МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

кафедра компьютерных технологий

«Допущен к защите»
Заведующий кафедрой Курмалбаев З.К.
д. ф.-м. н., проф.
(Ф.И.О., ученая степень, звание)
_____ «_____»_____ **20__ г.**
(подпись)

ДИПЛОМНЫЙ ПРОЕКТ

На тему: „Разработка структуры базы знаний для прогноза футбольных ставок"

Специальность „5В070400 - Вычислительная техника и ПО

Выполнил (а) Жарлухалбетов А.Б. ПО-12-2
(Фамилия и инициалы) группа

Научный руководитель Ахметова М.А., к.т.н., доцент
(Фамилия и инициалы, ученая степень, звание)

Консультанты:

по экономической части:
Бекенова А.Н., к.э.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «05» 05 2016 г.
(подпись)

по безопасности жизнедеятельности:
Тихобаева М.Г. д.х.н., профессор
(Фамилия и инициалы, ученая степень, звание)
_____ «21» 04 2016 г.
(подпись)

по применению вычислительной техники:
Ахметова М.А, к.т.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «31» 05 2016 г.
(подпись)

Нормоконтролер: Ахметова М.А, к.т.н., доцент
(Фамилия и инициалы, ученая степень, звание)
_____ «31» 05 20__ г.
(подпись)

Рецензент: Балгабаева Л.Ш., к.т.н.,
(Фамилия и инициалы, ученая степень, звание)
_____ «_____»_____ 20__ г.
(подпись)

Алматы 2016 г.

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ КАЗАХСТАН

Некоммерческое акционерное общество
АЛМАТИНСКИЙ УНИВЕРСИТЕТ ЭНЕРГЕТИКИ И СВЯЗИ

Факультет Аэрокосмических и информационных технологий
Специальность вычислительная техника и программное обеспечение
Кафедра Компьютерных технологий

ЗАДАНИЕ
на выполнение дипломного проекта

Студент Шарлухамбетов Арман Болатович
(фамилия, имя, отчество)

Тема проекта Разработка структуры базы знаний для прогноза футбольных ставок

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

Перечень подлежащих разработке дипломного проекта вопросов или краткое содержание дипломного проекта:
1. Принципы проектирования
2. Классификация моделей представления знаний
3. Разработка базы знаний
4 Технико-экономическая часть
5 Безопасность жизнедеятельности

Перечень графического материала (с точным указанием обязательных чертежей)

Рисунки структуры приложения

Рисунки моделей представления знаний

Рисунки, описывающие требования к приложению

Рисунки и таблицы базы знаний

Таблицы технико-экономической части

Рисунки из части безопасности жизнедеятельности

Рекомендуемая основная литература

Гаврилова - Базы знаний интеллектуальных систем: Пособие 2003. - С 234

Ахметова М.А - Системы искусственного интеллекта: Конспект лекций. 2013 - 46 с.

Нильсон Н. - Искусственный интеллект. Методы поиска решения, 1973 - 202 с.

Консультанты по проекту с указанием относящихся к ним разделов

Раздел	Консультант	Сроки	Подпись
БЖД	Приходько Н.Г.	01.04 - 21.04.16	
Экономич. часть	Бекенеева А.Ч.	21.04 - 05.05.16	
Принципы проектир.	Ахметова М.А	24.02.16	
Модели предст. знаний	Ахметова М.А.	08.03.16	
Разработка базы знаний	Ахметова М.А.	15.04.16	
Нормоконтроль	Ахметова М.А	31.05.16	

Г Р А Ф И К
подготовки дипломного проекта

№ п/п	Наименование разделов, перечень разрабатываемых вопросов	Сроки представления руководителю	Примечание
1.	Принципы проектирования	24.02.16	
	База знаний и интеллектуальные системы		
	Технологии и языки разработки баз знаний		
	Исследование предметной области		
2.	Классификация моделей представления знаний	08.03.16	
	Продукционная модель		
	Фреймовая модель		
	Семантические сети		
3	Разработка базы знаний	15.04.16	
	Подготовка к работе		
	Проектирование базы знаний		
	Результаты работы приложения		
4	Технико-экономическая часть	05.05.16	
5	Безопасность жизнедеятельности	21.04.16	

Дата выдачи задания «14» сентября 2015 г.

Заведующий кафедрой _____ Курмалбаев З.К.
(подпись) (Фамилия и инициалы)

Руководитель _____ Ахметова М.А.
(подпись) (Фамилия и инициалы)

Задание принял к исполнению
студент _____ Жорлумхалбетов А.Б.
(подпись) (Фамилия и инициалы)

Аңдатпа

Осы дипломдық жұмыстың негізгі мақсаты Android платформасындағы мобильдік қосымшаға арналған білім қорын құру.

Аталған мақсатты орындау үшін болжау және білімді ұсыну әдістерін зерттеу міндеттері мен білім үлгісін таңдау мәселесі қойылды. Осындай білім үлгісі негізінде құрылған мобильдік қосымша пайдаланушыларға спорт әлеміндегі жаңалықтарды көруге, сарапшылардың болжамын алуға, спорттық оқиғаға ұтыс тігуге мүмкіндік береді. Осы қосымша "Play Market" қосымшалар дүкенінде жариялануы мүмкін.

Дипломдық жұмыста, әзірлеудің шығындар көлемі көрсетілген және экономикалық тиімділігін растайтын есептеулер жүргізілген. Сонымен қатар, қосымша әзірленген бөлмеге зияндылық жайлы талдау өткізілген.

Аннотация

Целью данной дипломной работы является разработка базы знаний для мобильного приложения под платформу Android.

Для выполнения цели поставлены задачи исследование методов прогнозирования и представления знаний, выбор модели знаний. Мобильное приложение с такой базой знаний дает возможность пользователям просмотреть новости из мира спорта, получить прогноз от экспертов, сделать ставку на спортивное событие. Данное приложение может быть опубликовано в магазине приложений «Play Market».

В завершении были произведены расчеты, которые показывают объем затрат на разработку и подтверждают экономическую эффективность. Кроме того, был проведен поверхностный анализ помещения, в котором велась разработка.

Annotation

In this graduation project considered the questions about development of knowledge base for mobile application for the Android operating system.

The main part of the work is to study the methods of forecasting and the study of knowledge representation models. The mobile application allows users to view sport news from over the world, a forecast of experts, to make a bet on a sporting event. This application may be published in the store «Play Market» applications.

Finally was made the calculations that show development cost and confirming economic efficiency. Also was analyzed the working conditions in the office where project was developed.

Содержание

Введение

В нынешнее время значительно увеличилось количество букмекерских контор и возросла популярность ставок в области спорта. Сфера ставок у букмекеров растет и затрагивает большое количество людей в нашей стране. Аналитики и эксперты составляют прогнозы и делятся мнением и взглядом на матч, игроки ищут информацию, совещаются и делают свои ставки.

Высокий показатель пользования смартфонов и мобильных ПК говорит о том, что все чаще люди пользуются гаджетами и регулярно устанавливают приложения для них. Человек в любой момент может воспользоваться смартфоном, это очень удобно и вся необходимая информация всегда под рукой. На этом основании было принято решение разработать мобильное приложение.

Для достижения указанной цели необходимо проделать анализ знаний экспертов, которые делают прогноз футбольных ставок. Поэтому необходимо было сделать постановку задач исследование методов прогнозирования и представления знаний, выбор модели знаний. Мобильное приложение с такой базой знаний дает возможность пользователям просмотреть новости из мира спорта, получить прогноз от экспертов, сделать ставку на спортивное событие.

Разработка базы знаний является важным компонентом для любого мобильного приложения, в котором используются знания экспертов и логические правила. База знаний содержит в себе информацию о человеческом опыте и знаниях в определенной предметной области, в данном случае в области прогнозирования футбольных ставок. Современные базы знаний работают совместно с системами поиска информации, для которых требуется формат представления знаний.

Создание данного мобильного приложения востребовано, так как оно поможет всем людям, заинтересованных в сфере ставок, находить всю необходимую новостную информацию, узнавать мнения экспертов, просматривать таблицу коэффициентов на матч. Данное мобильное приложение упрощает жизнь людям, оно объединяет различные информационные спортивные сайты в единое приложение на телефоне, пользователям которого более не потребуется вести поиск информации одновременно в нескольких источниках.

1 Принципы проектирования

1.1 База знаний и интеллектуальные системы

Термин «искусственный интеллект» - ИИ – был впервые использован в 1956г на семинаре в США. Семинар был проведен в целях создания методов изучения и вычисления логических, а не вычислительных задач. В ближайшее время после подтверждения искусственного интеллекта самостоятельной областью науки состоялось разделение этой области на пару направлений[1]:

– нейрокибернетика;

– кибернетика «черного ящика».

Названные направления прогрессируют почти что независимо, видным образом имеют отличия как в методологических аспектах, так и в технологических. И только в нынешнее время пробиваются ростки объединения направлений вновь в общую область.

Зарождение нейрокибернетики. Центральную идею данного направления следует озвучить следующим образом: единственный объект, способный мыслить, - это человеческий мозг. Исходя из этого, любая рассуждающая машинная система должна быть аналогична его структуре. Таким образом, нейрокибернетика направлена на составление моделей программно-аппаратных структур, похожих на структуру мозга. Физиологами было рассчитано, что фундаментом человеческого мозга является огромное количество (до 10^{21}) переплетенных и взаимно работающих нервных клеток – нейронов. Из-за этого, направление нейрокибернетики было централизовано на создании микрочастиц, имеющих аналогичные нейронам функции, и их соединение в рабочие сетевые системы. Этим системам дали название «нейронные сети», или нейросети.

Первые нейросети были созданы американскими нейрофизиологами Френком Розенблаттом и Мак-Каллоком в 1956-1965 гг. Их стремление сотворить системы, имитирующие человеческий глаз и его связь с мозгом. Устройство, созданное ими в те года, было названо персептрон (perceptron).

Искусственные нейронные сети (ИНС) — математические прототипы, а также их платформенные или аппаратные осуществления, спроектированные по принципу структуры и жизнедеятельности биологических нейронных сетей —одушевленного живого организма. Это определение появилось при рассмотрении процессов, происходящих в мозге при умственной работе, и при попытке создать модели этих процессов. Первой моделью такого типа был персептрон. В дальнейшем эти модели используются в целях практического изучения, обычно в задачах предсказания и прогнозирования.

ИНС предполагают собой систему объединенных и взаимосвязанный между собой обычных процессоров (искусственных нейронов). Эти процессоры традиционно достаточно элементарны, в особенности в сопоставлении с процессорами, которые применяются в персональных

компьютерах. Любой процессор схожей сети обрабатывается только сигналами, которые он периодически передает, и сигналами, которые он систематически принимает от других процессоров. И тем не менее, будучи объединенными в довольно огромную сеть с контролируемым взаимодействием, такие не сложные местные процессоры в совокупности могут разбираться с многоэтажными задачи.

Нейронные сети не пишутся кодом в обычном смысле этого слова, они обучаются. Умение обучаться — одно из главных достоинств нейронных сетей перед классическими алгоритмами. Технически обучение состоит в поиске коэффициентов взаимосвязей нейронов. В процессе обучения нейронная сеть может обнаруживать трудные зависимости между input данными и output, а также обобщать их. Это означает, что при удачном обучения, сеть сумеет возвращать правдивый итог на основании данных, которых не было в секции обучения.

От кибернетики «черного ящика» к ИИ. Фундаментом данного подхода был заложен принцип, обратный нейрокибернетике. Не важно, как работает «мыслящий» аппарат. Важно, чтобы на входные сигналы влияния оно давало реакцию такую же, как человеческий мозг. Последователи этого направления объяснили свой метод так, что человек не обязан беспрекословно подчинятся законам природы в своих научных достижениях. Данная направленность ИИ была сотворена для розыска алгоритмов решения закономерных задач с поддержкой созданных ранее моделей эвм. В конце 50-х годов появилась схема лабиринтного розыска, данный способ ставит задачу как некое пространство условий в виде графа, и в данном графе ведется розыск рационального пути от входящих данных к итоговым. Начало 60-х – эра эвристического создания программ. Эвристика – подход, теоретически никак не аргументированный, который дает возможность уменьшить численность переборов в месте розыска. Взяв начало с середины 80-х годов, везде где только можно проистекает коммерциализация искусственного разума. Растут ежегодные финансовые вложения, формируются промышленные экспертные системы. Вырастает энтузиазм к самообучающимся системам. Искусственный разум становится одной из более многообещающих и популярных областей информатики и программирования.

База знаний — логическая база, содержащая логические правила вывода и познания о человеческом опыте и знаниях в определенной предметной области. В самообучающихся системах основа познаний еще охватывает информацию, которая является итогом решения прошлых задач. Инновационные базы познаний действуют вместе с системами розыска и извлечения информации в виде текста или изображений. Для такого поиска потребуется некая модель классификации мнений и установленный формат представления. Иерархический метод представления в базе знаний комплекта мнений и их взаимоотношений именуется онтологией.

Онтологию некой сферы познаний совместно с данными о свойствах определенных элементов нередко именуют «основанием познаний». Совместно с тем, настоящие базы познаний (в различие от обыкновенной базы данных) имеют никак не лишь практическую информацию, однако и порядок вывода, предоставляющий возможность работать автоматические выводы о теснее наличествующих либо опять вводимых прецедентах и тем наиболее создавать семантическую (разумную) отделку информации.

Сфера наук об искусственном интеллекте, рассматривающая базы знаний и методы манипуляции со знаниями, называется инженерией знаний. База знаний — важнейшая составляющая интеллектуальной системы. Наиболее распространенный тип таких программ — это экспертные системы. Их суть состоим в поиске вариантов решения проблем из определенной предметной области, отталкиваясь от записей правил в БЗ и в описании ситуации на пользовательском уровне.

Обыкновенные базы знаний имеют все шансы употребляться для сотворения экспертных систем сохранения информации в некой компании: документы, руководства, заметки тех снабжения. Основная причина сотворения таковых баз — посодействовать наименее искусным людям отыскать теснее имеющееся отображение метода решения какой-никакой-или трудности. Двумя наиболее необходимыми условиями к данным, находящимся в складе познаний интеллектуальной концепции, считаются аутентичность определенных и общих сведений, наличествующих в складе данных и релевантность инфы, получаемой с поддержкой принципов вывода базы знаний.

Ниже приведены особенности, которые могут (но не всегда) быть у концепции, управляющей базами знаний.

– автоматическое доказательство (вывод). Умение концепции создавать обновленные знания из устаревших, обнаруживать закономерности в БЗ. Часто предполагается, что база знаний имеет различия относительно базы данных именно присутствием механизма вывода;

– доказательство заключения. Умение концепции после посыла ответа «объяснить» направление её логических суждений, причем «по первому требованию»;

– интроспекция. Обнаружение противоречий, нестыковок в БЗ, управление верной структурой БЗ;

– машинное обучение. Трансформация БЗ в приспосабливаемую систему, адаптирующейся к нужно предметной области. Схожа со способностью живых существ «набирать опыт».

Знания в базе знаний можно разделить на алгоритмические и неалгоритмические.

– алгоритмические (процедурные) знания – это методы (программы, процедуры), вычисляющие процессы, исполняющие преображения, вычисляющие конкретные определенные задачи;

– неалгоритмические знания – состоит из придуманных в голове элементов, называемых понятиями. Понятие обычно имеет имя, значение, модель (составные элементы), оно связано с остальными понятиями и входит в какую-то систему понятий. Другие неалгоритмические знания - это связи между понятиями или утверждения о свойствах понятий и связях между ними.

На деле почти во всех экспертных системах и СБЗ содержание базы знаний делится на «прецеденты» и «принципы». Прецеденты – простейшие крупицы знания (простые утверждения о параметрах элемента). Принципы важны для демонстрации взаимодействия, зависимостей между прецедентами и их совокупности.

Прямое внедрение познаний из базы познаний для решения задач гарантируется приспособлением получения решений (inference engine – машинка вывода) – операцией розыска, составления плана, решения. Устройство решения отчуждает вероятность получать из базы познаний ответы на вопросы, обретать решения, формулируемые в определениях мнений, хранящихся в складе.

Примеры запросов:

– найти объект, удовлетворяющий заданному условию;

– какие действия нужно выполнить в такой ситуации и т.д.

Интерфейс – гарантирует работу с основанием познаний и приспособлением получения решений на языке высочайшего значения, приближенном к специфическому языку профессионалов в той практический области, к которой относится СБЗ.

Классические языки программирования – C, Basic, Pascal, Lisp и др. Особенно в данном ряду стоит отметить язычок многофункционального программирования Lisp. Его главные характеристики: эти представляются в облике списков, для получения решений употребляется рекурсия. Языки представления познаний (эти как Prolog) – имеют специальные средства описания познаний и интегрированный устройство розыска вывода.

Введение в экспертные системы. Определение и структура. В качестве рабочего определения экспертной системы примем последующее. Экспертные системы (ЭС) — наиболее трудные программные ансамбли, трансформирующие познания профессионалов в определенных предметных областях и тиражирующие данный экспериментальный процесс для консультаций с наименее обученными людьми.

В общем путь работы ЭС можно предоставить последующим образом: человек, заинтересованный в получении нужной информации, при помощи пользовательского интерфейса отправляет запрос к ЭС; решатель, воспользовавшись основанием познаний, генерирует и дает человеку пригодный совет, изъясняя ход собственных размышлений при поддержки подсистемы разъяснений.

Так как область исследования ЭС перманентно преображается, выведем основные термины в рамках данной работы.

Юзер — спец предметной области, для которого специализирована система. Традиционно его квалификация низка, и потому он имеет необходимость в поддержке и помощи собственной трудоспособности со стороны ЭС.

Эксперт по знаниям — спец в области искусственного происхождения разума, выступающий в роли промежуточного буфера между знатоком и основанием познаний.

Интерфейс юзера — набор программ, составляющих разговор юзера с ЭС как на стадии ввода данных, так и при получении итогов.

Основа познаний (БЗ) — ядро ЭС, система познаний предметной области, загруженная на механический обладатель в виде, ясном знатоку и обычному юзеру (традиционно на специфичном языке, похожем на природный). Синхронно этому «человеческому» виду есть БЗ во внутреннем «машинном» виде.

Решатель —подпрограмма, имитирующая цепь размышлений профессионала на основании познаний, наличествующих в БЗ. Синонимы: дедуктивная машинка, машинка вывода, блок закономерного вывода.

Подсистема разъяснений — подпрограмма, дозволяющая юзеру заполучить ответы на запросы: «Как была получена та либо другая рекомендация?» и «Отчего система обрела это заключение?» Протест на запрос «как» — наверное трассировка только процесса получения решения с указанием принятых на вооружение фрагментов БЗ, то имеется всех шагов цепи выводов. Протест на запрос «отчего» — гиперссылка на заключение, конкретно предшествовавшее приобретенному решению, то имеется отход на шаг обратно. Развитые подсистемы разъяснений поддерживают и остальные разновидности запросов.

Умственный редактор БЗ — подпрограмма, представляющая знатоку по знаниям вероятность проектировать БЗ в диалоговом режиме. Подключает в себя систему вложенных рацион, шаблонов языка представления познаний, подсказок («help» — режим) и остальных гарантийных средств, делающих легче работу с основанием.

Классифицирование систем, основанных на познаниях. Классифицирование решаемого задания.

1. Интерпретация данных. Наверное одна из обычных задач для экспертных систем. Перед интерпретацией понимается процесс определения значения данных, итоги которого обязаны существовать согласованными и корректными. Традиционно учитывается альтернативный тест данных.

2. Диагностика. Перед диагностикой понимается процесс соотнесения объекта с неким классом объектов и/либо обнаружение поломке в некой системе. Поломка — это аномалия от общепризнанных мерок. Таковая трактовка дозволяет с единичных теоретических позиций разглядывать и поломка оснащения в технических системах, и болезни живых организмов, и различные естественные странности. Принципиальной особенностью

считается тут надобность осмысливания многофункциональной текстуры («анатомии») диагностирующей системы.

3. Мониторинг. Основная задача мониторинга — постоянное преображение данных в действительном размере времени и оповещение о выходе тех или иных параметров за допустимые рамки. Главные недостатки — «пропуск» опасной ситуации и обратный процесс «ложного» срабатывания. Сложность этих недостатков в неопределенности признаков внештатных моментов и обязанность записи временного контекста.

4. Проектирование. Конструирование состоит в подготовке спецификаций на творение «объектов» с заблаговременно явными качествами. Перед спецификацией понимается целый комплект нужных документов — чертеж, пояснительная записка и т. д. Главные трудности тут — приобретение точного структурного описания познаний о объекте и неувязка «отпечатка». Для организации действенного проектирования и в еще большей ступени перепроектирования нужно сформировывать никак не лишь сами проектные решения, однако и темы их принятия. Таковым образом, в заданиях проектирования тесновато связываются 2 главных процесса, исполняемых в рамках соответственной ЭС: процесс вывода решения и процесс разъяснения.

5. Прогнозирование. Предсказание дозволяет предвещать результаты неких событий либо явлений на основании разбора наличествующих данных. Прогнозирующие концепции логически выдают вполне возможные следствия из данных обстановок. В прогнозирующей системе традиционно употребляется параметрическая динамическая модель, в которой смысла характеристик «подгоняются» перед установленную обстановку. Выходящие из данной модели следствия сочиняют базу для мониторингов с оценками шанса.

1.2 Технологии и языки разработки базы знаний

Этот этап описывает деятельность, предшествующую решению начать разрабатывать конкретную ЭС. Он включает:

- определение проблемной области и задачи;
- поиск эксперта, согласившегося работать вместе при изучении проблемы;
- определение предварительного подхода к решению проблемы;
- анализ затрат и прибыли от создания проекта;
- подготовку подробного плана разработки.

Верный отбор трудности дает, наверняка, самую критическую дробь исследования в целом. Если избрать неуместную дилемму, разрешено совсем скоро вязнуть в "топком месте" проектирования задач, которые никто никак не понимает, как улаживать. Неуместная неувязка имеет возможность еще привести к творению экспертной системы, стоящая гораздо более, нежели хранящая. Ремесло станет обстоять еще ужаснее, если создать систему, которая действует, однако никак не применима для юзеров. В том числе и

если исследование производится лично организацией для личных целей, данная фаза считается пригодным фактором для получения советов снаружи, чтоб обеспечивать успешно избранный и легко осуществимый с тех. точки зрения начальный план.

При выборе области внедрения надлежит учесть, что если познание, нужное для решения задач, неизменное, отчетливо формулируемое, и соединено с вычислительной отделкой, то обыденные алгоритмические компилятор, следуя всей вероятности, станут наиболее подходящим методом решения нестыковок в данной области. Экспертная система ни в коем случае никак не уничтожит надобность в реляционных базах данных, статистическом программном трансляторе, электрических таблицах и системах текстовой отделки. Однако если действенность задания находится в зависимости от познания, которое считается необъективным, изменяющимся, символьным либо вытекающим отчасти из суждений здорового значения, тогда область имеет возможность продуманно ходить кандидатом на экспертную систему.

Найдем факторы, заключающие о надобности создания и использования экспертных систем:

- недостаток знатоков, которые тратят значимую часть времени для предоставления помощи остальным;
- потребность в многочисленной группе экспертов, поскольку ни один из них не располагает идеальным полным знанием;
- уменьшенная производительность труда, поскольку задание просит полного анализа - сложное набора условий, а рядовой эксперт не в состоянии просмотреть (за предоставленное время) все эти условия;
- значимая разница между решениями сильнейших и худших исполнителей;

Подходящие задачи имеют следующие характеристики:

- являются узкоспециализированными;
- не зависят в какой-либо степени от общих человеческих познаний или здравых рассуждений;
- не являются для знатока ни слишком простыми, ни слишком трудными (время, которое потратит эксперт для того, чтобы справится с проблемой, может составлять от двух часов до трех-четырех недель);
- условия выполнения задания выбираются самим юзером системы;
- имеет результаты, которым можно дать оценку.

Традиционно экспертные системы разрабатываются маршрутом получения специфичных познаний от профессионала и ввода их в систему. Некие системы имеют все шансы владеть стратегиями индивидуума. Следственно, отыскать пригодного профессионала - наиболее главный шаг в разработке экспертных систем. В процессе исследования и следующего расширения системы эксперт по знаниям и разработчик традиционно действуют совместно. Эксперт по знаниям подсобляет профессионалу

структурировать познания, предопределять и формализовать мнения и принципы, нужные для решения трудностей. Во время начальных разговоров они решают, станет ли их совместная работа эффективной. Очень важно, так как две стороны станут действовать совместно, согласно наименьшей мерке, в продолжении одного года. Не считая их, в коллектив создателей будет целенаправленно подключить возможных юзеров и профи разработчиков программного обеспечения.

Подготовительный подготовка к программной разработке задания ориентируется на основе параметров задания и ресурсов, доступных на ее заключение. Знаток по знаниям выдвигает традиционно некоторое количество разновидностей, связанных с внедрением наличествующих в сфере программных средств. Конечный отбор вероятен только на шаге исследования макета. В завершении того, как задание определено, нужно подсчитать затраты и выгоды от исследования экспертной системы. В затраты относятся издержки на плату труда коллектива проектировщиков. В побочные затраты покупаемого программного инвентаря, с поддержкой которого проектируется экспертная система.

Выручку можно получить с помощью понижения цены конечного продукта, увеличения производительности труда, расширения бумажных документов продукта или услуг или даже создание новых типов продукта или услуг в данной области. Надлежащие затраты и выгоды от системы ориентируются опираясь на время, на протяжении которого окупятся финансы, затраченные на исследование. На нынешней ступени крупная часть компаний, развивающих огромные экспертные системы, выбрали создавать дорогие планы, дающие значимые выгоды.

Наметились зависимости исследования наименее дорогих систем, несмотря на более долгий сроком возвращаемости затраченных на них финансов, так как платформенные методы исследования экспертных систем постоянно улучшаются. Впоследствии, эксперт по знаниям удостоверился, что предоставленное задание имеет возможность быть решенной с поддержкой экспертной системы:

- экспертную систему можно спроектировать предоставленными на рынке инструментами;
- доступен необходимый эксперт;
- предложенные параметры разработки не являются невозможными;
- расходы и время их возвращаемости согласованы с заказчиком.

Этапы разработки экспертных систем. На рисунке 1.1 показана схема последовательности этапов разработки и проектирования экспертных систем.

Рисунок 1.1 – Этапы разработки

При проектировании экспертных систем нередко употребляется теория скорого макета. Сущность ее в последующем: сначала формируется не сама экспертная система, а ее макет, обязанный исправлять узкий набор проблем и запрашивать на собственное исследование небольшие сроки. Макет обязан показывать целесообразность грядущей экспертной системы для предоставленной предметной области, испытать верность шифровки прецедентов, взаимосвязей и стратегий размышления профессионала. Помимо этого, он предоставляет шанс знатоку по знаниям пригласить профессионала к функциональной роли в исследовании экспертной системы. Величина макета –несколько десятков принципов.

На сегодняшний день появилась закономерная тенденция создания экспертных систем, содержащая 6 этапов.

Этап 1. Идентификация

Рассматриваются задачи, которые доступны для решения. Составляется план процесса создания модели экспертной системы, ведется расчет нужных ресурсов (сроки, разработчики, вычислительные машины и т.д.), источники знаний (книжные данные, дополнительные эксперты, методики), существующие аналогичные экспертные системы, цели (передача опыта, механизация повседневных движений и др.), классы поставленных задач и т.д. Этап идентификации – это приветствие и обучение группы проектировщиков. Средняя продолжительность такого этапа обычно составляет 1-2 недели. На этом же этапе проектировки экспертных систем проводится выявление знаний. Знаток по знаниям помогает профессионалу получить и упорядочить знания, необходимые для существования экспертной системы, с употреблением всевозможных способов: анализ статей, разговоров, логические игры, лекционный материал, дискуссирующий монолог, наблюдение и другие. Выявление знаний – это получение знатоком по

знаниям более углубленного мнения о предметной области и способах выбора решения в ней.

Этап 2. Концептуализация

Определяется модель выявленных знаний о предметной области. Выясняется: терминология, список главных обозначений и их параметров, вид входных и выходных данных, тактика выбора решений и т.д. Концептуализация –это исследование неофициального изображения познаний о настоящем участке в облике графа, таблицы, диаграммы или слова, что отображает основные пункты и связи между мнениями предметной области. Продолжительность этапа 2-4 недели.

Этап 3. Формализация

В этапе формализации все главные мнения и связи, обнаруженные в прошлом этапе, отображаются в некоем специфичной речи, порекомендованной (избранной) знатоком по знаниям. Далее он принимает решение, то что нужная единица наличествующего приборного имущества соответствует имеющейся трудности либо нужен отбор иного инвентаря, либо потребуются уникальные исследования. Продолжительность этапа 1-2 месяца.

Этап 4. Реализация

Проектируется модель экспертной системы, содержащей базу знаний и другие подпункты. На данном этапе используются следующие инструментальные технологии: программирование на объектно-ориентированных языках (Паскаль, Си и др.), программирование на специфических языках, используемых в задачах искусственного интеллекта (LISP, FRL, SmallTalk и др.) и др. Четвертый этап создания в некотором смысле может называться основным, потому что здесь протекает возникновение программной концепции, показывающей жизнеспособность способа в целом. Продолжительность этапа 1-2 месяца.

Этап 5. Тестирование

Модель проходит проверку на практичность и адекватность аналогов ввода-вывода, эффективность тактики управления, количество проверочных заданий, грамотность базы знаний. Тестирование – это нахождение неточностей в данной методике, поиск багов в реализации модели, а также создание советов по приведению продукта до промышленного уровня.

Этап 6. Опытная эксплуатация

Проверяется работа экспертной системы для конечных юзеров. По итогам шестого этапа возможно будет необходимо значимое обновление экспертной системы.

Ход создания экспертной системы не заключается в строгий порядок перечисленных этапов. В процессе труда необходимо регулярно обращаться к ранним этапам и анализировать принятые до этого решения.

Языки создания экспертных систем. Для создания экспертных систем пригодны те же языки и методы программирования, что и для рядовых программ, но наличие таких особенных для искусственного интеллекта составных частей, как логический вывод, естественно-языковый интерфейс,

располагает к использованию для проектирования экспертных систем таких языков, как Лисп, Пролог, Клипс и особенных ресурсов поддержки создания.

Самой удобной для экспертных систем стала возникновение языка Пролог [2]. Ключевой смысл логического программирования заключается в разделении логики платформы от контроля процессом слежения, что делает ход созидания утилиты более легким и понятным. Пролог – язык высокого уровня, направленный на задействование моделей и методов математических законов. Разработан во Франции в Марсельском университете в 1972 году. Ключевым преимуществом Пролога, возвышающего его над остальными языками, является показная форма содержащихся в нем предикатов. Он специализирован для создания систем и подпрограмм искусственного интеллекта; классифицируется к классу языков пятого этапа. При работе с ним писателям кода не нужно расписывать пошагово процедуры — достаточно лишь выявить факты и установить их взаимоотношения. С помощью этих взаимоотношений процедуры, доступные в языке, располагают логическими выводами. Эта фишка делает Пролог наиболее адаптированным к созданию экспертных систем.

Язык Лисп [3] изобретен в Массачусетском научно-техническом вузе в истоке 60-х лет. Языки разработки Лисп и Пролог обладают интегрированными интерпретаторами для манипулирования знаниями. Лисп считается все пригодным языком программирования высочайшего значения и владеет возможностью производить списковые текстуры. Он классифицируется к декларативным языкам многофункционального вида и специализирован на отделках символьных данных, презентованных в облике списков.

Клипс был создан в лаборатории космических изучений NASA в середине 80х годов. Клипс (Clips) является аббревиатурой от C Language Integrated Production System. Он интегрирует в язык точку зрения образующихся прецедентов и язык охарактеризования процедур. Клипс пользуется продукционной моделью представления знаний и поэтому включает в себя три главных пункта:

- список фактов
- базу знаний
- блок вывода

Базисным различием предоставленного порядка от аналогов считается то, что он целиком создан на языке С. При этом начальный код обнародован в Веб-паутине. В Клипс употребляется уникальный LIPS-схожий язык программирования, направленный в исследование экспертной системы. Не считая этого, Клипс сопоставляется с еще двумя направлениями программирования: объектно-направленное и процедурное.

Помимо Лиспа, Пролога и Клипса существует множество различных языков [4], нацеленных на переработку символьной информации и создание экспертных систем: Smalltalk, FRL, Interlisp. Кроме этих специфичных языков для проектирования экспертных систем могут эксплуатироваться и

стандартные языки программирования общего плана: Си, Ассемблер, Паскаль, Фортран, Бейсик и др.

Главным недостатком языков программирования для создания экспертных систем это: значительные затраты времени создания полноценной системы, необходимость задействования высококвалифицированных инженеров-кодеров, неувязки с обновлением итогового продукта. Все это усложняет применение языков программирования для создания экспертных систем и делает весь процесс весьма финансово затратным и трудновыполнимым.

1.3 Исследование предметной области

1.3.1 Общая классификация методов и моделей прогнозирования

Метод прогнозирования выглядит как определенная хронология этапов, которые нужно проделать для того, чтобы добиться результата - модели прогнозирования. По аналогии с написанием программы есть порядок движений, совершая которые, создается программа — то есть делается прогноз.

Образец моделирования имеется активное понятие, правильно обрисовывающее разбираемый ход и изображающее базой про извлечения его грядущих ролей. В той же программной аналогии модель есть список библиотек и их соотношение, необходимых для программы — прогноза. Совокупность метода и модели образуют итог.

Классификация методов прогнозирования. Если подумать углубленно, то очевидно, что понятие «метод прогнозирования» намного обширнее понятия «модель прогнозирования. В взаимосвязи с данным в главном шаге систематизации традиционно разделяют способы в две категории: интуитивные и формализованные, что изображено на рисунке 1.2.

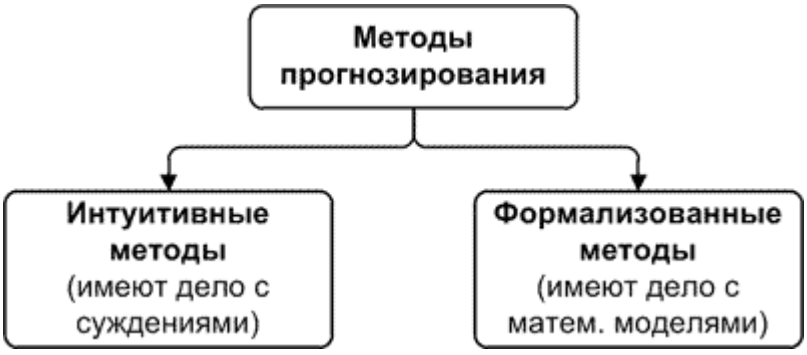

Рисунок 1.2 – Методы прогнозирования

Интуитивные методы прогнозирования основываются на рассуждениях и мнениях экспертов. В нынешнее время они часто употребляются в маркетинге, экономике, политике, так как механизм, действия которого

требуется спрогнозировать, или очень труден и не описывается математически, или очень прозрачен и в таком разборе нет смысла.

Формализованные методы — показанные в учебниках методы прогнозирования, по итогам которых создается модель прогнозирования, то есть определяются такая символьная зависимость, которая дает возможность просчитать грядущее значение хода, то есть сделать прогноз.

В данном общественное классифицирование способов моделирования окончено. Дальше создадим единое классифицирование модификаций. Тут нужно перебегать к систематизации модификаций моделирования. На рисунке 1.3 видно, что первому этапу модели следует разделиться на две группы: модели предметной области и модели временных рядов.

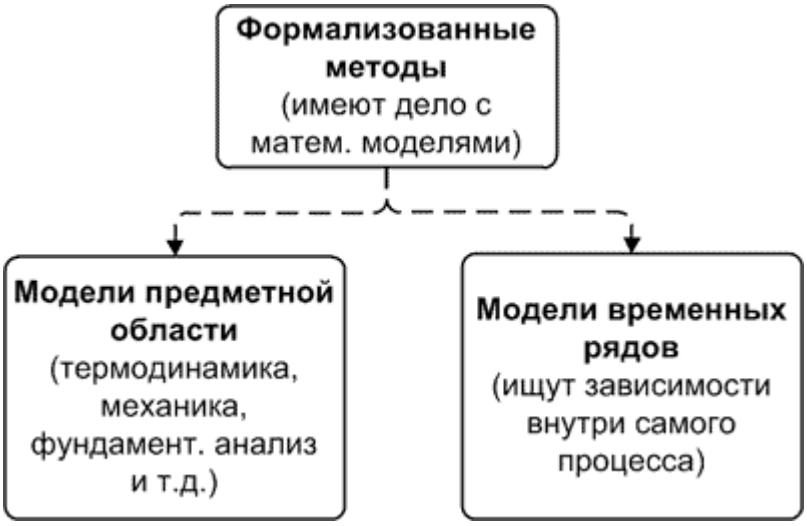

Рисунок 1.3 – Разделение формализованных методов

Модификации предметного участка — эти точные модификации моделирования, для их возведения употребляют законы предметной области. К примеру, образец, на основе которого совершается мониторинг погоды, охватывает уравнения динамики жидкостей и термодинамики. Мониторинг становления популяции мастерится в модификации, возведенной в отличительном уравнении. Мониторинг планки сахара в крови, страдающего диабетом, мастерится в базе порядка отличительных уравнений. Одним словом, в таковых модификациях употребляются связи, характерные определенной предметной области. Модификациям такового семейства характерен личный подход в исследовании.

Модели временных рядов — математические модели прогнозирования, которые стремятся найти зависимость будущего значения от прошлого внутри самого процесса и на этой зависимости вычислить прогноз. Эти модели универсальны для различных предметных областей, то есть их общий вид не меняется в зависимости от природы временного ряда. Мы можем использовать нейронные сети для прогнозирования температуры воздуха, а

после аналогичную модель на нейронных сетях применить для прогноза биржевых индексов.

Модификации мимолетных линий — точные модификации моделирования, которые настроены отыскать связь грядущего смысла с прошедшим изнутри движения и определить мониторинг. Данные модификации все пригодны для разных предметных областей, то есть их совместный разряд никак не изменяется в связи с природой временного ряда. Нейронные структуры могут применяться для моделирования температуры климата, а после подобный образец в нейронных сетях можно использовать для мониторинга биржевых индексов.

Классифицируем модели временных рядов. Модели временных рядов можно разделить на две группы, как показано на рисунке 1.4: статистические и структурные.

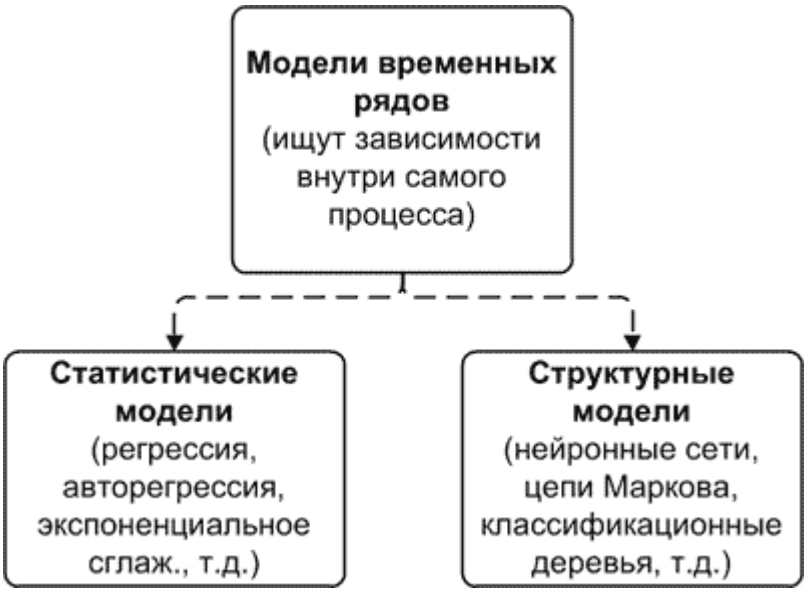

Рисунок 1.4 – Разделение моделей временных рядов

В статистических моделях связь будущего и прошлого выражена в виде некоторого уравнения [5]. К ним относятся:

- регрессионные модели (линейная регрессия, нелинейная регрессия);
- авторегрессионные модели (ARIMAX, GARCH, ARDLM);
- модель экспоненциального сглаживания;
- модель по выборке максимального подобия;

В структурных моделях связь будущего и прошлого показана в виде некоторой модели и правил перехода по ней. К ним относятся:

- нейросетевые модели;
- модели на базе цепей Маркова;
- модели на базе классификационно-регрессионных деревьев;

Для обоих разделений предоставлены основные, то есть наиболее популярные и подробно разъясненные модели прогнозирования. Однако на

сегодняшний день моделей прогнозирования временных рядов существует очень большое количество и для создания прогнозов, например, стали пользоваться такими технологиями, как SVM (support vector machine) модели, GA (genetic algorithm) модели и многие другие.

Таким образом, на рисунке 1.5 отображена полученная классификация моделей и методов прогнозирования.

Рисунок 1.5 – Методы и модели прогнозирования

1.3.2 Букмекеры и тотализаторы

Основное художество букмекера заключается в том, чтобы составить как можно больше вариантов ставок и при этом никак не запамятовать про собственные денежные интересы. В теории [6] данность смотрится последующим способом: владелец прорабатывает систему ставок таким образом, чтобы процент доходной части был всегда удобен ему. Наиболее обычным имеет возможность работать таковой экземпляр: в забеге семи фактически схожих коней владелец дает ставки 5 к 1 на любого участника. Если допустить, что на каждого из коней будет назначено одинаковое число средств, то, независимо от того, тот или другой конь финиширует первым, выручка хозяина станет эквивалентна одной седьмой всех поставленных средств.

Однако на деле не все так легко. Очень непросто добиться такого расклада, при котором владелец не имел бы денежной заинтересованности в итогах гонки и был беспристрастен к тому, какой участник придет первым, поскольку так или иначе добьется свой прибыли. По факту, игнорируя

перманентное колебание вероятности по мере возникновения очередных ставок, расход и доход владельца в значительной степени зависит от итогов заезда или поединка. Количество прогнозируемых выигрышей в момент ставки являются утвержденными и не могут уменьшаться, и, хотя букмекер может изменить сумму премий в случае победы фаворита при соответствующих ставках, он будет вынужден оплатить каждый заклад в соответствии с изначальными условиями.

В последнем случае он имеет возможность реквизировать победителя из собственного перечня и отрешиться воспринимать последующие ставки на данного участника заезда (если идет речь о бегах) либо убавить общий риск либо часть его, сам поставив на потенциального победителя у остальных букмекеров.

Типы ставок улучшились с выходом в свет этак именуемых аукционных пулов, на каких габариты выигрышей инсталлируются автоматом самими соучастниками ставок. Любой соучастник боев, любая лошадь "продается" на этом аукционе тому, кто оплатит огромную необходимую сумму. В итоге "торговец" приобретает всю необходимую часть выигрыша. Недочетом данного метода считается то, что выигрывает лишь один игрок, тот, кто выполняет высочайшую ставку.

В итоге этот недоработок был упразднен. В 1872 году парижский лавочник, по имени Пьер Олер, разработал модификацию аукционного пула, где билеты ставок можно было приобретать в любом составе по требованию безграничного количества игроков. После удаления из пула комиссионных, причитавшихся Олеру, остающуюся прибыль разделили между владельцами купонов на победившую лошадь относительно числу купонов, которыми обладал каждый. Метод Олера, который он назвал "пари мютюэль" (pan mutuel), что означает "пари между собой", оказался очень интересным и стал быстро распространяться. Несмотря на то, что вначале правительство Франции не одобрило это нововведение, пари мютюэль — ставки "между собой" — были объявлены законным способом заключения споров и внесения ставок. В настоящее время во всех операциях по системе пари мютюэль используются те или иные виды тотализаторов, обычно интегрированные с электронными компьютерами.

В базе предоставленной системы пролегают два главных принципа. Во-первых, все ставки, поставленные перед заездом, предусматриваются и складываются. Во-вторых, при завершении гонки протекает высчитывание процента выплаты и расчет объемов определенных выигрышей за вычетом конкретного процента для уплаты особого правительственного налога. Естественно, рядовому любителю не интересны различные математические отношения и методы подсчетов, нужных для действенного функционирования данной наитруднейшей технологической структуры. Основное для него — итог в образе приобретенной прибыли.

И тем не менее букмекерство не намерено целиком отдать все свои позиции. Не говоря уж о том, что людской народ более доверительно

относился к имению дел с другими людьми, а не с машинами, стоит выделить и то, что многие постоянные игроки споров не особо доверяли ранним тотализаторам (иногда не беспричинно).

В некоторых странах, где существуют и букмекерские компании, и ставки типа пари, букмекерство состоит наиболее распространенным видом заключения пари. Те, кто ставит ставки на бега крупными суммами, не довольны тем, что в тотализаторе им приходится спорить самим с собой, поскольку чем выше установленная ими сумма, тем меньше прибыль. Мелкие участники пари также имеют дела с букмекерами, находящимися вне системы. Там они могут делать ставки в маленьких количествах, которые в тотализаторе не принимаются. Помимо этого, крупные и мелкие вкладчики извлекают от этих букмекеров еще один тип побочной услуги — умение вносить ставки телефонным путем и, более того, в долг. Даже в тех странах, где букмекерство запрещено законом, подпольные букмекеры процветают.

Крупнейшая в мире компания — букмекерская фирма Уильяма Хилла в Англии, которая, как принято считать, работает с более 2 000 000 любителями скачек и футбольных матчей ежегодно, имеет в штабе более трехсот высококвалифицированных телефонных работников. (Фирма сама занимается подготовкой и обучением специалистов и другого обслуживающего персонала приемных пунктов.) Их работа заключается в приеме и утверждении ставок, приходящих по телефону, фиксация четкого времени, когда был создан запрос на ставку. Компания Хилла — прекрасный пример профессионального подхода к индустрии азартных игр.

Как и в основной массе государств СНГ, букмекерский бизнес в Казахстане подлежит неотъемлемому лицензированию. Лицензия включает в себя возможность «живого» беттинга и работу в Интернете. Условия выдачи правоустанавливающих бумаг – не очень высокие, поэтому множество участников экономического рынка работает на легальных основаниях. Главная особенность ставок в Казахстане – игрок не выплачивает каких-либо отчисления с победы. Все налоги в пользу государства забирают букмекеры. В последнее время лицензированные сотрудники принесли в казну государства около 5 миллионов долларов. Отсутствие налогов у бетторов – значимый аргумент в пользу развития индустрии. Минимальный возраст возможности внесения ставки – 21 год, и он отбивает несознательных граждан.

На сегодняшний день в нашей стране функционирует множество букмекерских контор. Рассмотрим наиболее известные из них.

БК «Леон». ЛеонБК Леон – это всемирно известный бренд с чистой репутацией. В онлайне контора имеет миллионы игроков по всему миру, включая страны СНГ. Казахстан для Леонбетс – особенный рынок, лицензия на деятельность в сфере игорного бизнеса получена ещё в 2008. Фирма владеет десятками пунктов приема ставок во всем Казахстане. Игроки имеют возможность пополнить баланс в тенге, а также просмотреть адреса пунктов приема. Принципиально, что букмекер бережет высочайшие нормативы сервиса – скорые выплаты, и ежедневная круглосуточная русскоязычная

система поддержки игроков, и моментальный расчет чека. Никуда не исчезла и известная всем система одобрения: 100% бонус на начальный депозит, а также приятные призы в обмен на бонусные очки (леоны). Продвинутых юзеров в Леоне ожидает мобильный беттинг.

БК Olimp. В Казахстане Олимп это один из лидеров рынка и быстро распространился по всей стране и получил высокую популярность (сотни ППС по стране). В Интернете бк «Олимп» выглядит неподобающе крупной конторе. Архаичный вид в совокупности с неуклюжей панелью навигации. Разнообразие событий на ww.olimp.kz также оставляет желать лучшего: упор делается на футбол и теннис, а остальные виды спорта – просто на просто игнорируются. Недоработан и лайв-раздел, ему элементарно не объемен. Контора Олимп не предлагает бонуса на начальный депозит, зато существуют: 10%ый бонус на экспресс с коэффициентов от 2.5, а также возврат от 15% до 70% от ставки в случае неудачи. Мобильная версия сайта весьма примитивна. Несмотря на положительную динамику развития, Олимпу в Казахстане ещё есть, куда расти.

БК «Гол+Пас». Контора Гол плюс пас – типичный представитель казахского букмекерского рынка. Компания имеет неплохое количество клиентов, но не направленна на повышение стандартов качества. Работа в оффлайн точках не вызывает неодобрения, другое дело – онлайн. Golpas предоставляет скромный выбор событий: дополнительных ставок нет. Положительный момент – наличие тотализатора, предлагаемого для футбола, хоккея и баскетбола.

Profitbet.kz – молодая казахстанская компания, получившая лицензию в 2011 году. Крупной аудиторией она пока что не обзавелась, но настрой к развитию виден. Сайт бк Профитбет радует современным видом, но на поиск интересующего события затрачивается немало времени. Линия разнообразна по количеству видов спорта, имеются ставки на политику и светские мероприятия. Зато коэффициенты уступают среднеевропейским показателям.
Гордостью букмекерской компании Профитбет является наличие трансляций к ставкам в режиме реального времени. Зарегистрированные пользователи могут получить доступ к ним в любой точке. Количество способов пополнения баланса – ограничено. Среди бонусных плюсов виднеется возврат 25% от количества проигранного счета.

1.3.3 Анализ факторов спортивного прогноза

Успешный спортивный прогноз построен на непосредственном присутствии трех компонентов в определенной очередности [7].

1. Статистика. Игровая статистика — это точка отсчета спортивного прогноза, как игрока, так и букмекера. Причем, последние на некоторые спортивные соревнования выставляют коэффициенты, исходя исключительно из статистических данных. Поэтому, чем более точными и достоверными будет информация, тем вероятней успешность прогноза. Практика

показывает, что ее объем не должен быть чересчур большим, то есть вполне достаточно сведений о трех-четырех последних играх. В командных видах спорта значение имеют не только данные о показателях уже проведенных матчей, но и командный состав – как прошедшей, так и предстоящей игры. Анализ статистики позволяет определить более точную расстановку сил претендентов на победу, а, следовательно, влияет на весь спортивный прогноз.

2. Аналитика. Правильный анализ исходных статистических данных определяет вероятность успеха всего спортивного прогноза. При анализе необходимо учесть все факторы, которые способны повлиять на ход и результат игры. И в первую очередь, психологические – отношения и атмосферу внутри команд, насколько команды уверены в своих силах и своей победе, где происходит игра и т.д. Также не стоит забывать и про факторы, которые касаются физического состояния игроков перед началом спортивного состязания – расстановка сил и самочувствие спортсменов, а также наличие у них травм, полученных незадолго до соревнований. Косвенные факторы, влияющие на итог игры, называют форс-мажорными. И хотя прогнозировать их сложно, общая тенденция наблюдается и здесь. К примеру, в разгар засушливого лета вряд ли стоит беспокоиться о сильном ливне, который затруднит передвижение игроков по полю, а вот ухудшение самочувствия у членов команды, не привычной к жаркому климату и палящему солнцу, – вполне возможно.

3. Интуиция. То, без чего не существует ни один успешный игрок на спорт-пари и, тем более, профессиональный беттер. Прислушиваться к своей интуиции, учиться ее понимать и делать правильные выводы, исходя из ее подсказок – это целое искусство. Тем не менее, овладеть им следует, если Вы стремитесь к успеху и победам в букмекерской конторе. Не нужно слепо следовать подсказкам, которые предлагает внутренний голос, к ним нужно лишь прислушиваться. Чем больше у Вас будет опыта, тем более «продуктивно» будет работать на Вас интуиция, потому как наработанные знания не просто усваиваются человеческим мозгом, но и закладываются в подсознание, чтобы в нужный момент прийти на помощь. Прислушиваться к себе нужно на протяжении всего процесса игры – и при составлении прогноза, и при выборе нужного игрового события из общей линии. У опытных игроков и профессиональных беттеров взаимодействие с собственным подсознанием отточено до уровня мастерства. Даже в случае, когда спортивное событие кажется Вам прозрачным, а прогноз – предельно простым и ясным, не стоит пренебрегать мнением собственного внутреннего голоса. Если он говорит Вам «не то», значит, в чем-то здесь кроется подвох и в итоге, скорее всего, Вы окажетесь в проигрыше.

Грамотный анализ спортивных событий**.** Не существует ни единой успешной тактики на игру у букмекеров с основанием лишь на статистику. Пользоваться ею необходимо при анализе предстоящего матча, но не брать

основным фактором для прогноза. Следует рассмотреть моменты, которые эксперты обычно берут в расчет при анализе футбольного события.

1. Положение в турнирной таблице. Смотреть на турнирную таблицу нужно лишь тогда, когда прошла хотя-бы треть сезона. Некоторые команды хорошо усилились в это время и имели трудовую предсезонку, тогда как другие лишились ведущих игроков.

2. Форма команд. Следует брать в расчет то, как команды играли в пяти-шести последних матчах. Также следует заострить внимание на последние домашние и выездные матчи, количество забитых и пропущенных голов в них.

3. Игроки. На итог матча может сказаться количество резервистов клуба, количество травмированных игроков перед матчем и игроков, которые рискуют пропустить следующий матч из-за перебора карточек.

4. Мотивация. Существует множество случаев, когда явный аутсайдер побеждал потенциального фаворита. В большинстве таких моментов этому способствовала сверх мотивация команды. Когда команда в шаге от поражения и вылета из розыгрыша, она может открыть в себе второе дыхание для победы. Также следует дотошно рассматривать принципиальные поединки или так называемые дерби. Когда приходит время дерби, команды выкладываются на максимум своих возможностей, их подстегивают вперед свои болельщики, поэтому в таких играх имеют место быть сенсационные итоги.

5. Статус турнира. К примеру, в России статус кубка не значителен, поэтому большинство клубов верхних дивизионов ликвидируются из него, чтобы облегчить свой календарь. Английские же команды не ценят статус Лиги Европы и, зачастую, выставляют на игры этого турнира резервные составы.

6. Время года. Все европейские футбольные чемпионаты начинаются в середине августа. В течение первых пяти туров регулярно наблюдается сенсационные итоги. Лишь по истечению трети чемпионата игра клубов стабилизируется, и результаты становятся более предсказуемы.

7. Статистика лиги. Всегда рекомендуется просматривать общие показатели турнира. К примеру, лиги Голландии и Германии всегда соблюдают тенденцию много забивать, в то время как в Италии команды обычно играют в сухую.

8. Атмосфера в команде. Стороннему человек трудно ощутить, какая атмосфера царит в клубе, и тем не менее, об этом можно догадаться логически. Такие аспекты как интервью игроков могут дать предмет для размышлений.

Умение выводить грамотный анализ игры не является врожденным талантом. Абсолютно каждый способен обучится обнаруживать те ситуации, на которых будут основаны ближайшие результаты игры. Грамотного проанализировав все факторы [8], существует возможность добиться перевеса над линией букмекеров и регулярно оставаться в плюсе.

2 Классификация моделей представления знаний

2.1 Продукционная модель представления знаний

Экспертные системы [9], работающие на правилах, знаниях о вычислении заданий подаются в изображении правил "если..., то...". Такой способ именуется как один из старейших способов представления знаний о предметной области в экспертной системе. В структуре, построенной на правилах, пары "условие-действие" имеют вид правил "если..., то...", в которых перенаправление (часть если) равно условию, а результат (часть то) - действию. Если условие подтверждается, экспертная система начинает действие, является правдивость результата. Данные особенных моментов следует беречь в рабочей памяти. Механизм вывода предоставляет цикл продукционной системы "распознавание-действие". При этом контроль может выполнятся или при помощи данных, или основываясь на цели. Самый используемый метод представления знаний - в виде определенных фактов и правил, по которым из хранящихся фактов способны быть добавлены обновленные. Факты показаны, к примеру, в виде троек: (АТРИБУТ ОБЪЕКТ ЗНАЧЕНИЕ). Факт такого вида обозначает, что выбранный элемент имеет определенные параметры (свойства) с указанным значением. Например, тройка (ТЕМПЕРАТУРА ПАЦИЕНТ1 37.5) изображает факт «температура больного, названного ПАЦИЕНТ1, равна 37.5». В более распространенных моментах факт преподносится неявным значением параметра, а каким-либо понятным термином, который может быть правдой или ложью. Правила в базе знаний имеют вид: ЕСЛИ A ТО S, где A - условие; S - действие. Действие S исполняется, если A истинно. Наиболее часто действие S так же, как и условие, обозначает из себя мнение, которое может быть создано машиной, если правдиво условие правила A. Правила в базе знаний нужны для изображения эвристических знаний, т.е. не официальных правил суждения, придумываемых экспертом основываясь на его опыте в определенной деятельности. В качестве условия A может выступать либо факт (как в данном примере), либо несколько фактов A1, ... , AN, соединенные логической операцией «и»: «A1 и A2 и ... и AN». В вычислительной логике такое уравнение считается конъюнкцией. Оно будет правдиво в таком случае, если правдивы все его составляющие. Продукционная модель имеет вид:

(i); P; Q: Ai => Bj ; N;

где i - номер правила; P- приоритет правила; Q - область применения правила Ai => Bj – ядро продукции; i, j - с какой части берется утверждение, чаще всего i = база данных, база знаний, диалог, О (блок объяснения); j- тоже самое и помимо этого добавляется блок приобретение знаний. АБД => ВБД, N- комментарии к продукции. Исходя из этого, движения структуры

продукционного типа создаются с помощью применения логических правил вывода, смысл которых заключается в следующем: допустим дано, что экземпляр А правдив и где-то есть правило вида «Если А, то В», тогда экземпляр В так же правдив. Правила срабатывают, когда найдены факты, согласные с их левыми частями: если правдиво начало, то должен быть правдив и результат. На первый взгляд, такой вывод просто может быть создан на ЭВМ, но, на деле человеческий мозг является более действенным при решении задач. Если экспертную систему исследовать как продукционную, то базу знаний о предметной области в праве брать как порядок продукционных правил. В структурах, работающих на правилах, если условие подтверждается, экспертная система реализует действие, показывающее правдивость результата. Механизм вывода реализует цикл продукционной системы "распознавание-действие". При этом контроль может выполнятся или при помощи данных, или основываясь на цели. Многие предметные области наиболее предназначены прямому поиску. Например, в проблеме подготовки к сравнению значимая часть информации обозначает собой начальные данные, при этом нередко тяжело понять гипотезы или цель. Это влияет напрямую на процесс логического мышления, при котором факты доставляются в рабочую память, и система организует поиск для них интерпретации. В экспертной системе на базе цели в рабочую память располагается целевое уравнение. Система сравнивает итоги правил с необходимым уравнением и посылает их ссылки в рабочую память. Это равно разбиению недочета на элементарные подцели. В дальнейшем этапе работы продукционной системы ход протекает дальше, эти ссылки преобразуются в обновленные подцели, которые сравниваются с правилами. Система функционирует до того времени, как все подцели в рабочей памяти не преобразуются в правдивые, одобряя гипотезу. Из этого следует, что противоположный процесс нахождения в экспертной системе максимально близко похож на процесс изучения гипотез при вычислении недостатков человеком-знатоком.

В экспертной системе, чтобы добиться подцели, нужно потребовать данные у пользователя. В особенных экспертных системах создатели решают, какие подцели будут выполнены путем запроса к человеку. Другие системы спрашивают у человека, если не могут определить правдивость подцели на основании правил из базы знаний. Экспертная система должна быть способна легко проверятся, быстро обновляться и являться эвристической по типу. Конструкция продукционной системы считается основным аспектом для каждого из этих требований. Быстрота обновления, к сведению, определяется синтаксической способностью продукционных правил: каждое правило считается "фундаментом" знаний, который может обновляться непосредственно. И тем не менее имеются семантические рамки, потому как частные правила сопряжены по цели. Следовательно, они должны быть слаженны в любом разработке изменения или обновления. В дальнейшем изучается создание разъяснений контролем вывода.

Объяснения и прозрачность при изучении на основе цели. Продукционная система основана на нахождении данных в графе. Программы подсистемы толкований регулируют ход нахождения на графе и эксплуатируют эти данные, чтобы выдавать ответ на вопросы человека. При поддержке продукционных правил каждый этап хода рассуждений записывается самостоятельно. Чаще всего экспертные системы, спроектированные на принципах, реагирующий на два вопрошания-"почему?" и "как?". Первый - "почему?" появляется, когда программа подает запрос данных у юзера, и его реакция подразумевает "почему вы интересуетесь этими данными?". Результатом будет последующее правило, которое система предпримет попытку задействовать. Реакцией на вопрос: "Как вы добились такого ответа?" - будет рядом принципов, задействованных для получения итога. По итогам, структура, созданная на знаниях, реагирует на запросы "почему?", показывая последующее правило, которое она предпримет попытку задействовать. Реакцией на вопросы "как?" - она развернет порядок логических выводов, которые добились результата. Даже если данные механизмы считаются структурно легкими, они владеют широким функционалом, если база знаний спроектирована логически верно. Если выводы являются логичными, важно не только, чтобы база знаний предоставляла соответствующий результат, но и чтобы каждое правило зависело от каждого пункта общего хода избавления от неувязок. Если в одиночном правиле базы знаний содержится несколько пунктов или принципы показаны в свободной форме, добиться корректных результатов на вопросы "как?" и "почему?" будет значительно сложнее. Это не только разрушает доверие пользователя к структуре, но и делает программу более трудоемкой для восприятия и обновления создателями.

Применение продукционной системы для обсуждения на основе данных. Пример анализа состояния автомобиля демонстрирует использование продукционной системы для организации обнаружения цели. В ней задействовался поиск вглубь, так как перед переносом к близлежащим целям для каждой найденной в базе правил подцели создавался полноценный поиск. Но все же продукционная система считается также пригодной структурой и для обследования на основании данных. В таком обследовании на основе данных обычно используется поиск в ширину. Алгоритм очень понятен: материалы рабочей памяти сопоставляются с пунктами каждого принципа в структурной базе правил. Если данные в рабочей памяти ведут к появлению принципа, итог переводится в рабочую память, и контроль отправляется последующему принципу. После обозрения имеющихся правил поиск перезапускается.

Рассмотрим такой экземпляр. При создании экспертной системы «Услуги советчиков» задействована продукционная модель представления знаний. К примеру представим ситуацию, словесно описанать которою модно таким образом: «Если процессор = «Celecronы» и память = 555, то выдать

информацию обо всех ЭВМ с такими данными». На языке запросов SQL данное правило выглядит следующим методом:
SELECT * FROM basicc WHERE prrocc like 'Celecrons%' and memorysize='555' ORDER BY art

Ядро продукции для исследованного правила имеет вид:
А1БД И А2БД принадлежит ВБЗ, то есть левая часть берется из БД, а правая - из БЗ. При создании экспертной системы «Услуги советчиков» была задействована методика поиска, основанная на реакции на вопрос «Почему?». Ответ системы будет смотреться следующим образом: «Потому что «вы запросили размер дисплея, равный 17» (посылка А1) ИЛИ «вы не задавали тип процессора» (посылка А2), поэтому «не указан объем жесткого диска» (заключение В1), «вы захотели объем памяти =1555» (посылка А3) ИЛИ «вы не указали модель видеокарты» (посылка А4), поэтому «не указана стоимость платы» (заключение В2), так как В1 ИЛИ В2 не истинны, поэтому «не доступна реальная стоимость компьютера» (заключение В4).

Для вывода допустимо использовать сеть вывода, состоящую из нескольких правил. Например, сеть вывода состоит из трех правил [10]:
- П1: Если А1 И А2 истина, то В1- истина;
- П2: Если А3 И А4 истина, то В2- истина;
- П3: Если В1 И В2 истина, то В3- истина.

В данном случае, помимо стандартной сети вывода, показанной на рисунке 2.1, есть побочная сеть вывода по стандартам. В каждый миг диалога можно заставить систему вывести информацию с любыми рамками, используя имена полей - шаблонов.

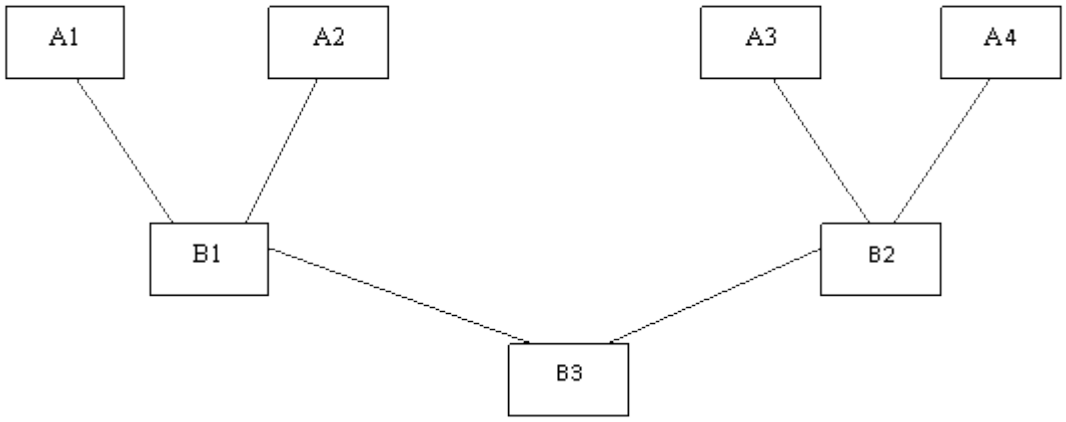

Рисунок 2.1 - Схема блока объяснение

Подведем слабые и сильные моменты популярных систем продукций. Сильные достоинства: легкость написания и распознавания отдельных принципов; простота пополнения, обновления и удаления; простота механизма логического вывода. Противоположные стороны: неясность

взаимных отношений правил; сложность оценки целостного образа знаний; крайне низкая эффективность обработки; отличие от человеческой структуры знаний; отсутствие гибкости в логическом выводе.

В итоге, если целью стоит маленькое задание, обнаруживаются только показательные стороны системы продукций. В обратных моментах увеличения количества знаний, надобности вычисления крупных заданий, организации разновидных выводов или ускорение процесса вывода понадобится конструкция базы данных.

2.2 Фреймовая модель представления знаний

Главная мысль фреймового расклада к представлению знаний содержится в том, что все, что обхватывает мнения либо моменты, не «размывается по сети», а отображается во фрейме.

Фреймом именуется конструкция для описания мнения либо момента, созданная из параметров данной ситуации и их атрибутов. Фрейм также разумно расценивать как отрывок семантической сети [11], созданный для растолкования терминов со всей совокупностью принадлежащих им свойств. Так, к примеру, мнение о деловой записке в структуре, созданной на фреймах, может выглядеть аналогично приведенном на рисунку 2.2.

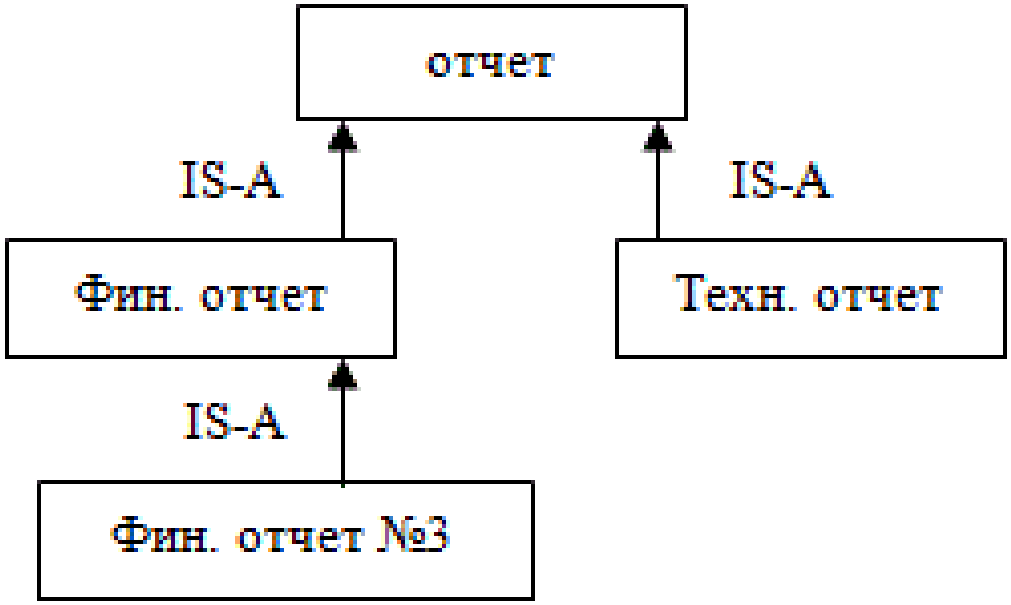

Рисунок 2.2 - Модель описания деловой записки

Графически это смотрится подобно семантической сети, однако главное различие заключается в том, что каждый узел во фреймовой структуре имеет универсальную структуру, соответствующую представленной на рисунке 2.3

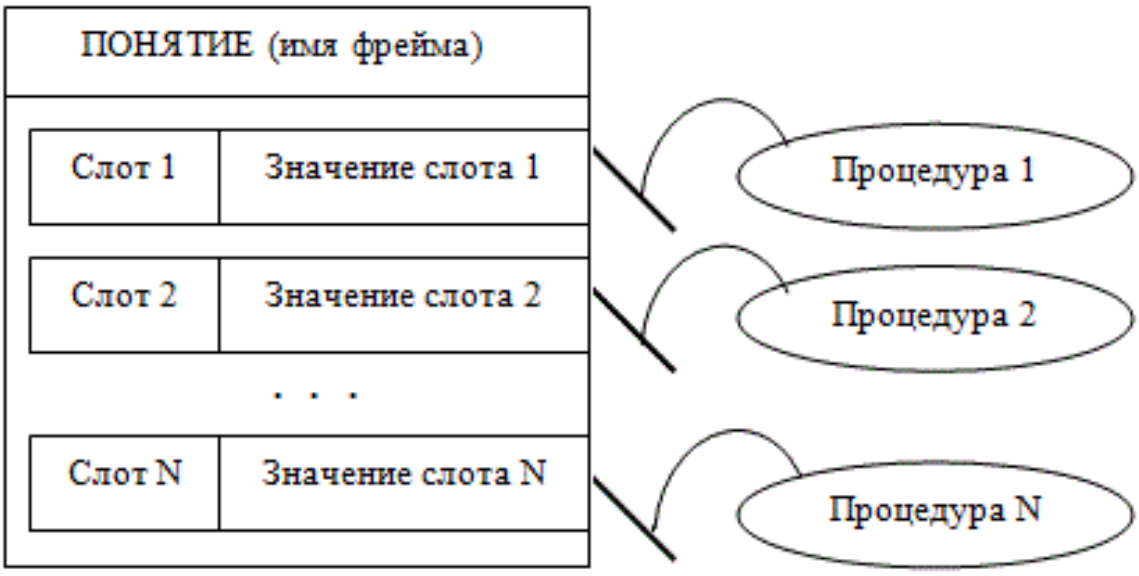

Рисунок 2.3 - Обобщенная структура фрейма.

Создатель принципа фреймового расклада Марвин Минский поделился таким определением: «Фрейм – это конструкция данных, демонстрирующая типичную обстановку, вроде пребывания в жилой комнате или подготовка похода на гулянку. К каждому фрейму приобщается некоторое количество видов информации. Немного информации о том, как эксплуатировать фрейм и немного о том, чего стоит ожидать в дальнейшем, и что следует предпринимать, если ожидания не станет реальностью».

Достоинством фреймовых моделей представления знаний заключается в том, что все термины, располагающиеся в каждом из узлов структуры, могут быть представлены комплексом атрибутов и их параметрами, которые находятся в слотах фрейма.

Слот — это атрибут [12], который сопрягается с узлом в структуре, построенной на фреймах. Он является главным аспектом фрейма. Имя слота дается исходя из четкого типа атрибута, а значением слота можно представить:

- экземпляр этого атрибута;
- ссылку на другой фрейм;
- ссылку на другой фасет.

С каждым уникальным слотом может быть сопряжена одна или несколько процедур, которые запускаются, когда трансформируются показатели слотов. Обычно со слотами сопрягаются такие процедуры:

1. ЕСЛИ-ДОБАВЛЕНО (IF-ADDED) — запускается тогда, когда новая инфа появляется в слоте;
2. ЕСЛИ-УДАЛЕНО (IF-REMOVED) — запускается при удалении инфы из слота;

3. ЕСЛИ-НУЖНО (IF-NEEDED) — запускается при запросе инфы из слота в том случае, когда значение слота равно нулю.

Эти процедуры отслеживают преображение инфы, соответствующей данному узлу, и отслеживают, чтобы при преображении тех или иных параметров выполняются необходимые действия.

Фреймовые системы и их функционирование. Фреймовые системы разрешают применять объектно-направленный метод к представлению знаний. Для демонстрации работы данного класса структур, созданных на знаниях, изучается иерархия понятия отчета, конструкция которого была рассчитана ранее, на рисунке 2.3.

При этом взгляды и мнения об объекте "Отчет" обосновываются множеством концепций, которые выведем через комплект атрибутов и вероятными разновидностями их параметров. Все атрибуты помещены в слоты фрейма, придуманные специально для каждого из них, что демонстрирует рисунок 2.4.

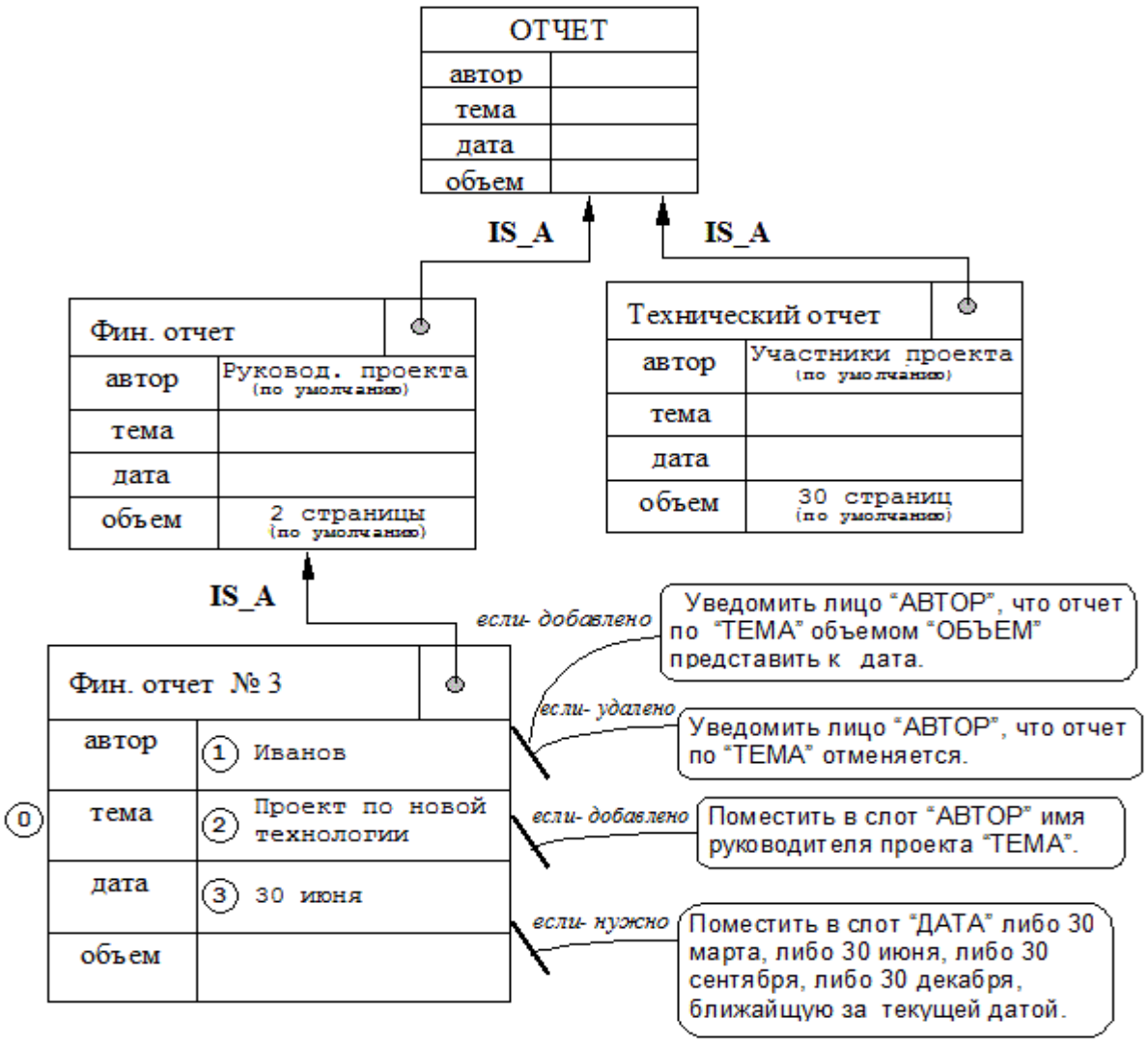

Рисунок 2.4 - Функционирование фреймовой системы.

Каким же образом разрешено применять настолько структурированные данные? Для наглядности проанализируем вариант, когда в систему введен запрос: «Мне нужен материальный доклад об исполнении плана согласно новейшей технологии».

Интерфейсная часть системы изучает структуру модели и отталкиваясь от фрейма–экземпляра «Фин. доклад» дописывает в свою структуру пустой фрейм «Фин. отчет №_», а после завершения его создания в слот «ТЕМА» этого фрейма, исходя от начального запроса, записывает строку «Проект по новейшей технологии».

Далее все происходит автоматически:

1. Операция «если–добавлено», сплетенная со слотом «Содержание», исполняет розыск согласно собственной базе данных управляющего данного плана. Предположим, будто его имя Иванов. Операция вписывает его фамилию в разъем «Создатель» денежного отчета №3. Если управляющий данной темы не отыщется, то в разъем «Создатель» станет наследовано смысл класса, а конкретно контент «Управляющий Плана». Операция «если–добавлено», сплетенная со слотом «Создатель», запускает старт, т.к. в разъем было добавлены свежие данные. Данная операция запускает составитель известия, чтоб выслать его Иванову, однако сразу же понимает, что недостает подходящей даты выполнения.

2. Процедура «если–добавлено», изучая слот «ДАТА» и найдя его значение, равным нулю, запускает процедуру «если–нужно», сопряженную с данным слотом. Эта процедура, исследуя нынешнюю дату, например 26.05.16, выявит, что «31 мая» ближайший к ней конец экономического квартала и добавить запись этой даты в слот «ДАТА».

3. Теперь процедура «если–добавлено», сопряженная со слотом «АВТОР», выявит, что еще один указатель, который нужно добавить в посылку, т.е. объем отчета, недостает. Слот «ОБЪЕМ» не сопряжен с процедурами и ничем поспособствовать не может. Однако главнее узла № 3 расположен узел обобщенной модели материального доклада, включающий в себя показатель объема.

4. Процедура, употребляющая теорию унаследования функций и параметров класса, включает в себя показатель объема и выводит следующий текст: «Господин Иванов, подготовьте материальный доклад по проекту новейшей технологии к 31 мая объемом 2 страницы». Если в некоторый момент фамилия Иванов будет вырезана из слота «АВТОР», то концепция автоматически доставит ему оповещение, что его отчет не нужен.

В приведенном сравнении заполнителями слотов фрейма работали конкретные единицы атрибутов, среди них и заполняемые по дефолту (IF-DEFAULT). Помимо этого, заполнителями слотов могут быть:

– имена других фреймов структуры, на которые создается ссылка;

– фасеты («агрегат», «диапазон», «по умолчанию» и др.).

Увеличим изученный раннее экземпляр для демонстрации этих значений (рис. 2.5):

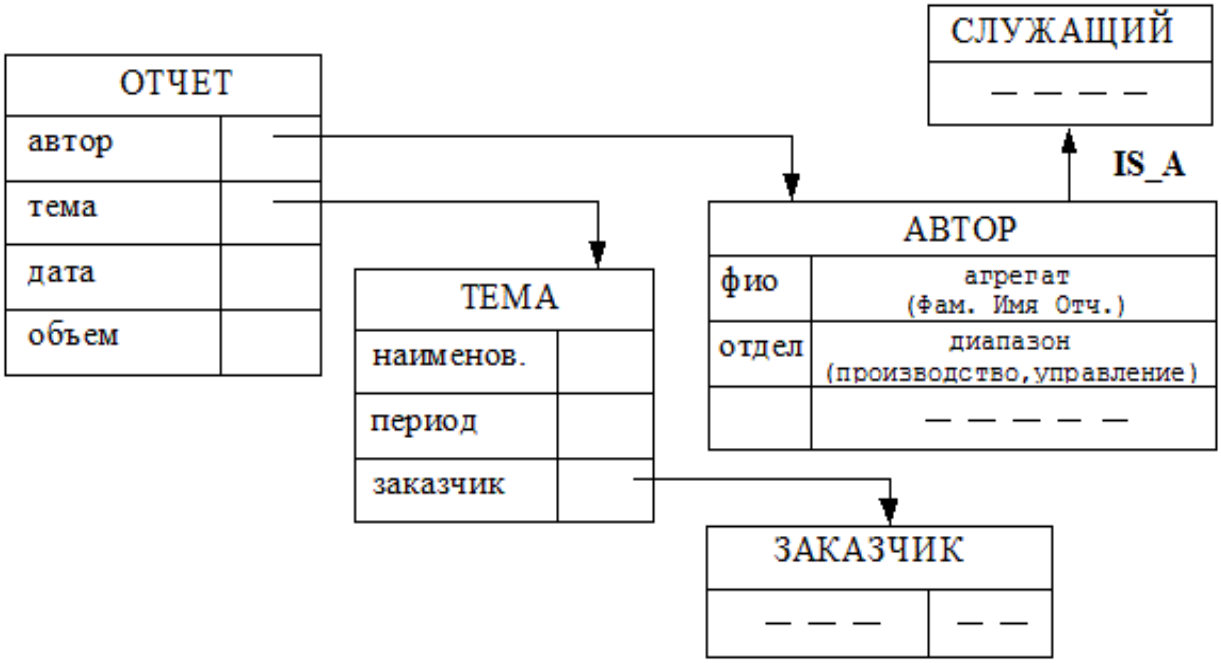

Рисунок 2.5 - Обобщенная фреймовая структура.

При обнаружении фреймов фасет «агрегат» обращает внимание на то, что необходимо описать требуемые элементы, а «интервал» - на то, что нужно избрать один из совокупности элементов. Данная инфа употребляется интерфейсной оболочкой при вводе данных.

Обобщенная структура фрейма**.** В общепринятой реальности фрейм отображается как матрица, теория и принципы которой считаются развитием понятия отношения в реляционной модели данных. Обобщенная структура фрейма имеет вид, изображенный в таблице 2.1.

Таблица 2.1 – Конструкция фрейма

Название фрейма			
Имя слота	Параметр наследия	Параметр класса	Смысловое обозначение
слот 1 слот 2 - - - - слот N			

Слотом фрейма именуется объект данных для содержания познаний об элементе, которому назначен выделенный фрейм. В список главных пунктов выявления слотов входят:

a) Имя слота. Любой разъем обязан обладать неповторимым именем во фрейме, к которому он относится. Имя слота в некоторых случаях может быть служебным. Среди служебных имен могут быть:

– имя пользователя, определяющего фрейм;
– дата определения или модификации фрейма;
– комментарий.

b) Указатель наследования. Он определяет, какие данные о параметрах слотов во фрейме высшей ступени наследуют слоты с теми же именами во фрейме нижней ступени. Типичные указатели наследования:

– S (тот же) - слот наследуется с теми же значениями данных;
– U (уникальный) - слот наследуется, но данные в каждом фрейме могут принимать любое значение;
– I (независимый) - слот не наследуется.

c) Указатель типа данных. Типом данных, включаемых в слот, могут быть:

– FRAME (указатель) - указывает имя фрейма верхнего уровня.
– ATOM (переменная);
– TEXT (текстовая информация);
– LIST (список);
– LISP (присоединенная процедура).

С поддержкой приспособления управления изучением согласно взаимоотношениям IS_A исполняется самодействующий обнаружение и классификация понятий слотов фрейма высшей ступени и подключенных операций.

Языки представления знаний на основе фреймов. Специфические языки представления знаний в сетях фреймов дают возможность продуктивно выстраивать промышленные структуры. К списку таких языков можно соотнести: FRL (Frame Representation Language), KRL (Knowledge Representation Language), фреймовая оболочка Kappa, PILOT/2 и другие программные средства.

Пример 1. В частности запись фрейма [13] на языке FRL будет иметь вид, показанный на рисунке 2.6:

```
(frame СТОЛ
     (purpose (value(размещение предметов для
       деятельности рук)))
     (type (value(письменный)))
     (colour (value (коричневый))))
```

Рисунок 2.6 – Код фрейма FRL

Пример 2. Запись фреймов на продукционно-фреймовом языке описания знаний PILOT/2, где фреймы John и Mary являются экземплярами фрейма-прототипа Person, будет иметь вид, продемонстрированный на рисунке 2.7:

```
[Person is_a prototype;
Name string, if_changed ask_why();
Age int, restr_by >=0;
Sex string, restr_by (=="male" || =="female"),
by_default "male";
Children {frame}];
[John is_a Person; if_deleted bury();
Name = "Johnson";
Age = 32;
Children = {Ann, Tom}];
[Mary is_a Person;
without Age;
Name = "Smirnova";
Sex = "female";
Children = empty];
```

Рисунок 2.7 – Код создания фрейма типа Pilot

Главным достоинством фреймов как модели представления знаний считается то, что она показывает концептуальный фундамент компании памяти человека, а также ее видоизменяемость и четкость. Особенно видны сильные стороны фреймовых систем представления знаний в тех случаях, если родовидовые связи трансформируются редко и предметная область проявляет несколько упущений.

В последнее время понятие «фреймовый» нередко подменяют значением «объектно-направленный». Данный подход считается прогрессом фреймовый структур. Шаблон фрейма разрешено принимать как класс, экземпляр фрейма — как объект. Языки объектно-направленного программирования (ООП) дают средства разработки классов и элементов, а также средства для описания операций обработки объектов (способы).

Языки ООП, не имеющие рецепта создания сопряженных операций, не разрешают осуществить разностороннее устройство логического вывода, поэтому созданные на них подпрограммы или предполагают собой объектно-направленные базы данных [14], или настоятельно просят интеграции с иными ресурсами отделки знаний (например, с языком PROLOG). Объектно-направленная методика представления знаний создана в таких структурах, как G2, RTWorks.

2.3 Семантические сети

Семантической сетью называется конструкция данных, содержащая установленное значение как сеть. Обычной терминологии семантической сети не существует, но традиционно под ней предполагают последующее: Семантическая сеть — это структура знаний, содержащая установленное значение в облике целостного вида сети, узлы которой связаны с мнениями и элементами, а дуги — взаимоотношениям между ними.

Следственно, различные сети должно исследовать как сети, содержащиеся в составе семантических сетей. К ним могут быть определены и сетевые концепции моделей баз данных. Сама по себе семантическая сеть считается блоком памяти и не объясняет, с помощью чего происходит представление знаний. Поэтому в процессе ознакомления с СОЗ семантические сети обязаны исследоваться как способ представления знаний со способностями выстраивания этих знаний, процессами их применения и приспособлением вывода.

Исследование иерархической конструкции условий и диаграмма представления. В иерархической конструкции условий доступны отношения, по крайней мере, двух типов:

- отношение начала или сравнение (IS - A);
- отношение «целое – часть» (PART - OF).

Например, в предположении «ноутбук» (IS - A) «компьютер» главной особенностью считается, что устройство ноутбук классифицируется к компьютерам. Это означает, что имеет тут присутствует отношение начала или сравнение. То есть, ноутбук представляет собой один из элементов множества различных устройств, образующих класс компьютеров.

Для этих отношений знаковым является то, что экземпляры понятий нижнего уровня имеют все атрибуты мнений верхнего уровня. Это свойство называется наследованием атрибутов между уровнями иерархии (IS - A). Для рассматриваемого примера это означает, что ноутбук, как понятие более низкого уровня, будет обладать всеми свойствами (атрибутами), определенными для понятия компьютер.

Отношение «целое – часть» можно описать словами, которые демонстрируют то, что примеры понятия «процессор» считаются частью любого примера мнения «компьютер». «Процессор» (PART - OF) «компьютер».

Отношения типа (PART - OF) дают возможность увидеть некоторые общие умения (набор атрибутов) для особенного класса мнений. В частности, показанный ранее метод отношения рассказывает, что частным свойством всех компьютеров является расположение в них процессоров. И это свойство будет передаваться всем примерам мнения ноутбук.

Особенно часто используется графическое отображение семантических сетей в виде диаграммы. Также условие «все ласточки – птицы» можно

показать графом, имеющим две вершины согласованные понятиям и дугу, демонстрирующую отношение между этими понятиями рисунок 2.8.

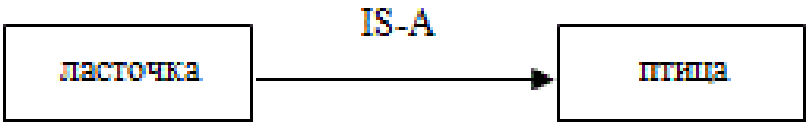

Рисунок 2.8 – Простейшая семантическая сеть

Если ласточка имеет конкретное имя, например, "Ласта", то семантическая сеть может быть расширена, что изображено на рисунке 2.9

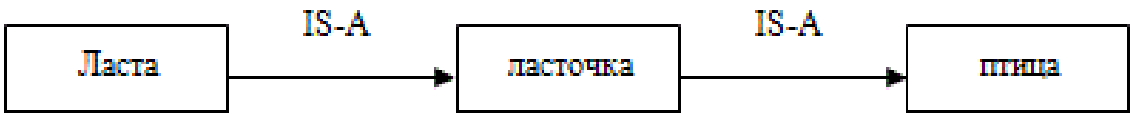

Рисунок 2.9 - Семантическая сеть с иерархией двух понятий

Наряду с тем, что с помощью данной сети описаны два факта: «Ласта – ласточка» и «ласточка – птица», из нее можно вынести, используя отношение наследования, новый факт: «Ласта – птица». Этот факт показывает, что способ представления семантической сетью позволяет легко делать выводы благодаря иерархии наследования между уровнями. Семантическими сетями можно также представлять знания, касающиеся атрибутов объекта. Например, факт «Птицы имеют крылья» можно отобразить в виде рисунке 2.10.

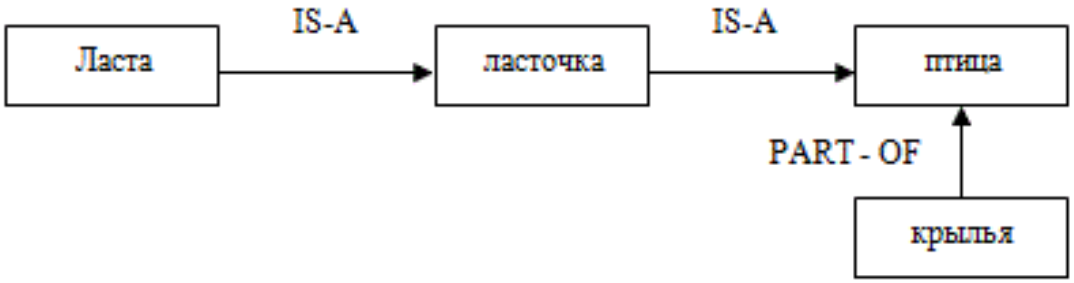

Рисунок 2.10 - Семантическая сеть с понятиями и атрибутами

Это означает, что, используя отношения (IS – A) и (PART – OF) можно создать очередной факт, а именно «Ласта имеет крылья». Вершины в семантической сети обычно иллюстрируют объект исследуемой области, концепт, мнений и т. п., а дуги - это отношения между ними. При расширении семантической сети в ней появляются побочные отношения. Например, если исследуемую сеть обновить фактами: «Ласта владеет гнездом» и «Ласта

владеет гнездом с весны по осень», то выйдет семантическая сеть, показанная на рисунке 2.11.

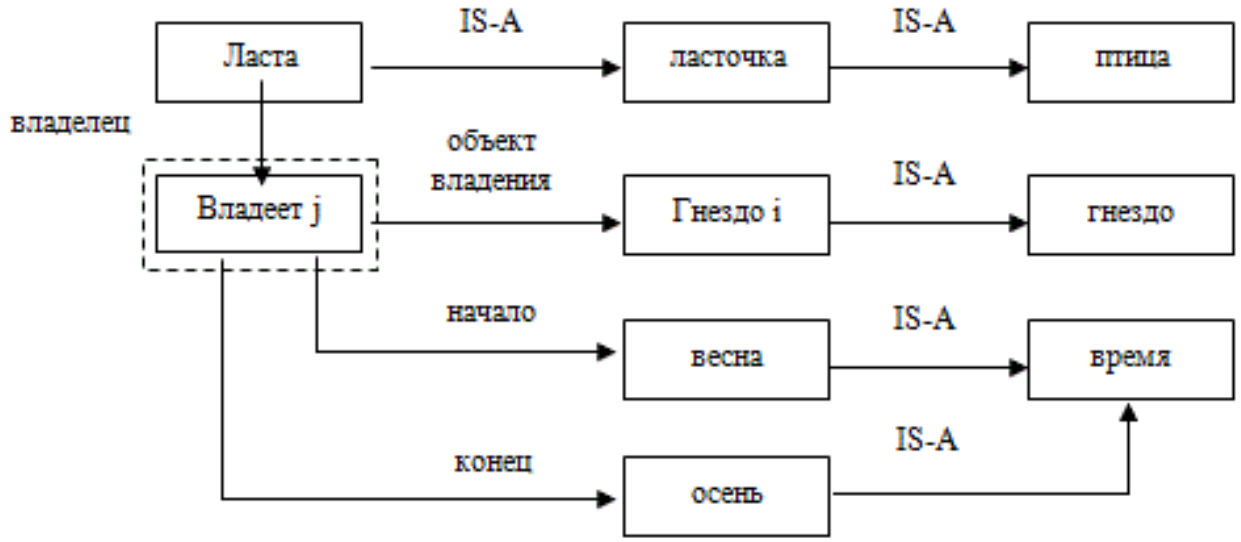

Рисунок 2.11 - Структура расширенной семантической сети

Здесь "Гнездо i" – это конкретное гнездо, которым владеет "Ласта", а для вершины ситуации (например, "Владеет j") определено несколько связей. Такая вершина называется падежной рамкой и определяет различные аргументы предиката ситуации.

Элементы семантической сети. Семантическая сеть изображает собой ориентированный граф с названными дугами и вершинами. Основными элементами сети считаются вершины и дуги. При этом вершинам семантической сети соответствуют понятия, события и свойства, показанные на рисунке 2.12

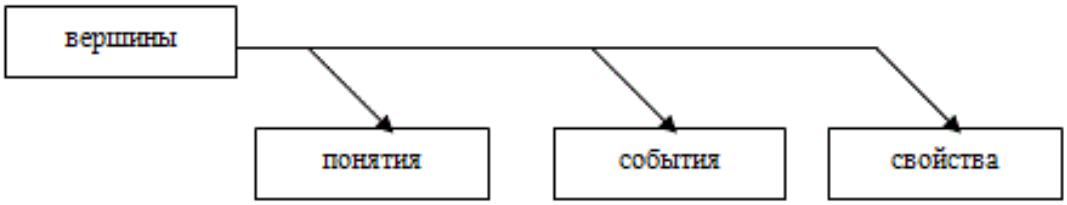

Рис. 2.12. Вершины семантической сети.

Понятия — представляют собой сведения об абстрактных или физических объектах предметной области или реального мира.

События — представляют собой действия происходящие в реальном мире и определяются:

- указанием типа действия;
- указанием ролей, которые играют объекты в этом действии.

Свойства — используются для уточнения понятий и событий. Применительно к понятиям они описывают их особенности и характеристики (цвет, размер, качество), а применительно к событиям — продолжительность, время, место.

Дуги графа семантической сети — отображают многообразие семантических отношений, которые условно можно разделить на четыре класса (рисунок 2.13).

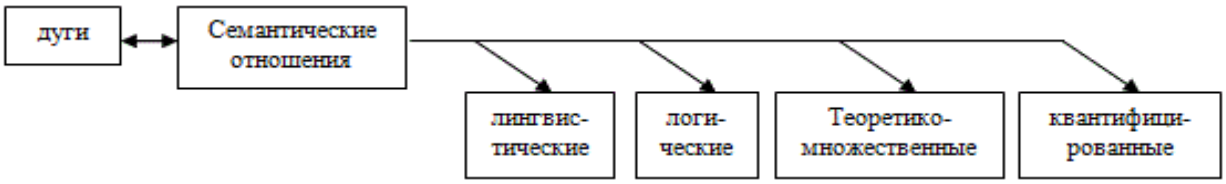

Рисунок 2.13 - Классификация семантических отношений

Лингвистические отношения — отображают смысловую взаимосвязь между событиями, между событиями и понятиями или свойствами. Лингвистические отношения бывают:

– глагольные (время, вид, род, залог, наклонение);

– атрибутивные (цвет, размер, форма);

– падежными (см. ниже).

Логические отношения — это операции, используемые в исчислении высказываний: дизъюнкция, конъюнкция, инверсия, импликация.

Теоретико-множественные — это отношение подмножеств, отношение части и целого, отношение множества и элемента. Примерами таких отношений являются "IS-A" и "PART-OF".

Квантифицированные отношения — это отношения, которые эксплуатируют логические кванторы общности и существования. Они нужны для демонстрации таких знаний как «Любой станок подлежит ремонту», «Существует работник А, обслуживающий склад Б».

В заключении необходимо выделить ряд преимуществ использования семантических сетей:

– описание понятий и событий производится на уровне, очень близком к естественному языку;

– обеспечивается возможность сцепления различных фрагментов сети;

– отношение между понятиями и событиями образуют достаточно небольшое и хорошо формализованное множество;

– для каждой операции над данными и знаниями можно выделить из полной сети, представляющей всю семантику (или все знания), некоторый ее участок, который охватывает необходимые в данном запросе смысловые характеристики.

3 Разработка базы знаний

3.1 Подготовка к работе

Проектирование базы знаний для мобильного приложения [15] является непростой задачей, выполняется разработчиком самостоятельно и состоит из нескольких этапов. База знаний, содержащая логические правила и знания экспертов в области прогнозирования спортивных событий, является важнейшим компонентов в работе мобильного приложения, которое включает в себя помимо базы знаний так же базу данных и пользовательский интерфейс.

Процесс разработки проводится на технологии Android Studio v143.2790544-windows на современном объектно-ориентированном языке программирования Java. Для приступления к работе требуется установить выбранную технологию (рисунок 3.1):

Рисунок 3.1 – Установка Android Studio

Установка состоит из нескольких пунктов, в которых указывается путь установки, выбирается тип установки – полный либо частичный, указываются библиотеки, которые нужны для полнофункциональной работы для создания мобильного приложения. После установки, следует скачать утилиты для корректной работы Android Studio, такие как jdk и sdk.

Следующий пункт – это регистрация на портале GitHub, основная часть изображена на рисунке 3.2. Данное действие необходимо для хранения бэкапов проекта на сервере портала, также на форуме данного портала можно найти множество подсказок и полезных статей по проектированию различных работ для мобильного приложения.

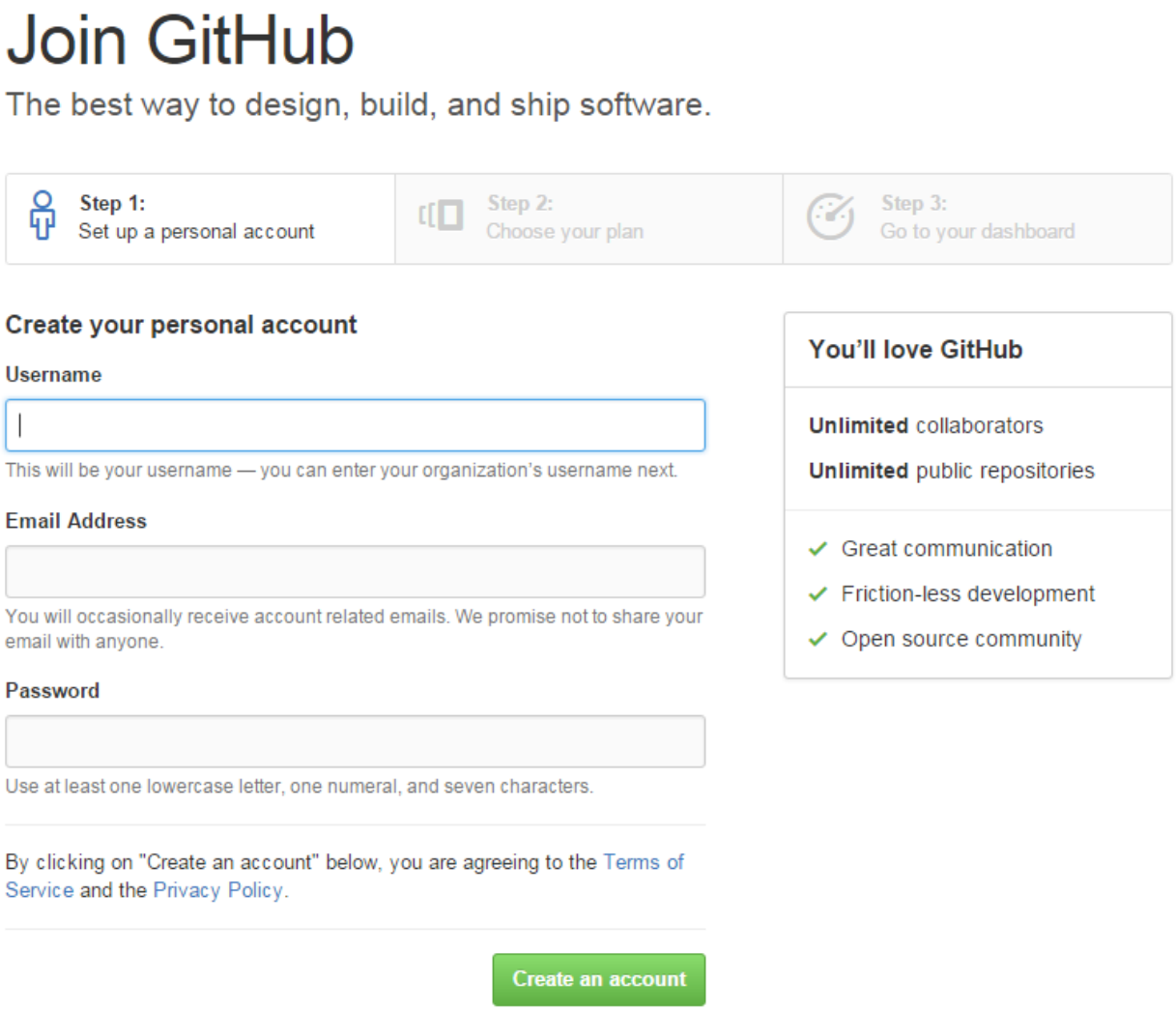

Рисунок 3.2 – Регистрация в GitHub

После регистрации появляется возможность связать свою работу в Android Studio с компьютера разработчика с аккаунтов на GitHub, и проект будет хранится в хранилище, как показано на рисунке 3.3. В нужный момент разработчик может получить доступ к своему проекту на сервере, даже если у него отсутствует личный ПК, разработчик может зайти на аккаунт портала и скачать собственный проект на иной переносной либо персональный компьютер, необходимо лишь наличие Android Studio соответствующей или новейшей версии и доступа к сети интернет.

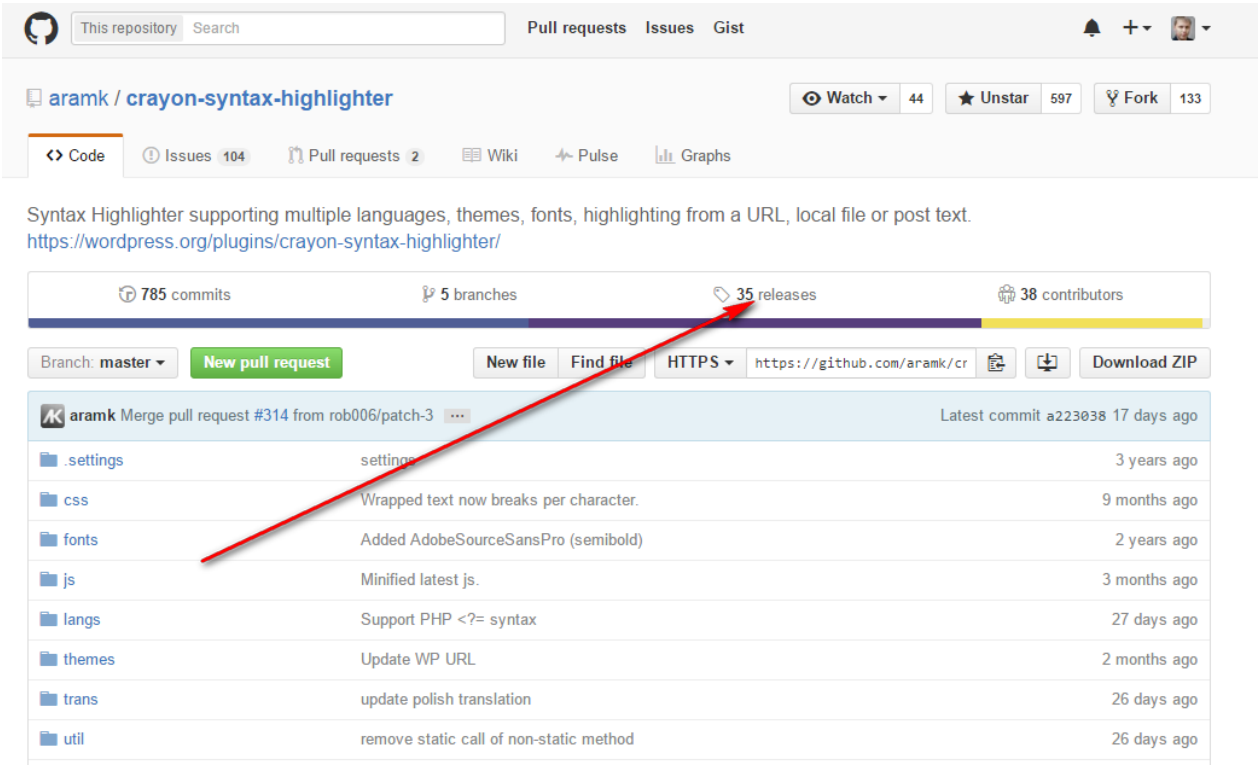

Рисунок 3.3 – Хранилище проектов на GitHub

3.2 Проектирование базы знаний

Начало работы в Android Studio, создаем новый проект, даем название и на экране выводится главное окно программы, которое включает в себя панель управления, панель инструментов, панель отладки, панель структуры приложения и эмулятор смартфона по центру экрана. Все вышеперечисленное отображено на рисунке 3.4.

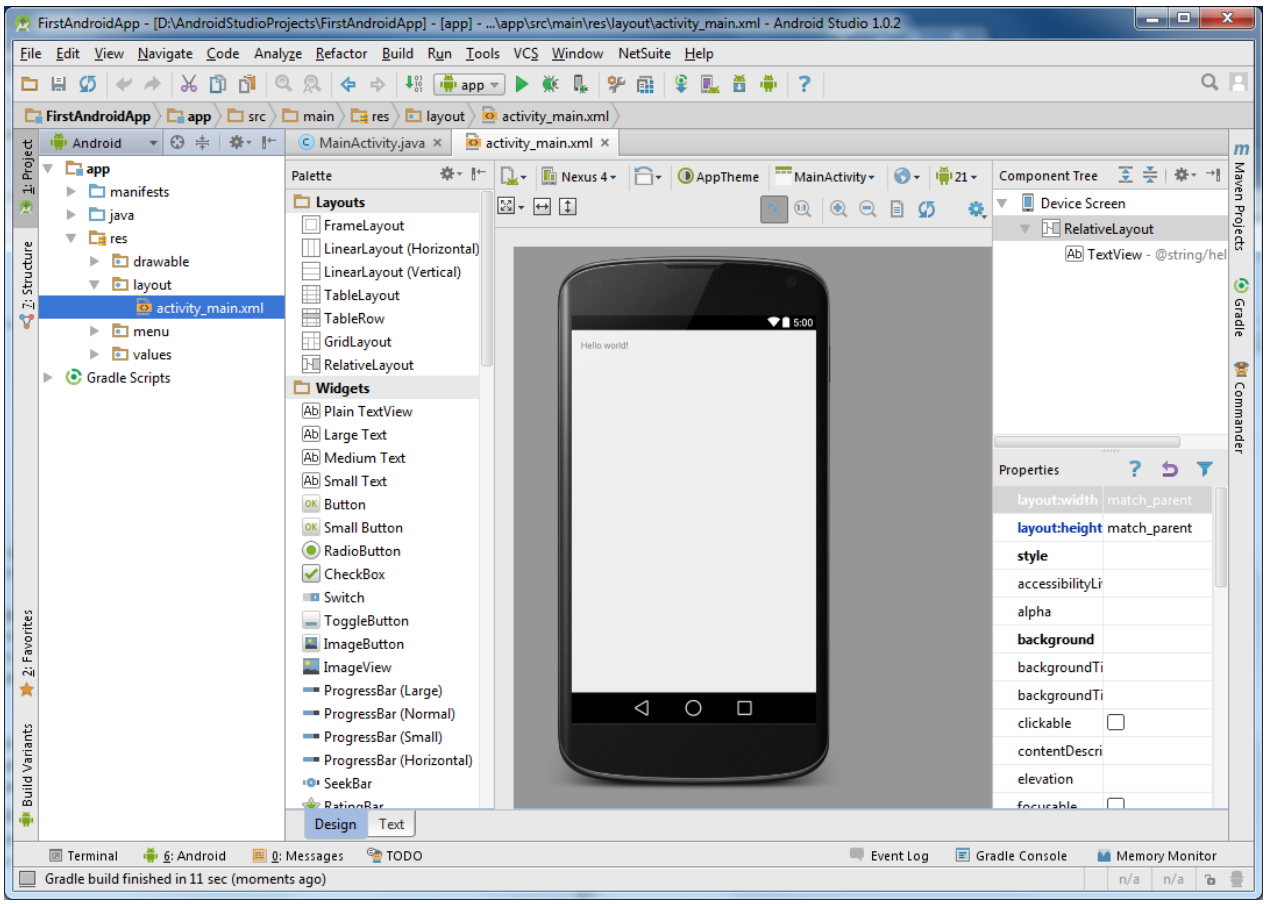

Рисунок 3.4 – Основное меню работы в Android Studio

Начало разработки состоит в том, что программируется код, который: включает в себя подключение библиотек;

– создается класс ListItemOfExperts, который наследует функции класса Fragment;
– создается список матчей, для которых есть мнения экспертов;
– создается список для вывода текста из hashMap.

Все выше перечисленное изображено на рисунке 3.5, на котором показана часть кода

```
import android.os.Bundle;
00 mport android.support.v4.app.Fragment;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

/**...*/
public class ListItemOfExperts extends Fragment {

    String text_experts;
    TextView txt;


    public ListItemOfExperts() {
        // Required empty public constructor
    }


    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View v =inflater.inflate(R.layout.list_item_of_experts, container, false);
        text_experts = this.getArguments().getString("expert");
        txt = (TextView) v.findViewById(R.id.expert_text);
        txt.setText(text_experts);
        return v;
    }
```

Рисунок 3.5 – Создание класса ListItemOfExperts

На приложенном рисунке 3.6 видно, что программой генерируется hashMap<string, string>. Далее идет подключение к сайту stavochka.com и запрашивается класс .b-forecast_th a. В следующем шаге выводится ссылка для данного класса. В hashMap Put кладется текст для данного класса в виде ключа, а в виде значения – ссылка перехода для полного объяснения матча.

```
class ParseTitle_experts extends AsyncTask<Void,Void,HashMap<String,String>>{

    @Override
    protected HashMap<String, String> doInBackground(Void... params) {
        HashMap<String,String> hashMap = new HashMap<>();
        try {
            Document document = Jsoup.connect("http://stavochka.com/forecasts/").get();
            Elements elements = document.select(".b-forecasts__th a");
            for(Element element :elements){
                Element element1 = element.select("a[href]").first();
                hashMap.put(element.text(),element1.attr("abs:href"));
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return hashMap;
    }
}
```

Рисунок 3.6 – Парсим информацию с сайта

Рисунок 3.7 демонстрирует работу бэкграунд процесса, реализованного с помощью класса ParseText_experts наследованного от класса AsyncTask. Демонстрация класса AsyncTask, на вход подается ссылка выбранного матча. После этого мы берем полную информацию обоснования матча и присваиваем это значение строке str.

```
class ParseText_experts extends AsyncTask<String,Void,String>{

    @Override
    protected String doInBackground(String... params) {
        String str = " ";
        try {
            Document document = Jsoup.connect(params[0]).get();
            Element elements = document.select(".b-forecast-text").first();
            str = elements.text();
        } catch (IOException e) {
            e.printStackTrace();
        }

        return str;
    }
}
```

Рисунок 3.7 – Переход непосредственно на мнение эксперта

3.3 Результаты работы приложения

По итогам работы получена часть мобильного приложения, отвечающая за предоставление списка матчей, на которые существуют авторитетные

мнения экспертов-знатоков в предметной области. Эта часть доступна в готовом проекте, как показано на рисунке 3.8

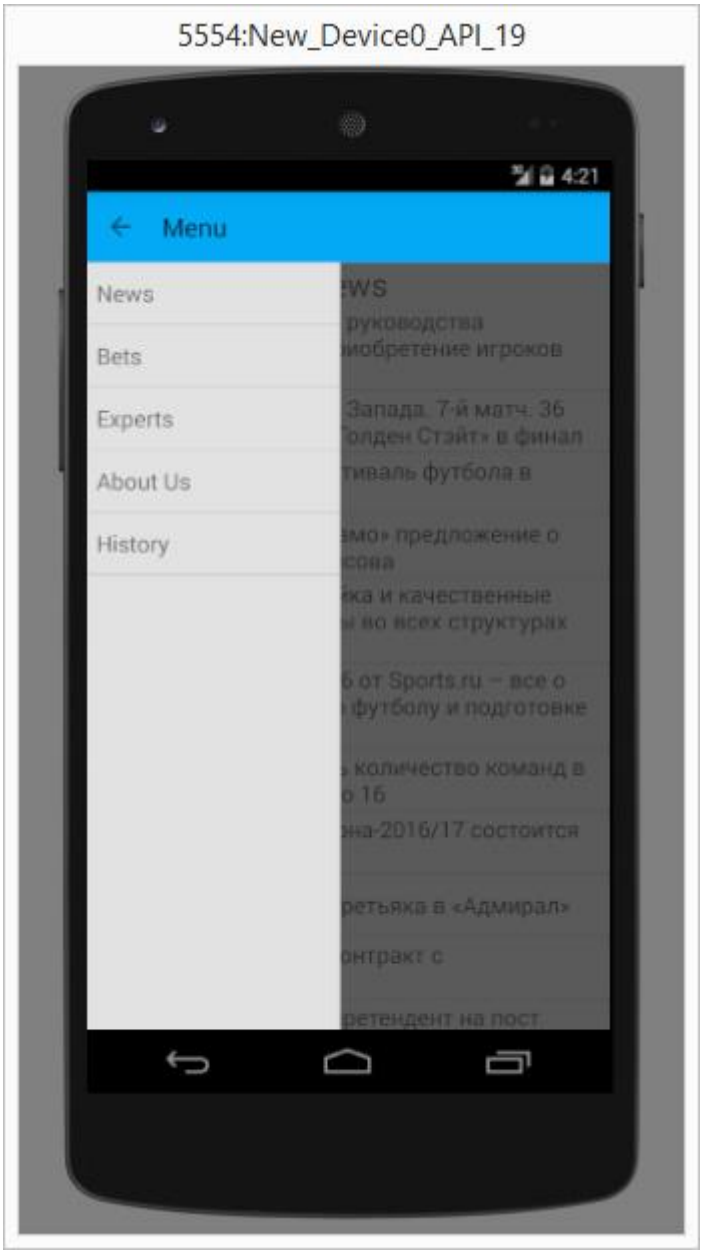

Рисунок 3.8 – Главное меню

Главное меню [16] включает в себя пять разделов, которые доступны пользователю. За прогнозы отвечает вкладка «Experts», перейдя на которую, нам станет доступен список матчей, на которые составлены обоснованные точки зрения от экспертов и знатоков данной футбольной области. На рисунке 3.8 изображен принскрин эмулятора, на котором виден список ближайших футбольных матчей.

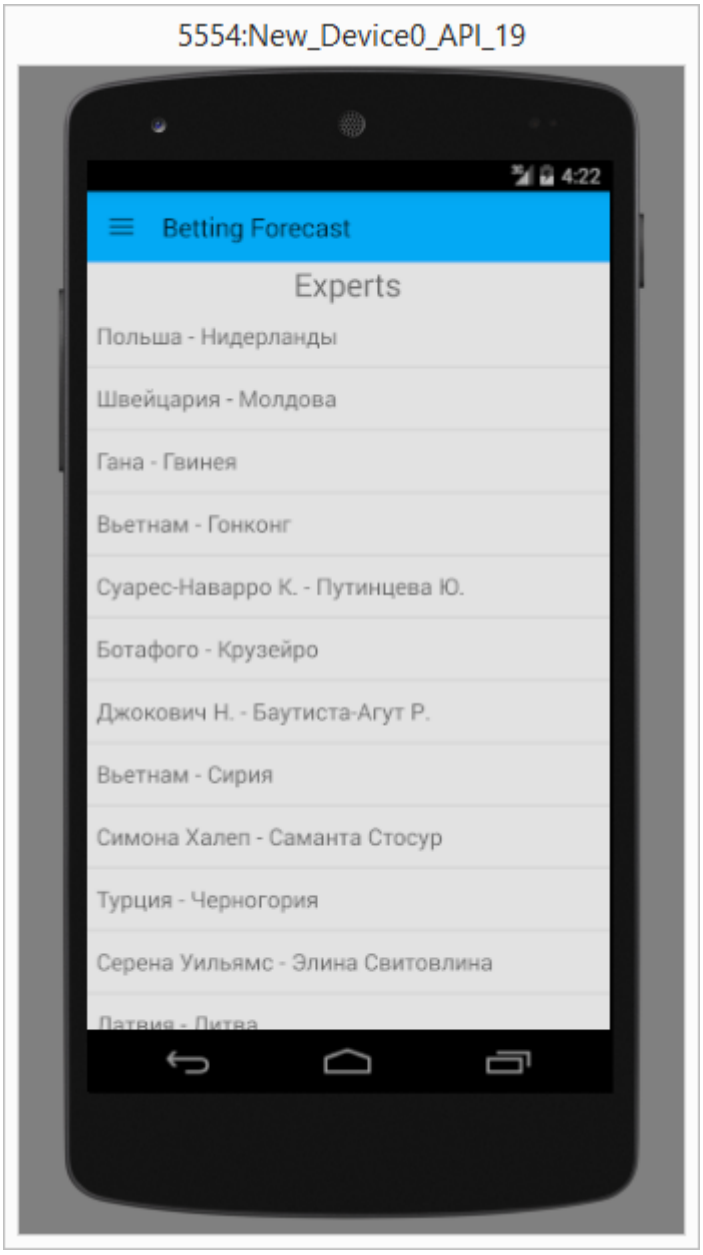

Рисунок 3.9 – Вкладка Experts

Далее, после выбора нужного матча пользователь может перейти на непосредственно само мнение эксперта, нажав на соответствующий интересующий матч. Переход осуществляется мгновенно и пользователю сразу доступен текст, обосновывающий точку зрения некоторого эксперта, как показано на рисунке 3.9.

Рисунок 3.10 – Переход на мнение эксперта

Как видно на принскрине, речь идет о матче между странами международного уровня и имеет тип «товарищеский». На экране полностью и развернуто расписано мнение знатока, и пользователь может учесть это мнение перед выбором ставки на данное спортивное событие в букмекерской конторе.

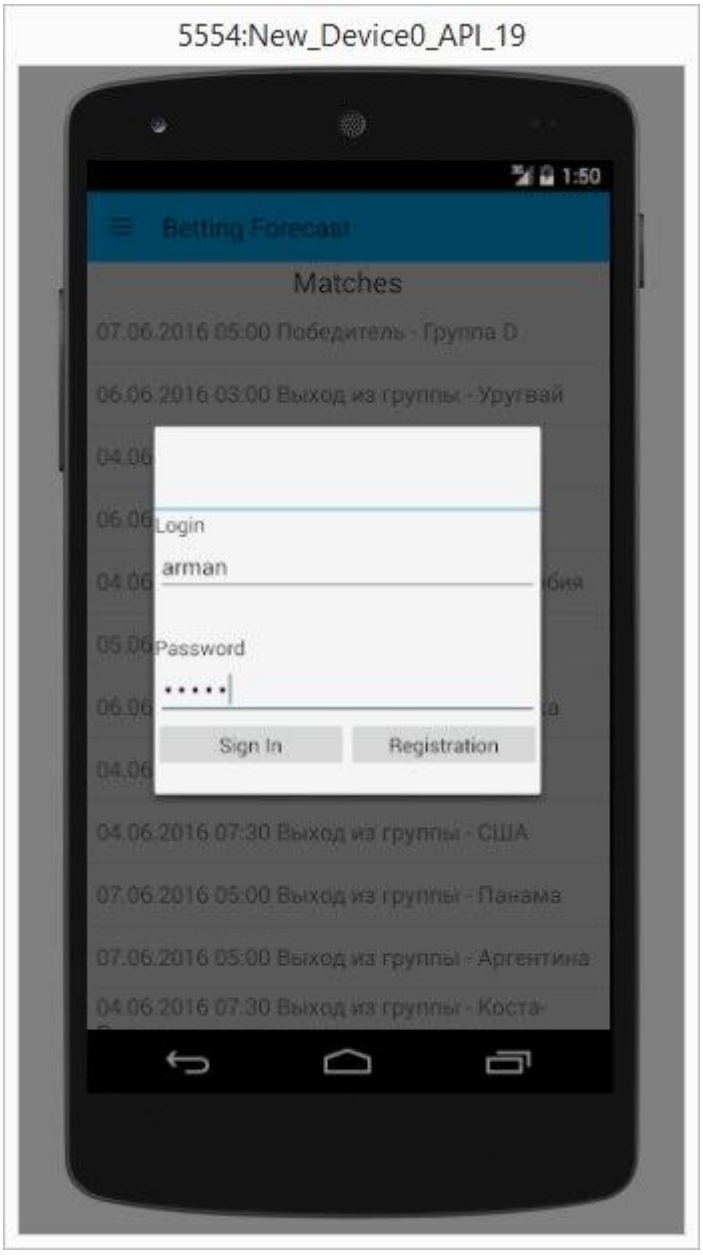

Рисунок 3.11 – Форма аутентификации

В приложении разработана форма запроса, имени пользователя и пароля, для хранения данных и действий прошлого сеанса, показанная на рисунке 3.11. В случае, если не имеется личная учетная запись, доступна кнопка регистрации.

В этой главе была описана подготовка к разработке, сама разработка и принцип работы приложения, включающего в себя базу знаний. Описание подкреплено письменной инструкцией и рисунками с демонстрацией работы мобильно приложения.

4 Технико-экономическая часть

4.1 Определение задач и объяснение эффективности

Тема данного дипломного проекта – Разработка базы знаний для мобильного приложения «Прогнозирование спортивных событий».

В наше время резко повысилась популярность букмекерских контор. Люди пользуются их услугами и рассчитывают получить прибыль, основываясь на предположениях и догадках о некотором спортивном событии. Данный проект поможет людям в достижении успеха, предоставляя всю возможную информацию и статистику. Экспертная система, основываясь на своей базе знаний, помогает человеку рассчитать риски и выбрать оптимальную ставку на спортивное событие.

4.2 Расчет окончательных материальных затрат на все компоненты программного продукта

Расчет итоговых затрат [17] на разработку компонентов дипломного проекта в виде информационных технологий ($С_{пі}$) производится по формуле:

$$C_{пі} = З_{фот} + З_{сзі} + M_i + Р_{сі} + Р_{мі} + П_{зі} + Р_{ні} \quad (4.1)$$

где $З_{фот}$ – полный объем награждения работы разработчиков, тенге;

$З_{сзі}$ – платеж по социальному налогу, тенге;

M_i – издержки на сырье и материалы, тенге;

$Р_{сі}$ – издержки на особенные программные компоненты, нужные для разработки проектного решения, тенге;

$Р_{мі}$ – платежи, основанные на пользовании машинной аппаратурой, тенге;

$П_{зі}$ – побочные затраты, тенге;

$Р_{ні}$ – накладные расходы, затраты на содержание средств тенге.

Размер полного объема награждения работы разработчиков ($З_{ФОТ}$) находится по формуле:

$$З_{фот} = З_{оі} + З_{ді} \quad (4.2)$$

где $З_{оі}$ – основная заработная плата, тенге;

$З_{ді}$ – дополнительная заработная плата, тенге.

Затраты на оплату труда определяются от объема и трудоемкости создания программного обеспечения. Полное количество (V_0) программной

части дипломной работы находится на вычислениях массы и размера функций, реализуемых программой:

$$V_0 = \sum_{i=1}^{n} Vi \quad (4.3)$$

где V_i - объем отдельной функции ПО;

n − общее число функций.

Из приложения устанавливаем объем ПО (строки исходного кода, LOC) составляют 4770. Таким образом:

$$V_0 = 4770 \text{ строк кода}$$

Общая трудоемкость проекта находится по формуле:

$$T_o = T_н \cdot K_c \cdot K_т \cdot K_н \quad (4.4)$$

где K_c – коэффициент, учитывающий трудность создания ПО;

$K_т$ – коэффициент поправки, который берет в расчет степень эксплуатации при создании условных модулей;

$K_н$ – коэффициент, берущий в расчет новизну ПО;

$T_н$ – нормированная трудоемкость.

Коэффициент трудности определяется относительно данных, представленных в таблице 4.1 и имеет значение K_c = 0,18, т.к. в проекте имеют место быть более 3-х характеристических параметров – программный функционал, интерактивное взаимодействие, обеспечение недосягаемости, управления и поиска данных в многоуровневых структурах.

Таблица 4.1 – Побочные значения трудности создания ПО

Характеристика ПО	Значения K_c
1 Работоспособность ПО в общей операционной сфере (связь с другими ПО)	0,08
2 Интерактивный допуск	0,06
3 Гарантия сохранности, ведения и розыска инфы в многосложных конструкциях	0,07
4 Содержание в ПО в одно время нескольких параметров по табл.Г4.1, приложение Г	
4.1 2 характеристики	0,12
4.2 3 характеристики	0,18
4.3 Свыше 3-х характеристик	0,26

Коэффициент поправки, который принимает во внимание частоту употребления при работе над проектом дефолтных модулей ($K_т$),

высчитывается на основании данных, доступных в таблице 4.2 и имеет значение 0,7.

Таблица 4.2 – Показания уточняющего значения

Размер охвата используемых функций создаваемого ПО классическими модулями, типовыми продуктами и ПО	Значения $К_т$
1 От 62 % и выше	0,6
2 От 42 % до 61	0,7
3 От 18 % до 41 %	0,8
4 До 17 %	0,9
5 Стандартное ПО не применяемое для проектировки функционала создаваемого ПО	1,0

Уточняющее значение, берущий в расчет свежесть создаваемого продукта ($К_н$) определяется на основе данных, представленных в таблице 4.3 и составляет 0,9.

Таблица 4.3 – Поправочные коэффициенты

Классификация свежести	Коэффициент свежести	Использование		Значение $К_н$
		На основе свежего типа ПК	В среде специфичной ОС	
А	Абсолютно новые ПО, к которым не существует аналогов	+ – + –	+ + – –	1,75 1,6 1,2 1,0
Б	ПО, которые считаются развитием конкретного характерного ряда ПО	+ – +	+ – –	1,0 0,9 0,8
В	ПО, считающиеся развитием конкретного характерного ряда ПО, созданных для заранее изученных видов конфигурации ПК и ОС	–	–	0,7

Базой для определения нормативной трудоемкости являются, укрупненные нормы времени на разработку проекта в зависимости от уточненного объема ПО и группы сложности.

Нормативная трудоемкость проекта ($Т_н$) определяется на основе принятого в расчет объема ПП и классификации трудности, которая уточняется с учетом сложности и новизны проекта и степени использования стандартных модулей при разработке.

Основываясь на информации из стандарта: для верхнего класса трудности ПО:

$$Т_н = 159$$

Целый размер трудоемкости высчитывается по формуле (4.4)

$$Т_о = 159 \cdot 0{,}18 \cdot 0{,}7 \cdot 0{,}9 = 18 \text{ чел./час.}$$

Учитывая сложность работы, устанавливаются рекомендуемое количество проектировщиков ($Ч_р$) и рекомендуемые границы, выделенные для создания проекта в итоге ($Т_р$). Помимо этого поставляются определенные задания:

– подсчет числа разработчиков при установленных рамках выполнения работы;

– предположение срока проектирования при заданном числе разработчиков.

Количество разработчиков проекта ($Ч_р$) высчитывается по формуле:

$$Ч = Т_о / (Т_р \cdot Ф_{эф}) \quad (4.5)$$

где $Ф_{эф}$ – значимый объем срока деятельности одного проектировщика на протяжении года (дн.);

$Т_о$ – целая трудоемкость разработки проекта (чел./дн.);

$Т_р$ – временные рамки создания проекта.

Длительность разработки продукта ($Т_р$) находится по методу:

$$Т_р = Т_о / (Ч_р \cdot Ф_{эф}) \quad (4.6)$$

где $Ч_р$ – рекомендуемое количество проектировщиков.

Эффективный фонд времени работы одного работника ($Ф_{эф}$) рассчитывается по формуле:

$$Ф_{эф} = Д_г – Д_п – Д_в - Д_о \quad (4.7)$$

где $Д_г$ − количество дней в году;

$Д_п$ – количество праздничных дней в году;

$Д_в$ − количество выходных дней в году;

$Д_о$ − количество дней отпуска.

В соответствии с общепринятым календарем на 2016 год:

- $Д_г$ = 366;
- $Д_п$ = 15;
- $Д_в$ = 112;
- $Д_о$ = 14, эффективный фонд времени одного работника составит:

$$Ф_{эв} = 366 – 15 – 112 – 14 = 225 \text{ дней}$$

Установленное число разработчиков $Ч_р$ = 1, следовательно, по формуле (4.6)

$$Т_р = 18 / (1 \cdot 225) = 0{,}08 \text{ лет } = 30 \text{ дней}$$

Таким образом, учитывая выше представленные расчеты и в соответствие с формулой (4.5)

$$Ч = 18 / (0{,}08 \cdot 225) = 1 \text{ чел}$$

Основная заработная плата разработчиков на конкретизированное ПО рассчитывается по формуле:

$$З_{оі} = \sum_{i=1}^{n} Т_{чi} \cdot Т_ч \cdot К \qquad (4.8)$$

где n − количество исполнителей, участвующих в разработке конкретного ПО; $Т_{чі}$ − часовая тарифная ставка некоторого проектировщика программного продукта (тыс. тенге);

$Ф_п$ − плановый фонд рабочего времени і–го исполнителя (дней), число рабочих дней в месяц;

$Т_ч$ − количество часов работы в день (час), 8 часов; К − коэффициент премирования, составляет 1,38.

На разработку уйдет 1 месяц, из них рабочие дни составят:

$$Ф_п = 23 \text{ раб. дней}$$

Учитывая специфику и трудность функционала составляется обязательный график работы программиста–проектировщика, принимающего участие в создании ПО, с классификацией знаний, специальности, уровня подготовки и статуса (таблица 4.4).

Таблица 4.4 - Данные по сотрудникам, задействованным в работе

Специалист-Исполнитель	Количество, чел-к	Заработная плата в месяц, тенге
Разработчик	1	100 000,00
Итого	1	100 000,00

Почасовая регулярная сумма высчитывается путем разделения тарифной суммы, предложенной при еженедельном стандарте рабочего срока (Фр)

$$Т_ч = \frac{Т_м}{Ф_р} \quad (4.9)$$

где $Т_ч$ − часовая регулярная сумма (тыс.тенге); $Т_м$ − месячная регулярная сумма (тыс.тенге).

Целый временной капитал:

$$Ф_р = Т_ч \cdot Ф_п \quad (4.10)$$

Из этого следует:

$$Ф_р = 8 \cdot 23 = 184 \text{ часов}$$

Регулярная сумма ставки проектировщика:

$$Т_ч = 100000 / 184 = 543 \text{ тенге в час}$$

Согласно формуле (4.8) основная зарплата проектировщика будет равна

$$З_{оi} = 543 \cdot 1{,}38 \cdot 184 = 137878{,}5 \text{ тенге}$$

Результаты главной зарплаты представлены в виде таблице 4.5.

Таблица 4.5 – Приведенные итоги вычисления издержек

Наименование содержания работ	Исполнитель	Трудоёмкость норма–час	Заработная плата за час работы, т/час	Сумма заработной платы, тенге
ТЗ	Разработчик	45	543	24435
Моделирование	Разработчик	30	543	16290
Программирование	Разработчик	60	543	32580
Тестирование	Разработчик	29	543	15747
Внедрение	Разработчик	20	543	10860
Итого		184		99912

Дополнительная бонусная зарплата насчитывает 23% от ключевой зарплаты и ищется по формуле:

$$З_{дi} = З_{oi} \cdot Н_д / 100 \quad (4.11)$$

где $Н_д$ – коэффициент бонусной заработной платы разработчика

$$З_{дi} = 137878,5 \cdot 23 / 100 = 31712 \text{ тенге}$$

Итоговая сумма зарплаты деятельности проектировщика равен:

$$З_{фот} = 158700 + 36501 = 169590,5 \text{ тенге}$$

Социальные отчисления составляют 11% (ст. 358 п. 1 НК РК) от дохода работника, и рассчитываются по формуле:

$$З_{сзi} = (ФОТ – ПО) \cdot 11\% \quad (4.12)$$

где ПО – пенсионный налог, который составляет 10% от ФОТ и социальными отчислениями не облагаются:

$$ПО = ФОТ \cdot 10\ \% \quad (4.13)$$

Таким образом:

$$ПО = 169590,5 \cdot 0,1 = 16959 \text{ тенге}$$

$$З_{сзi} = (169590,5 – 16959) \cdot 0,11 = 16789,47 \text{ тенге}$$

Затраты на ресурсы определяются по формуле:

$$М_i = (З_{oi} \cdot Н_{мз}) / 100\% \quad (4.14)$$

где $Н_{мз}$ − норма расхода материалов от основной заработной платы (3–5%)

$$М_i = 137878,5 \cdot 0,03 = 4136,36 \text{ тенге}$$

Расходы по статье «Аппаратное время» ($Р_{мi}$) включают оплату техники, необходимой для создания и отладки ПО, время использования которого определяется в соответствии с условными нормативами (в машино – часах) на 100 строк исходного кода ($Н_{мв}$) аппаратного времени в зависимости от классификации поставленных задач и типа ПК

$$Р_{мі} = Ц_{мі} \cdot (V_{оі} / 100) \cdot Н_{мв} \quad (4.15)$$

где $Ц_{мі}$ − цена одного машино–часа (тыс.тенге);

$V_{оі}$ − общий объем ПО (строк исходного кода);

$Н_{мв}$ − норматив расхода машинного времени на отладку 100 строк исходного кода (машино–часов).

Норматив расхода аппаратного времени на отладку 100 строк исходного кода определяется на основе таблицы Д.1 (Приложения Д) и составляет 12 ч/100 строк кода

$$Р_{мі} = 543 \cdot (4770 / 100) \cdot 12 = 310813{,}2 \text{ тенге}$$

Издержки по пункту «Второстепенные расходы» ($П_{зі}$) на конкретно ПО содержат в себе расходы на закупку и установку особенной научно–технической информации и специальной литературы. Определяются по стандарту, составленному в обобщенном виде по компании, в процентах к главной зарплате:

$$П_{зі} = З_{оі} \cdot Н_{пз} / 100 \quad (4.16)$$

где $Н_{пз}$ − норматив прочих затрат в целом по организации в (20%)

Таким образом:

$$П_{зі} = 137878{,}5 \cdot 0{,}2 = 27575{,}7 \text{ тенге}$$

Издержки по пункту «Накладные затраты» ($Р_{ні}$), рассчитываются по стандарту ($Н_{рн}$) в определенном количестве к главное заплате проектировщиков. Стандарт задается обычно в компании

$$Р_{ні} = З_{оі} \cdot Н_{рн}/100\% \quad (4.17)$$

где $Р_{ні}$ – накладные расходы на конкретную ПО (тыс.тенге);

$Н_{рн}$ – стандарт накладных затрат в общем по компании семьдесят процентов. После расчетов, имеем, что накладные затраты равны:

$$Р_{ні} = 137878{,}5 \cdot 0{,}7 = 96514{,}95 \text{ тенге}$$

Итоговые затраты на проектировку базы знаний, формула (4.1)

$$С_{пі} = 169590{,}5 + 16789{,}47 + 4136{,}36 + 310813{,}2 + 27575{,}7 + 96514{,}95 = 625420 \text{ тенге}$$

Заключительные результаты расчета затрат на проектировку базы знаний и расходы в табличном и структурированном виде представлены в таблице 4.6 и на рисунке 4.1.

Таблица 4.6 Результаты расчета затрат на разработку базы знаний

Затраты на разработку	Краткое выражение	Количество, тенге	В процентах от итогового числа
Объем награждения работы	$З_{фот}$	169590,5	27
Социальные издержки	$З_{сзі}$	16789,47	3
Ресурсы	$М_і$	4136,36	1
Аппаратные сроки	$Р_{мі}$	310813,2	50
Побочные издержки	$П_{зі}$	27575,7	4
Накладные затраты	$Р_{ні}$	96514,95	15
Результат		625420	100

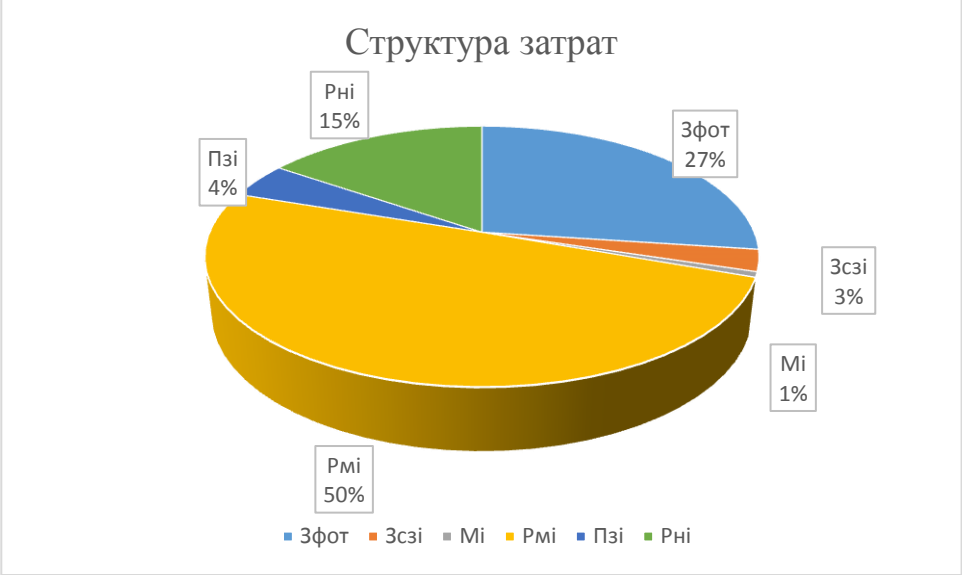

Рисунок 4.1 – Структура затрат на разработку экспертной системы

4.3 Расчет цены программного продукта

Расчет цены продукта, который создан самостоятельно проектировщиком и не собирается тиражироваться, находится с помощью:

$$Ц_{пп} = З_{рпр} + П_п + НДС \quad (4.18)$$

где $Ц_{пп}$ – цена программного продукта, тенге;

$З_{рпр}$ – затраты на создание продукта, в данном случае программного продукта, тенге;

$П_{п}$ – ожидаемый доход, тенге;

НДС – налог на дополнительную стоимость, тенге.

Запланированный доход равен (20%) от собственной стоимости проектировки

$$ПП = 625420 \cdot 0,2 = 125084 \text{ тенге}$$

НДС, начисленный на ПП, считается так:

$$НДС = (З_{рпр} + П_{п}) \cdot k_{НДС} \quad (4.19)$$

где $k_{НДС}$ – ставка налога на добавленную стоимость.

Применив данные в формуле (4.19), выводим

$$НДС = (625420 + 125084) \cdot 0,12 = 90060,5 \text{ тенге}$$

Подставив данные в формулу (4.18) выводим результат

$$Ц_{пп} = 625420 + 125084 + 90060,5 = 840564,5 \text{ тенге}$$

Вывод. Проектирование базы знаний является дорогостоящим проектом, требующего значимых интеллектуальных и денежных вложений, а также обязательного использования компьютерной техники и программного обеспечения. В экономической части дипломной работы был произведен расчет затрат на разработку базы знаний и расчет цены программного продукта. Итоговая стоимость разработки базы знаний для прогнозирования спортивных событий составила 625420 тенге, которая состоит из всех возможных затрат при разработке проекта.

Программный продукт создан для пользования на платной основе. Проект является целесообразным, так как в настоящее время область азартных игр и ставок имеет очень высокую популярность. Огромное количество людей рассчитывают получить прибыль на ставках, данный программный продукт даст им больше возможностей. Разработка данного программного продукта является социально эффективной, так как он поможет людям в достижении успеха, предоставляя всю возможную информацию и статистику. Экспертная система, основываясь на своей базе знаний, помогает человеку рассчитать риски и выбрать оптимальную ставку на спортивное событие.

5 Безопасность жизнедеятельности

5.1 Анализ условий труда

Главными задачами БЖД являются: распознавание вида и степени опасности, разработка средств для защиты от опасности и устранение либо предотвращение опасности. Целью дипломного проекта является разработка базы знаний, при помощи которой мобильное приложение будет способно с определенной долей вероятности оценивать результат спортивного события. База знаний будет разрабатываться в офисе, на персональном компьютере. Исходя их этого, проводится анализ рабочего места и создаются все условия для комфортной работы.

Основными пунктами в оценке рабочего места являются:

- естественное и искусственное освещение;
- вентиляция;
- уровень шумового фона;
- вредное воздействие компьютера на организм;
- режим работы и отдыха, нагрузка на глаза и мышцы;
- чрезмерно высокий уровень статического электричества;
- чрезмерно высокий уровень ионизирующего излучения;
- пожаробезопасность помещения.

Рассмотрим офис, в котором ведется разработка проекта, изображенного на рисунке 5.1. Размеры помещения: длина 4м, ширина 5м, высота потолка 3м. Имеется окно, двойное, размеры 2х2м. Искусственное освещение – два светодиодных светильника с двумя лампами каждый. Рекомендуемое количество рабочих мест – 2. Каждому из сотрудников выделены стол (2х1.5м) и офисное кресло.

Таблица 1 – Показатели оптимальных величин микроклимата в офисе

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

Место разработки по зрительным условиям работы относится к классу легких работ [18] (легкая физическая, категория Iа, работа производится сидя и не требует физического напряжения). Тип работы – проектирование базы знаний на персональном компьютере. Оборудование малошумящее – вред от шума отсутствует.

Основным источником электронных полей считаются заряды на дисплее персонального компьютера. Заряды электризуют взвешенные пылинки, повышая концентрацию электричества возле рабочего места. Это может являться фактором возникновения недугов и заболеваний дыхательной системы, может способствовать развитию аллергических болезней, что имеет отрицательное влияние на систему здоровья пользователей. При несоблюдении технической безопасности, рабочий кабинет может поддаться возгоранию. Причиной пожара может служить неполадки в электрической технике, перегрузка проводов, неисправности в переключателях и розетках. На случай чрезвычайной ситуации в коридоре перед рабочим офисом расположен в подвешенном состоянии щит пожарного инвентаря, включающий в себя из: лопаты, ведра, лом, топор, ящик с песком. На этаже в коридоре находится внутренний противопожарный кран. Для тушения незначительного возгорания используют ручные углекислотные огнетушители типа ОУ–8, емкостью 8л. Такие огнетушители чрезвычайно эффективны для борьбы с огнем в закрытых помещениях и используются для аппаратов, находящихся под напряжением.

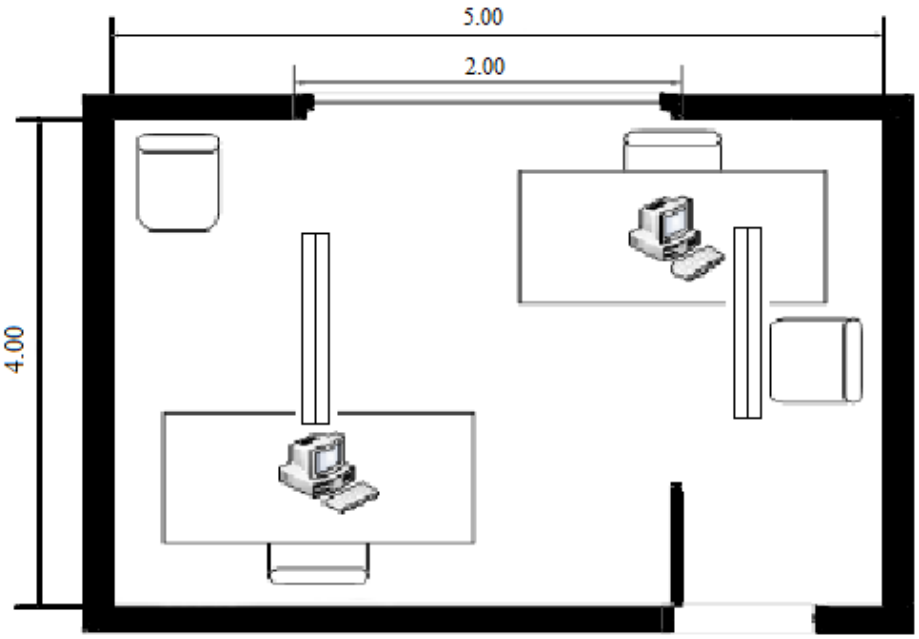

Рисунок 5.1 – План рабочего места

Проанализировав условия труда, рассмотрим более подробно следующие моменты.

5.2 Расчет системы кондиционирования

Кондиционирование оказывает влияние на улучшение микроклимата в помещении и должно выполняться в соответствии с нормами [19]. Стандарт пространства комнаты на одного разработчика равен 6 м^2. В офисе установлены воздуховоды, которые размещены за подвесными потолками. Воздух плавно и равномерно поступает и удаляется из офиса либо через вентиляционные решетки в стенах, либо через воздухопровод, установленный на потолке. Также установлен кондиционер, который освежает воздух и регулирует температуру в помещении.

Для расчета примем размеры рабочего помещения в соответствии с рисунком 5.1.

- длина a = 4 м;
- ширина b = 5 м;
- высота h = 3 м;
- окно = 2x2 м.
- $V_{вент}$ – объем воздуха, необходимый для обмена;
- $V_{пом}$ – объем рабочего помещения.

Соответственно объем помещения равен

$$V_{пом} = a \cdot b \cdot h = 60 \text{ м}^3 \quad (5.1)$$

Необходимый для обмена объем воздуха $V_{вент}$ определим исходя из уравнения теплового баланса

$$V_{вент} \cdot C \, (t_{уход} – t_{приход}) \cdot Y = Q \cdot 3600 \quad (5.2)$$

где $V_{вент}$ - объем воздуха, необходимый для обмена;

$Q_{избыт}$ - избыточная теплота (Вт);

C - удельная теплопроводность воздуха, C=1000 (Дж/кгК);

Y - плотность воздуха, Y=1,2 мг/см^3.

$t_{приход}$ - 18 градусов;

Температура уходящего воздуха определяется по формуле:

$$t_{уход} = t_{р.м.} + (h – 2)t \quad (5.3)$$

где t = 1–5 градусов – превышение t на 1м высоты помещения;

$t_{р.м.}$ = 26 градусов – температура на рабочем месте;

h = 3 м – высота помещения;

$t_{уход} = 26 + (3 – 2) 2 = 28 \, ^oC$

$$Q_{избыт} = Q_{изб.1} + Q_{изб.2} + Q_{изб.3} \quad (5.4)$$

где $Q_{изб.}$ – избыток тепла от электрооборудования и освещения;

$$Q_{изб.1} = E \cdot P \quad (5.5)$$

где E – коэффициент потерь электроэнергии на теплоотвод E=0,55;

P – мощность;

$$P = 60 \text{ Вт} \cdot 5 = 300 \text{ Вт}$$

$$Q_{изб.1} = 0.55 \cdot 300 = 165 \text{ Вт}$$

$Q_{изб.2}$ – теплопоступление от солнечной радиации

$$Q_{изб.2} = m \cdot S \cdot k \cdot Q_c \quad (5.6)$$

где m – число окон, m = 1;

S – площадь окна, S = 4 $м^2$

K – коэффициент, учитывающий остекление. Для двойного остекления k = 0,6;

Q_c = 200 $Вт/м^2$ – теплопоступление от окон.

$$Q_{изб.2} = 4 \text{ м}^2 \cdot 1 \cdot 0.6 \cdot 200 \text{ Вт/м}^2 = 480 \text{ Вт}$$

где $Q_{изб.3}$ – тепловыделения людей

$$Q_{изб.3} = n \cdot q \quad (5.7)$$

где q = 80 Вт/чел,

n – число людей, в данном случае, n = 2

$$Q_{изб.3} = 2 \cdot 80 = 160 \text{ Вт}$$

$$Q_{избыт} = 165 + 480 + 160 = 805 \text{ Вт}$$

Из уравнения теплового баланса следует

$$V_{вент} = 3600 \cdot 805 / (1000 \cdot 1{,}2 \cdot (28 - 18)) = 289{,}8 \text{ м}^3 = 290 \text{ м}^3$$

Необходимо тщательно продумать месторасположение кондиционера в офисе. Можно установить канальный кондиционер за подвесным потолком и направить потоки воздуха в разные точки комнаты через воздуховоды. Эти изменения обеспечат равномерную циркуляцию воздуха и позволят

урегулировать температуру. При этом, за счет объединения с системой вентиляции, снижается сумма затрат на вентиляционное оборудование. Блоки могут располагаться на расстоянии до 20 м друг от друга.

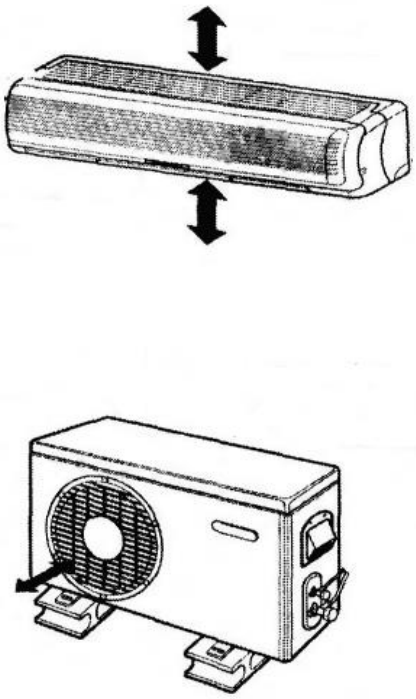

Рисунок 5.2 - Схема кондиционера сплит-системы

В нашем случае высота потолков не позволяет установить канальный кондиционер, поэтому можно выбрать установку сплит системы, показанная на рисунке 5.2. Конструкция кондиционера сплит-системы включает в себя наличие двух агрегатов - наружного и внутреннего блока. Как следует из названия, внутренний блок должен быть установлен внутри офиса, в то время как наружный блок будет располагаться снаружи здания. Кондиционирующую системы следует установить в соответствии с рисунком 5.3. Достоинством сплит-системы является низкий уровень шума, небольшие размеры и разнообразие функций.

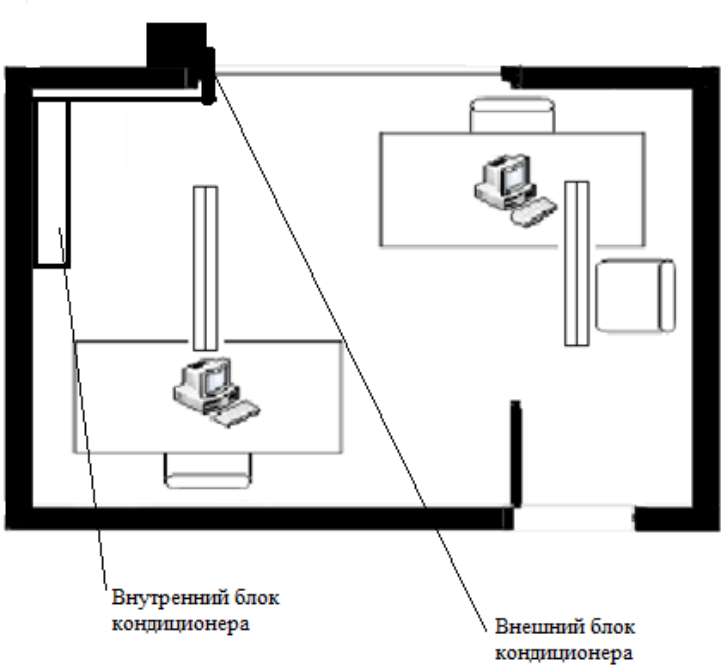

Рисунок 5.3 – Расположение кондиционера в помещении

5.3 Расчет системы автоматического пожаротушения

Автоматическое пожаротушение подразумевает под собой наличие средств оповещения о пожаре, устройств обнаружения пожара, звуковой сигнализации и средств пожаротушения. Комплекс названных средств, установленных специалистами–пожарниками, позволяет считать данную систему пожаротушения автоматической. В такой система роль человека сводится к минимальным задачам: проведению регулярных диагностических работ и замене либо ремонту средств пожаротушения по истечению сроков годности. Риск воспламенения исходит только от попадания постороннего напряжения на платы, или же прохождение токов высоких номиналов по проводникам, что вызовет возгорание изоляции. Учитывая тот факт, что оборудование является весьма дорогостоящим и уязвимым к прямому попаданию воды, то автоматическая система пожаротушения должна применять соответствующие безвредные огнетушащие вещества и иметь высокий уровень надежности.

Внедрение системы автоматического пожаротушения (САП) основано на использовании установки газового пожаротушения, что значительно снизит материальные затраты от поломки аппаратуры. САП газового пожаротушения использует следующие вещества:

- двуокись углерода;
- хладон 114В(2) / тетрофтордибромэтан;
- хладон 13В(1) / бромтрифторметан;
- комбинированный углекислотно–хладоновый состав;
- азот;

– аргон.

САП должна отвечать следующим требованиям, рисунок 5.4:

– дистанционное и локальное включение;

– иметь функции звукового оповещения пожарной опасности, сигнализация;

– соответствовать нормам и требованиям помещения, где будет установлена.

Тип САП и противопламенные вещества выбираются с учётом пожарной опасности и физико–химических свойств, а также в зависимости от классификации помещения по СНиП (Строительные нормы и правила).

В данном случае необходимо выбрать соответствующую САП, подходящую для применения в условиях наличия установок, находящихся под напряжением и в закрытом помещении.

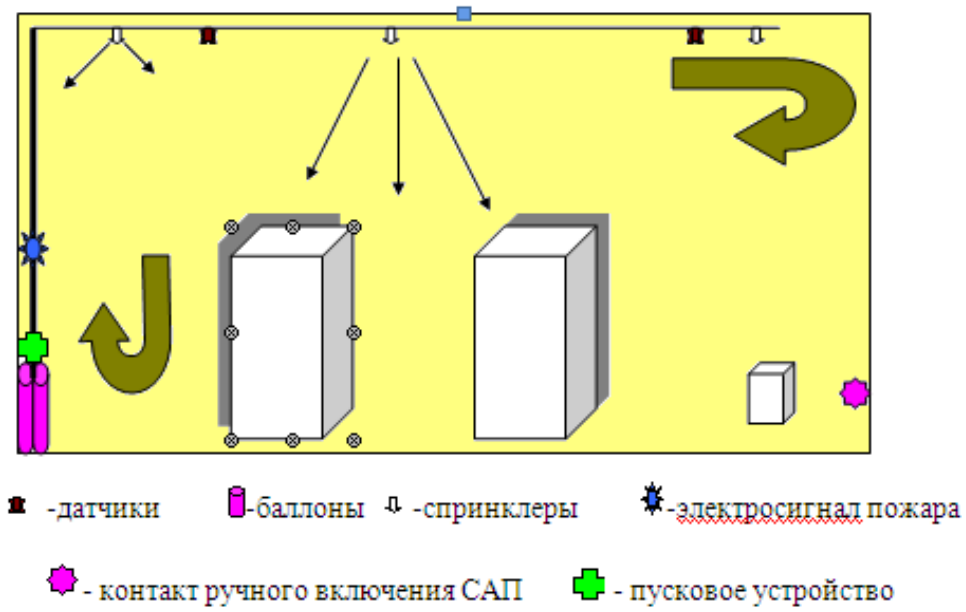

Рисунок 5.4 – Схема системы автоматического пожаротушения

Расчетная масса комбинированного углекислотно–хладонового состава md кг, для объемного пожаротушения определяется по формуле

$$md = k_6 \cdot gn \cdot V \qquad (5.8)$$

где k_6 – коэффициент компенсации не учитываемых потерь состава, k_6=1,23;

gn – нормативная массовая концентрация состава – 0,27 кг/м,

V – объём защищаемого помещения ($м^3$).

$$md = 1{,}23 \cdot 0{,}27 \cdot 60 = 20кг.$$

Расчетное число баллонов E_2 определяется из расчета доступного места в 40– литровой форме 25 кг углекислотно–хладонового состава. Количество форм –1(25кг). Внутренний диаметр магистрального трубопровода di, мм, определяется по формуле

$$d_i = d1 \cdot E_2 \quad (5.9)$$

где d_i – диаметр сифонной трубки баллона, мм;

E_2– число одновременно разрежаемых баллонов.

$$d_i = 10 \cdot 1 = 10 \text{ мм.}$$

Эквивалентная длина магистрального трубопровода l_2 м, определяется по формуле

$$l_2 = k_7 \cdot l \quad (5.10)$$

где k_7 – коэффициент увеличения длины трубопровода для компенсации

l – длина трубопровода по проекту, l=55 м.

$$l_2 = 1{,}35 \cdot 55 = 74{,}5 \text{ м}$$

Площадь сечения выходного отверстия оросителя определим по формуле, $м^3$:

$$A_3 = S / E_1 \quad (5.11)$$

где S – площадь сечения магистрального трубопровода, $мм^2$;

E_1 – число оросителей;

$$A_3 = 60 / 6 = 10 \text{ мм2}$$

Расход углекислотно–хладонового состава Q кг/с, зависит от соответствующей длины и диаметра трубопровода, Q=0,8. Расчетное время подачи углекислотно–хладонового состава t, мин, опредется по формуле

$$t = md / 60Q \quad (5.12)$$

$$t = 20 / 60 \cdot 0{,}8 = 0{,}26 \text{ мин.}$$

Масса основного запаса углекислотно–хладонового состав m, кг, определяется по формуле

$$m = 1{,}1 \cdot md \cdot (1 + k_8 / k_6) \quad (5.13)$$

где k_8 – коэффициент, учитывающий остаток углекислотно – хладонового состава в баллонах и трубопроводах, k_8=0,25.

$$m = 1,1 \cdot 20 \cdot (1+0,25/1,23) = 26,4 \text{ кг.}$$

Исходя из расчетов, в офисе с аппаратным оборудованием, для соблюдения требуемых норм и установки системы автоматического пожаротушения, необходимо установить 2 баллона с углекислотно–хладоновым составом.

В разделе безопасность жизнедеятельности был проведен анализ условий труда и была произведена оценка рабочего помещения. Также были рассмотрены все факторы, негативно влияющие на деятельность разработчика и были произведены расчеты системы кондиционирования и системы авто пожаротушения.

Заключение

В результате выполнения дипломной работы была спроектирована база знаний для мобильного приложения «Прогнозирование футбольных ставок» под платформу Android.

В ходе проектирования базы знаний были исследованы модели представления знаний и выбрана система поиска информации, которая имеет формат продукционной модели представления знаний. Продукционная модель – модель, основанная на правилах, одна их самых распространенных моделей представления знаний. Основное достоинство такой систем заключается в том, что задачи представления знаний и организация логического вывода являются достаточно простыми. Кроме того, продукционная модель хороша тем, что можно легко вносить изменения и дополнения в систему и она обладает высокой модульностью.

Были изучены современны принципы и требования разработки приложений для мобильных телефонов под платформу Android. В ближайшее время планируется разместить данный проект в интернет магазине приложений «Play Market» и получать прибыль от продажи данной интеллектуальной собственности. Также было установлено, что данная предметная область является распространенной и актуальной в нашей стране, поэтому приложение является социально эффективным и наверняка будет пользоваться популярностью.

В технико-экономической части работы была рассчитана общая стоимость программного продукта и количество финансовых затрат на его разработку. В результате расчетов выяснилось, что разработка приложения является экономически эффективным и целесообразным решением.

В разделе безопасности жизнедеятельности был проведен анализ трудовых условий и оценка рабочего помещения, а так же предоставляется расчет систем пожаротушения и кондиционирования.

В процессе выполнения дипломной работы была достигнута цель разработки и выполнены все поставленные задачи.

Список используемой литературы

1. Гаврилова – Базы знаний интеллектуальных систем: Пособие. 2003. – С 234.

2. Обзорная статья: Язык программирования Пролог

3. Лисп. Русскоязычное общество программистов lisp.ru

4. Ахметова М. -Системы искусственного интеллекта: Конспект лекций. 2013. – 46 с.

5. Armstrong J.S. Forecasting for Marketing // Quantitative Methods in Marketing. London: International Thompson Business Press, 1999. P. 92 – 119.

6. Тихонов Э.Е. Прогнозирование в условиях рынка. Невинномысск, 2006. - 221 с.

7. Jingfei Yang M. Sc. Power System Short-term Load Forecasting: Thesis for Ph.d degree. Germany, Darmstadt, Elektrotechnik und Informationstechnik der Technischen Universitat, 2006. -139 с.

8. Ретабоуил Сильвен. Android NDK. Создание приложений под Android на C/C++ – Москва: ДМК Пресс, 2012. – 296 с.

9. Нильсон Н. - Искусственный интеллект. Методы поиска решений, 1973 – 202с

10. Рассел С. Искусственный интеллект. Современный подход С. Рассел, П. Норвиг. — М.: Вильямс, 2007. — 1410 с.

11. .Заенцев И. В. Нейронные сети: основные модели И. В. Заенцев. — Воронеж: Изд-во Воронежского госуд. ун-та, 1999. — 76 с.

12. Хорошевский В Базы знаний интеллектуальных систем. Учебник. — СПб.: Питер, 2000, - 225 с.

13. Джонс М. Т. Программирование искусственного интеллекта в приложениях М. Тим Джонс; Пер. с англ. Осипов А. И. — М.: ДМК Пресс, 2006. — 312 с.

14. Сатимова Е.Г. Проектирование баз данных. Методические указания к выполнению лабораторных работ (для студентов всех специальностей) – Алматы: АИЭС, 2009. - 37с.

15. Медникс Зигард, Дорнин Лаирд, Мик Блэик, Накамура Масуми Программирование под Android – Питер:Москва, 2013. – 651 с.

16. Android A Programmers Guide - М. ИФРАН, 2011 – 399 с

17. Методические указания к выполнению экономической части дипломных работ для студентов специальности 5В070400 – Вычислительная техника и программное обеспечение. З.Д. Еркешева, Г.Ш. Боканова Алматы: АУЭС, 2013 – 40 с.

18. ГОСТ 12.1.005-88 ОБЩИЕ САНИТАРНО-ГИГИЕНИЧЕСКИЕ ТРЕБОВАНИЯ К ВОЗДУХУ РАБОЧЕЙ ЗОНЫ

19. ГОСТ 30494-96 ПАРАМЕТРЫ МИКРОКЛИМАТА В ПОМЕЩЕНИЯХ

20. habrahabr.ru

21. stavochka.com

Приложение А

Листинг программной части

```
public class Fragment_SIgn extends DialogFragment implements OnClickListener {

    EditText signPassword, signLogin;
    String str_signPassword,str_signLogin;
    Button signIN, registration;
    Cursor cursor = null;

    public Fragment_SIgn() {
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View v =inflater.inflate(R.layout.fragment_sign_in, container, false);
        signLogin = (EditText) v.findViewById(R.id.login);


        signPassword = (EditText) v.findViewById(R.id.password);


        signIN = (Button) v.findViewById(R.id.SignIn);
        v.findViewById(R.id.SignIn).setOnClickListener(this);

        registration = (Button) v.findViewById(R.id.Registration);
        v.findViewById(R.id.Registration).setOnClickListener(this);

        return v;
    }

    @Override
    public void onClick(View v) {
        if(v.getId() == signIN.getId()){
            DbHelper mDbHelper = new DbHelper(getActivity(),"myDataBase.db",null,1);
            SQLiteDatabase sdb = mDbHelper.getWritableDatabase();
            str_signLogin = signLogin.getText().toString();
            str_signPassword = signPassword.getText().toString();

            String selection ="Login LIKE ? AND Password LIKE ?";
            String[] selectionArgs = new String[]{str_signLogin,str_signPassword};

            cursor = sdb.query("User", null, selection, selectionArgs, null, null, null);

            if(cursor != null && cursor.getCount()>0){
                cursor.moveToFirst();
//                String user_login =
cursor.getString(cursor.getColumnIndex(DbHelper.User_Login));
```

```
//              String user_password =
cursor.getString(cursor.getColumnIndex(DbHelper.User_Password));
//              Log.d("###########", "User_login =" + user_login + "User_password =" +
user_password);
                MainActivity.USER_LOGIN = str_signLogin;
                MainActivity.USER_PASSWORD = str_signPassword;
                Log.d("****Cursor****",""+cursor.getCount());
                Log.d("****USER_LOGIN",MainActivity.USER_LOGIN);
                Log.d("USER_PASSWORD",MainActivity.USER_PASSWORD);
                cursor.close();
                sdb.close();
                dismiss();

            }else{
                Log.d("****Cursor****",""+cursor.getCount());
                cursor.close();
                sdb.close();
                Toast.makeText(getActivity(),"Неверныйй Login или
Password",Toast.LENGTH_SHORT);}
        }else{
            DialogFragment dlg_reg = new Fragment_Registration();
            dlg_reg.show(getFragmentManager(),"dlg_reg");
            Log.d("onclick srabotal", "Dialog: OnCLick");
            dismiss();}
        }

}
package com.example.batman.toolbarnavigationversion100500;

import android.os.Bundle;

import android.support.v4.app.Fragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;

/**
 * A simple {@link Fragment} subclass.
 */
public class FragmentAbout extends Fragment {

TextView text_about;

    public FragmentAbout() {
        // Required empty public constructor
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
```

```
                        Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment_about, container, false);
        Log.d("CONSTANTAAAAAAAAAH",MainActivity.USER_LOGIN);
        text_about = (TextView) v.findViewById(R.id.about_text);
        return v;
    }

}
package com.example.batman.toolbarnavigationversion100500;

import android.content.ContentValues;
import android.database.sqlite.SQLiteDatabase;
import android.os.AsyncTask;
import android.os.Bundle;
import android.support.v4.app.DialogFragment;
import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentTransaction;
import android.support.v4.app.ListFragment;
import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;
import android.widget.Toast;

import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.Map;
import java.util.concurrent.ExecutionException;

public class FragmentBets extends Fragment {
   // String[] sports = {"Footbal","Tennis","Basketbal","Hokey","Voleyball"};
//    ArrayAdapter<String> adapter;
//
//
//    Elements content,cont,con;
//    Document document,docum,doc;
//
//
//    ArrayList<String> leagues_matches_value = new ArrayList<String>();
//
//
```

```
//    ArrayList<String> footbal_leagues = new ArrayList<String>();
//    ArrayList<String> footbal_matches = new ArrayList<String>();
//    ArrayList<String> footbal_cofficient = new ArrayList<String>();
//    ArrayList<String> footbal_promejuto4nii = new ArrayList<String>();
//    Map<String,ArrayList<String>> league_mathces = new
HashMap<String,ArrayList<String>>();
//    Map<String,ArrayList<String>> matches_cofficient = new
HashMap<String,ArrayList<String>>();
//
    private ListOfMatches fragment_list_of_matches;
    private FragmentTransaction fragmentTransaction;

ListView list_bets;
HashMap<String,ArrayList<String>> hasj = new HashMap<>();

    public FragmentBets() {
    }
    @Override
    public void onActivityCreated(Bundle savedInstanceState) {
        super.onActivityCreated(savedInstanceState);

//        adapter = new
ArrayAdapter<String>(getActivity(),android.R.layout.simple_list_item_1,footbal_leagues)
;
//        if(footbal_leagues.isEmpty())
//            new JsoupBets().execute();
    }

    @Override
    public View onCreateView(LayoutInflater inflater, ViewGroup container,
                             Bundle savedInstanceState) {
        View v = inflater.inflate(R.layout.fragment_bets, container, false);
        final ParseTitle_bets parseTitle_bets = new ParseTitle_bets();

parseTitle_bets.execute("http://olimp.kz/mobile/index.php?page=line&action=1&time=0&
line_nums=0&sel[]=1");
        list_bets = (ListView) v.findViewById(R.id.list_bets);
        try {
            final HashMap<String,String> hashMap_bets = parseTitle_bets.get();
            final ArrayList<String> arrayList_bets = new ArrayList<>();

            for(Map.Entry entry : hashMap_bets.entrySet()){
                arrayList_bets.add(entry.getKey().toString());
            }
            ArrayAdapter<String> adapter = new
ArrayAdapter<String>(getActivity(),android.R.layout.simple_list_item_1,arrayList_bets);
            list_bets.setAdapter(adapter);
            list_bets.setOnItemClickListener(new AdapterView.OnItemClickListener() {
                @Override
                public void onItemClick(AdapterView<?> parent, View view, int position, long
id) {
```

```
//                ParseText_bets parseText_bets = new ParseText_bets();
//                parseText_bets.execute(hashMap_bets.get(arrayList_bets.get(position)));
                if ( MainActivity.USER_LOGIN == "" ) {
                    DialogFragment frg_signin = new Fragment_SIgn();
                    frg_signin.show(getFragmentManager(),"frg_signin");
                }
                ParseText_bets parseText_bets = new ParseText_bets();
                parseText_bets.execute(hashMap_bets.get(arrayList_bets.get(position)));
                ArrayList<String> keyseti = new ArrayList<>();
                try {
                    hasj = parseText_bets.get();
                    for(String key:hasj.keySet()){
                        keyseti.add(key);
                    }
                    fragment_list_of_matches = new ListOfMatches();
                    Bundle args22 = new Bundle();
                    args22.putStringArrayList("mat4i",keyseti);
                    fragment_list_of_matches.setArguments(args22);
                    fragmentTransaction = getFragmentManager().beginTransaction();
                    fragmentTransaction.replace(R.id.fragment_holder,
fragment_list_of_matches);
                    fragmentTransaction.addToBackStack(null);
                    fragmentTransaction.commit();

//
//                  for (Map.Entry entry: hasj.entrySet()) {
//                          String key = (String) entry.getKey();
//                          Log.d("Mat4", key);
//                          ArrayList<String> value = (ArrayList<String>) entry.getValue();
//                      for(String s:value){
//                          Log.d("kefi",s);
//                }
//              }
                } catch (InterruptedException e) {
                    e.printStackTrace();
                } catch (ExecutionException e) {
                    e.printStackTrace();
                }

            }
        });
    } catch (InterruptedException e) {
        e.printStackTrace();
    } catch (ExecutionException e) {
        e.printStackTrace();
    }
    return v;
}

class ParseTitle_bets extends AsyncTask<String,Void,HashMap<String,String>>{
```

```
        @Override
        protected HashMap<String, String> doInBackground(String... params) {
            HashMap<String, String> hashMap_bets = new HashMap<>();
            try {
                Document document = Jsoup.connect(params[0]).get();
                Elements elements = document.select(".row");
                for(Element element: elements){
                    Element element1 = element.select("a[href]").first();
                    hashMap_bets.put(element.text(),element1.attr("abs:href"));
                }
            } catch (IOException e) {
                e.printStackTrace();
            }
            return hashMap_bets;
        }

    }
    class ParseText_bets extends
AsyncTask<String,Void,HashMap<String,ArrayList<String>>>{

        @Override
        protected HashMap<String,ArrayList<String>> doInBackground(String... params) {
            String str = " ";
            HashMap<String,ArrayList<String>> hashMap = new HashMap<>();
            ArrayList<String> array = new ArrayList<>();
            array.clear();
            int i;
            try {
                Document document = Jsoup.connect(params[0]).get();
                Elements elements = document.select(".row");
                for(Element element:elements){
                    Element element11 = element.select("a[href]").first();
                    Document docum1 = Jsoup.connect(element11.attr("abs:href")).get();
                    Elements elem1 = docum1.select(".coefficient");
                    i =0;
                    for(Element kef: elem1){
                        array.add(kef.text());
                        i++;
                        if(i==3) break;
                    }
                    hashMap.put(element.text(),new ArrayList<String>(array));
                    array.clear();
                }
            } catch (IOException e) {
                e.printStackTrace();
            }

            DbHelper mDbHelper = new DbHelper(getActivity(),"myDataBase.db",null,1);
            SQLiteDatabase sdb = mDbHelper.getWritableDatabase();
            sdb.delete("Bets", null, null);
            ContentValues cv = new ContentValues();
```

```
            for(String key :hashMap.keySet()){
                cv.put(DbHelper.Bets_match,key);
                  ArrayList<String> values = hashMap.get(key);
                  for(String val:values){Log.d("#######",val);}
                     cv.put(DbHelper.Bets_P1, values.get(0));
                     cv.put(DbHelper.Bets_X, values.get(1));
                  if(values.size() == 3) {
                     cv.put(DbHelper.Bets_P2, values.get(2));
                  }
                  sdb.insert("Bets", null, cv);
               }
               sdb.close();
            return hashMap;
         }
      }

//    public class JsoupBets extends AsyncTask<Void,Void,ArrayList<String>>{
//
//       @Override
//       protected ArrayList<String> doInBackground(Void... params) {
//          try {
//             footbal_leagues.clear();
//             footbal_matches.clear();
//             footbal_cofficient.clear();
//             footbal_promejuto4nii.clear();
//             leagues_matches_value.clear();
//             int i =0;
//             int b=0;
//             int c=0;
//             String url;
//             String url2;
//             document =
Jsoup.connect("http://olimp.kz/mobile/index.php?page=line&action=1&time=0&line_num
s=0&sel[]=1").get();
//             content = document.select(".row");
//             for(Element element:content) {
//
//
//                String leagues_matches_key = element.text();
//
//
//                footbal_leagues.add(element.text());
//                url = "http://olimp.kz/mobile/"+element.attr("href");
//                docum = Jsoup.connect(url).get();
//                cont = docum.select(".row");
//                for(Element element1:cont){
//                   leagues_matches_value.add(element1.text());
//
//                  // footbal_promejuto4nii.add(element1.text());
//                   String matches_cofficient_key = element1.text();
//
```

```
//                    //Log.d("footbal_matches", element1.text());
//                    url2 = "http://olimp.kz/mobile/"+element1.attr("href");
//                    doc = Jsoup.connect(url2).get();
//                    con = doc.select(".coefficient");
//                    b=0;
//                    for(Element element2:con){
//                       footbal_cofficient.add(element2.text());
//                       b++;
//                       if(b==3) break;
////                       Log.d("footbal_cofficient",element2.text());
////                       Log.d("provero4_kakefov",""+b);
//                    }
//
//                    matches_cofficient.put(matches_cofficient_key,new
ArrayList<String>(footbal_cofficient));
//                    footbal_cofficient.clear();
//                    matches_cofficient_key = "";
//                 }
//
//                 league_mathces.put(leagues_matches_key, new
ArrayList<String>(leagues_matches_value));
//    //String league = league_mathces.getKey(leagues_matches_key);
//                  // ArrayList<String> prova = league_mathces.get(leagues_matches_key);
//                   leagues_matches_value.clear();
//                   leagues_matches_key = "";
//    // Log.d("@@@@@@@@",url);
//                  // Log.d("footbal_leagues",element.text());
//                   i++;
//                 if(i==8)break;
//              }
////               for (Map.Entry entry: league_mathces.entrySet()) {
////                  String key = (String) entry.getKey();
////                  Log.d("ЛИГАААААААААААА", key);
////                  ArrayList<String> value = (ArrayList<String>) entry.getValue();
////                  for(String s:value){
////                     Log.d("МАТЧИИИИИИИИИИИ",s);
////                  }
////               }
//
//              DbHelper mDbHelper = new DbHelper(getActivity(),"myDataBase.db",null,1);
//              SQLiteDatabase sdb = mDbHelper.getWritableDatabase();
//              sdb.delete("Bets", null, null);
//
//              ContentValues cv = new ContentValues();
//              for(String key :matches_cofficient.keySet()){
//                 cv.put(DbHelper.Bets_match,key);
//                 ArrayList<String> values = matches_cofficient.get(key);
//                 for(String val:values){Log.d("#######",val);}
//
//                    cv.put(DbHelper.Bets_P1, values.get(0));
//                    cv.put(DbHelper.Bets_X, values.get(1));
```

```
class ParseTitle_experts extends AsyncTask<Void,Void,HashMap<String,String>>{

    @Override
    protected HashMap<String, String> doInBackground(Void... params) {
        HashMap<String,String> hashMap = new HashMap<>();
        try {
            Document document = Jsoup.connect("http://stavochka.com/forecasts/").get();
            Elements elements = document.select(".b-forecasts__th a");
            for(Element element :elements){
                Element element1 = element.select("a[href]").first();
                hashMap.put(element.text(),element1.attr("abs:href"));
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return hashMap;
    }
}

    class ParseText_experts extends AsyncTask<String,Void,String>{

        @Override
        protected String doInBackground(String... params) {
            String str = " ";
            try {
                Document document = Jsoup.connect(params[0]).get();
                Element elements = document.select(".b-forecast-text").first();
                str = elements.text();
            } catch (IOException e) {
                e.printStackTrace();
            }

            return str;
        }
    }

}
package com.example.batman.toolbarnavigationversion100500;

import android.os.AsyncTask;
import android.os.Bundle;

import android.support.v4.app.Fragment;
import android.support.v4.app.FragmentTransaction;
import android.support.v4.app.ListFragment;

import android.util.Log;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.AdapterView;
```

```
import android.widget.ArrayAdapter;
import android.widget.ListView;
import android.widget.TextView;

import org.jsoup.Jsoup;
import org.jsoup.nodes.Document;
import org.jsoup.nodes.Element;
import org.jsoup.select.Elements;

import java.io.IOException;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;
import java.util.concurrent.ExecutionException;


      View v =inflater.inflate(R.layout.fragment_news, container, false);

      listik = (ListView) v.findViewById(R.id.listik);

      ParseTitle parseTitle = new ParseTitle();
      parseTitle.execute();
      try {
         final HashMap<String,String> hashMap = parseTitle.get();
         final ArrayList<String> arraylist = new ArrayList<>();
         for(Map.Entry entry : hashMap.entrySet()){
            arraylist.add(entry.getKey().toString());
         }
         ArrayAdapter<String> adapter = new
ArrayAdapter<String>(getActivity(),android.R.layout.simple_list_item_1,arraylist);
         listik.setAdapter(adapter);
         listik.setOnItemClickListener(new AdapterView.OnItemClickListener() {
            @Override
            public void onItemClick(AdapterView<?> parent, View view, int position, long
id) {
               ParseText parseText = new ParseText();
               parseText.execute(hashMap.get(arraylist.get(position)));
               fragment_news_item = new ListItemOfNews();
               Bundle args11 = new Bundle();
               try {
                  args11.putString("item_of_news",parseText.get());

               } catch (InterruptedException e) {
                  e.printStackTrace();
               } catch (ExecutionException e) {
                  e.printStackTrace();
               }
               fragment_news_item.setArguments(args11);

      fragmentTransaction = getFragmentManager().beginTransaction();
```

```
    fragmentTransaction.replace(R.id.fragment_holder, fragment_news_item);
    fragmentTransaction.addToBackStack(null);
    fragmentTransaction.commit();
            }
        });
    } catch (InterruptedException e) {
        e.printStackTrace();
    } catch (ExecutionException e) {
        e.printStackTrace();
    }
    return v;
}

class ParseTitle extends AsyncTask<Void,Void,HashMap<String,String>>{

    @Override
    protected HashMap<String, String> doInBackground(Void... params) {
        HashMap<String,String> hashMap = new HashMap<>();
        try {
            Document document = Jsoup.connect("http://www.sports.ru/topnews/").get();
            Elements elements = document.select(".short-text");
            for(Element element :elements){
                Element element1 = element.select("a[href]").first();
                hashMap.put(element.text(),element1.attr("abs:href"));
                Log.d("&&&&&&&&&&&",element1.attr("abs:href"));
            }
        } catch (IOException e) {
            e.printStackTrace();
        }
        return hashMap;
    }
}

class ParseText extends AsyncTask<String,Void,String>{

    @Override
    protected String doInBackground(String... params) {
        String str = " ";
        try {
            Document document = Jsoup.connect(params[0]).get();
            Element elements = document.select(".news-item__content").first();
            str = elements.text();
        } catch (IOException e) {
            e.printStackTrace();
        }

        return str;
```