0011796 * pow(kren, 2) - 0.080414 * kren + 3.9805)

        sub2_grad_nab_vys_10_blok11_l = Double(-0.0016 * pow(10, 2) - 0.0638
* 10 + 7.992)
        sub_grad_nab_vys_10_blok11_l = Double((sub2_grad_nab_vys_10_blok11_l
- (-0.0011796 * pow(10, 2) - 0.080414 * 10 + 3.9805)))
        grad_nab_vys_10_blok11_l =  sub_grad_nab_vys_10_blok11_l * (0.25 *
b17_blok11_l - 1) + (-0.0011796 * pow(10, 2) - 0.080414 * 10 + 3.9805)

        sub2_grad_nab_vys_20_blok11_l = Double(-0.0011796 * pow(20, 2) -
0.080414 * 20 + 3.9805)
        sub_grad_nab_vys_20_blok11_l = Double(((-0.0016 * pow(20, 2) -
0.0638 * 20 + 7.992) - sub2_grad_nab_vys_20_blok11_l))
        grad_nab_vys_20_blok11_l = sub_grad_nab_vys_20_blok11_l * (0.25 *
b17_blok11_l - 1) + (-0.0011796 * pow(20, 2) - 0.080414 * 20 + 3.9805)

        sub2_grad_nab_vys_30_blok11_l = Double(-0.0011796 * pow(30, 2) -
0.080414 * 30 + 3.9805)
        sub_grad_nab_vys_30_blok11_l = Double(((-0.0016 * pow(30, 2) -
0.0638 * 30 + 7.992) - sub2_grad_nab_vys_30_blok11_l))
        grad_nab_vys_30_blok11_l = sub_grad_nab_vys_30_blok11_l * (0.25 *
b17_blok11_l - 1) + (-0.0011796 * pow(30, 2) - 0.080414 * 30 + 3.9805)


        //БЛОК максимально допустимая посадочная масса
```

```
        var m_land_max_fin: Double
        if min(m_ogr_pos_dist_blok10, m_grad_l) > m_land_max {
            m_land_max_fin = m_land_max
        } else {
            m_land_max_fin = min(m_ogr_pos_dist_blok10, m_grad_l)
        }

        //БЛОК crosswind max
        var crosswind_l_max: Double
        if b3_blok4_l {
            crosswind_l_max = 5
        } else {
            crosswind_l_max = (33.3333 * k_br_l - 5)
        }

        //ОКРУГЛЕНИЕ
            //округление скоростей вверх до 5
        v_l_18 = ((v_l_18 / 5).rounded(.up)) * 5
        v_l_26 = ((v_l_26 / 5).rounded(.up)) * 5
        v_l_37 = ((v_l_37 / 5).rounded(.up)) * 5

        crosswind_l_max = (crosswind_l_max * 10).rounded()/10
        crosswind_l = (crosswind_l * 10).rounded()/10
        headwind_l = (headwind_l * 10).rounded()/10
        qfe_land = (qfe_land * 10).rounded()/10
        mm_rt_st_land = (mm_rt_st_land * 10).rounded()/10
        vysota_land = (vysota_land).rounded()
        //m_land_max_fin = (m_land_max_fin * 10).rounded()/10
        grad_nab_vys_blok11_l = (grad_nab_vys_blok11_l * 10).rounded()/10
        grad_nab_vys_10_blok11_l = (grad_nab_vys_10_blok11_l *
10).rounded()/10
        grad_nab_vys_20_blok11_l = (grad_nab_vys_20_blok11_l *
10).rounded()/10
        grad_nab_vys_30_blok11_l = (grad_nab_vys_30_blok11_l *
10).rounded()/10
        podreb_dl_vpp_blok8_l = (podreb_dl_vpp_blok8_l).rounded()


        // ВЫВОД РЕЗУЛЬТАТОВ
       labelTest.text = " Crosswind MAX = \(Double(crosswind_l_max)) ,
Crosswind ACTUAL = \(Double(crosswind_l)) \n HEADWIND =
\(Double(headwind_l)) \n \n QFE = \(Double(qfe_land)) /
```

```
\(Double(mm_rt_st_land)) мм.рт.ст.    Elev = \(Double(vysota_land)) \n \n
MAX LANDING MASS = \(Double(m_land_max_fin)) \n \n Град.ухода на 2-й:
\(Double(grad_nab_vys_blok11_l))%, крен10 =
\(Double(grad_nab_vys_10_blok11_l))%, крен20 =
\(Double(grad_nab_vys_20_blok11_l))%, крен30 =
\(Double(grad_nab_vys_30_blok11_l))% \n \n Vref = \(Double(v_l_37)) \n V18 =
\(Double(v_l_18)), V26 = \(Double(v_l_26)) \n Потребная длина ВПП =
\(Double(podreb_dl_vpp_blok8_l))"
    }

    override func viewDidLoad() {
        super.viewDidLoad()
        // Do any additional setup after loading the view, typically from a
nib.
    }

    @IBAction func ldaTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "LDA
(Landing Distance Available) - Располагаемая посадочная дистанция",
preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func elevTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message:
"Elevation - превышение аэродрома - высота самой высокой точки ВПП
относительно уровня моря", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }


    @IBAction func slopeTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "Slope
- средний уклон между двумя концами или точками на ВПП (разность высот между
двумя указанными точками, деленная на расстояние между ними). Выражается в
```

```
процентах, перед которыми ставится знак «плюс», если уклон восходящий, или «минус», если уклон нисходящий", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler: nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func hdgTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "Heading - курс оси ВПП согласно магнитному курсу", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler: nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func kscepTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "Коэффициент сцепления ВПП", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler: nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func oatTap(_ sender:
        UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "OAT - Outside Air Temperature - температура наружного воздуха", preferredStyle: .alert)
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler: nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }

    @IBAction func qnhTap(_ sender: UITapGestureRecognizer) {
        let helpalert = UIAlertController(title: "Справка", message: "QNH - атмосферное давление в районе аэродрома, приведенное к среднему уровню моря (MSL). Указывается в гектопаскалях. Сообщается по ATIS и органами УВД по запросу экипажа", preferredStyle: .alert)
```

```
        let okbutton = UIAlertAction(title: "OK", style: .cancel, handler:
nil)
        helpalert.addAction(okbutton)
        self.present(helpalert, animated: true, completion: nil)
    }
    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }
}
```

Приложение 5. Исходные данные для ранжирования


```
{
  "fuzzy_sets": [
    {
        "name": "Недостаточная",
        "polyline": [[0,0],[32,1],[128,0]]
    },
    {
        "name": "Достаточная",
        "polyline": [[16,0],[128,1],[256,1]]
    },
    {
        "name": "Маленький экран",
        "polyline": [[0,1],[7,1],[9,0]]
    },
    {
        "name": "Средний экран",
        "polyline": [[8,0],[9,1],[11,0]]
    },
    {
        "name": "Большой экран",
        "polyline": [[9,0],[12,1],[13,1]]
    },
    {
        "name": "Менее 3 лет",
        "polyline": [[13,0],[14,1],[17,1]]
    },
    {
        "name": "Более 3 лет",
        "polyline": [[10,1],[13,1],[14,0]]
    },
    {
        "name": "Плохая эргономика",
        "polyline": [[1,1],[2,0],[3,0]]
    },
    {
        "name": "Хорошая эргономика",
        "polyline": [[1,0],[2,0.7],[3,1]]
```

```
    },
    {
        "name": "Низкая стоимость",
        "polyline": [[0,1],[25000,1],[30000,0]]
    },
    {
        "name": "Средняя стоимость",
        "polyline": [[20000,0],[30000,1],[60000,0]]
    },
    {
        "name": "Высокая стоимость",
        "polyline": [[30000,0],[50000,0.9], [150000,1]]
    },
    {
        "name": "хор",
        "polyline": [[2,0],[3,1]]
    },
    {
        "name": "отл",
        "polyline": [[3,0],[5,1]]
    },
    {
        "name": "уд",
        "polyline": [[0,0],[1,1],[2,1],[4,0]]
    },
    {
        "name": "неуд",
        "polyline": [[0,1],[1,0],[2,0]]
    },
    {
      "name": "Да",
      "membership": [["Да",1]]
    },
    {
      "name": "Нет",
      "membership": [["Нет",1]]
    }
  ],
  "criterions" : [
```

```
    {
      "name": "Память",
      "sale": ["Недостаточная", "Достаточная"]
    },
    {
      "name": "Интернет 3G/LTE",
      "scale": ["Да","Нет"]
    },
    {
      "name": "GPS/ГЛОНАСС",
      "scale": ["Да","Нет"]
    },
    {
      "name": "Диагональ экрана",
      "scale": ["Маленький экран","Средний экран", "Большой экран"]
    },
    {
      "name": "Год выпуска",
      "scale": ["Менее 3 лет", "Более 3 лет"]
    },
    {
      "name": "Эргономика в кабине",
      "scale": ["Плохая эргономика", "Хорошая эргономика"]
    },
    {
      "name": "Текущая  стоимость",
      "sale": ["Низкая стоимость", "Средняя стоимость", "Высокая
стоимость"]
    },
    {
      "name": "Итоговая оценка",
      "sale": ["неуд","уд","отл"],
      "rules": [
        { "conditions": {"Эргономика в кабине": "Плохая эргономика",
"Интернет 3G/LTE": ["Да","Нет"],
                                  "GPS/ГЛОНАСС":["Да","Нет"],
"Диагональ экрана":["Маленький экран","Средний экран", "Большой
экран"],
```

```
                                "Год выпуска": ["Более 3 лет", "Менее
3 лет"], "Память": ["Недостаточная","Достаточная"],
                                "Текущая  стоимость" :[ "Низкая
стоимость","Средняя стоимость", "Высокая стоимость"]},
          "result": "неуд"
        },
        { "conditions": {"Эргономика в кабине": ["Плохая эргономика",
"Хорошая эргономика"],  "Интернет 3G/LTE": ["Да","Нет"],
                                 "GPS/ГЛОНАСС":["Да","Нет"],
"Диагональ экрана":["Маленький экран","Средний экран", "Большой
экран"],
                                "Год выпуска": ["Более 3 лет"],
"Память": ["Недостаточная","Достаточная"],
                                "Текущая  стоимость" :[ "Низкая
стоимость","Средняя стоимость", "Высокая стоимость"]},
          "result": "неуд"
        },
       { "conditions": {"Эргономика в кабине": ["Плохая эргономика",
"Хорошая эргономика"],  "Интернет 3G/LTE": ["Да","Нет"],
                                 "GPS/ГЛОНАСС":["Да","Нет"],
"Диагональ экрана":["Маленький экран","Средний экран", "Большой
экран"],
                                "Год выпуска": ["Более 3 лет", "Менее
3 лет"], "Память": ["Недостаточная"],
                                "Текущая  стоимость" :[ "Низкая
стоимость","Средняя стоимость", "Высокая стоимость"]},
          "result": "неуд"
        },
        { "conditions": {"Эргономика в кабине": ["Хорошая
эргономика"],  "Интернет 3G/LTE": ["Да","Нет"],
                                 "GPS/ГЛОНАСС":["Да","Нет"],
"Диагональ экрана":["Маленький экран","Средний экран", "Большой
экран"],
                                "Год выпуска": ["Более 3 лет", "Менее
3 лет"], "Память": ["Недостаточная"],
                                "Текущая  стоимость" :[ "Низкая
стоимость","Средняя стоимость", "Высокая стоимость"]},
          "result": "уд"
        },
```

```
        { "conditions": {"Память": ["Достаточная"], "Интернет
3G/LTE": "Да",
                                  "GPS/ГЛОНАСС":"Да", "Диагональ
экрана":["Средний экран", "Большой экран"],
                                  "Год выпуска": "Менее 3 лет",
"Эргономика в кабине": "Хорошая эргономика",
                                  "Текущая  стоимость" :["Средняя
стоимость", "Высокая стоимость"]
                                  },
          "result": "хор"
        },
       { "conditions": {"Память": ["Достаточная"], "Интернет 3G/LTE":
"Да",
                                  "GPS/ГЛОНАСС":"Да", "Диагональ
экрана":["Средний экран", "Большой экран"],
                                  "Год выпуска": "Менее 3 лет",
"Эргономика в кабине": "Хорошая эргономика",
                                  "Текущая  стоимость" : ["Средняя
стоимость","Низкая стоимость"]
                                  },
          "result": "отл"
        }
      ]
    }
  ],
  "alternatives" : [
{"Память": 32,  "Интернет 3G/LTE": "Нет",  "GPS/ГЛОНАСС": "Нет",
     "Диагональ экрана": 7.9,   "Год выпуска": 2013,  "Эргономика в
кабине": 3,     "Текущая  стоимость": 19990},
{"Память": 32,  "Интернет 3G/LTE": "Да",   "GPS/ГЛОНАСС": "Да",
     "Диагональ экрана": 7.9,   "Год выпуска": 2013,  "Эргономика в
кабине": 3,     "Текущая  стоимость": 29990},
{"Память": 32,  "Интернет 3G/LTE": "Нет",  "GPS/ГЛОНАСС": "Нет",
     "Диагональ экрана": 7.9,   "Год выпуска": 2015,  "Эргономика в
кабине": 3,     "Текущая  стоимость": 29990},
{"Память": 32,  "Интернет 3G/LTE": "Да",   "GPS/ГЛОНАСС": "Да",
     "Диагональ экрана": 7.9,   "Год выпуска": 2015,  "Эргономика в
кабине": 3,     "Текущая  стоимость": 39990},
```

{"Память": 128, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 7.9, "Год выпуска": 2015, "Эргономика в кабине": 3, "Текущая стоимость": 36990},
{"Память": 128, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 7.9, "Год выпуска": 2015, "Эргономика в кабине": 3, "Текущая стоимость": 46990},
{"Память": 32, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 9.7, "Год выпуска": 2014, "Эргономика в кабине": 2, "Текущая стоимость": 29990},
{"Память": 32, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 9.7, "Год выпуска": 2014, "Эргономика в кабине": 2, "Текущая стоимость": 39990},
{"Память": 128, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 9.7, "Год выпуска": 2014, "Эргономика в кабине": 2, "Текущая стоимость": 36990},
{"Память": 128, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 9.7, "Год выпуска": 2014, "Эргономика в кабине": 2, "Текущая стоимость": 46990},
{"Память": 32, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 12.9, "Год выпуска": 2015, "Эргономика в кабине": 1, "Текущая стоимость": 58990},
{"Память": 128, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 12.9, "Год выпуска": 2015, "Эргономика в кабине": 1, "Текущая стоимость": 65990},
{"Память": 256, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 12.9, "Год выпуска": 2015, "Эргономика в кабине": 1, "Текущая стоимость": 72990},
{"Память": 128, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 12.9, "Год выпуска": 2015, "Эргономика в кабине": 1, "Текущая стоимость": 75990},
{"Память": 256, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 12.9, "Год выпуска": 2015, "Эргономика в кабине": 1, "Текущая стоимость": 82990},
{"Память": 32, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 9.7, "Год выпуска": 2016, "Эргономика в кабине": 2, "Текущая стоимость": 44990},
{"Память": 128, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 9.7, "Год выпуска": 2016, "Эргономика в кабине": 2, "Текущая стоимость": 51990},

{"Память": 256, "Интернет 3G/LTE": "Нет", "GPS/ГЛОНАСС": "Нет",
"Диагональ экрана": 9.7, "Год выпуска": 2016, "Эргономика в кабине": 2, "Текущая стоимость": 58990},
{"Память": 32, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 9.7, "Год выпуска": 2016, "Эргономика в кабине": 2, "Текущая стоимость": 54990},
{"Память": 128, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 9.7, "Год выпуска": 2016, "Эргономика в кабине": 2, "Текущая стоимость": 61990},
{"Память": 256, "Интернет 3G/LTE": "Да", "GPS/ГЛОНАСС": "Да",
"Диагональ экрана": 9.7, "Год выпуска": 2016, "Эргономика в кабине": 2, "Текущая стоимость": 68990}
]
}

Приложение 6. Результаты расчета ранга альтернатив аппаратной платформы системы


```
[
  {
    "Память": 32,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 7.9,
    "Год выпуска": 2013,
    "Эргономика в кабине": 3,
    "Текущая  стоимость":
19990,
    "Итоговая оценка":
1.6515151515151514,
    "mu": [
      [
        0.0,
        0.55002
      ],
      [
        0.5,
        0.27501
      ],
      [
        1.0,
        0.55002
      ],
      [
        2.0,
        0.55002
      ],
      [
        3.0,
        0.27501
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 7.9,
    "Год выпуска": 2013,
    "Эргономика в кабине": 3,
    "Текущая  стоимость":
29990,
    "Итоговая оценка":
1.6515151515151514,
    "mu": [
      [
        0.0,
        0.55002
      ],
      [
        0.5,
        0.27501
      ],
      [
        1.0,
        0.55002
      ],
      [
        2.0,
        0.55002
      ],
      [
```

```
        3.0,
        0.27501
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 7.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 3,
    "Текущая  стоимость":
29990,
    "Итоговая оценка":
1.6515151515151514,
    "mu": [
      [
        0.0,
        0.55002
      ],
      [
        0.5,
        0.27501
      ],
      [
        1.0,
        0.55002
      ],
      [
        2.0,
        0.55002
      ],
      [
        3.0,
        0.27501
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 7.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 3,
    "Текущая  стоимость":
39990,
    "Итоговая оценка":
1.6515151515151514,
    "mu": [
      [
        0.0,
        0.55002
      ],
      [
        0.5,
        0.27501
      ],
      [
        1.0,
        0.55002
```

```
      ],
      [
        2.0,
        0.55002
      ],
      [
        3.0,
        0.27501
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 7.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 3,
    "Текущая  стоимость":
36990,
    "Итоговая оценка": 0,
    "mu": [
      [
        0.0,
        0.0
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 7.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 3,
    "Текущая  стоимость":
46990,
    "Итоговая оценка": 0,
    "mu": [
      [
        0.0,
        0.0
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
```

```
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2014,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
29990,
    "Итоговая оценка":
1.6515173129477192,
    "mu": [
      [
        0.0,
        0.65001
      ],
      [
        0.5,
        0.32501
      ],
      [
        1.0,
        0.65001
      ],
      [
        2.0,
        0.65001
      ],
      [
        3.0,
        0.32501
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2014,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
39990,
    "Итоговая оценка":
1.8941881133108442,
    "mu": [
      [
        0.0,
        0.65001
      ],
      [
        0.5,
        0.32501
      ],
      [
```

```
        1.0,
        0.65001
      ],
      [
        2.0,
        0.65001
      ],
      [
        3.0,
        0.32501
      ],
      [
        3.56044,
        0.14286
      ],
      [
        4.0,
        0.14286
      ],
      [
        5.0,
        0.14286
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2014,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
36990,
    "Итоговая оценка": 0,
    "mu": [
      [
        0.0,
        0.0
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2014,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
46990,
    "Итоговая оценка":
3.733333333333333,
    "mu": [
      [
        0.0,
        0.0
      ],
```

```
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.65001
      ],
      [
        4.0,
        0.65001
      ],
      [
        5.0,
        0.65001
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 12.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 1,
    "Текущая  стоимость":
58990,
    "Итоговая оценка":
0.33333333333333337,
    "mu": [
      [
        0.0,
        0.90899
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 12.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 1,
    "Текущая  стоимость":
65990,
    "Итоговая оценка":
0.3333333333333333,
    "mu": [
      [
        0.0,
        0.91599
      ],
      [
        1.0,
```

```
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 256,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 12.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 1,
    "Текущая  стоимость":
72990,
    "Итоговая оценка":
0.3333333333333333,
    "mu": [
      [
        0.0,
        0.92299
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 12.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 1,
    "Текущая  стоимость":
75990,
    "Итоговая оценка":
0.3333333333333333,
    "mu": [
      [
        0.0,
        0.92599
      ],
      [
        1.0,
        0.0
      ],
```

```
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 256,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 12.9,
    "Год выпуска": 2015,
    "Эргономика в кабине": 1,
    "Текущая  стоимость":
82990,
    "Итоговая оценка":
0.33333333333333337,
    "mu": [
      [
        0.0,
        0.93299
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2016,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
44990,
    "Итоговая оценка":
1.6515173129477192,
    "mu": [
      [
        0.0,
        0.65001
      ],
      [
        0.5,
        0.32501
      ],
      [
        1.0,
```

```
        0.65001
      ],
      [
        2.0,
        0.65001
      ],
      [
        3.0,
        0.32501
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2016,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
51990,
    "Итоговая оценка": 0,
    "mu": [
      [
        0.0,
        0.0
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 256,
    "Интернет 3G/LTE": "Нет",
    "GPS/ГЛОНАСС": "Нет",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2016,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
58990,
    "Итоговая оценка": 0,
    "mu": [
      [
        0.0,
        0.0
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
```

```
        0.0
      ],
      [
        3.0,
        0.0
      ],
      [
        4.0,
        0.0
      ],
      [
        5.0,
        0.0
      ]
    ]
  },
  {
    "Память": 32,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2016,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
54990,
    "Итоговая оценка":
1.8941881133108442,
    "mu": [
      [
        0.0,
        0.65001
      ],
      [
        0.5,
        0.32501
      ],
      [
        1.0,
        0.65001
      ],
      [
        2.0,
        0.65001
      ],
      [
        3.0,
        0.32501
      ],
      [
        3.56044,
        0.14286
      ],
      [
        4.0,
        0.14286
      ],
      [
        5.0,
        0.14286
      ]
    ]
  },
  {
    "Память": 128,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2016,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
61990,
    "Итоговая оценка":
3.733333333333333,
    "mu": [
      [
        0.0,
        0.0
      ],
```

```
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.65001
      ],
      [
        4.0,
        0.65001
      ],
      [
        5.0,
        0.65001
      ]
    ]
  },
  {
    "Память": 256,
    "Интернет 3G/LTE": "Да",
    "GPS/ГЛОНАСС": "Да",
    "Диагональ экрана": 9.7,
    "Год выпуска": 2016,
    "Эргономика в кабине": 2,
    "Текущая  стоимость":
68990,
    "Итоговая оценка":
3.733333333333333,
    "mu": [
      [
        0.0,
        0.0
      ],
      [
        1.0,
        0.0
      ],
      [
        2.0,
        0.0
      ],
      [
        3.0,
        0.65001
      ],
      [
        4.0,
        0.65001
      ],
      [
        5.0,
        0.65001
      ]
    ]
  }
]
```