В соответствии с моделью конкуренции, приведенной в 1.1.1, на основании модели М. Портера с добавлением в качестве 6 силы комплементоров дано формальное представление конкуренции в сегменте ОАТ как сложной системы (СС) в виде концептуальной модели (рис. 14), определены подсистемы и элементы сложной системы мирового рынка вооружений для истребителей 5-го поколения.

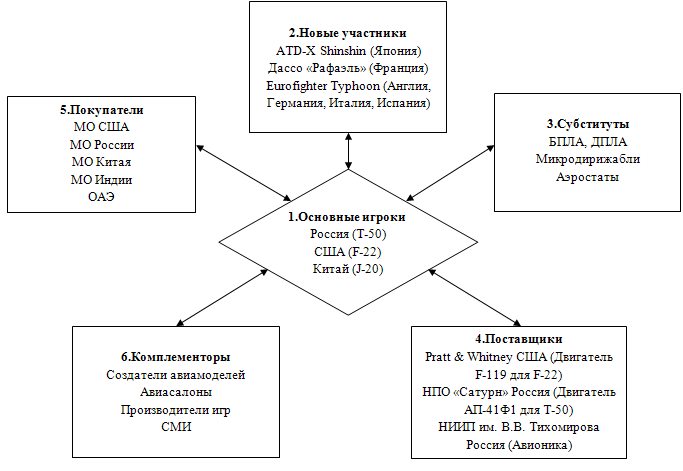


Рис. 14. Модель конкуренции в сегменте истребителей 5-го поколения

(на основе расширенной модели М. Портера)

Анализ тенденций развития мировой авиапромышленной индустрии показывает, что основные производители всё чаще стремятся к созданию альянсов. В этом случае целесообразно применить теорию компромиссных игр (Stirling W.C., 2003) к описанию таких ситуаций. На рынках высокотехнологичной продукции складывается обычно более сложная конкурентная ситуация, которую описать полностью в терминах агентов с индивидуальными целями и предпочтениями не удается. Поэтому необходимо строить схемы прогнозирования и разрешения конфликтов с индивидуальными, а также – с координированными целями. Теория компромиссных игр (Satisficing Game Theory – SGT) дает возможность построить такую теоретическую модель, в рамках которой игроки могут координировать свои действия друг с другом. Главное отличие SGT от CGT заключается в том, что рыночные агенты формируют собственные предпочтения, принимая во внимание предпочтения других участников рынка. Такое поведение называют ситуационным альтруизмом (situational altruism) – формой неэгоистичного поведения, при которой одни агенты жертвуют своими предпочтениями, чтобы другие смогли извлечь пользу от таких действий. Подход SGT дает возможность создания альянсов и коалиций, в которых каждому участнику уделяется достаточное внимание.

Для каждого агента на рынке строятся две функции:

1. функция преимуществ (ФП), описывающая достижение фундаментальной цели, например, завоевание лидерства на рынке, увеличение прибыли, без учета цены её достижения и не учитывающая риски;
2. функция риска (ФР), минимизирующая издержки при достижении фундаментальной цели.

На каждом шаге при принятии решений агентом вычисляется, будет ли выгоднее стремиться к достижению намеченной цели без сотрудничества с другими агентами или вступить с кем-то из них в содружество. В этом случае агент сможет получить лучшую технологию, используемую в производстве партнером, или разделить риски и тем самым снизить затраты на производство и продвижением товаров/услуг. Также, обмениваясь информацией, агенты смогут принимать решения, используя услуги одного поставщика, представляющего лучшие компоненты для производства продукта/услуги; а также совместно принимать меры против выхода на рынок товаров-заменителей (субститутов) и/или новых участников.

Каждый агент обладает упорядоченным по приоритету списком возможных действий. Возможное действие зависит от действий, которые предпринимают другие участники. В системе из N агентов i-ый агент имеет функцию полезности , где – действие, предпринимаемое k-ым агентом. ФП отражает эффективность управляющего действия (управления) агента для скорейшего достижения поставленной цели (без учета затраченных ресурсов). ФР агента отражает величину израсходованных ресурсов (затраты на производство, НИОКР, продвижение, человеческие ресурсы, потребление энергии, временные траты и т.п.). В многоагентной системе ФП и ФР являются функциями многих переменных.

Обозначим ФП через , ФР – через , а через – совместную функцию, которая учитывает атрибуты, влияющие на значение как ФП, так и ФР управляющего действия. Для описания n-агентной системы введем совместную функцию 2n переменных:

(1)

где соответствуют набору ролей, в которых функцией полезности выступает ФП, а соответствуют набору ролей, в которых функцией полезности выступает ФР.

Назовем её функцией взаимозависимости (ФВ), где переменные соответствуют управлениям, которые доступны i-ому агенту для достижения поставленной цели. Переменные , где i=1,...,N показывают управляющие воздействия (управления), которые доступны i-ому агенту для уменьшения издержек. Используя ФВ, можно учесть существующие отношения между агентами, выступающими в разных ролях. Следовательно ФВ описывает все варианты, которые могут повлиять на поведение участников многоагентной системы. При построении ФВ многоагентной системы (или ее части) можно руководствоваться влиянием ролей разных агентов друг на друга.

Для этого строим направленный ациклический граф, отображающий взаимные зависимости и влияния агентов друг на друга, при этом на графе изображаются как роли, преследующие достижение максимальной прибыли или лидерства на рынке, так и роли, стремящиеся к уменьшению издержек или снижению рисков. Т.к. высокотехнологичные рынки, как правило, являются олигополистичными, когда на рынке взаимодействует небольшое число агентов, то результаты анализа всех допустимых ролей (рис. 18) можно представить в виде графа.

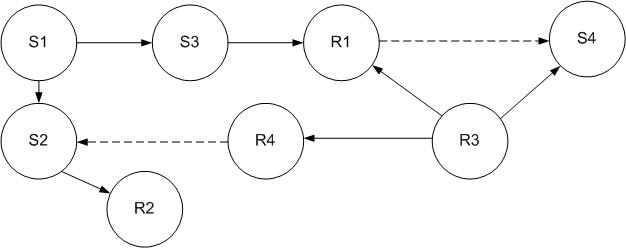


Рис. 18. Граф взаимного влияния четырех агентов

Наличие условных функций взаимозависимости позволяет существовать некоторой форме альтруизма в многоагентной среде. В противоположность категорическому альтруизму (categorical altruism), когда агент отказывается от своих предпочтений, чтобы другие агенты получили выигрыш, условные функции взаимозависимости вносят концепцию ситуационного альтруизма (situational altruism), благодаря которой агент может изменить свои предпочтения в зависимости от предпочтения другого, но уже не в одностороннем порядке.

Теория компромиссных игр предоставляет математически строгий метод описания упомянутого компромиссного поведения.

Определение 1.Рассмотрим систему из N агентов, и обозначим через множество управлений, доступных для выбора i-ому агенту, i=1,...,N. Компромиссной игрой будем называть тройку , где - функция взаимозависимостей, определяющая роли, доступные агентам, соответствуют набору ролей, в которых функцией полезности выступает ФП, а соответствуют набору ролей, в которых функцией полезности выступает ФР.

Для решения компромиссной игры необходимо рассчитать совместные ФП и ФР, руководствуясь следующими выражениями:

(2)

(3)

Соответственно для индивидуальных предельных значений ФП и ФР i-го агента используются выражения:

(4)

(5)

Определение 2. Решение компромиссной игры уровня q – подмножество всех векторов опций, удовлетворяющих условию того, что значение совместной ФП больше либо равно произведению значения совместной ФР на скалярный коэффициент q:

(6)

Показатель q отражает относительный вес между достижением цели и минимизацией издержек или рисков.

Индивидуальное компромиссное (достаточное) решение для каждого агента получается расчетом предельных значений ФП и ФР:

(7)

При формировании стратегии высокотехнологичной компании (агента) q рассматривается как параметр: его уменьшение увеличивает размеры множеств индивидуальных компромиссных решений, позволяя агенту «смягчить» правила для достижения компромисса. Значение q=1 наделяет одинаковыми весами роли достижения цели и минимизации издержек.

На основании сделанных предположений разработан алгоритм принятия решений по позиционированию агента (компании) на высокотехнологичном рынке в соответствии с теорией SGT:

Шаг 1: на первом шаге применения SGT описываются взаимодействия и взаимные влияния между агентами рынка. Причем учитываются влияния как основных конкурентов **A1** друг на друга, так и новых участников рынка **А2**, субститутов **А3**, поставщиков **А4**, покупателей **А5** и комплементоров (удовлетворенных покупателей) **A6**. Число агентов, с которыми возможно взаимодействие основного участника рынка, ограничивается выбранной им стратегией, политическим, экономическим и другими аспектами и задается заказчиком. Так как ситуация на рынке меняется, необходимо строить модель системы в виде динамического графа.

Шаг 2: т.к. при принятии решения на каждом шаге агенту нет необходимости анализировать взаимные влияния со всеми участниками рынка, происходит ранжирование всех агентов рынка по заданному критерию. Этот критерий определяется в зависимости от конкурентной стратегии компании заказчика конкурентного анализа.

Шаг 3: когда все участники рынка проранжированы по приоритету, становится возможным построение динамического графа, характеризующего взаимозависимости участников рынка. Далее выделяются те участники рынка, с которыми необходимо разрешать конфликтные ситуации. Это могут быть как основные конкуренты **А1**, так и представители других групп агентов **А2, .., А6**. На основании полученной из конкурентного анализа информации рассчитывается ФР и ФП каждого возможного действия (управления) для каждого агента в отношении других участников рынка . И выбираются действия (управления), для которых разница между значениями функций преимуществ и недостатков максимальна:

(8)

Другими словами, каждый агент всегда находится в поиске максимальных преимуществ, которые можно получить с наименьшим риском.

Шаг 4: после того как ФП и ФР для i-го агента определены, выбирается наиболее предпочтительное из возможных управлений. При этом если агент допускает существование риска, то он максимизирует свою ФП. Нерасположенный к риску агент, напротив, минимизирует свою ФР, хотя это и не гарантирует прогресса в продвижении к намеченной цели. Такой подход будет гарантировать наиболее быстрое продвижение к цели вместе с минимальным риском, который при этом возникает. Следовательно, при использовании теории компромиссных игр появляется возможность для участников рынка (агентов) выбирать решения, близкие к оптимальным.

**1.3.2. Разработка математической модели конкуренции в авиастроении**

Здесь приведена разработка теоретико-игровой модели взаимодействия элементов СС, позволяющей ЛПР от производителя ЛА принимать стратегические решения (рис. 19).

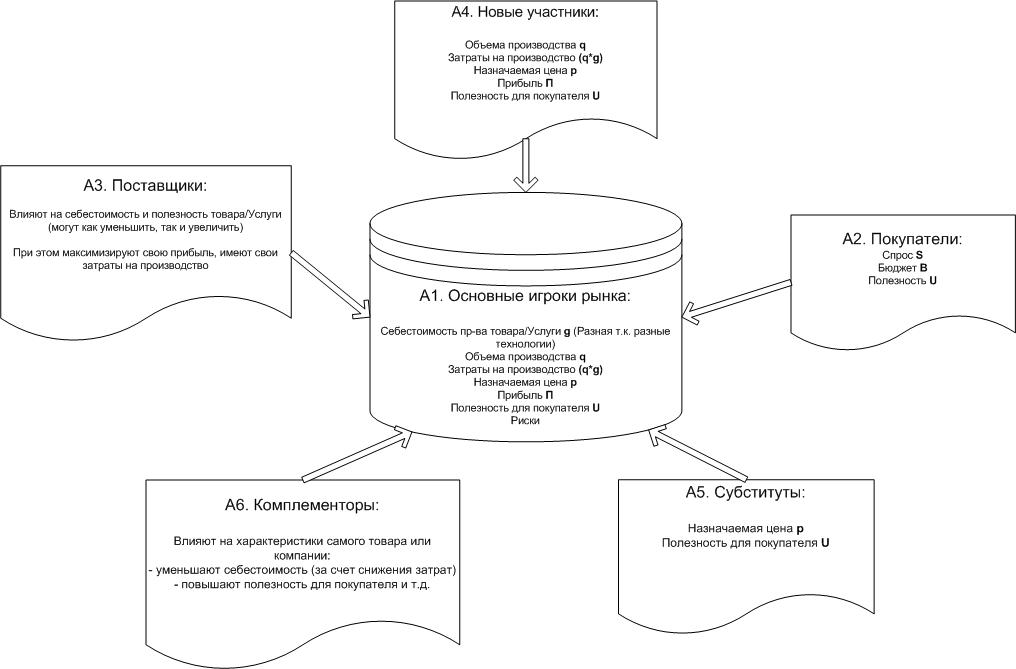


Рис. 19. Схема анализа конкуренции основных игроков отраслевого рынка

Задаются общие параметры игры: определяются функции спроса S(Q,p) и предложения P на рынке, прибыль П основных участников рынка в зависимости от объемов производства Q и затрат G. Затем исследуются состояния равновесия модели при стремлении достижения игроками своих целей (например, max прибыли или min издержек) при учете действий других игроков (однако без взаимного сотрудничества). Ставится задача: найти вектор X\* значений параметров KPI[[1]](#footnote-1) для построения оптимальной стратегии компании.

На рынке высокотехнологичной продукции каждая из N компаний-производителей имеет собственную удельную (на единицу производимого блага) себестоимость товара G\_i (i=1..n), т.к. компании используют разные технологии, разработанные ранее в результате инновационных циклов НИОКР, а также имеют собственных поставщиков комплектующих. При этом для поставщиков компонентов, новых участников рынка и т.д. строится аналогичная модель, данные из которой поступают в базовую модель.

Себестоимость складывается из затрат на НИОКР, распределяющихся на каждую единицу продукта, а также из общих затрат на производство

(9)

Объем производства определяется исходя из стратегии компании и потребностей рынка (т.е. покупательского спроса S). Общие затраты составляют .

Рынок характеризуется потребительским спросом S. Предполагается, что обратная функция спроса на производимую конечную продукцию – линейная [[2]](#footnote-2):

(10)

где – суммарный объем выпускаемой продукции или общее предложение на рынке P, – запретительная цена на товар (), – параметр, характеризующий емкость рынка (). Тогда прибыль главного для нас игрока рынка с учетом (1) и (2) определяется как

(11)

(12)

(13)

(14)

где – существующие постоянные затраты i-го участника рынка, не зависящие от объема продукции.

Допустим, что удельные затраты на производство единицы продукта или услуги не зависят от объема выпуска , тогда поиск состояния равновесия можно выполнить по модели Курно[[3]](#footnote-3). В частности, прибыль i-го игрока:

(15)

(16)

Пусть в модели биматричной игры, которая соответствует дуополии на рынке, участвуют производители В1 и В2 и . Тогда

(17)

(18)

Согласно гипотезе Курно, игрок, максимизирующий свою прибыль, не предполагает реакции конкурента, поэтому . Отсюда следует, что оптимальный выпуск производителя В1 при заданном выпуске производителя В2 можно представить следующим выражениями:

(19)

(20)

В этом случае процесс перехода в состояние равновесия по модели дуополии Курно можно рассматривать как бесконечную многошаговую динамическую игру: сначала производитель В2 определяет свой оптимальный выпуск при заданном выпуске производителя В1, на следующем шаге производитель В1 дает свой оптимальный ответ на предыдущий ход конкурента, и т.д. Как показано в ряде работ[28] по теории игр, вне зависимости от начальных условий, выпуски будут стремиться к равновесным значениям. Их можно определить, рассматривая уравнения реакции (19) и (20) как систему алгебраических уравнений, в которой неизвестные выпуски принимают равновесные значения. Таким образом, равновесные выпуски при производстве изделий, производимых B1 и B2, примут следующие значения:

(21)

(22)

Далее рассмотрим метод получения результатов конкурентного анализа. Итоги расчета будут представлены в виде графика прибылей основных игроков, рекомендаций ЛПР и диаграмме распределения долей высокотехнологичного рынка.

График прибылей для основных игроков (MP) строится на основании суммарной прибыли каждого отдельного игрока, которая считается как результат произведения прибыли за единицу товара на объем производства:

(23)

Для определения рекомендаций ЛПР требуется сравнить прибыль, объем производства и взаимодействия всех агентов До и После проведения конкурентного анализа. Список взаимодействий позволяет давать рекомендации по выбору возможных (новых) поставщиков и потребителей. Для этого нам потребуется представить в виде векторов взаимосвязи агентов (L) с начальными и конечными поставщиками (Ldн,Ldн) и потребителями (Lcн,Lcн), после чего вычислить их разницу:

(24)

(25)

Таким образом, мы получим вектора изменений списка поставщиков ( и потребителей (, где компонента 0 будет означать отсутствие изменений, 1 – появление новой связи и (-1) – исчезновение старой связи, что в свою очередь и является рекомендацией по выбору для ЛПР. Назначаемая цена и объем производства предлагаются ЛПР, исходя из установившегося равновесия на рынке данного продукта (изделия).

Рекомендации по выбору стратегии основываются на распределении долей рынка. Стратегия «Лидер» предлагается игроку с наибольшей занятой долей рынка. Выбор стратегии «Бросающий вызов лидеру» упрощен. Будем считать, что такой стратегии придерживаются 25% игроков, чьи доли рынка максимально приближены к доле, занимаемой лидером. Все остальные игроки считаются «следующими за лидером».

Рассмотрим диаграмму распределения долей рынка. Доля рынка для i-го основного участника (Di) вычисляется по формулам:

(26)

(27)

Здесь Pобщ – общее положении игроков на рынке. Оно складывается из суммы частных положений на рынке для i-го игрока, которые считаются как:

(28)

В этой формуле ki – это коэффициент, получаемый для каждого игрока в результате сравнения М одинаковых характеристик (Xi и Xj) товара i-го игрока со всеми его конкурентами (kij). Таким образом:

(29)

(30)

**1.3.3. Разработка модели поведения агентов при реализации**

**конкурентной стратегии в секторе микродирижаблей**

На основании теоретико-игровых модели, построенной в параграфе 1.3.2и фрактальной модели структуры АП, разработаны алгоритмы принятия решений агентами рынка и программно обеспечение для конкурентного анализа в секторе объектов авиационной техники. Алгоритм инициации модели для проведения конкурентного анализа представлен на рис. 20. Рассмотрим его:

1. Задаем число N уровней иерархии модели.
2. Задаем множества агентов: на верхнем уровне иерархии (); далее (), () и т.д. до N (перечисляются все агенты на всех уровнях иерархии до N).
3. Задаем множество {**S}** стратегий агентов в зависимости от целей i-го агента, i=1,…,6 из базовых конкурентных стратегий.
4. Для i-го агента задаем множество действий {**O}** для достижения стратегической цели.
5. Задаем множество взаимных влияний {**I**} агентов друг на друга (для описания технических, экономических, политических и др. аспектов).
6. В процессе выполнения модели т.к. ситуация на рынке может меняться, вносятся изменения в заданные в п. 1-4 множества.



Рис.20. Схема алгоритма инициации модели конкурентного анализа

Алгоритм принятия решения i-ым агентом представлен на рис. 21.

1. С учетом выбранной стратегии и сформированного множества {**O**} действий формируется профиль поведения агента как подмножество мн-ва {**O**} для достижения
2. Для каждого действия в отношении других агентов определяются его характеристики, и происходит ранжирование по заданному критерию, который может учитывать не только заданные характеристики, но и предпочтение агента[[4]](#footnote-4).
3. Определяются участники рынка, с которыми необходимо взаимодействовать для достижения цели.
4. На граф взаимодействий i-ый агент выбирает партнеров, заказчиков, поставщиков и др.
5. При этом кроме ответа других агентов, учитывается потребность агента в получении максимальных преимуществ с наименьшими рисками.
6. Другие агенты дают ответ, основываясь на собственных моделях взаимодействий, с учетом своих предпочтений и достижения своих целей.
7. Процесс продолжается до установления равновесия (каждый агент достиг своей цели и находится ситуации, когда любое следующее действие ухудшит его положение).

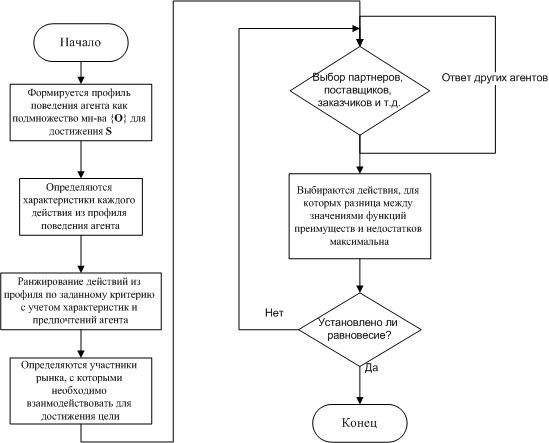


Рис.21. Схема алгоритма принятий решений агентом

**Выводы по теоретической части**

1. Проведен обзор классических и новых методов конкурентного анализа: аналитических, статистических, теоретико-множественных и графических, применяемых к исследованиям высокотехнологичных рынков.
2. Дан анализ основных видов конкуренции: чистой (совершенной), олигополистической и монополистической, в результате которого показано, что для объектов авиационной техники характерно взаимодействие внутри олигополистического рынка.
3. Проведено исследование классической модели «пяти сил» конкуренции М. Портера, показаны достоинства и недостатки данной модели, в результате предложена новая модель, включающая дополнительных участников рынка, какими являются т.н. фирмы или институты - «комплементоры».
4. Дано сравнение основных видов авиационной техники, конкурирующих между собой на трех уровнях иерархии, где на первом уровне рассматриваются истребители 5-го поколения, на втором – беспилотные летательные аппараты, на третьем – микродирижабли и аэростаты.
5. В работе рассмотрены методы и модели теории игр, среди которых представлены принцип минимакса, решение игр в смешанных стратегиях, графический метод, модели Бертрана, Курно и Штакельберга. Учитывая особенности рассматриваемой предметной области, акцент сделан на модели компромиссной игры.
6. Разработана математическая модель конкуренции в авиационной промышленности, как модель олигополистического рынка в секторах истребителей 5-го поколения, беспилотных летательных аппаратов, микродирижаблей и аэростатов.
7. Разработан алгоритм поведения агентов (субъектов рынка авиационной промышленности) при реализации конкурентной стратегии основных игроков данного рынка.

**Практическая часть**

**2.1. Разработка высокоуровневой архитектуры SaaS-приложения**

**2.1.1. Особенности использования модели SaaS**

Модель SaaS (Software as a Service) реализует концепцию исследования программного обеспечения (ПО), при которой поставщик разрабатывает Web-приложение и самостоятельно управляет им, предоставляя заказчикам доступ к ПО через Интернет. В отличие от модели SaaP (Software as a Product) в модели SaaS:

* доступ к ПО предоставляется удаленно по сетевым каналам через web-интерфейс;
* ПО развертывается в едином дата-центре (или дата-центрах), а не на мощностях каждого конкретного заказчика;
* ПО предоставляется на условиях аренды, и за него взимаются периодические платежи (стоимость технической поддержки обычно включается в стоимость арендной платы);
* обновления ПО устанавливаются централизованно на стороне провайдера.

Таким образом, модель SaaS обладает рядом существенных преимуществ как для пользователя ПО, так и для поставщика/разработчика (см. табл. 5).

Преимущества модели SaaS перед SaaP. Таблица 5.

|  |  |
| --- | --- |
| **Для пользователя** | **Для поставщика/разработчика** |
| * Отсутствие затрат, связанных с установкой, обновлением и поддержкой работоспособности программного обеспечения; | * Борьба с пиратством; * Уменьшение затрат на развертывание и внедрение ИС (за счет развертывания и развития на «родной» аппаратной среде); |

Продолжение таблицы 5.

|  |  |
| --- | --- |
| **Для пользователя** | **Для поставщика/разработчика** |
| * Отсутствие аппаратной платформы, связанной с развертыванием ПО; * Отсутствие забот о своевременном обновлении ПО; * Быстрота внедрения; * Понятный интерфейс; * Возможность получить более высокий уровень обслуживания ПО. | * Легкое проникновение на глобальные рынки; * Простота реализации web-приложений; * Привязка заказчика (заказчик не может отказаться от услуг поставщика и продолжать использовать систему); * В долгосрочном периоде доходы от продаж SaaS выше, чем от SaaP |

В то же время SaaS при всех своих преимуществах обладает следующими недостатками:

* клиент привязывается к единственному поставщику или разработчику, что может быть для него (клиента) нежелательно;
* возникают вопросы безопасности и утечки информации (решаются на 3-х уровнях: юридическом, технологическом, информационном);
* требуется постоянное подключение клиента к Интернету;

На российском рынке бизнес-ИТ модель SaaS уже предоставлена разработками компаний SugarCRMInc[[5]](#footnote-5) (*SugarCRM*), NetSuiteInc (*NetSuiteERP*), NetSuiteInc (*NetSuiteCRM*), Веспол (*BomaAnalyticOnline*), Логнекс (*МойСклад*), Альтсофт (*Кларис*), Мегаплан (*Мегаплан*), БОСС. Кадровые системы (БОСС-Кадровик). Таким образом, после изучения особенностей модели SaaS и данных о существующих на рынке решениях планируется разрабатывать Web-приложение, предоставляемое компаниям-пользователям (производителям ОАТ) именно на основе этой модели.

**2.1.2. Разработка архитектуры информационной системы SaaS-приложения КА в авиастроении**

Разрабатываемое SaaS-приложение имеет клиент-серверный (КС) тип архитектуры информационной системы (ИС). Для решения задачи, нами была выбрана трехуровневая клиент-серверная архитектура, включающая в себя следующие уровни: пользовательский интерфейс, сервер приложений и сервер баз данных. Данная модель представляет собой следующее:

* на стороне клиента располагается только пользовательский интерфейс, обеспечивающий взаимодействие и передачу данных на серверную часть;
* сервер приложений содержит в себе бизнес-логику, обрабатывает запросы от клиента, передает их серверу БД, получает ответ и возвращает полученный результат;
* сервер баз данных обеспечивает управление данными.

Реализация данного подхода изображена на рис. 22.

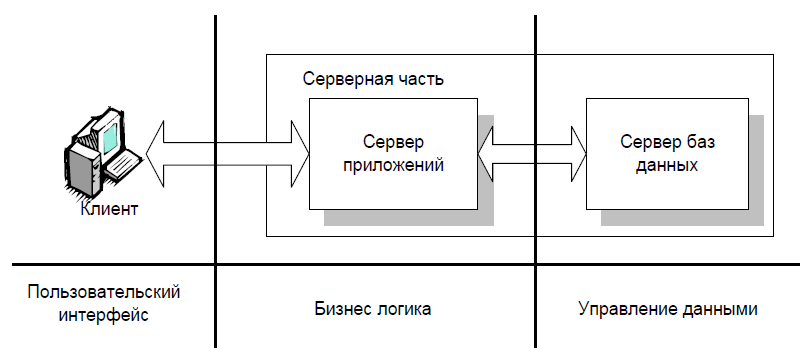


Рис. 22. Трехуровневая клиент-серверная архитектура

Перейдем к рассмотрению плюсов и минусов данной модели. Её плюсами являются:

* при использовании тонкого клиента снижаются требования к ПК клиента; на клиенте расположен только пользовательский интерфейс, доступ к которому пользователь, как правило, получает посредством браузера;
* между клиентской программой и сервером приложений передается лишь минимально необходимый поток данных (аргументы вызываемых функций и возвращаемые ими значения);
* снижение нагрузки на сервер данных и, как следствие, повышение скорости работы системы в целом;
* упрощение наращивания функциональности и обновления ПО.

Её минусами является:

* увеличение расходов на администрирование и обслуживание серверной части.

Разрабатываемое SaaS-приложение конкурентного анализа отвечает всем вышеописываемым характеристикам. Высокоуровневая архитектура[32] web-системы представлена на рис. 23. Перейдем к ее рассмотрению.

Доступ к пользовательскому интерфейсу (клиент) будет осуществляться через окно браузера (WebBrowser:MSInternetExplorer, MozillaFirefox, Opera, GoogleChrome, Safari). На наш взгляд, это оптимальное решение, т.к. оно требует от пользователя минимальных затрат, связанных с установкой и обслуживанием ПО. В первое время приложение планируется как бесплатный сервис, однако в дальнейшем возможно введение ежемесячной абонентской платы за использование. Собственно этим и исчерпываются возможные неудобства клиента. Реализация интерфейса будет организована стандартным образом при помощи XHTML-разметки (англ. ExtensibleHypertextMarkupLanguage — расширяемый язык разметки гипертекста), доступ к которой для скриптов (JavaScript) и программ осуществляется посредством DOM (от англ. DocumentObjectModel – «объектная модель документа»).

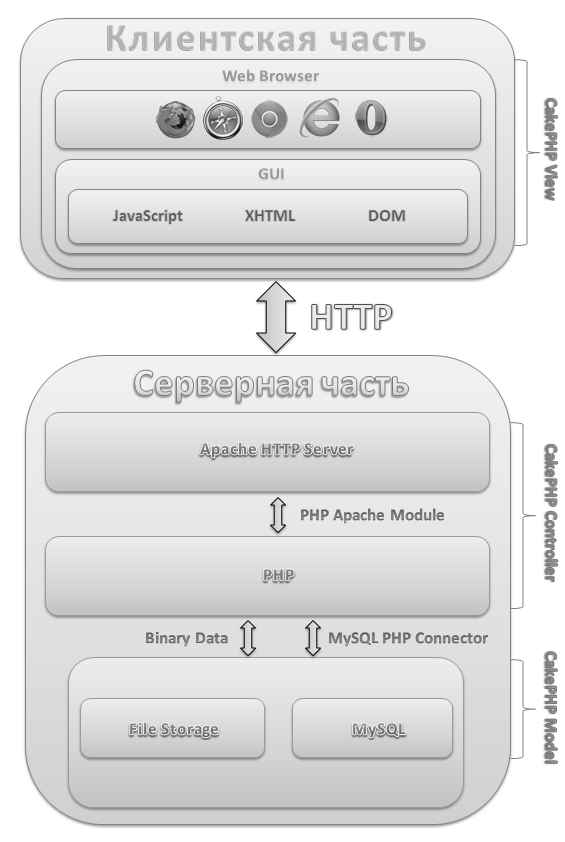


Рис. 23. Высокоуровневая архитектура системы

Передача данных от клиентской части к серверной будет осуществляться при помощи стандартного протокола прикладного уровня передачи данных HTTP (англ. HyperTextTransferPrоtocоl — «протокол передачи гипертекста»).

Серверная часть реализована на основе ApacheHTTPServer, к которому средствами PHPApacheModule подключен скриптовый язык программирования PHP (англ. PHP:HypertextPreprocessor – «PHP: препроцессор гипертекста»). Передача логических данных, обрабатываемых PHP в БД, осуществляется при помощи MySQLPHPConnector. Пользовательский контент передается в файловое хранилище в виде стандартных документов (BinaryData).

В качестве СУБД для сервера БД выбран MySQL. MySQL является решением для малых и средних приложений, что отвечает всем нашим требованиям. Пользовательский контент (файлы с таблицами и диаграммами, изображения, видео и т.д.) будет храниться в файловом хранилище.

Организация бизнес-логики будет отвечать требованиям модели MVC (ModelViewController), которая в свою очередь является проекцией трехуровневой КС-архитектуры. Сам же движок будет запрограммирован средствами PHP (FrameworkCakePHP). Подробнее см. раздел 2.2.3 «Особенности использования фреймворкаCakePHP».

**2.2. Выбор средств разработки программной системы**

**2.2.1. Концепция MVC**

Концепция MVC (Model - View - Controller/Модель - Вид (Представление) - Контроллер) – это проверенная и надежная схема, дающая возможность быстро разработать модульное и расширяемое приложение. Разбивка приложения на отдельные блоки (виды, модели и контроллеры) позволяет вести разработку прямо «на ходу» Новые возможности легко могут быть добавлены, а старые изменены. Модульная структура приложения позволяет разработчику и дизайнеру работать одновременно. Стандартная схема архитектуры «Модель-Вид-Контроллер» изображена на рис. 24:

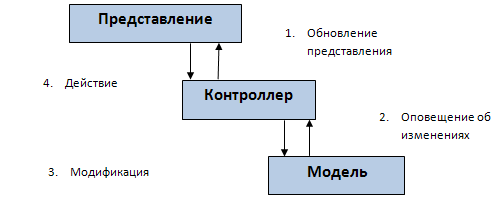


Рис. 24. Схема разработки ПО MVC

Рассмотрим данную схему подробнее. Шаблон MVC содержит в себе три основных компонента:

* *Модель* – отвечает за предоставление данных и методов работы с ними, а также реагирует на запросы, изменяя свое состояние. Модель отображает внутреннее устройство системы.
* *Вид (Представление)* **–** отвечает за отображение информации, поступающей из системы или в систему.
* *Контроллер –* обеспечивает связь между пользователем и системой. Он получает данные от пользователя (через Вид) и передает их в Модель, и наоборот, получает сообщения от Модели и передает их в Вид.

**2.2.2. Выбор СУБД для реализации БД**

Система управления базами данных (СУБД) – совокупность программных и лингвистических средств общего или специального назначения, обеспечивающих управление созданием и использованием баз данных.

Так как разрабатываемое нами SaaS-приложение конкурентного анализа имеет клиент-серверный тип архитектуры информационной системы, то при выборе СУБД мы будем ориентироваться на клиент-серверный способ доступа к БД. Клиент-серверная СУБД может рассматриваться как распределенная система, в которой все запросы создаются на одном узле (клиент), а вся обработка выполняется на другом (сервер), если считать для простоты, что имеется лишь один узел клиента и один узел сервера.

Выделим основные СУБД, отвечающие нашим требованиям:

* Oracle
* PostgreSQL
* MySQL

Мы остановили свой выбор на СУБД MySQL по следующим причинам:

* На подавляющем большинстве хостингов по умолчанию имеется доступ к СУБД MySQL, чего нельзя сказать о PostgreSQL и Oracle (доступ к ним предоставляется, как правило, за отдельную плату).
* MySQL является наиболее распространенной СУБД, вследствие чего образует вокруг себя активное сообщество и развитую службу поддержки.
* До начала разработки SaaS-приложения конкурентного анализа наибольший опыт работы мы имели именно в среде MySQL.

**2.2.3. Особенности использования фреймворка CakePHP**

Фреймворк CakePHP – это программный каркас (framework) для создания веб-приложений[[6]](#footnote-6), написанный на языке PHP и построенный на принципах открытого ПО. CakePHP реализует паттерн «Модель-Вид-Контроллер» (MVC).

Причин, по которым был выбран данный программный каркас, несколько:

* бесплатность и открытость программного кода ядра;
* MVC архитектура;
* простое добавление модулей;
* возможность локализации;
* гибкая система кэширования;
* механизм, обеспечивающий безопасность и сохранность данных;
* развитая система общения с БД (PostgreSQL, MySQL, SQLite);
* генерация программного кода по схеме базы данных;
* компоненты для генерации и заполнения форм;
* простое внедрение AJAX-технологии.

Важно отметить, что программирование на CakePHP базируется на особых соглашениях (наименование таблиц БД, файлов) и нотациях (например, lowerCamelCase[[7]](#footnote-7)). Может показаться, что подобные ограничения негативно сказываются на гибкости, однако на практике оказывается, что, при поддержке требуемых соглашений процесс разработки только ускоряется, а гибкость совершенно не страдает.

Как уже было сказано ранее, CakePHP следует паттерну разработки MVC, что позволяет разделить приложение на три основные части:

* Модель (Model) – включает в себя файлы из папки /models;
* Вид (View) – включает в себя файлы из папки /views;
* Контроллер (Controller) – включает в себя файлы из папки /controllers.

Хотелось бы особо подчеркнуть концептуальную сторону использования CakePHP. Представляется важным непрерывность процесса разработки SaaS-приложения. Включаясь в этот процесс на завершающем этапе, CakePHP позволяет перевести схему базы данных (которая будет транслирована из Microsoft Visio, на основе диаграммы классов) в программный код при помощи утилиты bake[[8]](#footnote-8). Таким образом, мы моментально получаем набор файлов: models, controllers, views, которые образуют каркас SaaS-приложения.

Еще одной важной особенностью использования CakePHP является упрощенное взаимодействие с базой данных. Дело в том, что стандартная функция PHP, посылающая запрос MySQL, возвращает дескриптор результата запроса[[9]](#footnote-9), который в дальнейшем нужно передавать в функцию, работающую с результатами запросов, которые в свою очередь возвращают данные не в самом удобном формате. CakePHP предлагает готовые решения для получения информации (find, query), сохранения информации (save, saveAll) и удаления информации (delete) из БД. Эти решения оперируют массивами и имеют возможность гибкой настройки для получения данных в наиболее подходящей форме.

**2.3. Программные средства проектирования SaaS-приложения**

**2.3.1. Представление системы с помощью унифицированного языка моделирования (UML)**

Первым средством, использованным при проектировании SaaS-приложения, стал инструментарий создания UML-диаграмм. До начала разработки каждой отдельной диаграммы был проведен комплексный анализ унифицированного языка моделирования (UML), рассмотрены для сравнения основные типы диаграмм с целью отбора наиболее актуальных из них. За основу взят стандарт UML 1.5, так как для решения нашей задачи, он является исчерпывающим.

UML 1.5 определяет двенадцать типов диаграмм, разделенных на три группы. Это диаграммы, представляющие:

* статическую структуру приложения
* поведенческие аспекты системы
* физические аспекты функционирования системы

Бабич А.В. в своем кратком обзоре UML[9], рекомендует остановиться на рассмотрении всего восьми типов диаграмм, а именно: прецедентов, классов, объектов, последовательностей, взаимодействия, состояний, активности, развертывания.Далее, по итогам проведенного сравнительного анализа, я сокращу этот набор до трех-четырех диаграмм, в полноте описывающих архитектуру SaaS-приложения. Итоги анализа представлены в табл. 6.

Рассмотрев назначение каждого из ранее выделенных типов диаграмм, мы решили, что для проектирования SaaS-приложения будет вполне достаточно применения следующих диаграмм: *прецедентов, классов, последовательностей* и *активности* (в таблице они выделены оливковым цветом). Остальные диаграммы на наш взгляд для решения поставленной задачи не требуются (они отмечены бледно-красным). Теперь перейдем к аргументации нашего выбора относительно каждого типа диаграмм.

|  |  |
| --- | --- |
| **Диаграмма** | **Назначение** |
| Прецедентов | * Определение границы и контекста моделируемой предметной области на ранних этапах проектирования; * Формирование общих требований к поведению проектируемой системы; * Разработка концептуальной модели системы для ее последующей детализации; * Подготовка документации для взаимодействия с заказчиком и пользователями системы; |
| Классов | * Анализ предметной области; * Составление словаря предметной области; * Прямое отображение в исходный код приложения; * Конечный результат проектирования и отправная точка процесса разработки; |
| Объектов | * Отображение множества объектов (экземпляров классов) и отношений между ними в некоторый момент времени; |
| Последовательностей | * Отображение временных особенностей передачи и приема сообщений объектами; * Уточнение диаграмм прецедентов; * Документирование поведения системы, детализация логики сценариев использования; |
| Взаимодействия | * Показывает поток сообщений между объектами системы и основные ассоциации между ними в определенный момент времени; * Альтернатива диаграммы последовательностей; |
| Состояний | * Демонстрация того, каким образом работают сложные объекты; * Показывает, как объект переходит из одного состояния в другое; * Описывает процесс изменения состояний только одного экземпляра определенного класса; |
| Активности | * Наглядная визуализация алгоритма решения поставленной задачи; |
| Развертывания | * Графическое представление инфраструктуры, на которую будет развернуто приложение; |

Сравнительный обзор UML-диаграмм. Таблица 6.

Диаграмма *прецедентов* включается в наш набор априори. С ее помощью мы определим границы и контекст моделируемой предметной области, а также сформируем общие требования к поведению проектируемой системы.

Аналогичная ситуация с диаграммой *классов*. С её помощью мы детализируем и конкретизируем составные части будущей системы. Более того, если диаграмма прецедентов отражает начальный этап проектирования, то диаграмма классов завершает проектирование и является отправной точкой процесса разработки.

Диаграмма *последовательностей* является вспомогательной для диаграммы прецедентов. Ее основные функции это детализация логики сценариев использования, а также отображение временных особенностей передачи и приема сообщений объектами.

Диаграмма *активности* представляет собой алгоритм решения поставленной задачи (аналог блок-схемы) и поэтому входит в основной набор используемых диаграмм.

Теперь, что касается оставшихся диаграмм:

Относительно диаграммы *объектов* Бабич пишет следующее: «…диаграмма объектов – это своего рода снимок состояния системы в определенный момент времени, показывающий множество объектов, их состояния и отношения между ними в данный момент». В нашем случае использование подобной диаграммы не целесообразно, потому как поведенческие особенности агентов слишком сложны, и при ее конструировании мы получим либо чрезмерно перегруженную диаграмму, либо напротив, крайне упрощенную диаграмму, описывающую тривиальный случай.

**2.3.1.1. Диаграммы прецедентов**

Перейдем к подробному рассмотрению взаимодействия нашей системы с экторами, в роли которых выступают как сотрудники, поддерживающие разрабатываемую систему, так и внешние пользователи и специалисты. Проанализируем поведение каждого из них, опираясь на диаграммы прецедентов внешней (рис. 25) и внутренней (рис. 26) частей системы.

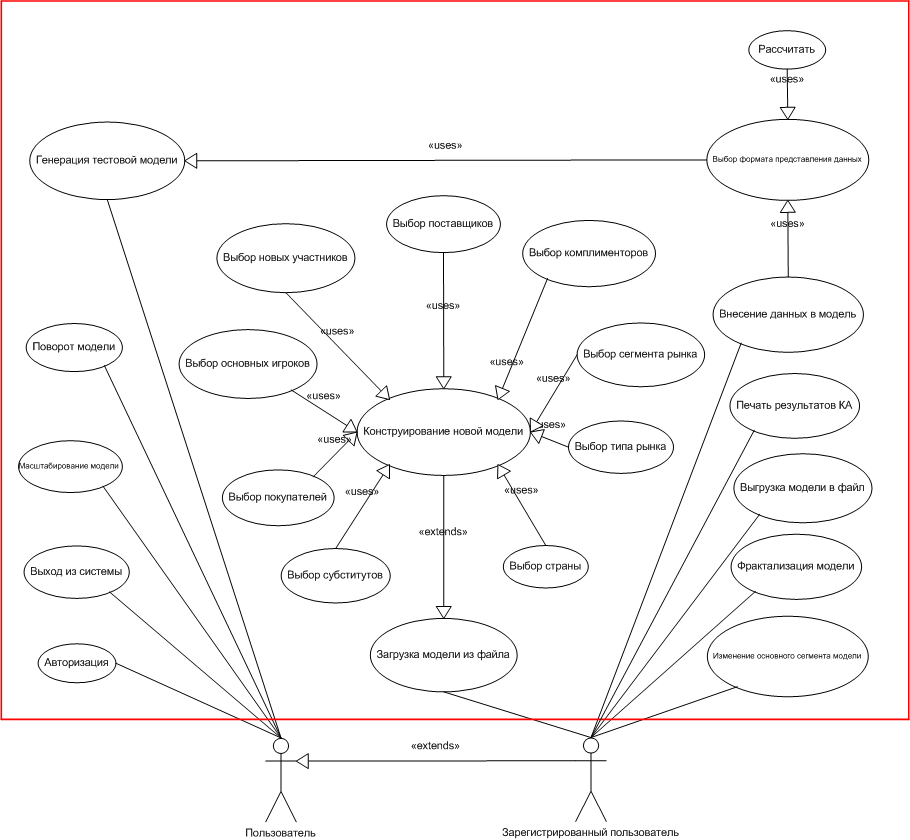


Рис. 25. Диаграмма вариантов использования внешней части системы

1. *Пользователь.* В роли Пользователя, как правило, выступает системный аналитик. Его действия ограничиваются генерацией тестовой модели и ряда простейших операций над ней. Пользователь может ознакомиться с ней, изменить графическое расположение сил и агентов, повернуть ее, масштабировать. Также Пользователь может пройти процедуру авторизации или выхода из системы.
2. *Зарегистрированный пользователь.* Зарегистрированный пользователь, является тем же самым Пользователем, имеющим расширенный спектр возможностей, таких как: генерация собственной модели из исходного набора агентов и параметров, изменение основного сегмента модели, ее фрактализация, возможность редактировать исходные параметры агентов (внесение собственных данных в модель), а также ряд функций по работе с моделью (выгрузка модели в файл, печать результатов конкурентного анализа, сохранение и загрузка модели из БД).

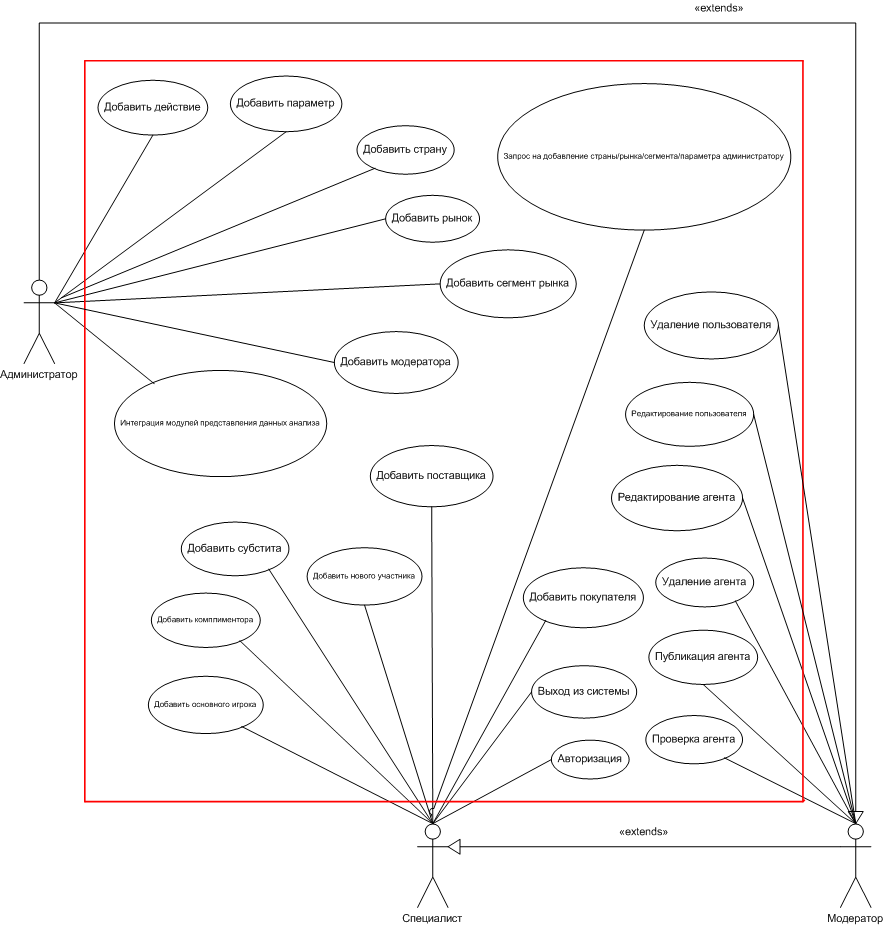


Рис. 26. Диаграмма вариантов использования внутренней части системы

1. *Специалист.* В роли Специалиста выступает системный аналитик. Он является основным поставщиком информации, вносимой в систему, которая в дальнейшем проверяется модератором. Если в роли Специалиста оказывается компания-заказчик моделирования, то его действия связаны с добавлением этой компании, ее основных конкурентов, поставщиков (если они не были внесены ранее) и т.д. в разработанную систему, с одной единственной целью – в дальнейшем самому же воспользоваться указанными данными для конкурентного анализа в интересующем его сегменте рынка. Специалист обладает так же и более широкими возможностями по добавлению страны/рынка или сегмента рынка в том случае, если ранее данная информация внесена не была.
2. *Модератор.* Его действия связаны с проверкой достоверности информации, вносимой Специалистом. Если информация оказывается подлинной (Специалистом предоставлены соответствующие документы, которые подтверждают наличие того или иного агента, или же подлинность установлена самим модератором), то она вносится в систему; в противном случае добавление информации отклоняется. Так же, Модератор следит за актуальностью информации об агентах и принимает соответствующие решения и действия связанные либо с редактированием агента, либо с его удалением из .
3. *Администратор.* Основная роль Администратора – добавление (назначение) Модераторов, а также контроль их деятельности. Помимо этого Администратор рассматривает и обрабатывает ряд запросов от Специалиста, например, запрос на добавление неучтенного ранее параметра какому-либо агенту, или добавление ранее отсутствующего в системе сегмента рынка или же исполняет их по собственной инициативе.

Роль Администратора содержит в себе все возможности Модератора и Специалиста. Роль Модератора содержит в себе все возможности Специалиста.

**2.3.1.2. Диаграмма классов**

Диаграмма классов представлена на рис.27 и играет важную роль в процессе разработки SaaS-приложения КА. Опираясь на нее, мы проектируем архитектуру БД, то есть самый фундаментальный уровень системы. Тщательная подготовка и проектировка необходимы, потому что чем дальше мы двигаемся в процессе разработки приложения, тем сложнее совершать какие-либо изменения в его архитектуре. Названия классов соответствует нотации CakePHP (см. модели CakePHP в разделе 2.2.3.).

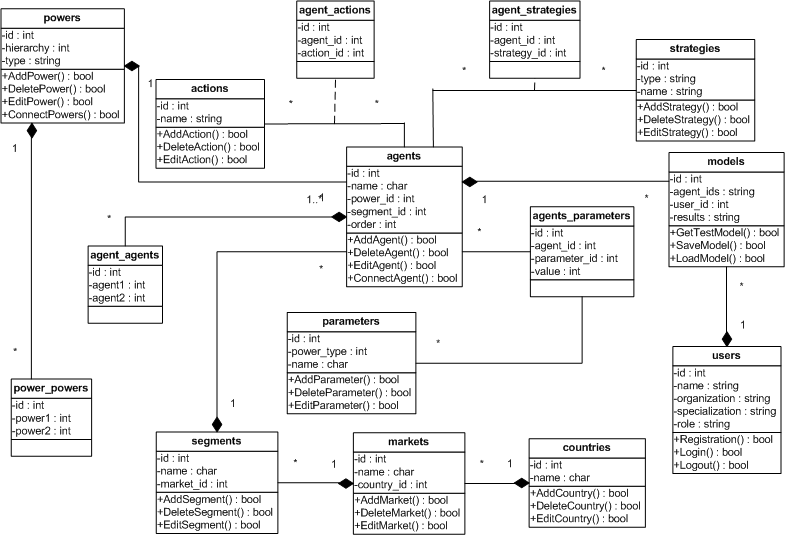
****

Рис. 27. Диаграмма классов

Перейдем к подробному описанию элементов данной диаграммы:

1. *Класс powers.*Содержит атрибуты hierarchy – степень фрактализации и type – тип силы и добавляемых агентов (напр., «Новые игроки», «Поставщики» и т.д.), а также стандартные операторы, доступные эктору «Пользователь» – добавить\_силу(), удалить\_силу(), отредактировать\_силу(), связать\_силы(). Одной силе соответствует несколько агентов.
2. *Класс powers\_powers.*Вспомогательный класс для powers. Содержит атрибуты power1 и power2, обозначающие связь между силой1 и силой2.
3. *Класс agents.*Основной класс проектируемого SaaS-приложения. Так или иначе, с ним связаны все остальные классы системы. Содержит атрибуты name – имя объекта, order – порядковый номер агента в силе, а также стандартные операторы, доступные эктору «Модератор» – добавить\_агента(), удалить\_агента (), отредактировать\_агента() и «Пользователь» – связать\_агентов(). Одному агенту соответствует несколько действий, параметров и связей, а также единственная стратегия и единственный сегмент рынка.
4. *Класс actions.*Содержит атрибут name – название действия, по которому вызывается соответствующая функция, участвующая в получении результата конкурентного анализа сегмента рынка, а также стандартные операторы, доступные эктору «Администратор» – добавить\_действие (), удалить\_действие(), отредактировать\_действие().
5. *Класс agent\_actions.*Вспомогательный класс для agents и actions. Содержит атрибуты agent\_id и action\_id, обозначающие привязку нескольких возможных действий к одному агенту.
6. *Класс agent\_agents.*Вспомогательный класс для agents. Содержит атрибуты agent1 agent2, обозначающие связь между агентом1 и агентом2.
7. *Класс strategies.*Содержит атрибуты type – тип стратегии и name – имя стратегии (например, стратегия «Лидер»), а также стандартные операторы, доступные эктору «Администратор», а именно: добваить\_стратегию(), удалить\_стратегию(), отредактировать\_стратегию(). Выбранная агентом стратегия, определяет его поведение (действия) на рынке. В каждый определенный момент времени одному агенту может соответствовать только одна стратегия.
8. *Класс agent\_strategies.*Вспомогательный класс для agents и strategies. Содержит атрибуты agent\_id и strategy\_id, обозначающие привязку стратегии к агенту.
9. *Класс parameters.*Содержит атрибуты power\_type – тип силы (поскольку тип агента зависит от типа силы, в которую он помещен) и name – название параметра, а также стандартные операторы, доступные эктору «Администратор» – добавить\_параметр(), удалить\_параметр(), отредактировать\_параметр().
10. *Класс agent\_parameters.*Вспомогательный класс для agents и parameters. Содержит атрибуты agent\_id и parameter\_id, обозначающие привязку параметра к агенту, а также значение параметра – value.
11. *Класс segments.*Класс, помогающий осуществлять фильтрацию агентов по определенному сегменту рынка (в нашем случае это БПЛА). Содержит атрибут name – название сегмента рынка, а также операторы, доступные эктору «Администратор», а именно: добваить\_сегмент(), удалить\_сегмент(), отредактировать\_сегмент().Связан с классом markets. Одному сегменту соответствует несколько агентов.
12. *Класс markets.* Класс, помогающий осуществлять фильтрацию агентов по определенному типу рынка (в нашем случае это Авиастроение). Одному рынку может соответствовать несколько сегментов рынка. Содержит атрибут name – название типа рынка, а также операторы, доступные эктору «Администратор», а именно: добваить\_рынок(), удалить\_рынок(), отредактировать\_рынок ().Связан с классом countries.
13. *Класс countries.*Класс, помогающий осуществлять фильтрацию агентов по стране (в нашем случае это Российская Федерация). Одной стране может соответствовать несколько типов рынка. Содержит атрибут name – название типа рынка, а также операторы, доступные эктору «Администратор», а именно: добваить\_страну(), удалить\_страну(), отредактировать\_страну ().
14. *Класс models.*Содержит атрибуты agent\_ids – набор агентов, содержащихся в модели (этой информации достаточно, чтобы восстановить всю оставшуюся информацию о сконструированной ранее модели) и results – результаты конкурентного анализа сегмента рынка, а также стандартные операторы, доступные эктору «Пользователь» – тестовая\_модель(), сохранить\_модель(), загрузить\_модель(). Одной модели соответствует набор из нескольких агентов и один пользователь.
15. *Класс users.*Содержит атрибуты name – имя пользователя, organization – организация или компания, сотрудником которой является пользователь, specialization – ориентация организации пользователя, role – роль пользователя (обычный пользователь, модератор, администратор), а также стандартные операторы, доступные эктору «Пользователь» – регистрация(), вход(), выход(). Одной модели соответствует набор из нескольких агентов и один пользователь.

**2.3.1.3. Диаграммы последовательностей**

Перейдем к рассмотрению последовательности добавления нового агента в проектируемую систему. Для этого мы продемонстрируем процесс взаимодействия и линии жизни четырех объектов, показанных на рис. 28, на примере совершаемых ими действий.

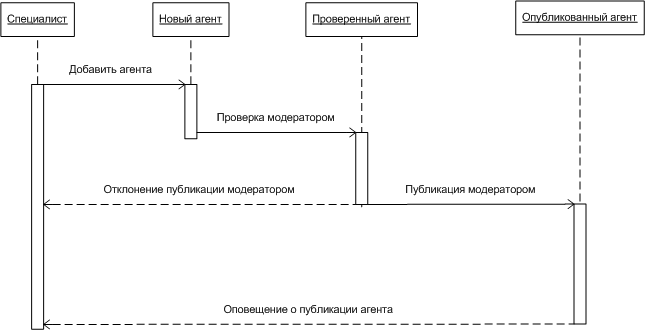


Рис. 28. Диаграмма последовательностей (добавление агента в систему)

1. *Специалист.*Деятельность Специалиста не прерывается на протяжении всего времени выполнения действий прочими объектами. Именно он совершает первое действие в последовательности – добавляет агента в систему, после чего ожидает результата проверки достоверности объекта, тем самым порождая объект «Новый агент».
2. *Новый агент.*Данный объект может соответствовать одному из шести типов сил модифицированной модели М. Портера, а именно: *основные игроки, покупатели, поставщики, новые игроки, субституты* или *комплементоры.* Этот объект содержит в себе информацию о реальном игроке того или иного сегмента рынка (принадлежность к сегменту, тип, имя, набор параметров, официальные документы, подтверждающие его достоверность). Существование Нового агента состоит из трех этапов: формирование и добавление информации об агенте в систему, получение и обработка информации модератором, вынесение решения о публикации или отклонении публикации агента модератором.

Если принимается решение об отклонении публикации, то об этом оповещается Специалист, и процесс завершается. В противном случае, если принимается решение о публикации агента в систему (установлена его достоверность), создается новый объект «Проверенный агент», полностью соответствующий Новому агенту, но с дополнительной пометкой об успешно пройденной проверке.

1. *Проверенный агент.*Время существования Проверенного агента не велико и соответствует тому времени, которое требуется для внесения соответствующей информации в базу данных и сохранении приложений (сканы официальных документов, подтверждающих достоверность агента) в файловое хранилище. После публикации этого объекта модератором, появляется новый объект – «Опубликованный агент».
2. *Опубликованный агент.*Появление данного объекта говорит об успешном прохождении последовательности действий, в ходе которых в систему был добавлен и опубликован новый агент.

**2.3.1.4. Диаграмма активности**

В данном разделе нами будет рассмотрена диаграмма активности (рис. 29), на которой представлен алгоритм принятия решений агентом, соответствующий блок-схеме, приведенной в разделе 1.3.3.

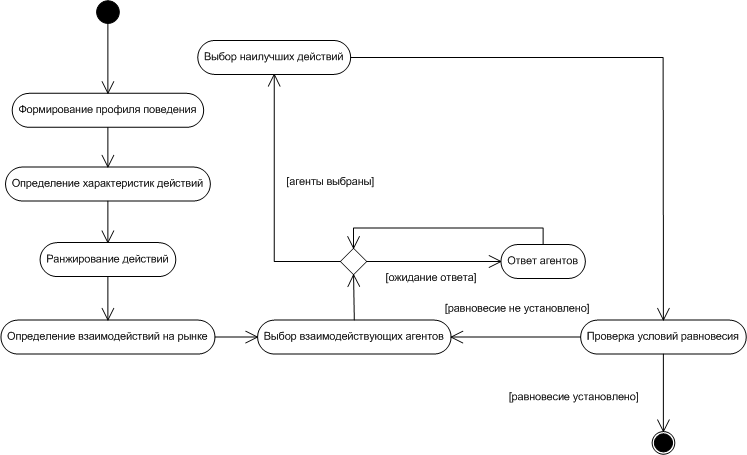


Рис. 29. Диаграмма активности

1. С учетом выбранной стратегии и сформированного множества действий формируется профиль поведения агента (ППА)
2. Для каждого действия из ППА определяются его характеристики.
3. Для каждого действия из ППА происходит ранжирование по заданному критерию.
4. Определяются участники рынка, с которыми необходимо взаимодействовать для достижения цели.
5. Агент выбирает партнеров, заказчиков, поставщиков и др., после чего ждет от них ответа (согласны ли они взаимодействовать с агентом).
6. Другие агенты дают ответ, основываясь на собственных моделях взаимодействий, с учетом своих предпочтений и достижения своих целей.
7. Выбираются и принимаются действия.
8. Процесс продолжается до установления равновесия

**2.3.2. Разработка функциональной модели КА**

Данная часть главы посвящена разработке функциональной модели конкурентного анализа для описания бизнес-процессов использования данного SaaS-приложения топ-менеджерами предприятий – производителей объектов боевой авиации. Для этого была выбрана методология IDEF0, позволяющая сделать акцент на взаимосвязи процессов. Бизнес-процесс представляется в детализированном виде, позволяющем эффективно взаимодействовать системным аналитикам, программным архитекторам, специалистам в предметной области и другим, занятым в проекте лицам. На верхнем уровне модели (рис. 30, диаграмма A0) определяются основные входы (Input, I1-I3) и выходы (Output, O1-O3), управляющие воздействия (Control, C1-C3) и исполнители (M1-M3).

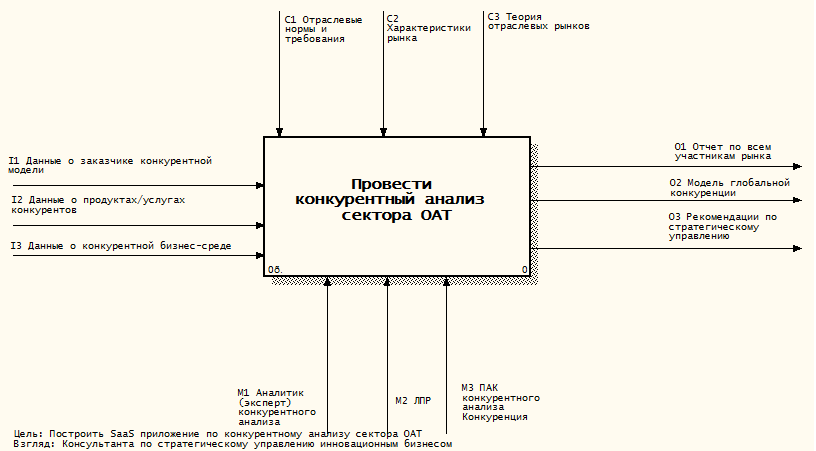


Рис.30. Функциональная модель конкурентного анализа

сектора ОАТ в виде IDEF0-диаграммы (A-0)

Декомпозиция диаграммы A0 отображает основные этапы конкурентного анализа (рис. 31) в интересах заказчика – лица, принимающего решения (ЛПР) от производителя ЛА:

1. проведение анализа данных о заказчике и его продукте;
2. анализ данных конкурентной среды рынка или сегмента рынка заказчика;
3. выявление параметров бизнеса основных конкурентов для использования в агентной модели;
4. построение модели глобальной конкуренции на рынке заказчика;
5. проведение вычислительного эксперимента с целью выработки рекомендаций по формированию конкурентной стратегии заказчика.

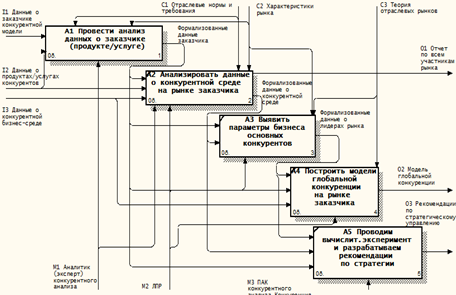


Рис. 31. Декомпозиция процесса конкурентного анализа (A0)

Рассмотрим декомпозицию четвертого и основного из перечисленных выше этапов конкурентного анализа (A4). Для построения модели глобальной конкуренции на рынке заказчика (рис. 32) требуется:

1. построить фрактальный нагруженный граф для сегмента или рынка заказчика;
2. определить все виды отношений между агентами и зафиксировать в виде таблицы;
3. задать и детализировать алгоритмы поведения агентов рынка;
4. построить на основании предыдущих этапов модель глобальной конкуренции.

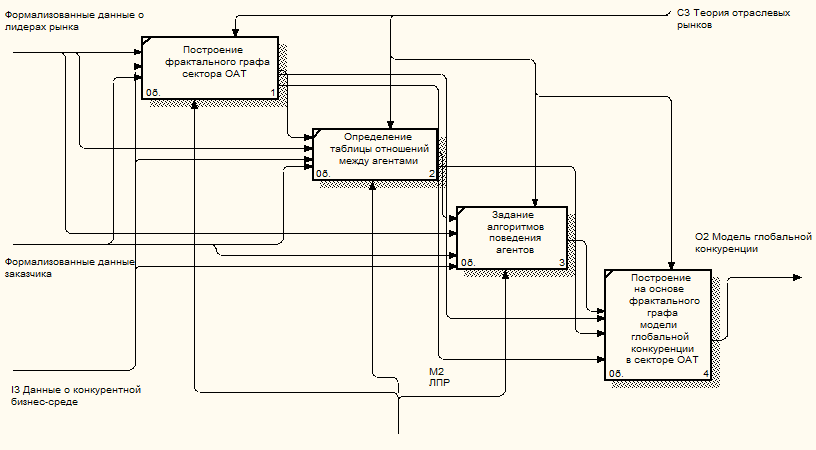


Рис. 32. Декомпозиция процесса построения модели глобальной конкуренции

на рынке заказчика

**2.3.3. Структурный анализ SaaS-приложения**

В данном разделе мы воспользуемся методологией графического структурного анализа для описания внешних по отношению к системе источников и адресатов данных, логических функций, потоков данных и хранилищ данных, к которым осуществляется доступ. Для этой цели будет использована методология DFD. Построение диаграмм потоков данных существенно поможет в проектировании SaaS-приложения, а также наглядно проиллюстрирует внешнее (рис. 33) и внутреннее (рис. 34,…,37) устройство информационной системы. Начнем с рассмотрения диаграммы внешнего устройства потоков данных web-системы.

Внешней сущностью является Пользователь. Он взаимодействует с процессами Авторизации и Конкурентного анализа расширенной модели М. Портера. При Авторизации Пользователь делится своей личной информацией и в случае, если эта информация уже содержится внутри БД системы – выполняется вход, если же информация оказывается новой, в таком случае Пользователь регистрируется в системе, а информация о нем сохраняется в накопитель данных – БД системы.

При Конкурентном анализе модели Портера пользователь передает процессу все данные и настройки сконструированной модели, а именно: *список агентов* и *отношения между ними*, *список сил* и *отношения между ними*, *формат представления результата* и *параметры сохранения модели в БД.* Параллельно с этими данными, процесс запрашивает следующие данные из накопителя данных – БД системы, а именно *стратегии агентов, параметры агентов* и *действия агентов*. На основе всех этих данных происходит их преобразование в результирующие данные, которые подаются двум следующим процессам: Сохранению модели в БД (в случае, если были заданы соответствующие параметры сохранения; при сохранении в БД, запоминаются также и входные данные) и Представлению результатов Конкурентного анализа на основе данных о формате представления результата.

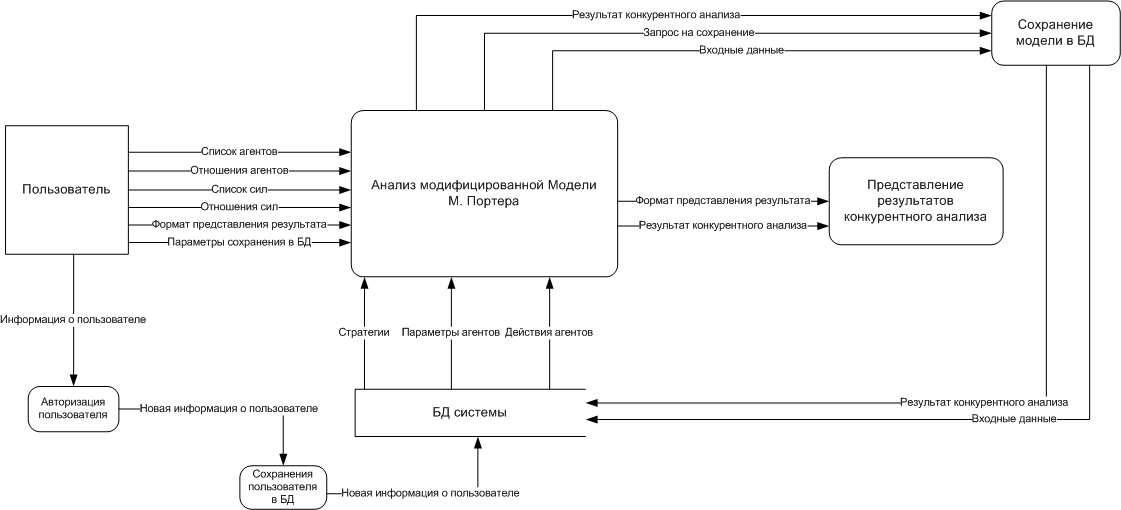


Рис. 33. DFD-диаграмма: внешнее устройство потоков данных приложения

Перейдем теперь к рассмотрению внутреннего устройства потоков данных программного комплекса. В Приложении А можно ознакомиться с панорамным представлением данной диаграммы. Нами были выделены следующие внешние сущности: Специалист и Разработчик ПО.

Разработчик ПО поставляет следующие данные в систему (см. рис. 34): разработанный для интеграции модуль или плагин с подготовленной документацией и описанием. Эти данные проходят через последовательность процессов, связанных с их получением, анализом актуальности использования, тестированием на предмет совместимости с разработанной web-системой и решением о внедрении.

Данные о результате решения о внедрении передаются Разработчику по электронной почте. В случае положительного решения сам модуль попадает в накопитель данных – файловую систему хостинга, а служебная информация (место его физического размещения и пр.) в накопитель данных БД системы.

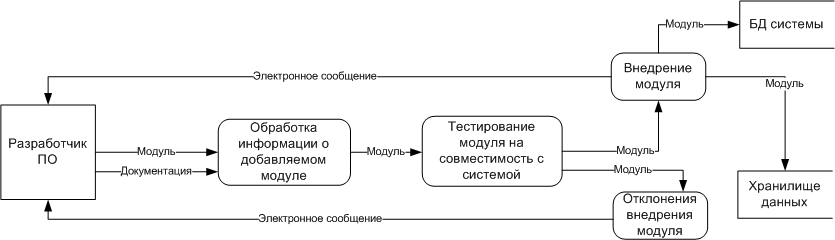


Рис. 34 Внутреннее устройство потоков данных приложения

Специалист подает в систему данные, обозначенные как «входной элемент» (см. рис. 35), которые передаются в процесс Обработки входного элемента, после чего принимается решение о внесении или отклонении внесения элемента в БД. В первом случае, обработанный элемент попадет в хранилище данных – БД системы, во втором случае – Специалист уведомляется об отклонении внесения элемента по электронной почте.

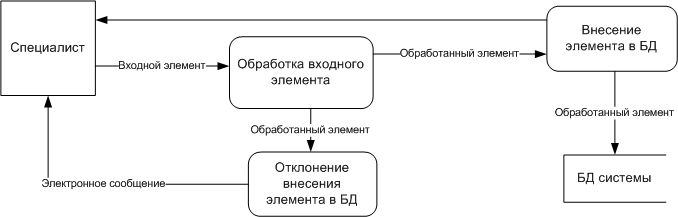


Рис. 35. Внутреннее устройство потоков данных приложения

На рис. 36 представлена декомпозиция обработки входного элемента. После получения данных о добавлении нового элемента в систему, определяется тип данных и составляется запрос на добавление нового элемента. Как только процесс определения типа данных обработал их и составил запрос на их добавление, определенные данные входят в процесс проверки их достоверности, в результате которого выносится решение о добавлении нового элемента в БД.

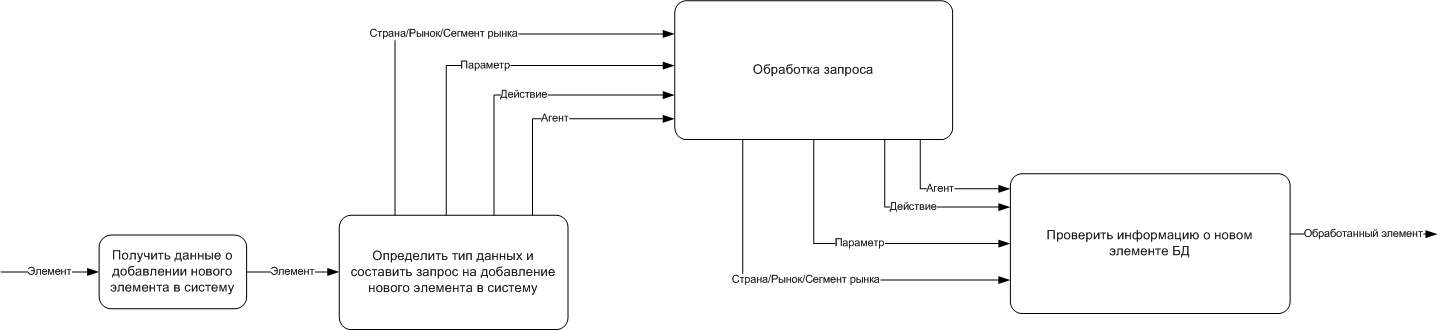


Рис. 36. Внутреннее устройство потоков данных приложения.

Декомпозиция обработки входного элемента.

На рис. 37 представлена декомпозиция определения типа данных. Элемент может быть *новым агентом, параметром, действием, сегментом рынка, рынком* или *страной.*

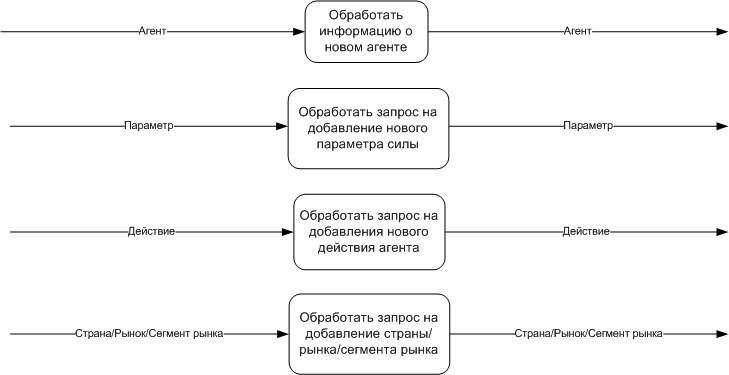
**

Рис. 37. Внутреннее устройство потоков данных приложения.

Декомпозиция определения типа данных.

**2.3.4. Проектирование архитектуры БД**

Здесь подробно представлено устройство базы данных приложения (рис. 38).



Рис. 38. Схема базы данных системы

Выделим в нашей схеме основные и вспомогательные таблицы и обсудим каждую из них. Наиважнейшей сущностью является таблица «Агенты». Именно вокруг нее и строится вся система, а на ее основе производится конкурентный анализ сегмента рынка.

1. *Силы.*Одна из основных таблиц. Является «каркасом» для данных из таблицы «Агенты». По сути, при расчете модифицированной модели М. Портера несет на себе функции определения дополнительных характеристик агента, его позиционирование (уровень иерархии (фрактализация) и тип агента). К одной силе может быть привязано несколько агентов.
   1. *Отношения между силами.*Данная таблица, по сути, является способом хранения разряженной матрицы отношений между различными силами. При конкурентном анализе, нам важно учитывать все связи между силами; именно эта информация и хранится в этой таблице.
2. *Агенты.*Основная таблица нашей web-системы. В ней хранится список агентов с указанием имени, позиции, уровня фрактализации (через таблицу Силы) и принадлежности к определенному сегменту рынка (для фильтрации агентов). Она является центральной, и с ней так или иначе связаны все прочие таблицы. Конкурентный анализ модифицированной модели М. Портера происходит на основе данных параметров агентов, выбранных ими стратегий (а следовательно, и действий) и отношений между ними.
   1. *Отношения между агентами.*Данная таблица, по сути, является способом хранения разряженной матрицы отношений между различными агентами. При конкурентном анализе, нам важно учитывать все связи между агентами; именно эта информация и хранится в этой таблице.
   2. *Параметры агентов.*Данная таблица связывает конкретного агента с набором его параметров и их значениями.
      1. *Параметры.*Это одна из наиболее важных таблиц, поскольку набор параметров того или иного агента играет основополагающую роль в ходе конкурентного анализа сегмента рынка. Для каждой силы определен свой набор параметров, одинаковый для каждого агента, привязанного к силе (различаются только значения). Например, для покупателей определены такие параметры, как *спрос, бюджет* и *полезность.*
   3. *Действия агентов.*Данная таблица связывает конкретного агента с набором его возможных действий.
      1. *Действия.*Еще одна крайне важная таблица. В ней содержатся имена различных действий, предусмотренных для агентов. Каждое действия, на основе его имени, соответствует той или иной функции, запрограммированной в вычислительном модуле на языке PHP. Таким образом, каждое действие по своей сути является вызовом определенной функции, работающей с параметрами агентов.
   4. *Стратегии агентов.*Данная таблица связывает конкретного агента с выбранной им стратегией.
      1. *Стратегии.*В данной таблице содержится набор из трех основных стратегий – «лидер», «следующий за лидером», «бросающий вызов лидеру», определяющих поведение, соответствующее таблице «Действия», того или иного агента.
3. *Сегменты рынка.*Вспомогательная таблица для фильтрации агентов. Определяет профиль агента (в нашем случае это БПЛА). Поскольку наша система будет содержать большое количество различных агентов из самых разных областей и сегментов рынка, то крайне важно точно знать, какой агент к какому сегменту принадлежит.
   * 1. *Рынки.*Более высокий уровень фильтрации, также вспомогательная таблица. Определяет тип рынка, к которому относится агент (в нашем случае это Авиастроение).
        1. *Страны.*Более высокий уровень фильтрации, также вспомогательная таблица. Определяет страну, к которой относится агент (в нашем случае это Российская Федерация).
4. *Пользователи.*Одна из основных таблиц приложения. В ней хранится информация о зарегистрированных в системе пользователях. С ее помощью происходит определение роли («Пользователь», «Зарегистрированный пользователь», «Специалист», «Модератор», «Администратор»), а следовательно и ограничение в правах того или иного пользователя. Одному пользователю может соответствовать несколько моделей.
5. *Модели.*В данной таблице хранится информация о сконструированных моделях М. Портера, сохраненных в системе. Одной модели соответствуют несколько агентов, через которых восстанавливается загружаемая модель (через их связи с параметрами, действиями и прочими характеристиками).

**2.4. Разработка GUI**

**2.4.1. Предварительная подготовка и описание основных концепций организации пользовательского интерфейса**

Важным этапом проектирования и реализации приложения является подготовка и составление макетов пользовательского интерфейса (GUI). На рис. 39 представлен основной макет интерфейса – главная страница с рабочей областью.

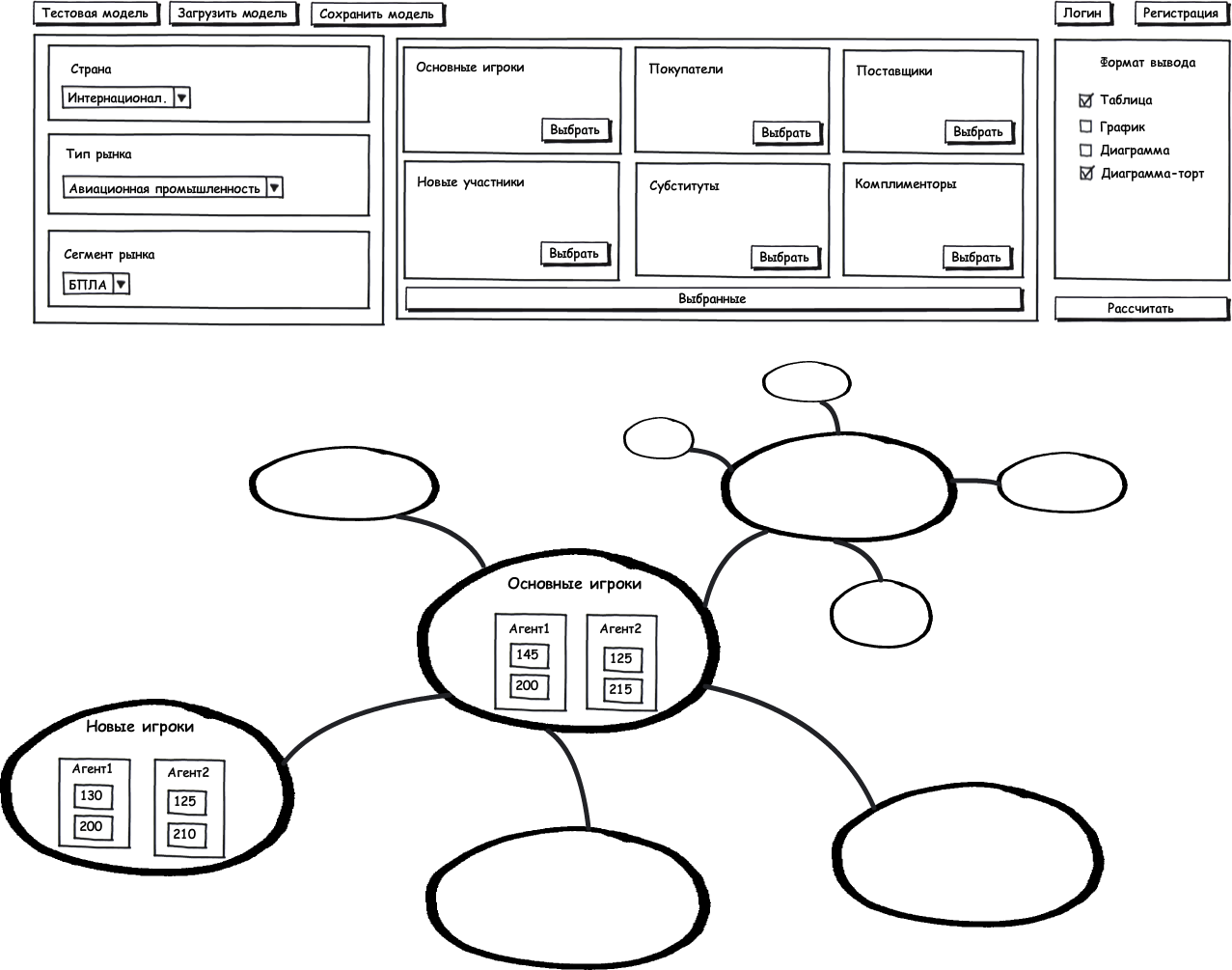
****

Рис. 39. Основной макет интерфейса: главная страница с рабочей областью

Рассмотрим макет подробнее, остановившись на каждом из спроектированных блоков. Верхнее меню для удобства разбито на две части:

* первая отвечает за работу с модифицированной моделью М. Портера («Тестовая модель», «Загрузить модель», «Сохранить модель»);
* вторая же отвечает за взаимодействием с пользователем («Логин», «Регистрация»; после входа в систему меняются на «Личный кабинет» и «Выход»).

Большой блок настроек и фильтров, следующий сразу за основным меню, разбит на три части:

* В первой части происходит фильтрация агентов по местоположению, типу рынка и сегменту рынка. В случае выбора страны «Интернационал» (что соответствует международному конкурентному рынку) предлагается возможность выбора как всех стран, так и только некоторых из них. Данный блок позволяет пользователю сузить круг возможных агентов до отдельного сегмента рынка. Такая возможность является необходимой, потому как наша web-система должна будет располагать очень большим количеством возможных для выбора агентов и возможностями детализации и масштабирования.
* Во второй части происходит отбор агентов для проведения конкурентного анализа. При добавлении того или иного агента к силе, происходит автоматическая фильтрация по типу силы (например, если мы работаем с силой «Новые участники», то среди агентов также будут только «Новые участники»). К каждой силе можно привязать несколько разных агентов.
* В третьей части выбирается формат представления данных, полученных в результате конкурентного анализа.

Далее разворачивается сама модифицированная модель М. Портера, соответствующая выбранным силам и привязанным к ним агентам. Следующим этапом является связь между силами и агентами, после чего возможно проведение конкурентного анализа в заданном сегменте рынка и вывод результатов в выбранном формате.

На данном макете блок настроек и фильтров представлен в развернутом виде, однако при реализации дизайна средствами HTML, CSS и JavaScript он будет свернут в один единый блок, навигация по которому будет производиться с помощью перехода между вкладками, вынесенными в подменю. На рис. 40 показано вспомогательное всплывающее окно, разворачивающееся по нажатию на кнопку «Выбрать».

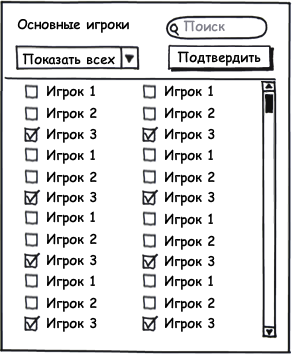


Рис. 40. Окно выбора агентов

В конкретном случае речь идет о вкладке «Основные игроки». Здесь представлен полный список позиций (в данном случае – игроков), предварительно внесенных в базу данных системы, с возможностью фильтрации по поисковой строке на основе полнотекстового поиска с использованием ajax-технологий. Также, помимо отображения всех позиций, предусмотрено отображение только отмеченных или, напротив еще *не* отмеченных позиций. Кнопка «Подтвердить» сохраняет выбранные позиции для последующего проведения конкурентного анализа.

#### Структура портала http://competitive-analysis.ru

Для успешного продвижения проектируемого SaaS-приложения, а также с целью создания активного сообщества вокруг него, было решено разработать портал http://competitive-analysis.ru. По итогам анализа готовых SaaS-приложений (см. список в разделе 2.1.1.), а также, основываясь на опыте посещения веб-сайтов и познаниях в области информационной архитектуры, была продумана структура портала. Она включает в себя следующие необходимые разделы:

* **Посмотреть –** страница, на которой представлена тестовая модель М. Портера; предусмотрена для незарегистрированных пользователей и является ознакомительной.
* **Начать работу –** страница с рабочей областью, позволяющая конструировать, загружать и сохранять модели М. Портера; предусмотрена для зарегистрированных пользователей.
* **Скачать –** страница с документацией, видеоуроками, обучающими материалами, ранее сконструированными моделями (если их пожелают выложить пользователи) и прочими полезными файлами.
* **Новости.** В разделе содержатся последние новости, связанные с поддержкой и улучшением SaaS-приложения, конкурентным анализом, полезными ресурсами на смежную тематику и пр., архив новостей, формы поиска с расширенной фильтрацией. В опубликованной новости предусмотрено использование хэштегов.
* **Партнеры –** согласованный список организаций, использующих разработанное SaaS-приложение.
* **Контакты –** электронные почтовые адреса и телефоны службы поддержки, отделов по работе с пользователями, организациями и пр.

**2.4.3. Разработка форм ввода данных пользователем**

Предварительно подготовив и описав основные концепции GUI SaaS-приложения, перейдем к разработке и описанию форм ввода данных пользователем. Перечислим основные блоки, на которые подразделяются формы:

* основной блок;
* блок конструирования модели;
* блок редактирования характеристик и связей агента.

В состав основного блока входит работа с моделью, настройки рынка, работа с агентами, формат вывода. Дизайн составных частей основного блока представлены на рис. 41,…,43.

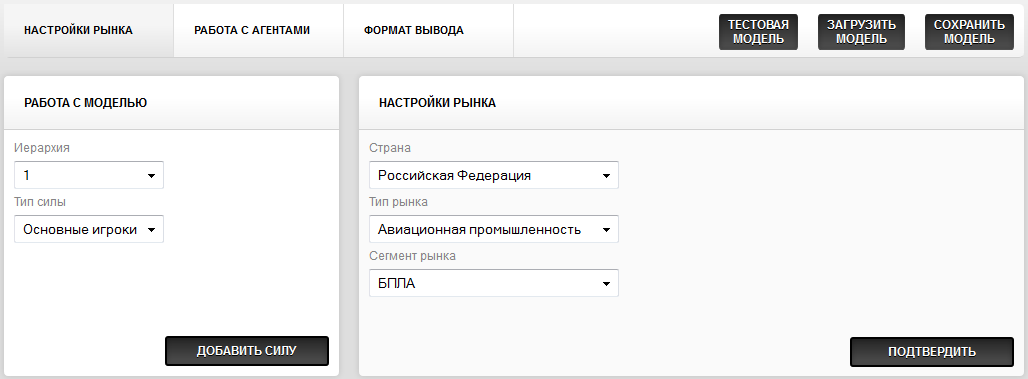


Рис. 41. Основной блок ввода данных. Работа с моделью и настройки рынка

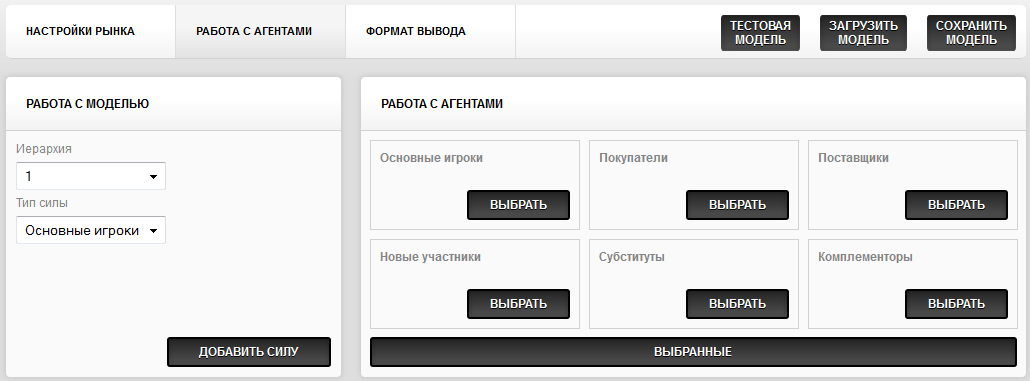


Рис. 42. Основной блок ввода данных. Работа с моделью и работа с агентами

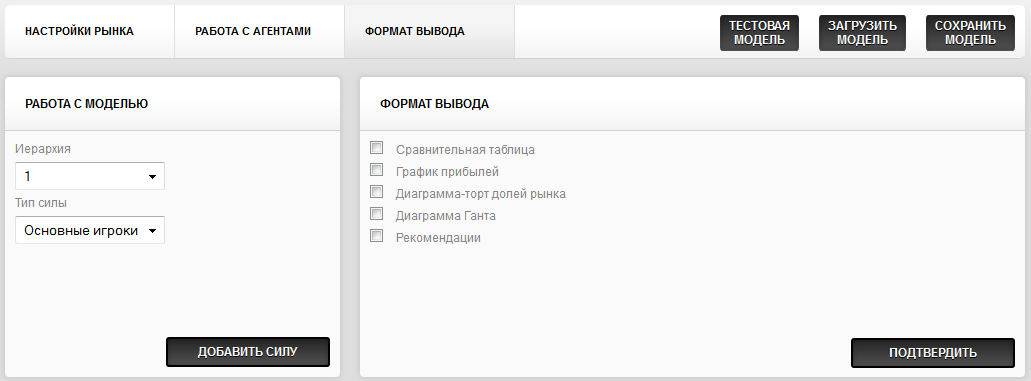


Рис. 43. Основной блок ввода данных. Работа с моделью и настройки рынка

Возможности настроек рынка, работы с агентами и формата вывода описаны в разделе 2.4.1. Работа с моделью позволяет добавлять силы в конструируемую модифицированную модель Портера. Существует возможность выбора типа силы и уровня иерархии. Сила, входящая в различные уровни иерархии маркируются соответствующим уникальным цветом. Для нашего набора тестов мы подготовили три уровня, которые выделены красным, зеленым и синим цветами. Дизайн блока конструирования модели представлен на рис. 44.

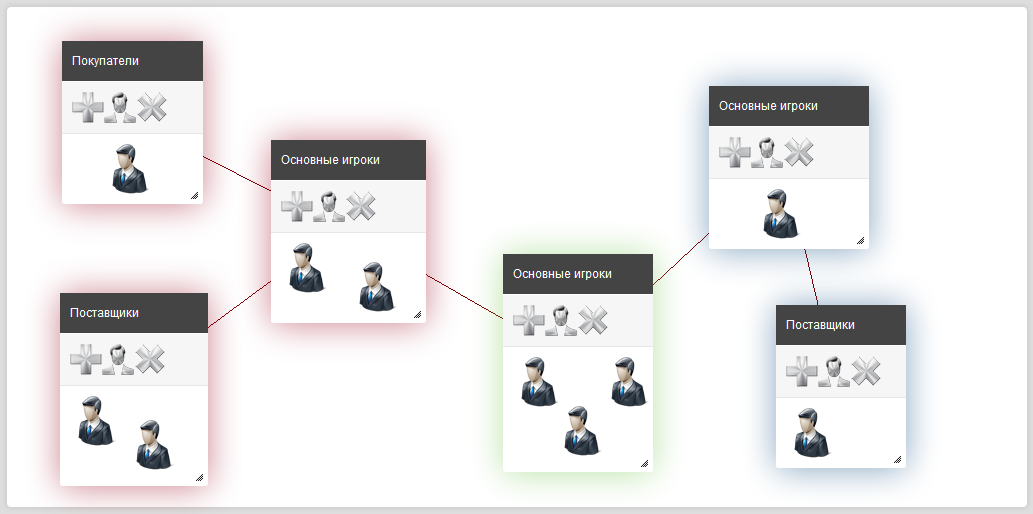


Рис. 44. Блок конструирования модели

В данном блоке расположен набор из шести взаимосвязанных сил на трех уровнях иерархии. Внутри каждой силы размещены 1-3 агента (например, основные игроки первого уровня иерархии содержат в себе двух агентов). Добавление связи между силами осуществляет по нажатию на кнопку «+» в панели действий силы. Следующая кнопка с изображением агента позволяет выбрать и добавить агента из списка. Кнопка «Х» удаляет силу, все ее связи и агентов, расположенных в ней. Данный блок имеет возможность масштабироваться для приближения или удаления определенного сегмента модели. Также, каждая сила и агент могут перетаскиваться по рабочему полю; силы поддерживают опцию растяжения и сжатия. При клике на агента, вызывается блок редактирования характеристик и связей агента, проиллюстрированный на рис. 45.



Рис. 45. Блок редактирования характеристик и связей агента

В данном блоке указывается имя агента, существует возможность редактирования заданных характеристик. По клику на «+», активируется опция связи агентов. Каждому агенту можно задать свой уникальный цвет для отслеживания межагентных связей. На рис. 45 проиллюстрированы эти связи (отмечены красным). Агент, на которого навелась мышь, или который находится в процессе редактирования – заливается своим уникальным цветом, тогда как все связанные с ним агенты обрамляются в рамку того же цвета. Нами было решено, что подобная схема позволяет максимально разгрузить рабочую область и сохранить весь функционал по отображению связи между силами и агентами.

#### Описание формы вывода результатов

Перейдем к описанию формы вывода результатов КА. Данная часть разрабатываемого SaaS-приложения является ключевой, поскольку цель работы – получить инструмент, анализирующий ситуацию на рынке (в конкретном его сегменте). Важно заметить, что анализ проводится как для выбора стратегии конкретной организации (как правило, это лицо, конструирующее модель М. Портера), так и для получения представления о рынке в целом. Исходя из этих соображений, мы разбили форму вывода результатов на две большие части: *рекомендации организации* и *конкурентный анализ ситуации в сегменте рынка.* Рекомендации организации содержат в себе предложения по:

* выбору стратегии;
* выбору поставщиков;
* выбору покупателей;
* изменению назначаемой цены;
* изменению объема производства.

Конкурентный анализ ситуации в сегменте рынка дает общую оценку состояния рынка с учетом всех его игроков. Предлагаются следующие решения:

* график прибылей игроков;
* распределение долей рынка по игрокам (диаграмма-торт/Гант).

Таким образом, лицо, проводящее конкурентный анализ получает полную и подробную информацию о ситуации в сегменте рынка, сравнительную таблицу характеристик конкурентов, рекомендации, относительно выбора стратегии и принимаемых решениях .

**2.5. Разработка вычислительного модуля**

**2.5.1. Реализация SaaS-приложения с использованием CakePHP**

Опишем процесс реализации SaaS-приложения следуя концепции MVC (см. раздел 2.2.1. Концепция MVC). В разделе 2.3.4. нами была спроектирована архитектура БД, основанная на диаграмме классов, в среде Microsoft Visio 2007. Это позволило нам организовать импорт из Visio непосредственно в СуБД MySQL. Структура БД и список таблиц представлены на рис. 46 в MySQL GUI SQLyog.

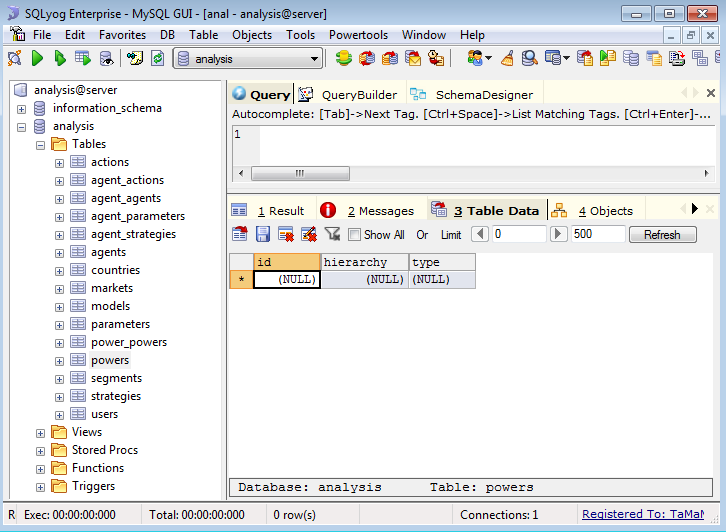


Рис. 46. Структура БД в MySQL GUI SQLyog

Следующим этапом стала генерация Моделей, Контроллеров и Представлений для нашего SaaS-приложения при помощи утилиты cake bake. Результат работы изображен на рис. 47 в среде NetBeans IDE 7.0.1.

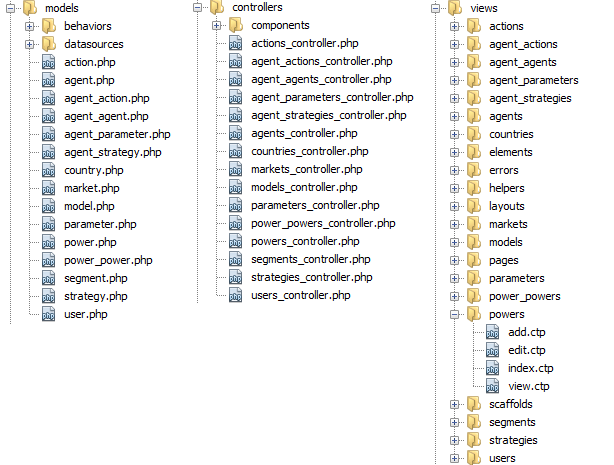


Рис. 47. Сгенерированные файлы в среде NetBeans IDE 7.0.1.

Рассмотрим компоненты Модели. Главной сущностью будем считать class Agents. Именно Agents является связующим классом для всех остальных классов. На рис. 48 изображена инициация данной сущности средствами CakePHP.



Рис. 48. Класс Agents

Взаимодействие и порядок инициации основных сущностей изображен на рис. 49. В первую очередь задается набор элементов системы (силы в расширенной модели М. Портера): где *k* - тип силы (основные конкуренты, поставщики, покупатели и т.д.), *i -* уровень иерархии, и связи между ними (класс power\_powers (PP)). Затем в соответствие к каждому элементу систем (силе) создается агент , где индексы *k* и *i* соответствуют индексам силы, *j* – порядковый номер агента в множестве подобных. Далее для каждого агента задается набор влияний (класс agent\_agents (AA)). Классы PP и AA по сути представляют собой двумерные матрицы (рис. 50).

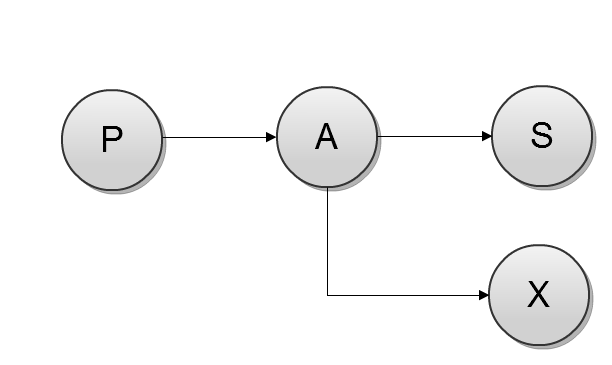


Рис. 49. Основные сущности при переходе от CC к программному приложению

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |
|  | - |  |  |  |  |
|  |  | - |  |  |  |
|  |  |  | - |  |  |
|  |  |  |  | - |  |
|  |  |  |  |  | - |

Рис. 50. Матрица взаимных влияний агентов

Набор стратегий задан строго – тремя отдельными строками в БД. Действия задаются по параметру name таблицы actions. При выгрузке данных из БД происходит сравнение по данному параметру, и вызывается соответствующая функция.

Все связи задаются через вспомогательные таблицы. Каждому агенту с их помощью ставится в соответствие набор действий, а так же стратегий.

Названия классов соответствует нотации CakePHP (см. модели CakePHP в разделе 2.2.3. Особенности использования фреймворка CakePHP).

При проведении конкурентного анализа мы должны обходить всех агентов сконструированной модифицированной модели Портера до тех пор, пока не наступит состояние равновесия (см. раздел 1.1.3.). При таком обходе постоянно выбираются взаимодействующие агенты к которым применяются наилучшие из выбранных действий. Функция, отвечающая за данное поведение, расположена в модели Agents и приведена на рис. 51.

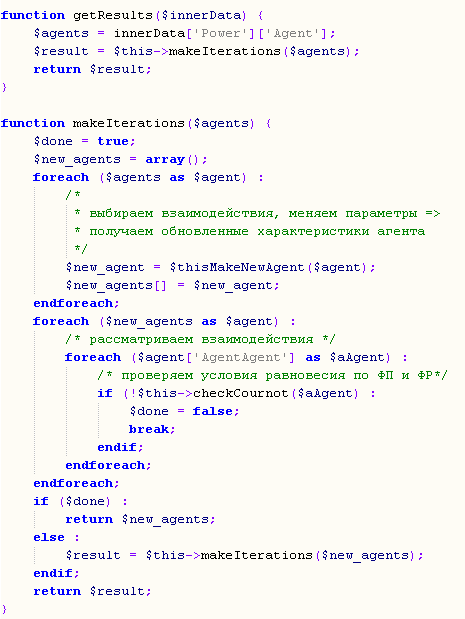


Рис. 51. Основная функция вычислительного модуля

Перейдем к рассмотрению Контроллеров. Контроллер служит средством связи Представления и Модели. В powers\_controller размещена функция index(), отвечающая за обработку и передачу данных об агентах, силах и связях между ними, заключенных в переменной $this->data, в основную функцию вычислительного модуля. Ее программный код приведен на рис. 52.

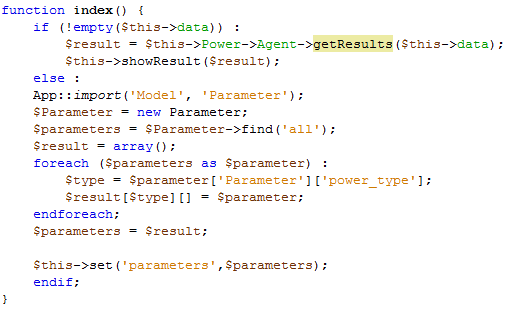


Рис. 52. Функция обработки и передачи данных из Представления в Модель

Компонент Представления содержится в файле index.ctp. Форма ввода данных базируется на стандартных html-тегах <form>, <input>, <submit> и пр. Для реализации масштабирования и подвижности всех элементов, а также для возможности связывать, добавлять и удалять элементы модели, было написано значительное количество JavaScript кода на основе библиотеки jQuery. В случае добавления новых элементов, происходила генерация новых <input> тегов с соответствующими параметрами name, в которых помещались данные. Ниже приведем кусок кода html-разметки и несколько скриптовых функций (рис. 53,…55).



Рис. 53. Листинг index.ctp. Изменение параметров агента

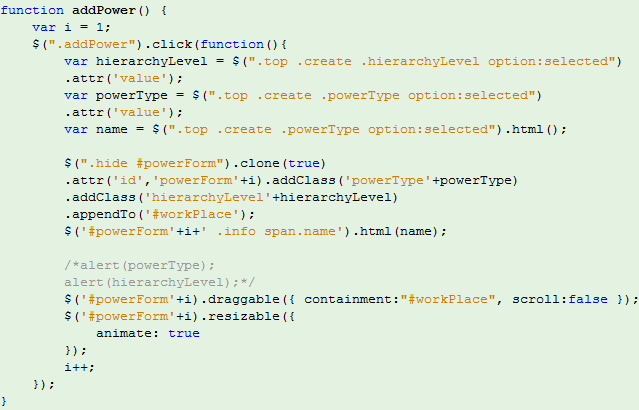


Рис. 54. Листинг index.ctp. Добавление силы



Рис. 55. Листинг index.ctp. HTML-разметка

Завершающей частью работ на вычислительным модулем была верстка страницы с использованием CSS в соответствии с макетами и дизайном сайта.

Реализованный прототип SaaS-приложения конкурентного анализа представлен в приложении B.

**2.6. Тестирование программного комплекса и результаты исследования**

**2.6.1. Задача тестирования и подготовка входных данных**

На базе математической модели конкуренции появилась возможность подготовить и сгенерировать тестовую по Портеру для решения задачи тактической разведки с помощью ЛА, состоящую из трех уровней иерархии: истребители 5-го поколения, БПЛА и микродирижабли. Входные данные модели представлены в шести таблицах (см. табл. 7,…,табл. 12) и соответствующих шести силам модифицированной модели Портера: F1 основные игроки, F2 покупатели, F3 поставщики, F4 новые участники, F5 субституты и F6 комплементоры. Первый уровень иерархии на экране АРМ’а аналитика маркирован красным, второй – зеленым и третий – синим цветами. В результате счета по программе тестирования будет получен график прибылей основных игроков на каждом уровне иерархии, составлен график распределения долей рынка До и После проведения конкурентного анализа; в результате ЛПР должна быть предложена наиболее предпочтительная стратегия, которая сводится к выбору того или иного типа ЛА для решения данной задачи.

Параметры основных игроков. Таблица 7.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Себестоимость пр-ва товара (млн. $) | Объем производства (ед.) | Затраты на пр-во (млн. $) | Назначаемая цена (млн.$) | Прибыль (млн. $) | Полезность для покупателя (%) | Риски (%) |
| Т-50 (Россия) | 50 | 100 | 20 | 130 | 60 | 70 | 30 |
| F-22 (США) | 200 | 220 | 190 | 410 | 70 | 80 | 20 |
| J-20 (Китай) | 30 | 120 | 25 | 100 | 45 | 60 | 25 |
| Скат | 0.2 | 550 | 0.15 | 0.7 | 0.35 | 50 | 35 |
| MQ-1 Predator | 0.2 | 650 | 0.2 | 0.7 | 0.3 | 55 | 25 |
| HERMES 1500 | 0.2 | 500 | 0.15 | 0.6 | 0.35 | 50 | 20 |
| Ирбис | 0.001 | 1200 | 0.001 | 0.0065 | 0.0045 | 50 | 25 |
| Рысь | 0.0015 | 1150 | 0.001 | 0.0075 | 0.005 | 50 | 25 |
| Гепард | 0.002 | 800 | 0.001 | 0.01 | 0.007 | 50 | 30 |

Параметры покупателей. Таблица 8.

|  |  |  |  |
| --- | --- | --- | --- |
|  | Спрос (ед.) | Бюджет (млн. $) | Полезность (%) |
| МО США | 200 | 15000 | 70 |
| МО России | 95 | 6000 | 60 |
| МО Китая | 115 | 5000 | 50 |
| ОАЭ | 10 | 1000 | 45 |
| МО Индии | 20 | 2000 | 40 |
| МО США | 650 | 195 | 65 |
| МО Израиля | 500 | 175 | 60 |
| МО России | 550 | 100 | 60 |
| МО России | 3150 | 20 | 60 |

Параметры поставщиков. Таблица 9.

|  |  |  |
| --- | --- | --- |
|  | Объем производства (ед.) | Затраты на производство (тыс. $) |
| Pratt & Whitney (Двигатель F119 для F-22) | 200 | 1000 |
| НПО «Сатурн» Россия (Двигатель АЛ-41Ф1 для Т-50) | 100 | 800 |
| ФГУП «НЦП газотрубостроения Салют» (Двигатели) | 40 | 800 |
| Northroop Grumman Corporation США (Авионика) | 100 | 1300 |
| EuroRadar (Авионика) | 35 | 1000 |
| НИИП им. В.В. Тихомирова Россия (Авионика) | 50 | 900 |
| «Фазатрон-НИИР» Россия (Авионика) | 20 | 950 |
| ФГУП «ГосНИИАС» Россия  (Авионика, электроника, навигация) | 30 | 1100 |
| Швеция (Авионика для истребителя JAS39 Grippen) | 30 | 1250 |
| Израиль (Авионика) | 60 | 1400 |
| Япония (Авионика для истребителя F-22) | 100 | 1200 |
| Китай (Авионика для истребителя J-10A, J-20) | 120 | 900 |
| Hindustan Aeronautics Limited Индия  (Электроника для Т-50) | 100 | 900 |

Параметры новых участников. Таблица 10.

|  |  |  |
| --- | --- | --- |
|  | Назначаемая цена (млн.$) | Прибыль (млн.$) |
| F-35 (США) | 300 | 100 |
| ATD-X Shinshin (Япония) | 100 | 40 |
| KF-X (Южная Корея) | 120 | 45 |
| Дассо «Рафаэль» (Франция) | 150 | 60 |
| Eurofighter Typhoon (Англия, Германия, Италия) | 140 | 55 |

Параметры субститутов. Таблица 11.

|  |  |
| --- | --- |
|  | Полезность для покупателя (%) |
| БПЛА | 60 |
| Микродирижабли | 60 |

Параметры комплементоров. Таблица 12.

|  |  |  |
| --- | --- | --- |
|  | Назначаемая цена (тыс.$) | Полезность для покупателя (%) |
| СМИ | 1000 | 70 |
| Авиасалоны | 5000 | 50 |
| Производители игр | 1500 | 40 |
| Создатели авиамоделей | 900 | 40 |
| Производители игр | 1000 | 40 |
| Создатели авиамоделей | 500 | 40 |
| Создатели авиамоделей | 500 | 30 |

В табл. 13 приведено сравнение пересекающихся параметров основных игроков и вычислен коэффициент k для каждого из них. Вычисление коэффициента происходит на этапе инициации модели.

Сравнение пересекающихся параметров основных игроков. Таблица 13.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | F-22 | T-50 | J-20 | Скат | MQ-1 | HERMES | Ирбис | Рысь | Гепард |
| Скорость, км/ч | 2400 | 2500 | 2500 | 800 | 217 | 305 | 10 | 12 | 7 |
| Высота, км | 20 | 20 | 20 | 2.7 | 2.1 | 2.4 | 0.9 | 1 | 1 |
| Полезная нагрузка, кг | 2000 | 2100 | 2000 | 600 | 512 | 450 | 25 | 150 | 300 |
| Длительность полета, ч | 1.5 | 2 | 2 | 1 | 1 | 1  Продолжение таблицы 13 | 72 | 72 | 72 |
| Дальность полета, км | 1650 | 2750 | 2750 | 400 | 110 | 150 | 360 | 432 | 252 |
| k | 0.66 | 0.9 | 0.9 | 9 | 8.2 | 7.9 | 193 | 189 | 200 |

**2.6.2. Описание процесса тестирования**

Поставив задачу тестирования и подготовив входные данные в предыдущем разделе, нами была заполнена БД SaaS-приложения. Для каждого уровня иерархии были добавлены соответствующие агенты со значениями их параметров. Затем мы начали конструировать модифицированную модель М. Портера, описывающую конкуренцию в авиастроении. Она изображена на рис. 56. Процесс конструирования состоял из следующих этапов:

* фильтрация и выбор агентов по стране/рынку/сегменту рынка;
* добавление сил для каждого уровня иерархии;
* добавление агентов для каждой силы из БД;
* добавление связей между силами и агентами;
* установлены начальные стратегии агентов;
* установлены возможные действия агентов;
* установлены параметры вывода данных;
* установлено ЛПР (T-50).

Далее был запущен вычислительный модуль, обрабатывающий результат входных данных. Результат конкурентного анализа представлен на рис. 57. Он включает в себя график прибылей основных игроков рынка, список рекомендаций для ЛПР по выбору стратегии, поставщиков, покупателей, назначаемой цене продукта и изменению объема производства. В следующем блоке представлено распределение основных игроков по долям рынка до и после проведения КА. Таким образом, ЛПР может сравнить положение своей организации в данный момент с положением после следования рекомендациям, полученным в ходе вычислений.

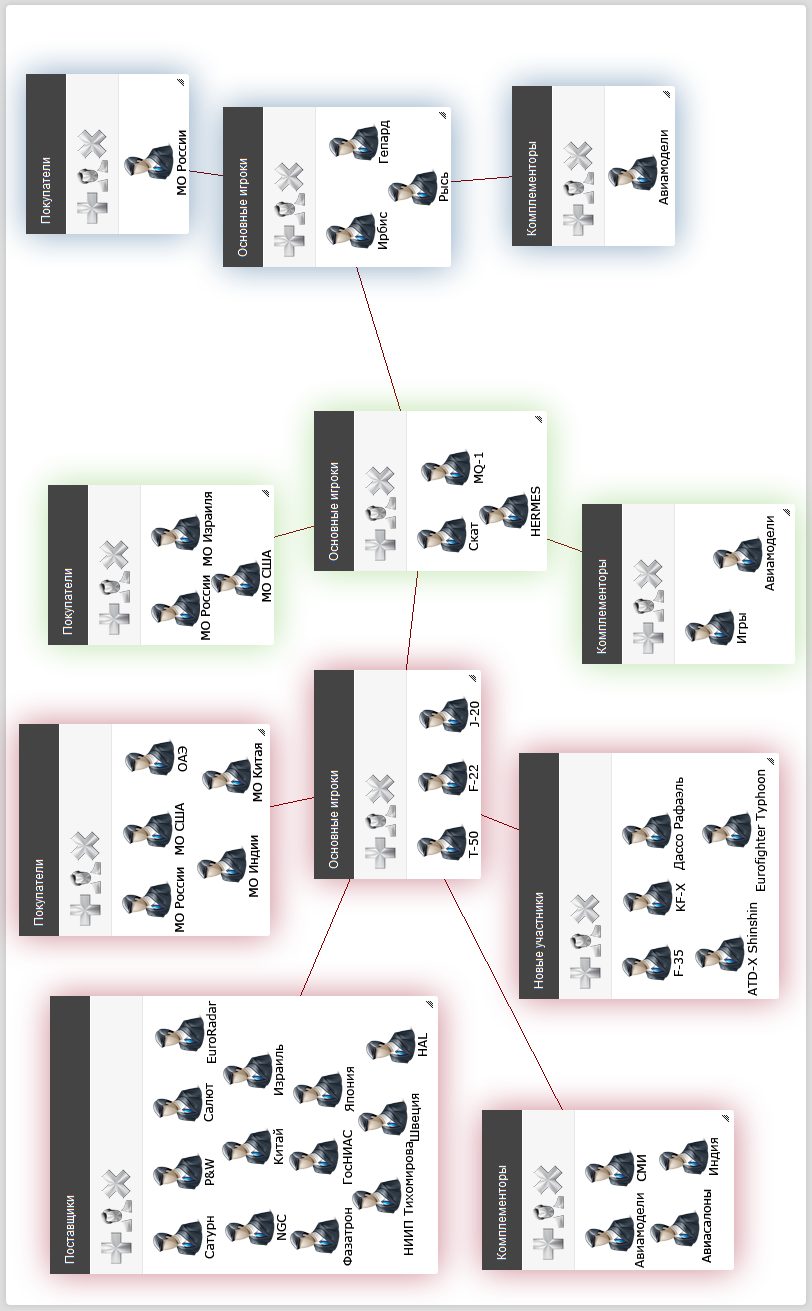


Рис. 56. Рабочая область. Сконструированная модель М. Портера в авиастроении

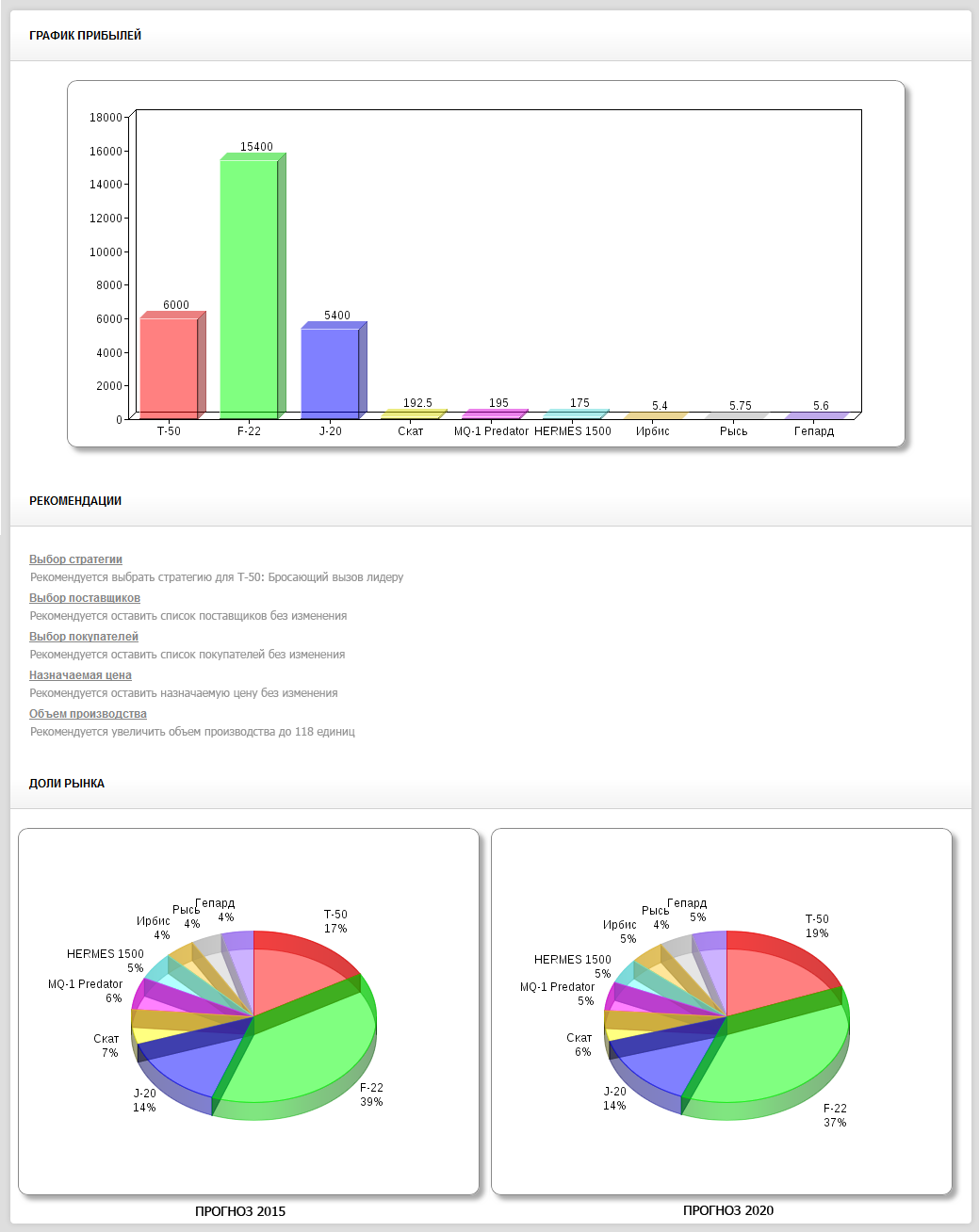


Рис. 57. Результат выполнения КА

**2.6.3. Анализ результатов тестирования**

В качестве результата тестирования SaaS-приложения рассмотрим график прибылей, как рекомендацию для ЛПР при распределении долей рынка основных его игроков. Этот график отобразил положение следующих агентов отрасли авиастроения: T-50, F-22, J-20, Скат, MQ-1 Predator, HERMES 1500, Ирбис, Рысь и Гепард.

Показано, что наиболее прибыльным агентом будет американский истребитель F-22; следом за ним идут T-50 и J-20. Наименее прибыльными являются агенты из сектора микродирижаблей. Это объясняется тем, что затраты на их производство сегодня существенно меньше затрат в секторах истребителей и БПЛА.

Рассмотрим рекомендаций для ЛПР (в данном случае для Т-50). ЛПР было предложена выбрать стратегию «Бросающий вызов лидеру», то есть стать прямым конкурентом фирмы, занимающей доминирующее положение на рынке отрасли авиастроения. Такой подход подразумевает расширение глобального спроса и направлен на обнаружение новых потребителей данного товара, что потребует увеличения объема производства. Однако, поскольку мы моделировали упрощенную версию рынка, то оказалось, что при заданном наборе поставщиков и покупателей система может предложить ЛПР новых взаимодействий. Поэтому увеличение объема производства произойдет за счет увеличения спроса существующих потребителей. Если ЛПР будет придерживаться данных рекомендаций, то руководимая им компания улучшит свое положение на рынке. Это продемонстрированно на диаграммах распределения долей рынка До и После проведения конкурентного анализа (см. рис. 57).

Результат вычислений показал, что для Т-50 возможен прирост доли рынка на 2%, что приведет к изменению соотношений для остальных игроков. Таким образом, Т-50 приблизится к позициям своего основного конкурента F-22 и увеличит отрыв от J-20.

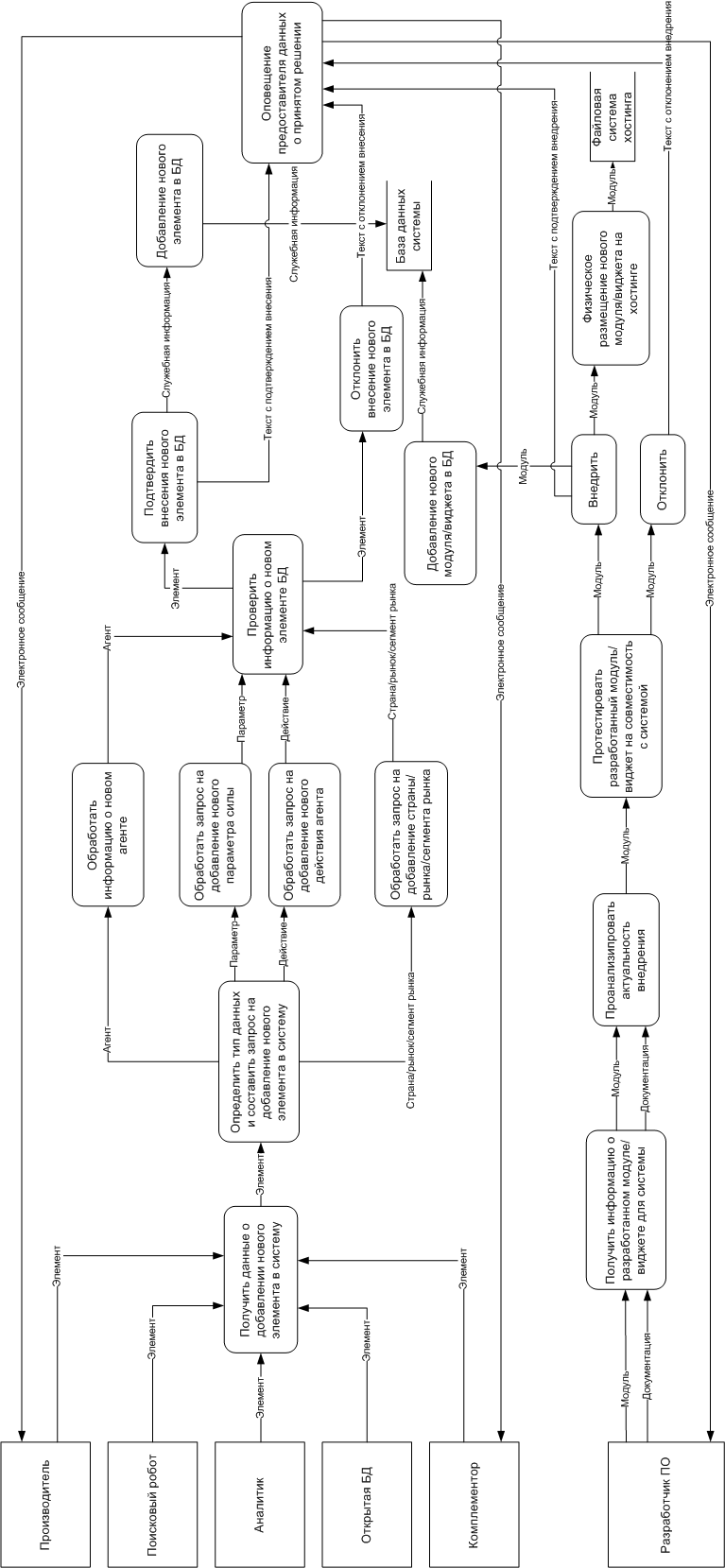
Диаграммы также демонстрируют улучшений позиций в секторе микродирижаблей за счет ослабления позиций беспилотных летательный аппаратов. Подобные изменения объясняются тем, что рассматриваемые привязные аэростаты и беспилотники частично пересекаются в своих практических областях использования: наблюдение, разведка, радиопередача, однако, вследствие ценовых различий предпочтение имеет смысл отдать более дешевым микродирижаблям.

**Список литературы**

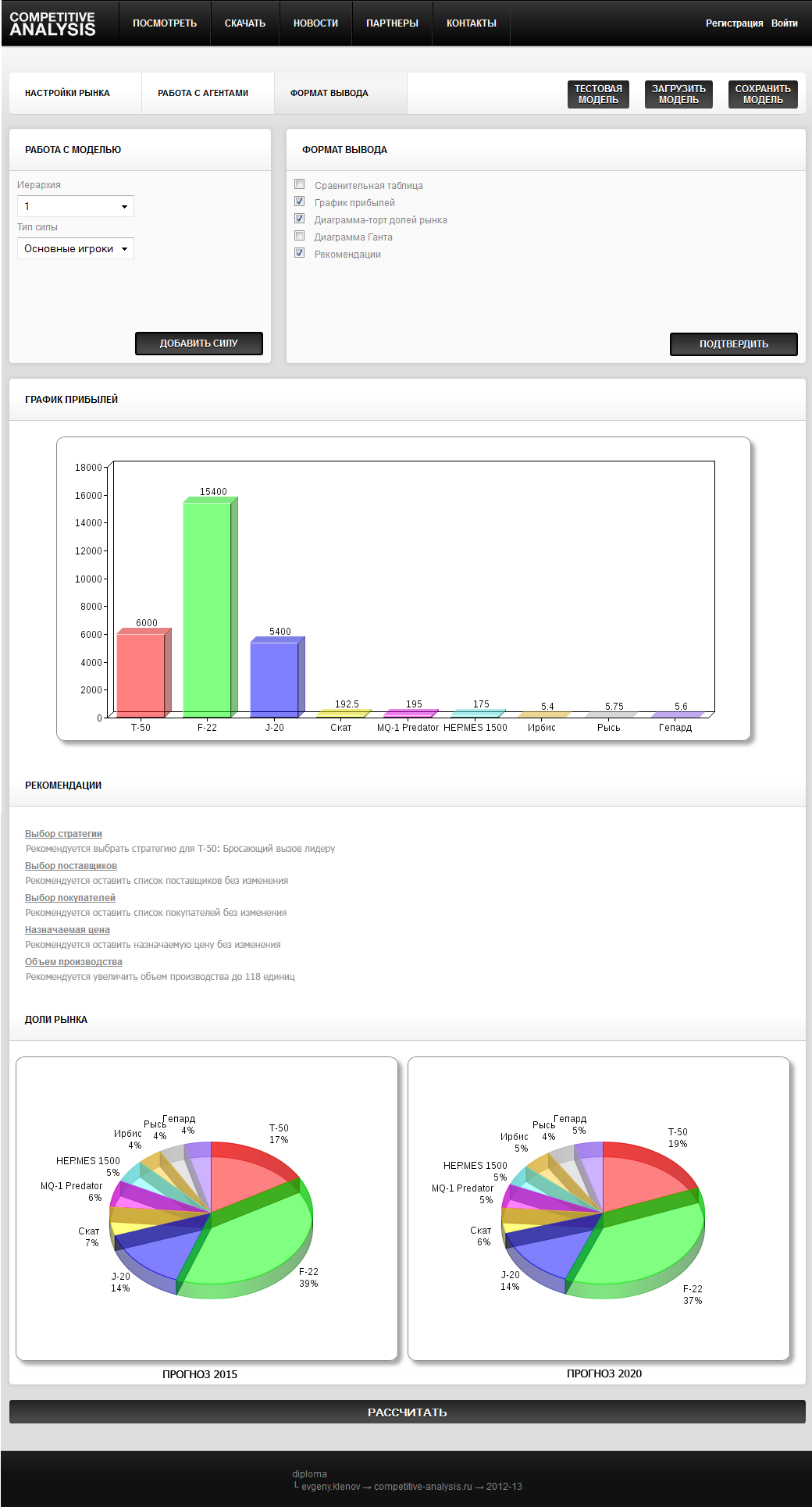
1. Bertrand, J. Book review of theorie mathematique de la richesse sociale and of recherches sur les principles mathematiques de la theorie des richesses // Journal de Savants. - 1883. - v.67. - P. 499–508.
2. *Brandenburger A.M., Nalebuff B.J.* Co-opetition: A revolutionary mindset that combines competition and cooperation. - NY: Currency Doubleday, 1998. - 288p.
3. *Hamel G., Prahalad C.K.* Competing for the Future. - Boston: Harvard Business School Press, 1994. - 357p.
4. Marktform und Gleichgewicht – Вена, 1934.
5. *Moore J.F*. The Death of Competition. - NY: Harper Business, 1996. - 297p.
6. *Porter, M. E.* Competitive strategy. New York: Free Press, 1980.
7. *Porter, M. E.* The competitive advantage of nations. -New York: Free Press, 1990. (Republished with a new introduction, 1998.)
8. *Treacy M., Wiersema F.* The Discipline of Market Leaders. Reading, Mass.: Addison Wesley, 1995, p.31.
9. *Бабич А. В.* UML: Первое знакомство. Пособие для подготовки к сдаче теста UM0-100 (OMG Certified UML Professional Fundamental). – М.: Интернет-Университет Информационных Технологий; БИНОМ. Лаборатория знаний, 2011. – 176с.
10. *Березин В.М., Дайнов М.И.* Защита от вредных производственных факторов при работе с ПЭВМ. Учебное пособие. -М.:Изд-во МАИ, 2003.
11. Би*берштейн Н., Боуз С., Джонс К., Фиаммант М., Ша Р.* Компас в мире сервис-ориентированной архитектуры (SOA): ценность для бизнеса, планирование и план развития предприятия. – М.: КУДИЦ-ПРЕСС, 2007. – 256с.
12. *Бобков Н.И. Голованова Т.В.* Охрана труда на ВЦ: Методические указания к дипломному проектированию. -М.:Изд-во МАИ,1995.
13. *Бьюзен Т. И Б.* Супермышление/ Пер. с англ. Е.А. Самсонов; Худ. Обл. М. В. Драко. – 2-е изд. – Мн.: ООО «Попурри», 2003. – 304с.
14. *Гантмахер Ф. Р.* Теория матриц. — Издание 4-е, дополненное. – М.: Наука, 1988. – 548с.
15. ГОСТ 12.1.045-84 (1988) ССБТ. Электростатические поля. Допустимые уровни на рабочих местах и требования к проведению контроля.
16. ГОСТ 12.1.003-83 (1984) ССБТ. Шум. Общие требования безопасности.
17. ГОСТ 12.2.032-78 (2001) ССБТ. Рабочее место при выполнении работ сидя. Общие эргономические требования.
18. ГОСТ 12.1.004-91. Пожарная безопасность.
19. *Дайнов М.И., Малько Л.И.,Яров В.М.* Борьба с шумами и вибрацией в авиационной промышленности. Методические указания к дипломному проектированию. -М.:Изд-во МАИ,1998.
20. *Дайнов М.И., Метечко Л.Б., Толоконникова В.В.* Методические указания к дипломному проектированию «Экологические платежи за загрязнение окружающей природной среды»-М.:Изд-во МАИ, 2001.
21. *Ершов Д.М.* Реферат к диссертационной работе на тему "Аналитическая web-система поддержки принятия решений при стратегическом управлении организацией", Москва, 2010
22. *Курбатов Б. Е.* Конспект лекций по курсу «Безопасность жизнедеятельности».-М: 2009.
23. *Круглов В.В.* Конкуренция: Учебное пособие. – М.: ТК Велби, Изд-во Проспект, 2004. – 80с.
24. *Малюта А.Н*. Гиперкомплексные динамические системы. - Львов: Вища школа, 1989. – 120с.
25. *Малюта А.Н.* Закономерности системного развития. - Киев: Наук.думка, 1990. – 136с.
26. *Моисеева Н.К., Конышева М.В.*. Управление маркетингом: теория, практика, информационные технологии. М., Финансы и статистика, 2002
27. Налоговый кодекс Российской Федерации.
28. Петросян Л.А., Зенкевич Н.А., Семина Е.А. Теория игр: Учеб. Пособие для ун-тов – М.: Высшая школа, Книжный дом «Университет», 1998. – 304с.
29. *Портер М.* Конкуренция. – М.: Издательский дом «Вильямс», 2005. – 608с.
30. *Портер М. Е.* Конкурентная стратегия: Методика анализа отраслей и конкурентов; Пер. с англ. – М.: Альпина Бизнес Букс, 2005. – 454 с.
31. ПУЭ – Правила устройства электроустановок (2002). Раздел 1 – Общие правила.
32. *Розенфельд Л., Морвиль П.* Информационная архитектура в Интернете, 2-е издание. – СПб: Символ-Плюс, 2005. – 544с.
33. СанПиН 2.2.2.542-96. Гигиенические требования к видеодисплейным терминалам (ВДТ), персональным электронно-вычислительным машинам (ПЭВМ) и организации работы. 1996.
34. *Смит А.* Исследование о природе и причинах богатства народов. - М.: ЭКСМО, 2007. - 960с.
35. СНиП 23-05-95. Естественное и искусственное освещение. 1996.
36. Темников Ф.Е., Волкова В.Н. Методы формализованного представления систем: учеб. пособие – М.: ИПКИР, 1974.
37. *Шатихин Л.Г.* Структурные матрицы и их применение для исследования систем. – М.: «Машиностроение», 1974. – 248 с.
38. *Бабенко Е.А., Клёнов Е.А.* Разработка SaaS-приложения конкурентного анализа сектора беспилотных летательных аппаратов. В трудах 11-ой Международной конференции «АВИАЦИЯ И КОСМОНАВТИКА – 2012», Москва 13 ноября 2012.
39. Программно-аппаратный комплекс Competition конкурентного анализа сегмента рынка / Зарегистрирован в Государственном реестре Госстандарта России 25 дек 2012. Авторы: Бабенко Е.А., Клёнов Е.А., Ершов Д.М., Скородумов В.С. Москва 2012.

**Приложения**

**Приложение А. Панорама внутреннего устройства системы (DFD)**

****

**Приложение B. Демонстрация реализованного SaaS-приложения**



1. Параметры игровой модели определяются индивидуально для каждого сегмента рынка, как правило, на базе **ключевых показателей эффективности (KPI – Key Performance Indicators)** в сторонудостижения стратегических целей компании на данном рынке. [↑](#footnote-ref-1)
2. Герасименко В.В. Метод определения спроса на основе анализа цен и объемов продаж. URL: <http://www.elitarium.ru/2010/01/13/metod_opredelenija_sprosa.html> (Дата обращения: 07.10.12) [↑](#footnote-ref-2)
3. Cournot A. Recherches sur les principles mathematique de la theorie des richesses. Paris, 1938. Ch. VII. [↑](#footnote-ref-3)
4. Когда действия в отношении всех участников проранжированы, становится возможным построение динамического нагруженного графа взаимодействий для n-ого агента. [↑](#footnote-ref-4)
5. В скобках указаны названия предлагаемых ИТ-решений. [↑](#footnote-ref-5)
6. Определение CakePHP. Википедия. URL: <http://ru.wikipedia.org/wiki/CakePHP> (Дата обращения: 05.10.12) [↑](#footnote-ref-6)
7. Определение CamelCase. Википедия. URL: <http://ru.wikipedia.org/wiki/CamelCase> (Дата обращения: 11.10.12) [↑](#footnote-ref-7)
8. Документация по CakePHP. URL: <http://book.cakephp.org/1.3/ru/view/1522/Code-Generation-with-Bake> (Дата обращения: 11.10.12) [↑](#footnote-ref-8)
9. Документация по PHP. URL: <http://php.net/manual/ru/function.mysql-query.php> (Дата обращения: 11.10.12) [↑](#footnote-ref-9)