

# ВЫПУСКНАЯ РАБОТА БАКАЛАВРА

## Тема: Автогенерирование описания модели данных для интеграции системы бизнес-аналитики

Студент гр. 43501/3 А.Р. Раскин

Работа допущена к защите зав. кафедрой

В.М. Ицыксон

« » 2017 г.

# ВЫПУСКНАЯ РАБОТА БАКАЛАВРА

## Тема: Автогенерирование описания модели данных для интеграции системы бизнес-аналитики

Направление: 09.03.01 – Информатика и вычислительная техника

Выполнил студент гр. 43501/3 А.Р. Раскин Научный руководитель,

ст.преподаватель А.В. Зозуля

Дипломная работа, 67 стр., 20 рис., 5 табл., 11 ист., 1 прил.

ИНФОРМАЦИОННЫЕ СИСТЕМЫ, БИЗНЕС-АНАЛИТИКА, PENTAHO, VTIGER, МЕТАДАННЫЕ, MONDRIAN, ИНТЕГРАЦИЯ

В выпускной работе проводится разработка плагина для систе­ мы бизнес-аналитики Pentaho BI с целью интеграции CRM-системы Vtiger для расширения возможностей аналитики данных.

В работе проведён анализ рынка, обзор существующих систем и их сравнение. Был проведён анализ инструментов аналитики различ­ ных систем, в результате чего был выбран OLAP-движок.

Средством разработки является объектно-ориентированный язык программирования Java. Взаимодействие между CRM-системой и платформой Pentaho реализовано с помощью REST-интерфейса.

В результате работы получен полноценный модуль Pentaho с гра­ фическим интерфейсом, анализирующий метаданные системы и со­ ставляющий по ним структуру хранилища для анализа инструмента­ ми Pentaho BI Suit.

# СОДЕРЖАНИЕ

[ВВЕДЕНИЕ](#_bookmark0) . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 5

1. [Анализ существующих решений](#_bookmark1) . . . . . . . . . . . . . 7
   1. [Информационные системы](#_bookmark2) . . . . . . . . . . . . . . . . 7
      1. [Виды информационных систем](#_bookmark3) . . . . . . . . . 7
      2. [Способы аналитики ИС](#_bookmark4) . . . . . . . . . . . . . . 8
      3. [Инструменты аналитики в Vtiger](#_bookmark5) . . . . . . . . 9
   2. [Платформы бизнес-аналитики](#_bookmark10) 11
      1. [Виды бизнес-аналитики](#_bookmark11) 11
      2. [Анализ данных в Pentaho BI](#_bookmark14) 13
   3. [Интеграция BI и информационной системы](#_bookmark17) 16
   4. [Итоги](#_bookmark18) 17
2. [Постановка задачи](#_bookmark19) 18
3. [Архитектура системы и выбор средств разработки](#_bookmark20) . 19 [3.1. OLAP схема](#_bookmark21) 19
   1. [Соответствие схемы и хранилища CRM](#_bookmark23) 20
   2. [Разработка плагина Pentaho](#_bookmark25) 21
      1. [Разработка на языке Java](#_bookmark28) 22
      2. [Разработка с помощью Sparkl](#_bookmark29) 23
   3. [REST API сервис](#_bookmark31) 24
   4. [Разбор XML конфигурации](#_bookmark33) 25
   5. [Итоги](#_bookmark34) 26
4. [Проектирование и разработка системы](#_bookmark36) 27
   1. [Функциональность](#_bookmark37) 27
      1. [Получение конфигурации](#_bookmark38) 27
      2. [Обработка XML-файла](#_bookmark39) 28
   2. [Реализация](#_bookmark40) 28
      1. [Структура компонентов и описание](#_bookmark41) 28
      2. [REST-сервис](#_bookmark43) 29
      3. [Создание Mondrian схемы](#_bookmark46) 30
      4. [Графический интерфейс](#_bookmark51) 35
   3. [Итоги](#_bookmark53) 36
5. [Тестирование системы](#_bookmark54) 37
   1. [Тестирование внутренней реализации](#_bookmark55) 37
   2. [Тестирование REST-сервиса](#_bookmark57) 38
   3. [Тестирование веб-интерфейса](#_bookmark58) 38
   4. [Итоги](#_bookmark59) 39
6. [Анализ разработанной системы](#_bookmark61) 40

[ЗАКЛЮЧЕНИЕ](#_bookmark67) 44

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ](#_bookmark68) . . . 45 [ПРИЛОЖЕНИЕ А.](#_bookmark80) 46

[А.1. Общие сведения](#_bookmark81) 47

[А.2. Используемые технические средства](#_bookmark82) 47

[А.3. Входные данные](#_bookmark83) 47

[ПРИЛОЖЕНИЕ А. ЛИСТИНГИ](#_bookmark80) 48

# ВВЕДЕНИЕ

Почти все крупные организации сталкиваются с быстро ме­ няющейся деловой средой и постоянными сложностями в связи с огромным количеством поступающих данных, требующих анализа и хранящихся в компаниях.

Проблема сложности и качества аналитики таких данных для успешного решения проблем в настоящее время приводит всё больше компаний к решению использовать специальные информационные системы и средства аналитики в них.

Одним из таких средств является CRM(Customer relationship management) - термин, характеризующий практики, стратегии и технологии, используемые компаниями для управления и анализа данных и взаимодействий с пользователем. Такая система позволяет сосредоточиться на релевантных данных о какой-либо части бизнес­ процесса.

К сожалению, аналитические инструменты в большинстве ин­ формационных систем предоставляют скудный функционал и возможности аналитики данных. Во многих системах присутствует только табличный тип отчётов и столбиковый вид диаграм, чего во многих случаях не хватает для полноты анализа данных.

Проблемы, касающиеся мощности аналитики и прогнозирования являются очень актуальными. Для решения данной проблемы мно­ гие организации используют более мощные средства и платформы, например “бизнес-аналитику”. Business Intelligence (BI) - это подход, которые помогает улучшить наблюдаемость бизнес-процессов, упро­ щает анализ данных и создание отчётов.

Чаще всего, организациям приходится нанимать специалистов, занимающихся интеграцией информационных систем для использо­ вания систем бизнес-аналитики. В данной работе рассматривается возможность упрощённой интеграции CRM-системы Vtiger в плат­ форму бизнес-аналитики Pentaho BI Suit, с помощью внедрения данных из системы Vtiger в платформу.

В процессе работы будет разработан модуль, предназначенный для внедрения данных из CRM в платформу Pentaho BI. Данный мо­ дуль позволит импортировать и использовать данные из хранилища CRM-системы для глубокого анализа и построения отчётов.

# 1. Анализ существующих решений

## Информационные системы

Информационные системы - системы, которые включают в себя организацию и анализ информации о бизнесе с помощью внедрения технологий. Это смесь базовых идей менеджмента, операций и теории информационных систем с компьютерными технологиями и инженер­ ным подходом для управления данными в организациях.

* + 1. Виды информационных систем

Существует несколько видов информационных систем, которые наиболее часто используются в повседневном мире. Каждая из них построена вокруг определённого инструмента организации дан­ ных(хранилища), имеет набор инструментов для анализа, формирова­ ния отчётов, систему управления данными(Dashboard) и возможность интеграции со сторонними ресурсами.

* + - 1. CRM-системы. Customer Relationship Management (CRM) - си­ стемы для сбора и хранения всей информации, связанной со вза­ имодействиями c клиентами. Цель CRM - предоставить полный набор данных о клиентах, которые могут использоваться для увеличения продаж, удержания клиентов и повышения эффек­ тивности взаимоотношений с ними. Программное обеспечение CRM помогает организовать, автоматизировать и синхронизи­ ровать продажи, маркетинг и обслуживание клиентов. Примера­ ми таких систем являются: Microsoft Dynamics CRM, Salesforce, Vtiger.
      2. ERP системы. В отличии от CRM, ERP (Enterprise Resource Planning) системы ориентированы больше на сами бизнес-про­ цессы. Как и CRM, ERP-система позволяет быстро распреде­ лять стандартизованную информацию по всем подразделениям

. Все сотрудники компании вводят информацию в систему ERP, создавая моментальный снимок в масштабе всего предприятия в режиме реального времени. О любой проблеме автоматически оповещаются все связанные отделы. Это позволяет более широ­ ко видеть процессы, происходящие в организации и принимать

более точные решения. Таким образом, ERP система позволяет компаниям сосредоточиться на данных, а не на операциях. При­ меры ERP-систем: Microsoft Dynamics ERP, Epicor, Adaxa Suite.

* + - 1. BPM системы. Business Process Management (BPM) - позво­ ляют создавать и совершенствовать широкий спектр процессов, которые прямо или косвенно влияют на клиента. BPM системы автоматизируют процессы документооборота и работы с клиен­ тами. Основой является круги непрерывного улучшения, описы­ вающие ключевые шаги системы, включающие в себя наблюде­ ние, проектирование, моделирование, выполнение и мониторинг. Иногда круг улучшается за счет оптимизации и реинжинирин­ га. BPM обычно не используется изолированно. Примеры ERP­ систем: Apian, PromApp, Camunda.

В данной работе используется CRM-система Vtiger. [[1]Это](#_bookmark69) бесплатная система с открытым кодом и возможностью создания пользователь­ ских модулей.

* + 1. Способы аналитики ИС

Для всех трёх видов систем характерен большой минус - малые возможности в анализе собранных данных. В информационных систе­ мах возможны несколько способов анализа и визуального представле­ ния результатов в виде отчётов.

* + - 1. Работа мастеров - masters in management (MiM). Во всех рас­ смотренных выше системах есть набор инструментов для анали­ за данных. Это специальные программы, автоматизирующие со­ здание отчётов, диаграмм, срезов данных за счёт графического интерфейса и простоты использования. Внутренне такие отчёты представляют из себя простые выборки из базы данных с помо­ щью операторов Join и фильтрации по фиксированному набору элементов. Такие отчёты могут выполнять непосредственно ме­ неджеры, использующие систему.
      2. Специализированные отчёты. Мастера отчётов могут предоста­ вить лишь возможность составления простейших отчётов. В слу­ чаях, когда необходимо углубиться в анализ данных и предо­ ставить данные, требующие более широкого знания системы и

навыков написания на языке SQL, компании привлекают про­ граммистов-профессионалов. Профессионалы обычно использу­ ют SQL-язык для выборок из базы данных, фильтрации и со­ здания частей, требующих сложных арифметических операций. Обычно такой способ - это трудоёмкий ручной процесс, требую­ щий специальных навыков.

* + 1. Инструменты аналитики в Vtiger

Vtiger CRM для анализа и интеллектуальной обработки данных предоставляет специальные модули: Reports, Dashboards и различные виджеты. [[2]”Видж](#_bookmark70)ет - это небольшой графический элемент, выводи­ мый на главную страницу системы Vtiger CRM, либо на страницу “Виджеты” для любых Модулей системы. ”

Виджеты домашней страницы, как можно увидеть на рисунке [1.1,](#_bookmark6) помогают визуализировать представление данных в системе. На одном экране можно поместить и метрики, и результаты анализа. Это помогает следить за состоянием процессов и быстро принимать реше­ ния.

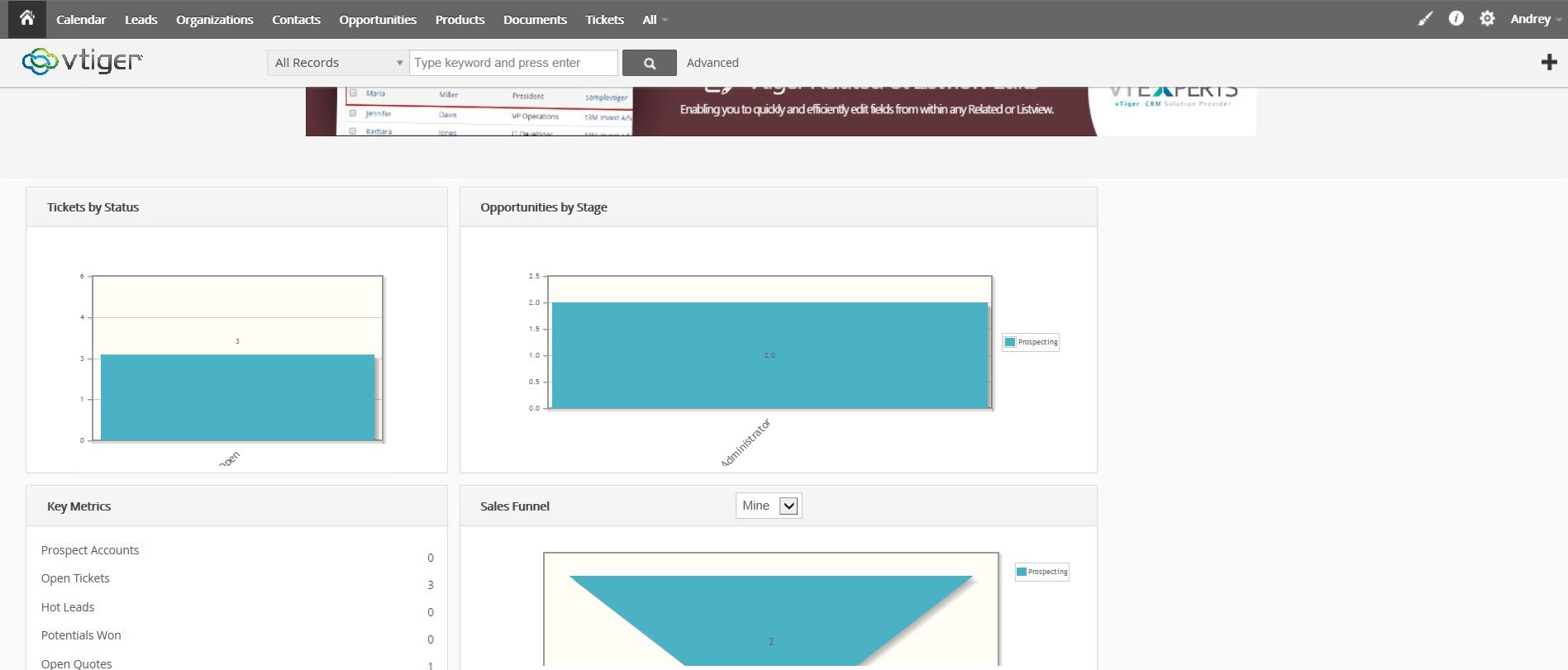


Рисунок 1.1. Пример виджета домашней страницы

Другим вариантом аналитики в представленной системе являет­ ся модуль Отчёты. Модуль Отчёты суммирует, обрабатывает и поды­ тоживает данные, хранящиеся в системе, с помощью отчётов. Vtiger предоставляет генератор и дизайнер отчётов:

* + - 1. Генератор отчётов предназначен для создания отчётов, которые будут отображаться на экране;
      2. Дизайнер отчётов помогает выбрать данные, которые будут отображаться в отчёте.

Как видно на рисунке [1.2,](#_bookmark7) стандартно Vtiger CRM предоставляет ши­ рокий выбор вариантов составления отчётов.

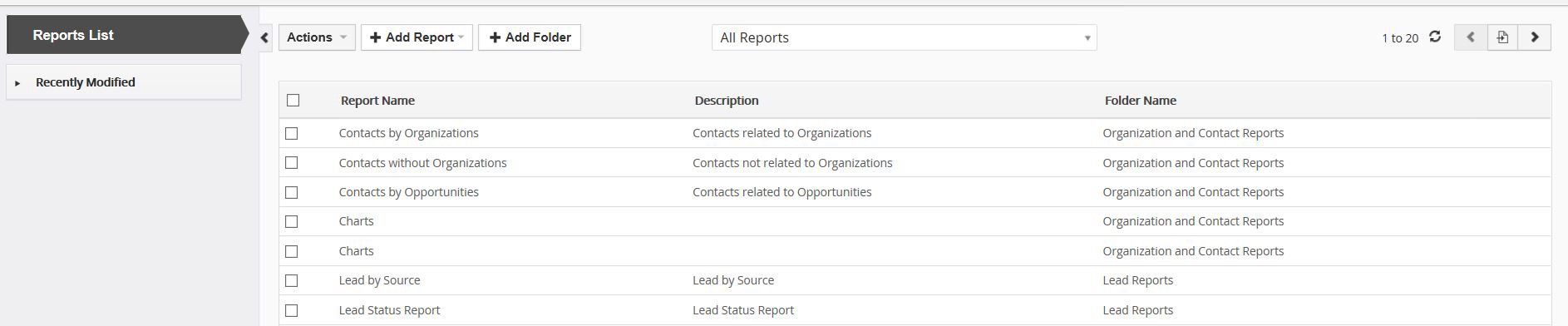


Рисунок 1.2. Список стандартных отчётов

Существует два вида отчётов: диаграммы и детальные отчё­ ты. Детальные отчёты позволяют выводить информацию в табличной форме и выделять из потока данных только релевантную информа­ цию. Пример такого отчёта на рисунке [1.3.](#_bookmark8)

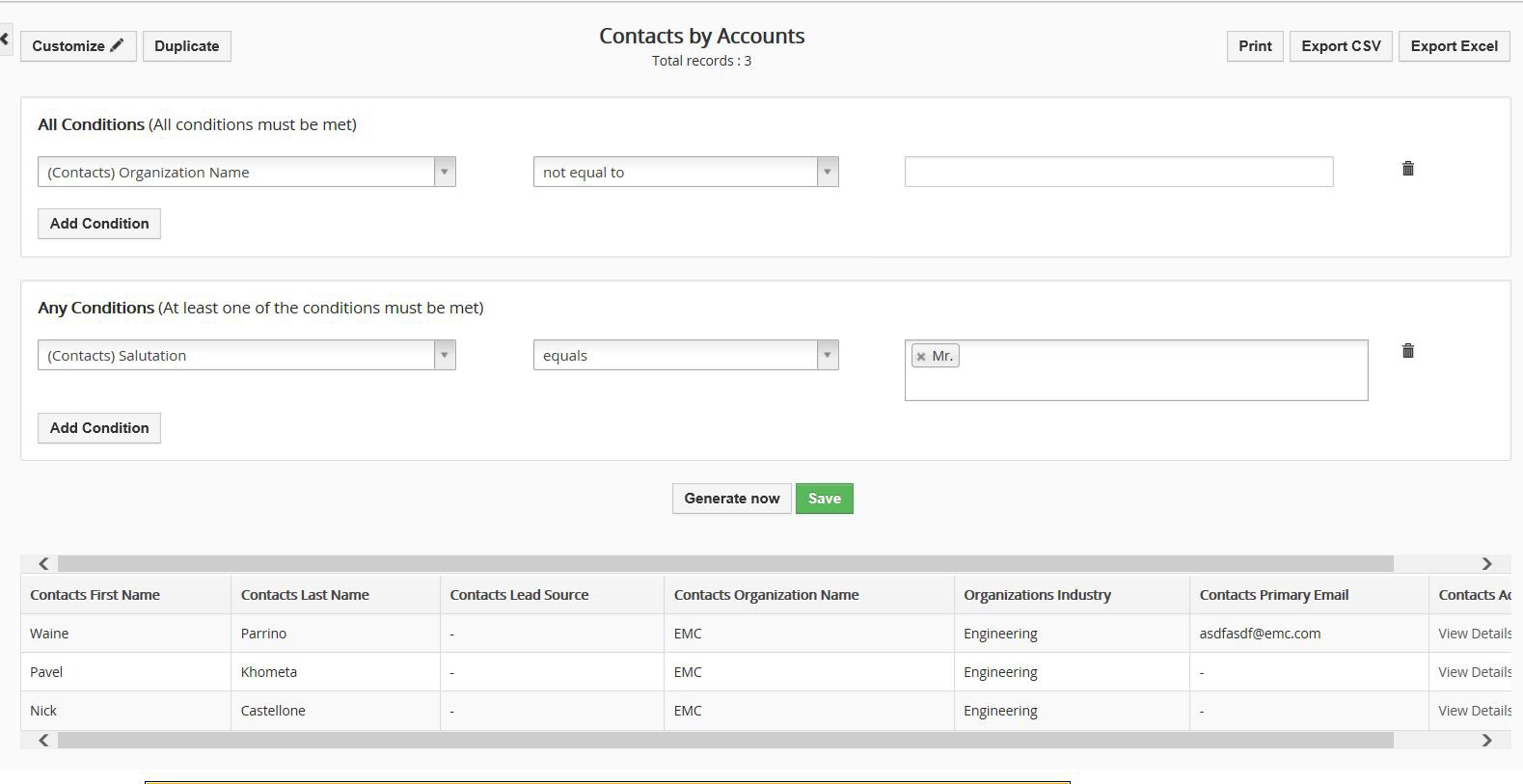


Рисунок 1.3. Пример детального отчёта

Диаграмма в свою очередь - это визуальное представление дан­

ных в виде круговой или столбиковой диаграммы, или линейного гра­ фика. Пример диаграммы на рисунке [1.4.](#_bookmark9)

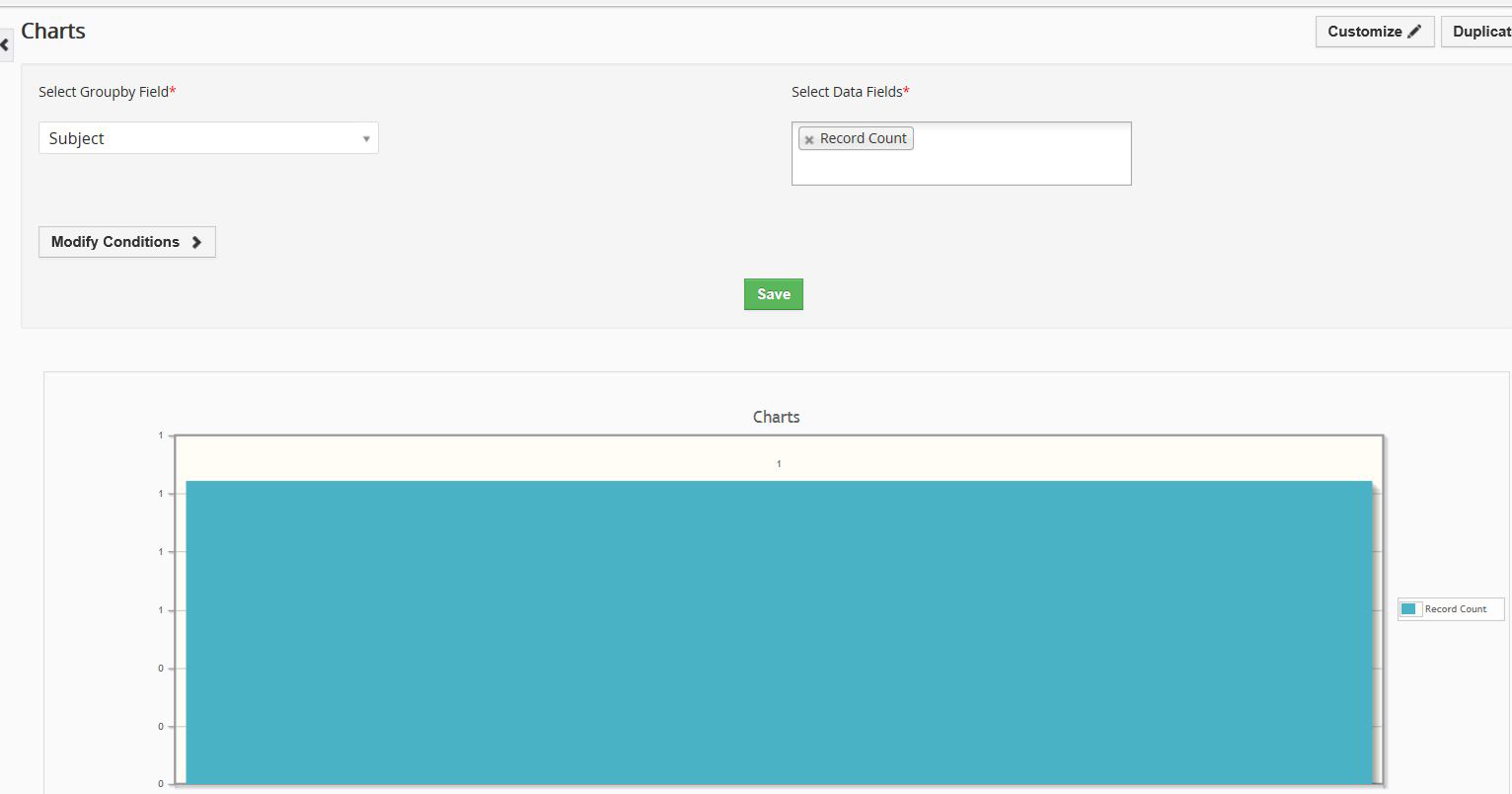


Рисунок 1.4. Пример отчёта-диаграммы

Каждый из двух видов отчётов требует дополнительной настрой­ ки и применения фильтров. Однако оба вида отчётов не всегда явля­ ются полными для аналитики.

## Платформы бизнес-аналитики

Business Intelligence (BI) - это платформа или система, построен­ ная из комбинаций различных компонентов. Эти компоненты могут формировать различные технологии и инструменты.

* + 1. Виды бизнес-аналитики

[[3]](#_bookmark71) Платформа BI(рисунок [1.5)](#_bookmark12) - это всё, что связано с данными. Во-первых, идентификация нужных данных и их источников; во-вто­ рых, хранение данных и их анализ; в-третьих, визуализация данных как информации, используя создание отчетов, запросы к хранилищу и т.д.

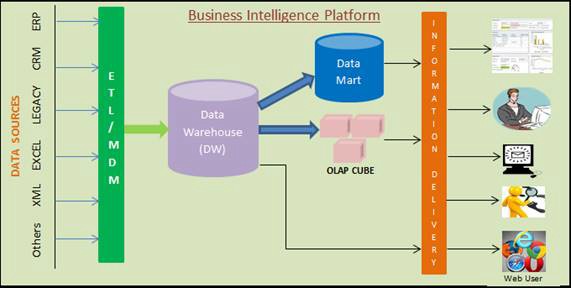


Рисунок 1.5. Структура BI платформы

Несмотря на то, что при внедрении CRM-систем в бизнес-про­ цессы компании обычно наблюдается прирост в продажах [20%+([4]),](#_bookmark72) такие результаты относятся к небольшим фирмам. В таких организа­ циях обычно основными и единственными метриками являются мет­ рики продаж, и использование CRM-систем даёт большой прирост к эффективности. Используя BI-решения, компании получают боль­ шие возможности для анализа данных и принятия своевременных ре­ шений, что влияет на доход и эффективность управления ресурсами компании. Инструменты аналитики BI можно разделить на три груп­ пы(рисунок [1.6),](#_bookmark13) каждая из которых реализует разные BI подходы и возможности:

* + - 1. Управляемый анализ и создание отчётов. В эту группу можно отнести отчёты, панели управления, таблицы. В таких инстру­ ментах метрики и данные установлены заранее, но фильтрация, визуализация, сравнение и анализ варьируются в зависимости от потребностей компании.
      2. Самостоятельный анализ BI системы. Эта категория включает в себя специализированный анализ данных, например рекуррент­ ный. В отличии от управляемого анализа, позволяет пользовате­ лям добавлять и редактировать данные без помощи программи­ стов. Такие инструменты включают в себя: OLAP технологии, Ad Hoc анализ, Data mining.
      3. Продвинутые способы аналитики. Включают в себя инструмен­

ты для прогнозирования, статистического моделирования и ин­ теллектуального анализа данных(Data mining).

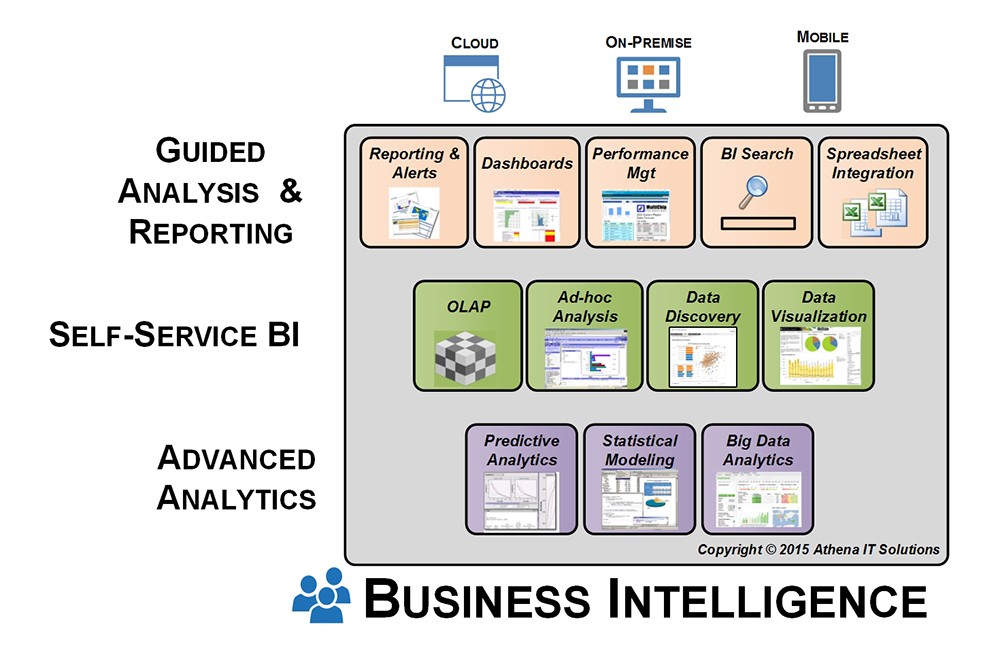


Рисунок 1.6. Инструменты BI аналитики

Примеров BI-платформ множество: IBM Cognos, Tableu, SAP BI, Oracle tools и др. Одним из лидеров также является Pentaho BI. Это полноценный набор инструментов, покрывающий все нужды большин­ ства компаний. Pentaho BI является open-source проектом, но также имеет проприетарную платную версию. Сервер платформы написан на языке Java и предоставляет инфраструктуру для отображения от­ чётов через веб-сервисы. Главным преимуществом является расши­ ряемая комплексная Data Platform, не имеющая пределов в бизнес аналитике и интеграции данных.

* + 1. Анализ данных в Pentaho BI

В платформе Pentaho есть два варианта умного анализа данных:

* + - 1. Pentaho Reporting - встраиваемый движок создания отчётов. [[5]](#_bookmark73) Это набор инструментов с открытым кодом, предоставляющий возможность создания реляционных и аналитических отчётов.

Благодаря Pentaho Reporting можно преобразовывать данные в значимую информацию, адаптированную для клиентов. Pentaho

Reporting может создавать HTML, Excel, PDF, Text или печат­ ные отчеты. Для разработчиков также имеется экспорт отчётов в форматах CSV и XML для передачи другим системам.

Одним из преимуществ является отсутствие необходимости зна­ ний о системе. Графический интерфейс отчётов прост в исполь­ зовании, как видно из рисунка [1.7,](#_bookmark15) и не требует специальных умений.

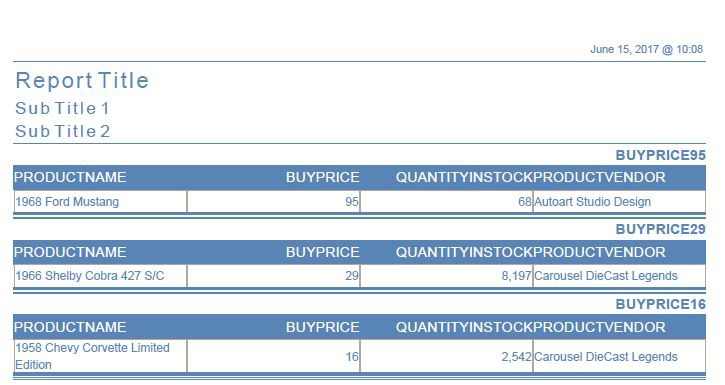


Рисунок 1.7. Отчёт в Pentaho Reporting

* + - 1. Mondrian OLAP Server. [[6]](#_bookmark74) OLAP - Online Analytical Processing

- позволяет пользователям делать срезы нескольких измерений, чтобы изолировать и найти нужные части данных. Например, гиперкуб (многомерная структура данных) может содержать данные о продажах, структурированных по региону, продукту и другие данные. Для работы необходима специальная Mondrian схема - структура хранилища данных. OLAP-аналитика позво­ ляет анализировать сразу огромные массивы данных, проводить аналитику с разных точек зрения, а также поддерживает приня­ тие решений. В отличие от статических отчётов, OLAP-инстру­ менты являются динамическими. Также пользователю не обя­ зательно знать языки выборки, достаточно лишь минимальных знаний о системе. Пример отчёта сделанного с помощью OLAP­ аналитики на рисунке [1.8.](#_bookmark16)

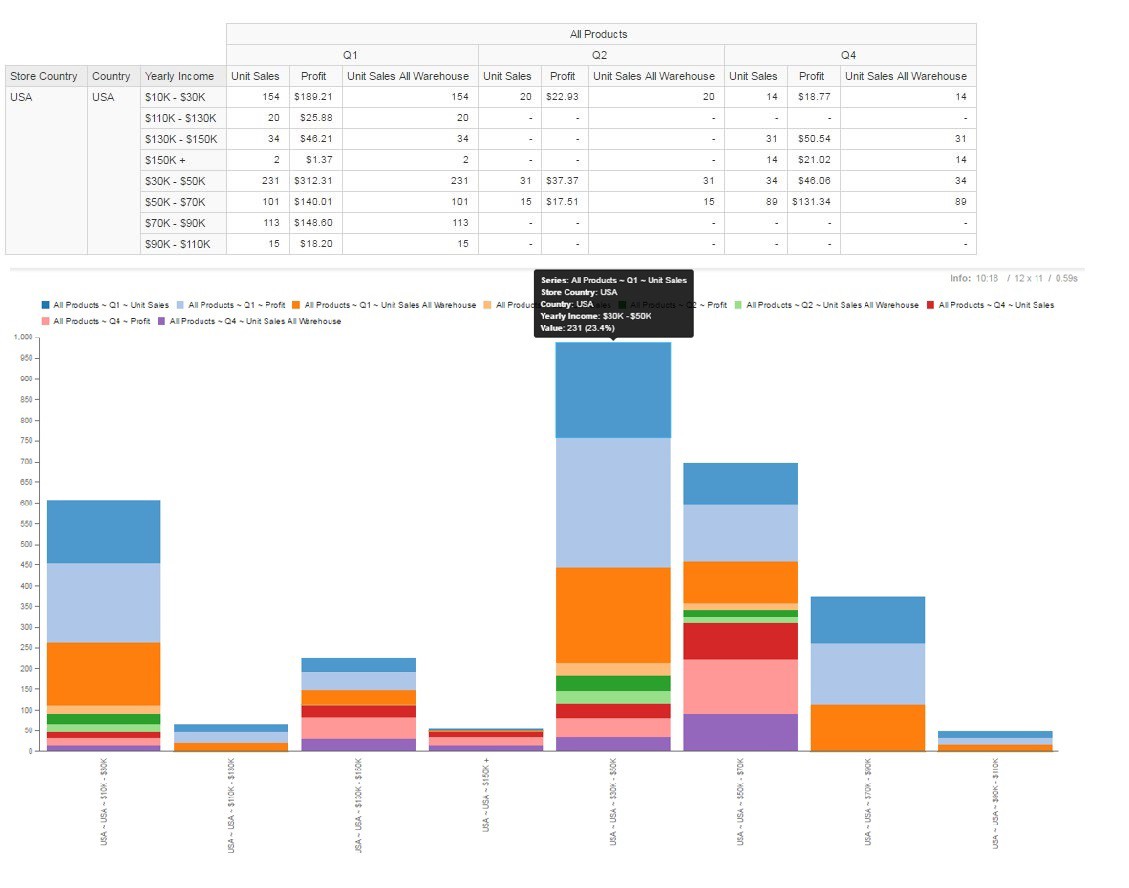


Рисунок 1.8. Анализ с помощью инструмента Saiku

Сравнение двух вариантов отчётов представлено в таблице [1.2.2.](#_bookmark16)

Таблица 1.1. Сравнение Reporting инструментов и Analysis инструментов

|  |  |
| --- | --- |
| Reporting | Analysis |
| Назначение | |
| Преобразование данных в  информацию | Элемент OLAP-куба |
| Показывает, что происходит в компании | Показывает, что происходит в  компании и что с этим можно сделать |
| Наводить на вопросы о  бизнес-процессе со стороны пользователя системы | Отвечать на вопросы с  помощью интерпретации результатов аналитики |
| Подход | |

|  |  |
| --- | --- |
| Пассивный подход | Активный подход |
| Вопросы посылаются  пользователям, которые затем извлекают содержательную уникальную информацию, касающуюся бизнес-процессов | Определённые данные извлекает пользователь для ответа на вопросы, связанные с бизнес-процессами компании |

При реализации варианта с OLAP-аналитикой, получаемая в про­ цессе схема может также использоваться в отчётах Pentaho Reporting. Несмотря на то, что OLAP не позволяет проводить анализ в режи­ ме реального времени, преимущество использования языка SQL для запросов данных неоспоримо. Большие объёмы данных также легче обрабатывать с помощью Mondrian. Система отвечает на запросы к базе данных достаточно быстро даже при миллионах строк и автома­ тизирована в отличие от ручных отчётов.

В данной работе будет проведена интеграция CRM-системы Vtiger с платформой Pentaho для получения возможности использо­ вания OLAP-технологий для анализа данных.

## Интеграция BI и информационной системы

И BI и CRM решения относятся к сбору и анализу данных, ко­ торые крайне важны для эффективного управления в компании. От­ личия однако заключаются в том, что происходит с собранными дан­ ными. В случае BI, принятие решений полностью лежит на пользова­ теле. CRM, напротив, позволяет пользователям участвовать в любом процессе анализа данных и автоматизирует их.

CRM-система, несмотря на свои минусы и основные этапы, может предложить действенные идеи при подключении BI. В большинстве организаций, связь приложения CRM с системой бизнес-аналитики, позволяет получить более эффективный анализ и сохранить лояль­ ность клиентов. Многие пакеты CRM-систем сегодня поставляются с небольшим количеством инструментов для анализа. Необходимость приобретения дополнительных программных средств ведёт к допол­ нительным финансовым расходам, что является проблемой, особенно для небольших кампаний.

Специалисты-аналитики, использующие системы бизнес-аналити­ ки, в свою очередь, не всегда имеют возможность изучить строение данных для анализа CRM-системы. Знание языка программирования,

на котором написано CRM-приложение, необходимо для добавления пользовательских модулей и построения нестандартных отчётов. Ис­ пользуя бизнес-аналитику можно сократить необходимость дополни­ тельных знаний в CRM и сосредоточиться непосредственно на анализе данных.

BI является стратегическим решением, с помощью которого ком­ пании производят глубокую аналитику. BI обрабатывает большие объ­ ёмы информации, а CRM-система в результате интеграции выступает провайдером данных. Бизнес-аналитика позволяет обходить ограни­ чения CRM-систем в способах и глубине анализа.

## Итоги

В современном мире объемы анализируемых данных и требова­ ния организаций к вариантам их анализа продолжают расти. Многие компании начинают понимать эффективность использование CRM­ систем в управлении и внедряют их. К сожалению, большинство ин­ формационных систем, в том числе и CRM-системы имеют довольно небольшой набор для анализа данных. Кроме того, создание отчётов в CRM требует привлечения специалиста и чаще всего трудоёмкой ручной работы. Расширить возможности, автоматизировать и упро­ стить процесс анализа позволяет интеграция информационной систе­ мы с BI платформой. Различные BI платформы предоставляют ин­ струменты для гибкого, многоцелевого анализа и прогнозирования. В данной работе произведена интеграция системы Vtiger CRM в плат­ форму Pentaho BI Suite.

# 2. Постановка задачи

Необходимо разработать модуль для системы Pentaho BI Suite, для автоматического составления физической модели предоставлен­ ной системе базы данных в виде XML-файла с набором таблиц, полей в них и связей между таблицами.

Плагин должен:

1. Получать конфигурацию базы данных на основе вводимой поль­ зователем ссылки на домен сервера информационной системы, логина и пароля к этой системе;
2. Преобразовывать конфигурацию в формат, принимаемый Mondrian OLAP сервером;
3. Публиковать сгенерированный файл с Mondrian схемой на сервер Pentaho BI, подключая его к указанному источнику данных.

Необходима простота использольвания для пользователя - воз­ можность установки плагина без дополнительных приложений, а так­ же интиутивно понятный интерфейс.

Конечная версия модуля должна представлять из себя "универ­ сальный"конвертор реляционной структуры базы данных в Mondrian схему.

# Архитектура системы и выбор средств разработки

Чтобы расширить возможности анализа CRM-системы с помо­ щью интеграции платформы Pentaho необходимо задать платформе структуру базы данных, используемую в системе Vtiger. Для досту­ па к базе данных в Mondrian используются MDX-запросы. Для их исполнения на сервере Mondrian находится Mondrian-схема, представ­ ляющая логическую структуру базы данных.

## OLAP схема

Почему Mondrian не читает данные напрямую из базы данных, а использует схему для хранения структуры? Ответ - Гибкость. В со­ временном мире базовые концепции хранения данных разнообразны. В компаниях использует и различные базы данных, и различные фор­ маты данных. Mondrian-схема - это логическая модель многомерной структуры данных. Она определяет один или несколько кубов в базе данных, каждый из которых содержит одно или несколько измерений и метрик. В Pentaho Mondrian схема соотносит концепт OLAP-техно­ логии со структурой таблиц и строк в базе данных, а также опреде­ ляет поля, которые будут рассчитываться в процессе выборок, но не хранятся в физической модели базы.

Формат схемы описывается с помощью языка XML. Каждая схе­ ма имеет набор кубов. Куб на рисунке [3.1](#_bookmark22) - это многомерная структу­

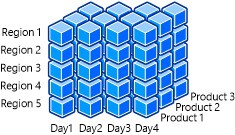


Рисунок 3.1. Пример OLAP-куба

ра, которую можно делить на части с целью отображения релевантной

информации по определённой теме. Каждый куб содержит несколько измерений(Dimension), обычно описывающих определённую бизнес­ проблему. Измерения позволяют фильтровать, группировать и марки­ ровать данные. Далее для более глубокого анализа существуют иерар­ хии с возможностью более детальных уровней анализа. Структура куба представлена в листинге [3.4.](#_bookmark32)

* + 1. D i m e n s i o n ( s h a r e d d i m e n s i o n )
    2. H i e r a r c h y
    3. L e v e l
    4. P r o p e r t y
    5. C u b e
    6. D i m e n s i o n U s a g e
    7. D i m e n s i o n
    8. M e a s u r e
    9. C a l c u l a t e d M e a s u r e s

Также в каждом кубе могут быть метрики - численные значения, ко­ торые пользователи делят, агрегируют и анализируют.

## Соответствие схемы и хранилища CRM

Для того, чтобы автоматически составить описанную выше схе­ му, необходимо проанализировать и соотнести структуру CRM-систе­ мы со схемой базы данных. Рассмотрим, что представляет из себя структура системы Vtiger. Данная CRM-система состоит из модулей, пример которого указан в листинге [3.2.](#_bookmark24)

1. < module >
2. < blocks >
3. < block >
4. < l abe l > L B L \_ A S S E T \_ I N F O R M A T I O N </ l abe l >
5. < s e q u e n c e >1 </ s e q u e n c e >
6. < fields >
7. < field >
8. < f i e l d n a m e > a s s e t \_ n o </ f i e l d n a m e >
9. < uitype >4 </ uitype >
10. < c o l u m n n a m e > a s s e t \_ n o </ c o l u m n n a m e >
11. < t a b l e n a m e > v t i g e r \_ a s s e t s </ t a b l e n a m e >
12. < c o l u m n t y p e > v a r c h a r ( 3 0 ) </ c o l u m n t y p e >

13 ...

14 </ field >

15 ...

1. </ blocks >
2. </ module >

Каждый модуль - это бизнес-процесс. Он состоит из набора полей, каждое из которых представляет определённый атрибут. Среди атри­ бутов находятся тип поля и его имя, метка, например, для локализа­

ции, указание и наличие соединённых с ней таблиц. В соотношении полей с физической моделью, поля являются колонками базы данных. Таким образом, каждый модуль - это куб, имеющий связи с несколь­ кими другими кубами. Для того, чтобы анализировать несколько ку­ бов, в Mondrian имеется структура виртуального куба. Это несколько совмещённых кубов, имеющих перекрёстные связи.

## Разработка плагина Pentaho

После разработки Mondrian схемы, следующим шагом является загрузка её на сервер. Это возможно несколькими способами. Самый простой способ - загрузка вручную. Платформа Pentaho оборудована инструментом загрузки файлов. В графическом интерфейсе сервера пользователь может управлять любыми источниками данных. При­ мер загрузки схемы через консоль можно увидеть на рисунке [3.2.](#_bookmark26)

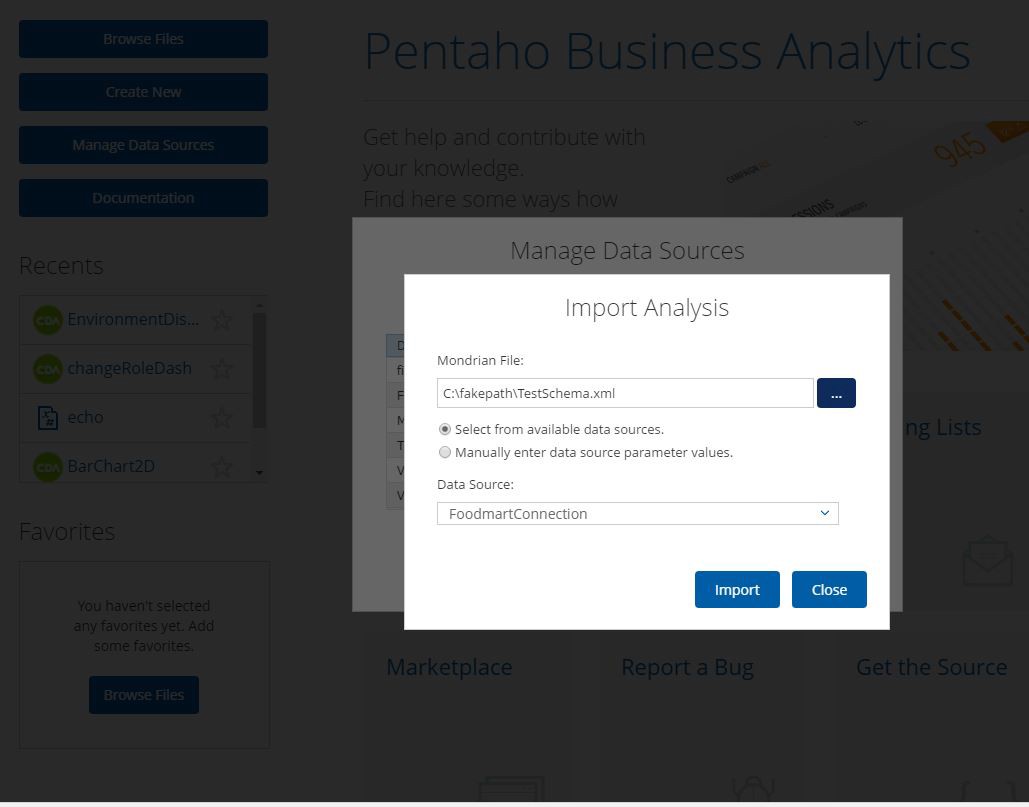


Рисунок 3.2. Загрузка схемы с помощью консоли

Другой способ это Kettle трасформации. Kettle или Pentaho Data Integration - это компонент платформы, отвечающий за основные про­ цессы управления данными ETL - Extract, Transform, Load — «извле­

чение, преобразование, загрузка». С помощью таких операций можно импортировать и экспортировать данные на сервер Pentaho.

Однако, Kettle требует изучения. Альтернативным способом яв­ ляется использование Pentaho A[PI[7].](#_bookmark75) В нём имеется набор точек вхо­ да, позволяющих манипулировать источниками данных и файлами, находищимися на сервере. В работе помещение источника описания на сервер реализовано с помощью таких URL. Пример можно увидеть в листинге [3.1.](#_bookmark27)

Листинг 3.1. "Команда добавления нового источника данных Pentaho API"

1 PUT p e n t a h o / p l u g i n / data - a c c e s s / api / d a t a s o u r c e / a n a l y s i s / c a t a l o g / S a m p l e S c h e m a

Pentaho BI - платформа с открытым кодом, написанная на языке Java. Плагины обладают следующими функциями:

1. Обслуживание другого приложения;
2. Визуализация данных в формате, описанном пользователем;
3. Регистрация новых файлов: схем, отчётов, параметров для сер­ вера;
4. Интеграция сервера Pentaho со сторонними источниками;
5. Улучшение консоли пользователя Pentaho User Console.

Для интеграции Pentaho BI с Vtiger СRM необходим REST-сер­ вис, обозначенный в пункте 1. Также следует разработать способ вза­ имодействия с пользователем, а именно, веб интерфейс.

Для разработки интерфейса предоставлено две воможности: раз­ работка на языке платформы - язык Java и разработка с помощью Pentaho Plugin Builder(SPARKL).

* + 1. Разработка на языке Java

С помощью языка Java платформа Pentaho предлагает разработ­ ку плагинов на основе платформы Spring и JAX-RS. Эти инструмен­ ты дают возможность создать REST-приложение, основанное на ан­ нотациях и XML-конфигурации зависимостей классов. Требуются на­ выки программирования на языке Java, понимание технологии Мо­ дель-Представление-Контроллер. Такой способ разработки освобож­ дает программиста от необходимости получения навыков в CTools, DI

и предоставляет более широкий выбор возможностей для разработки, например, использования более современных програмнных платформ и технологий. Java плагин обычно представляет из себя REST-сервис, запускающийся при старте сервера Pentaho и работающий на том же домене. Такой сервис может быть доступен извне с помощью задан­ ного URL-ссылки при наличии данных аутентификации. В момент обращения к точке входа плагина сервер принимает запрос, иденти­ фицирует пользователя и сам обращается к сервису.

* + 1. Разработка с помощью Sparkl

Sparkl сам является плагином для BI сервера и позволяет созда­ вать плагины и приложения для Pentaho. Приложение, созданное с

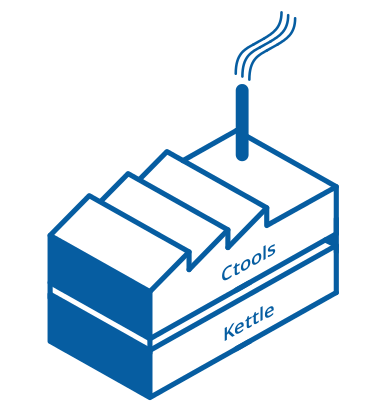


Рисунок 3.3. Структура Sparkl

помощью Sparkl, - это набор досок(dashboard), разработанных с помо­ щью инструментов CDE и Pentaho Data Integration(или Kettle) работ и трансформаций с определёнными характеристиками. Как видно на рисунке [3.3,](#_bookmark30) два слоя взаимодействуют друг с другом, предлагая ши­ рокий выбор возможностей для создания плагинов. Достоинством та­ кого способа является прямое взаимодействие с данными и файлами сервера, что обеспечивает высокую скорость работы при обработке больших объёмов данных. Недостатки Sparkl:

* + - 1. Требование знания созданий трансформацтй Kettle;
      2. Проблемы с выполнением Java кода из Kettle;
      3. Сложность создания и редактирования панелей интерфейсов.

## REST API сервис

[[8]](#_bookmark76) REST (сокр. от англ. Representational State Transfer — «пере­ дача состояния представления») — архитектурный стиль взаимодей­ ствия компонентов распределённого приложения в сети.

REST приложения используют HTTP запросы:

1. GET - возвращает указанный ресурс или коллекцию;
2. POST - создаёт ресурс на сервере. Также используется во всех остальных случаях;
3. PUT - обновляет указанный ресурс или коллекцию. Также ис­ пользуется в качестве аналога POST для создания;
4. DELETE - удаляет указанный ресурс.

[[9]P](#_bookmark77)entaho поддерживает Rest Api и предлагает его использовать для создания плагинов для сервера(рисунок [3.4).](#_bookmark32)

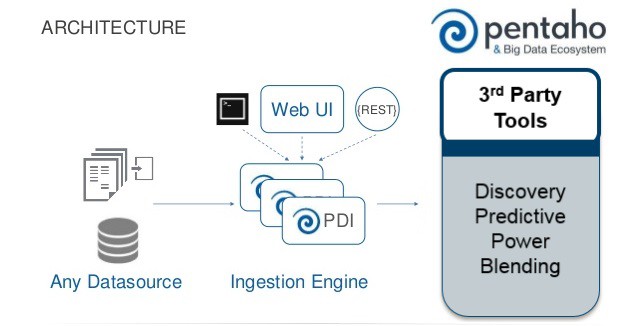


Рисунок 3.4. Схема платформы Pentaho Open BI

Pentaho API использует модель REST и технологию точек входа для доступа к данным. Точки входа позволяют легко соотнести URI и нужный HTTP метод. Иными словами, - это URL-ссылка, которую другая программа(приложение-клиент) будет использовать для обще­ ния с REST-сервисом.

Все точки входа - обычный набор данных. В качестве модели могут выступать различные медиа("MIME") типы в зависимости от назначения от точек входа. Данные описываются с помощью XML схемы, которая однозначно определяет XML отображения данных и помогает описать другие форматы, такие как JSON.

В работе в качестве способа анализа данных используется OLAP­ сервер. Структура базы данных в таком сервере описывается с помо­ щью формата XML, обладающего рядом достоинств:

1. Пространство имён позвляет обмениваться структурами;
2. Наличие стандартизированных способов выражения структуры документа: XML схема, DTD;
3. Использование стандартов синтаксического анализа: DOM, SAX и другие.

## Разбор XML конфигурации

Перед тем, как данные из XML становятся доступными, их необ­ ходимо загрузить в объекты. Для этих целей есть несколько видов инструментов анализаторов:

1. DOM - определяет древовидную структуру, содержащую элемен­ ты документа. Другими словами, создаёт объектно-ориентиро­ ванный граф - представление документа для навигации;
2. SAX - спецификация, определяющая основанный на событи­ ях анализ, когда по мере сканирования документа вызываются функции на основании определённых найденных частей данных;
3. JAXB - API, реализующее как привязку XML к Java объектам, так и наоборот. Является наиболее простым в реализации.

Так как для составления конфигурации базы данных необходимо иметь полную структуру, чтобы ссылаться на существующие табли­ цы, была выбрана технология DOM. Для составления же конечной Mondrian схемы будет использоваться JAXB, как наиболее гибкий и простой в использовании.

## Итоги

Разработка на языке Java проще и не требует дополнительных инструментов и знаний. В целях минимизации требуемых к установке средств платформы Pentaho для работы разрабатываемого плагина также не подходит использование SPARKL, для которого необходимо наличие CDE плагинов.

На рисунке [3.5](#_bookmark35) представлена схема архитектуры разрабатывае­ мого плагина. Через веб-интерфейс пользователь вызывает функции получения конфигурации из CRM-системы, с помощью разработанно­ го REST-сервиса на сервере Pentaho выполняется преобразование в Mondrian схему, и через API Pentaho загружается на сервер, подклю­ чая схему к указанному источнику данных(базе данных).

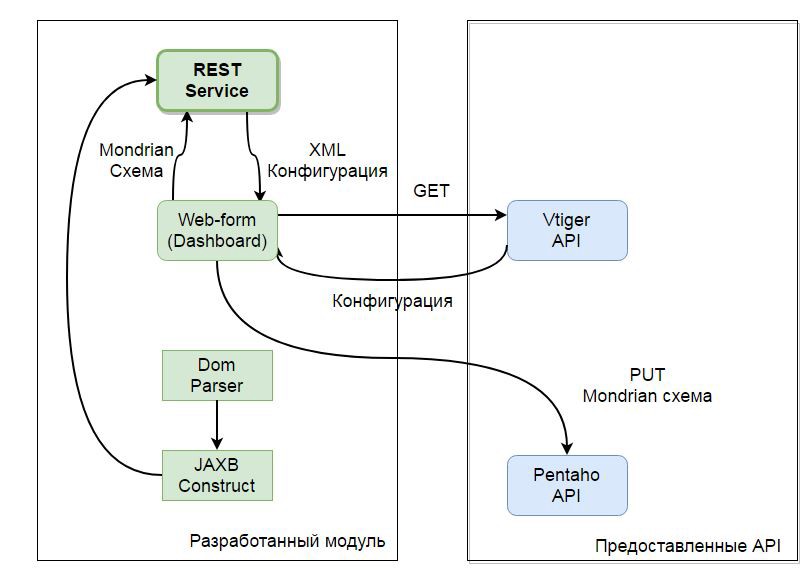


Рисунок 3.5. Архитектура разрабатываемого приложения

# Проектирование и разработка системы

Разработка плагина производилась на языке Java, с использова­ нием REST API функций в качестве точек входа. Для реализации веб-интерфейса использовался стандартный HTML 5 с Javascript.

## Функциональность

* + 1. Получение конфигурации

Для получения конфигурации Vtiger используется REST команда GET, обращающаяся к серверу Vtiger. Конфигурация предоставляет­ ся в формате XML и содержит описание модулей, их полей, связь между модулями и связи между таблицами в модулях. Окно ввода содержит поля:

* + - 1. VTiger URL - ссылка на расположение CRM;
      2. Username - имя пользователя для доступа к CRM;
      3. Password - пароль для доступа к CRM;
      4. Datasource - источник данных(база данных), к которому будет относится созданная схема.

В случае удачного подключения и обработки конфигурации, поль­ зователь видит оповещение в браузере с кодом ответа сервиса. В слу­ чае неудачного подключения, пользователю выводится соответствую­ щее сообщение об ошибке. Такими ошибками могут быть:

1. Неверный логин или пароль;
2. Недоступность сервера;
3. Неудачная попытка загрузки новой схемы;
4. Ошибка при синтаксического анализа конфигурации.
   * 1. Обработка XML-файла

Полученная от сервера Vtiger конфигурация приходит в формате XML. Для создания схемы необходимо сначала разобрать документ на классы, соответствующие элементам XML. Затем, имея полную кол­ лекцию модулей, необходимо установить между ними связи, как непо­ средственно между таблицами в одном модуле, так и между самими модулями. Затем, с помощью маршалинга Java-объектов в XML-схему получить необходимую Mondrian-схему. Для того чтобы обработать файл необходимо:

* + - 1. Загрузить дерево документа в память;
      2. Взять каждый элемент дерева;
      3. Создать на основе элемента соответствующий класс.

## Реализация

Главная часть приложения состоит из анализа и генерации Mondrian-схемы на основе конфигурации модулей информационной системы.

* + 1. Структура компонентов и описание

Основной набор компонентов относится к разбору XML конфи­ гурации, так как каждый элемент DOM дерева должен быть преоб­ разован в класс. В таблице [4.1](#_bookmark42) представлены основные элементы и соответствующие им элементы куба.

Таблица 4.1. Таблица соответствия элементов конфигурации и OLAP куба

|  |  |
| --- | --- |
| Название класса | Описание |
| PentahoMondrianService | Описание методов REST  сервиса |
| DatabaseConfiguration | Описание конфигурации |
| Schema | Отображение Mondrian схемы |
| XMLParser | Класс-синтаксический  анализатор, обрабатывающий XML |
| Классы-элементы XML | |

|  |  |
| --- | --- |
| FieldXML | Класс-описание поля Vtiger |
| ModuleXML | Класс-описание модуля Vtiger |
| VTigerXML | Класс-описание конфигурации  VTiger |
| Классы-элементы Mondrian | |
| Cube | Класс-описание куба  Mondrian, содержит коллекцию  элементов-измерений |
| Dimension | Класс-описание измерения,  содержит коллекцию иерархий |
| Hierarchy | Класс-описание иерархии,  содержит коллекцию уровней |
| Level | Класс-описание куба  Mondrian, содержит коллекцию  элементов-измерений |
| VirtualCube | Класс-описание набора кубов  Mondrian, содержит коллекцию измерений |
| TableAdapter | Класс-адаптер для  размещения специального тега таблицы |

* + 1. REST-сервис

В языке Java определена поддержка REST через Java Specification Request (JSR) 311. Эта специфик[ация([10])](#_bookmark78) называется JAX-RS. Для реализации веб-сервиса в данной работе использовалась библиотека Jersey. Jersey предоставляет классы для имплементации веб-сервера в виде сервлета.

Конфигурация сервиса находится в отдельном файле - plugin.spring.xml. Базовый адрес сервиса:

Листинг 4.1. "Базовый адрес сервиса"

1 h t t p ://your\_domain:port/display-name/url-pattern/path\_from\_rest\_class

Сервер анализирует входящий HTTP запрос и выбирает соответ­ ствующий ему класс и метод для ответа. Выбор осуществляется на

основе аннотаций в классах и методах. Рассмотрим используемые в работе аннотации, представленные в листинге [4.2:](#_bookmark44)

Таблица 4.2. Основные аннотации, используемые в работе

|  |  |
| --- | --- |
| Аннотация | Описание |
| @PATH(your\_path) | Устанавливает путь на конкатенацию  адреса домена и адреса метода |
| @GET | Метод является HTTP GET запросом |
| @Produces(MediaType) | Какой тип данных будет являться  ответом на запрос |
| @PathParam | Какие параметры будут внесены в  запрос |

В классе, представленном в листинге [4.2,](#_bookmark45) метод будет доступен по адресу base/mondrian. Он будет являться GET запросом и возвращать XML строку.

Листинг 4.2. "GET запрос сервиса"

1. @ G E T
2. @ P a t h ( " / m o n d r i a n " )
3. @ P r o d u c e s ( M e d i a T y p e . A P P L I C A T I O N \_ X M L )
4. public S t r i n g m o n d r i a n () throws P a r s e r C o n f i g u r a t i o n E x c e p t i o n , S A X E x c e p t i o n , I O E x c e p t i o n , J D O M E x c e p t i o n {

5 ...

6 }

Данный класс-ресурс будет доступен по адресу localhost:8080/pentaho/plugin/api/mondrian при запросе типа GET, и пользователю будет возвращаться сформированная Mondrian схема в виде XML строки. Далее происходит вызов функции преобразования Java-объектов в XML файл - генерация Mondrian схемы через вызов методов програмнной платформы JAXB. Подробно о работе синтак­ сического анализатора и преобразования конфигурации рассказано далее.

* + 1. Создание Mondrian схемы

Создание схемы делится на два этапа: синтаксический анализ кон­ фигурации системы и создание Mondrian схемы на основе обработки полученных объектов.

Разбор XML конфигурации

Реализация анализатора находится в классе XMLParser. Исполь­ зуется библиотека JDOM2, предоставляющая стандартизированные методы для разбора документа в формате XML в древовидную струк­ туру и обращения к элементам этой структуры. Для того, чтобы воспользоваться методами JDOM2, необходимо создать экземпляр SAXBuilder - строитель структуры документа. SAXBuilder использует стороннюю реализацию SAX(выбранный JAXP) или же можно ука­ зать, какой анализатор использовать вручную.

Рассмотрим часть работы синтаксического анализатора на приме­ ре получения элементов модулей. Каждый элемент содержит в себе также коллекцию элементов.

* + - 1. Получение конфигурации. Для этого создаётся подключение по HTTP URL ссылке, поданной в качестве аргумента к сервису.

Листинг 4.3. "Пример открытия соединения"

* + - * 1. //Создание ссылки из параметра
        2. URL u = new URL ( url ) ;
        3. //Открытие соединения
        4. U R L C o n n e c t i o n uc = u . o p e n C o n n e c t i o n () ;
        5. H t t p U R L C o n n e c t i o n c o n n e c t i o n = ( H t t p U R L C o n n e c t i o n ) uc ;
      1. Преобразование файла в документ JDOM. Необходимо создать объект анализатора, представленного JDOM2 и с помощью него преобразовать строку XML в древовидную структуру.

Листинг 4.4. "Инициализация анализатора"

1. S A X B u i l d e r b u i l d e r = new S A X B u i l d e r () ;
2. I n p u t S t r e a m in = c o n n e c t i o n . g e t I n p u t S t r e a m () ;
3. D o c u m e n t d o c u m e n t = b u i l d e r . b u i l d ( in ) ;
   * + 1. Анализ структуры. Для этого был реализован метод, представ­ ленный в листинге [4.5,](#_bookmark47) анализирующий поля документа и связы­ вающий между собой компоненты полученной конфигурации в схему. Обрабатывая в цикле все наборы элементов в структуре создаётся Mondrian схема.

Листинг 4.5. "Разбор документа"

1. E l e m e n t m o d u l e s E l e m e n t = c o n f i g u r a t i o n . g e t C h i l d ( R O O T \_ M O D U L E \_ E L E M E N T ) ;
2. List < E l e m e n t > m o d u l e s = m o d u l e s E l e m e n t . g e t C h i l d r e n () ; 3
3. f o r ( E l e m e n t m o d u l e E l e m e n t : m o d u l e s ) {
4. S y s t e m . out . p r i n t l n () ;
5. M o d u l e X M L m o d u l e X M L = p a r s e M o d u l e X M L ( m o d u l e E l e m e n t ) ;
6. v T i g e r x M L . a d d M o d u l e ( m o d u l e X M L ) ; 8

9 }

Каждому тегу конфигурации необходимо привести в соответ­ ствие элемент Mondrian схемы. Карта соответствия можно увидеть в таблице [4.3](#_bookmark48)

Таблица 4.3. Таблица соответствия элементов конфигурации и OLAP куба

|  |  |  |
| --- | --- | --- |
| Тег конфигурации | Описание тега | Элемент OLAP-куба |
| <module> | Тег, определяющий  границы модуля CRM-системы | Cube - Куб,  основной элемент схемы |
| <field> | Поле модуля | Dimension -  Измерение, по которому может производится срез. |
| <tablename> | Таблица, в которой хранится поле | Table - таблица, в  которой находится измерение |
| <columnname> | Имя колонки в таблице | Level column -  колонка в иерархии измерения |
| <uitype> | Тип поля | Указание, является  ли поле внешним ключом к другой таблице |
| <relatedlists> | Модули, связанные с выбранным | Virtual Cube -  элемент, связывающий несколько кубов |
| <entityidcolumn> | Тег, указывающий  на главный ключ сущности модуля | Hierarchy primary  key - главный ключ таблицы измерения |
| <translated> | Перевод наименования сущности на русский | Caption -  наименование, показываемое в графическом интерфейсе |

Каждому элементу Mondrian схемы создана функция генера­ ции дочерних элементов. Так, например, элемент “Куб” генерирует несколько элементов “Измерение” и так далее. Рассмотрим это на при­ мере “Куба”.

Генерация схемы

Листинг 4.6. "Компонент генерации куба"

1. @ X m l A c c e s s o r T y p e ( X m l A c c e s s T y p e . F I E L D )
2. @ G e t t e r
3. @ S e t t e r
4. public c l a s s C u b e {
5. @ X m l A t t r i b u t e
6. private S t r i n g n a m e ;
7. @ X m l A t t r i b u t e
8. private S t r i n g d e f a u l t M e a s u r e ;
9. @ X m l A t t r i b u t e
10. private S t r i n g c a p t i o n ;
11. @ X m l J a v a T y p e A d a p t e r ( T a b l e A d a p t e r . c l a s s )
12. private S t r i n g t a b l e ;
13. @ X m l E l e m e n t ( n a m e = " D i m e n s i o n " , t y p e = D i m e n s i o n . c l a s s )
14. private List < D i m e n s i o n > d i m e n s i o n ; 15
15. public C u b e g e n e r a t e ( M o d u l e X M L m o d u l e ) {
16. f o r ( F i e l d X M L f i e l d X M L : m o d u l e . g e t F i e l d X M L L i s t () ) {
17. D i m e n s i o n d i m e n s i o n G e n = new D i m e n s i o n ( f i e l d X M L . g e t F i e l d N a m e () ) ;
18. d i m e n s i o n G e n = d i m e n s i o n G e n . g e n e r a t e ( f i e l d X M L ) ;
19. d i m e n s i o n G e n . s e t F o r e i g n K e y ( f i e l d X M L . g e t C o l u m n N a m e () ) ;
20. d i m e n s i o n . add ( d i m e n s i o n G e n ) ; 22 }

23 return t h i s ; 24 }

25 }

В строках 17-21 листинга [4.6](#_bookmark49) в цикле рассматриваются получен­ ные модули. Для каждого поля модуля необходимо создать измерение. Каждое измерение - это срез многомерного куба, на основе которо­ го будет происходить фильтрация данных. Каждый дочерний класс имеет аннотации элементов и атрибутов. Элементы - поля, которые станут тегами в конечной Mondrian схеме, а аттрибуты - это парамет­ ры этих тегов. В листинге [4.7](#_bookmark50) представлен пример сгенерированного куба.

Листинг 4.7. "Пример сгенерированного куба"

1. < C u b e n a m e = " A s s e t s " >
2. < T a b l e n a m e = " v t i g e r \_ a s s e t s " / >
3. < D i m e n s i o n n a m e = " a s s e t \_ n o " f o r e i g n K e y = " a s s e t \_ n o " t y p e = " S t a n d a r d D i m e n s i o n " >
4. < t a b l e n a m e = " v t i g e r \_ a s s e t s " / >
5. < H i e r a r c h y n a m e = " a s s e t \_ n o " v i s i b l e = " t r u e " h a s A l l = " t r u e " >
6. < L e v e l n a m e = " a s s e t \_ n o " c o l u m n = " a s s e t \_ n o " v i s i b l e = " t r u e " u n i q u e M e m b e r s = " f a l s e " / >
7. </ H i e r a r c h y >
8. < D i m e n s i o n / >
9. < C u b e / >

В строке 3 примера “Dimension” является тегом, а “name”, “foreignKey”, “type” - атрибутами.

Основной задачей также является возможность объединения нескольких кубов для перекрёстного анализа модулей. Для этой функ­ ции в Mondrian существуют виртуальные кубы [[6]](#_bookmark74).

* + 1. Графический интерфейс

Проектирование графического дизайна не было главной задачей разработки. Главное в интерфейсе для плагина - интуитивность и ско­ рость работы.

Для реализации было решено использовать технологии HTML и Javascript. На рисунке [4.1](#_bookmark52) показано главное окно, которое открывается пользователь при выборе владки плагина.

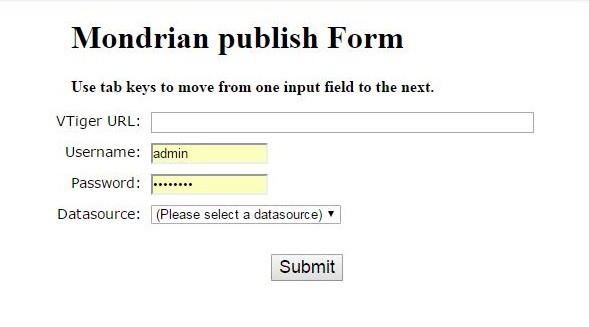


Рисунок 4.1. Окно интерфейса плагина

Страница выглядит как форма для заполнения с кнопкой под­ тверждения. За дизайн отвечает css код, находящийся в отдельном файле.

Важной частью приложения также является Javascript код, скры­ тый за HTML формой. Реализацию функций можно найти в прило­ жении [4.1.](#_bookmark52) В таблице указаны основные функции.

Таблица 4.4. Основные функции Javascript-части программы

|  |  |
| --- | --- |
| Функция | Комментарий |
| getSources | Функция получения  XML-конфигурации от Vtiger API |
| getMondrian | Функция обращения к  REST-сервису, обрабатывающему конфигурацию |
| publish | Функция загрузки  сгенерированной схемы на сервер Pentaho |
| getAnalysis | Функция получения  конфигурации с сервера Pentaho по идентификатору |

## Итоги

Было разработано приложение-плагин для Pentaho BI server, в частности, для Pentaho User Console. Данное приложение реализует интеграцию CRM-системы в платформу Pentaho посредством анализа и преобразования XML-конфигурации базы данных системы в сред­ ство анализа бизнес-аналитики - Mondrian схему.

# 5. Тестирование системы

Тестирование было разделено на backend(внутренняя реализа­ ция веб-сервиса и анализатора) и frontend(в данном случае Javascript часть) части.

## Тестирование внутренней реализации

Тесты инструментов разбора XML могут быть использованы для проверки работоспособности, надёжности и корректности работы ана­ лизатора. Такие тесты обычно содержат шаблон, который может быть отредактирован для использования с любым контекстом. В тестиро­ вании анализатора сравниваются две строки - идеальная и сгенериро­ ванная. Реализован простейший алгоритм:

1. Создана модель данных;
2. Сгенерирована XML строка в соответствии с моделью;
3. Производено сравнение строки с эталонной, указанной вручную. В данной работе используется библиотека XmlUnit. Она позволяет

сравнивать два XML файла, игнорировать лишние пробелы и переста­ новку элементов и другие, не влияющие на процесс анализа, неточно­ сти.

[5.1:](#_bookmark56)

Для сравнения существует класс Diff, представленный в листинге

Листинг 5.1. "Класс сравнения двух документов"

1. D i f f d i f f = D i f f B u i l d e r
2. . c o m p a r e ( e x p e c t X m l )
3. . w i t h T e s t ( m a r s h a l e d X m l )
4. //Ignore element order
5. . w i t h N o d e M a t c h e r (new D e f a u l t N o d e M a t c h e r ( E l e m e n t S e l e c t o r s . b y N a m e )

)

1. . i g n o r e W h i t e s p a c e ()
2. . i g n o r e C o m m e n t s ()
3. . c h e c k F o r S i m i l a r ()
4. . b u i l d ()
5. a s s e r t ! d i f f . h a s D i f f e r e n c e s ()

## Тестирование REST-сервиса

Jersey, используемый в проекте для тестирования предоставля­ ет минимальный веб-сервер для развёртывания пользователем. Для тестирования разработанного сервиса необходимо проверить работу точки входа для получения сгенерированный схемы. Так как в методе генерации схемы используется сторонний сервис, а именно, установле­ ние соединения с сервисом Vtiger, необходимо использовать заглушку (“Mock”).

Листинг 5.2. "Пример описания заглушки"

1. @ M o c k
2. private B a c k e n d S e r v i c e b a c k e n d S e r v i c e M o c k ;

Чтобы не оставлять сервер открытым после выполнения теста, можно использовать аннотацию @AfterClass, выполняющую часть ко­ да после исполнения всех запланированных в классе тестов.

Листинг 5.3. "Использования аннотации AfterClass"

1. @ A f t e r C l a s s
2. public s t a t i c void a f t e r C l a s s () throws E x c e p t i o n {
3. s e r v e r . c l o s e () ; 4 }

Для верификации результата теста нужно сравнить код ответа от сервера с ожидаемым(200 - OK), указанным в спецификации [[11]](#_bookmark79)

Листинг 5.4. "Сравнение ответного кода с ожидаемым положительным"

1 a s s e r t E q u a l s ( R e s p o n s e . S t a t u s . OK . g e t S t a t u s C o d e () , r e s p o n s e . g e t S t a t u s () ) ;

Были протестированы варианты кодов ответа: позитивного(200), негативного(505) и случай недоступности сервиса(404).

## Тестирование веб-интерфейса

Тестирование интерфейса плагина, в данном случае, представля­ ет из себя тестирование Javascript кода, отвечающего за ввод данных в форму на странице плагина. Для тестирования использовалась биб­ лиотека Jasmine, предоставляющая возможность создавать “заглуш­ ки” для элементов:

Листинг 5.5. "Пример использования библиотеки Jasmine"

1. var s u b m i t ;
2. b e f o r e E a c h () {

3

1. s u b m i t = j a s m i n e . c r e a t e S p y () ;
2. s p y O n ( d o c u m e n t ’ g e t E l e m e n t B y I d ’ ) . a n d R e t u r n ({ s u b m i t : s u b m i t }) 6 }

Тесты для интерфейса были включены в процесс сборки плагина и выполняются автоматически.

## Итоги

При написании любого приложения тестирование является важ­ ной частью разработки. Разделённое на две части: FrontEnd и BackEnd тестирование помогает обнаружить ошибки как в серверной части, в данном случае, парсере документа, так и в части интерфейса, в данном случае являющимся Javascript кодом.

Тесты для серверной части помогают:

1. Проверить корректность сгенерированного XML файла по схеме или другому грамматическому источнику;
2. Обнаружить пустые или отсутствующие элементы, которые мо­ гут нарушить порядок работы синтаксического анализатора.

В процесс тестирования было создано 4 Unit-теста, а также несколько Javascript-тестов. Как показано на рисунке [5.1,](#_bookmark60) все Unit­ тесты закончились удачно.

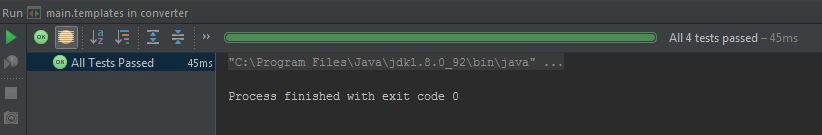


Рисунок 5.1. Результаты тестирования

Данные полученные в тестировании свидетельствуют о коррект­ ной работе сервиса. При некоректной поданной схеме конвертация не происходит и пользователю возвращается корректная ошибка. При от­ сутствии доступа к серверу Vtiger CRM для получения конфигурации выводится соответствующая ошибка.

# 6. Анализ разработанной системы

В результате разработки была реализована возможность интегра­ ции хранилища CRM-системы в платформу бизнес-аналитики.

Интеграция происходит с помощью веб-сервиса, находящегося на сервере Pentaho. С помощью приложения возможно преобразовать имеющуюся структуру базы данных CRM в многомерную структуру, необходимую для OLAP-аналитики.

В результате исследования было выяснено, что аналитические ин­ струменты CRM Vtiger недостаточно мощны для комплексного анали­ за, а также не позволяют использовать несколько источников данных. Реализованное приложение даёт возможность построения диаграмм, аналитики и прогнозирования в Pentaho BI с помощью инструментов платформы.

Сравним отчёты, сформированные в Pentaho и созданные с по­ мощью встроенных инструментов Vtiger. Создадим отчёт, отобража­ ющий прибыль по контрагентам. На графике необходимо отобразить сумму сделок у каждого контрагента.

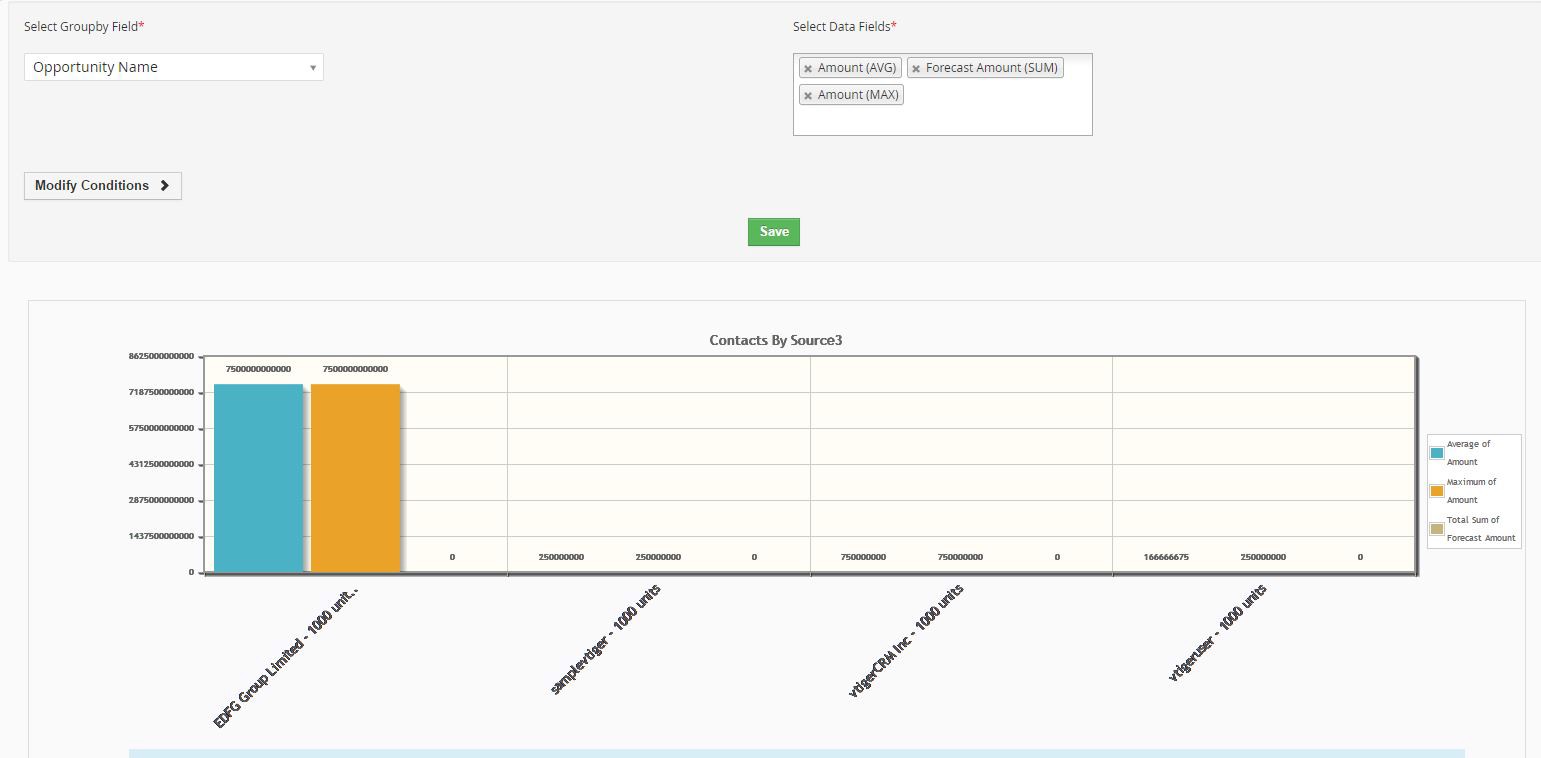


Рисунок 6.1. Пример отчёта в Vtiger модуле отчётов

Как видно на рисунке [6.1,](#_bookmark62) в модуле отчётов Vtiger доступно толь­ ко три параметра для одного графика. Для сравнения построим похо­ жий отчёт в Pentaho, а конкретно, в инструменте Saiku. На рисунке

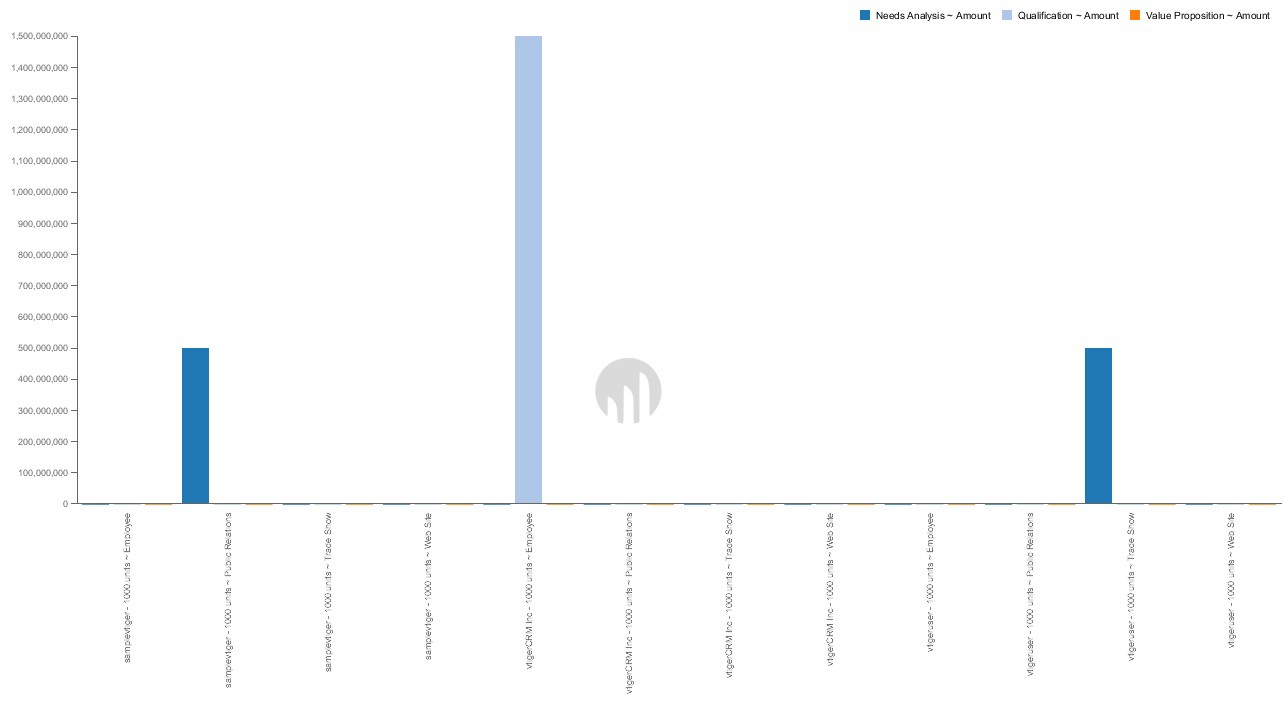


Рисунок 6.2. Пример отчёта в Saiku в системе Pentaho

[6.2](#_bookmark63) представлен график в Pentaho BI. OLAP-аналитика позволяет до­ бавлять любое количество параметров к отчёту, углублиться в какой­ либо из пунктов в отчёте, предоставив более подробную статистику. Также, как видно на рисунке [6.3,](#_bookmark64) в Pentaho можно легко переключать­ ся между видами отчётов, получая при этом возможность просматри­ вать нужную информацию с разных сторон.

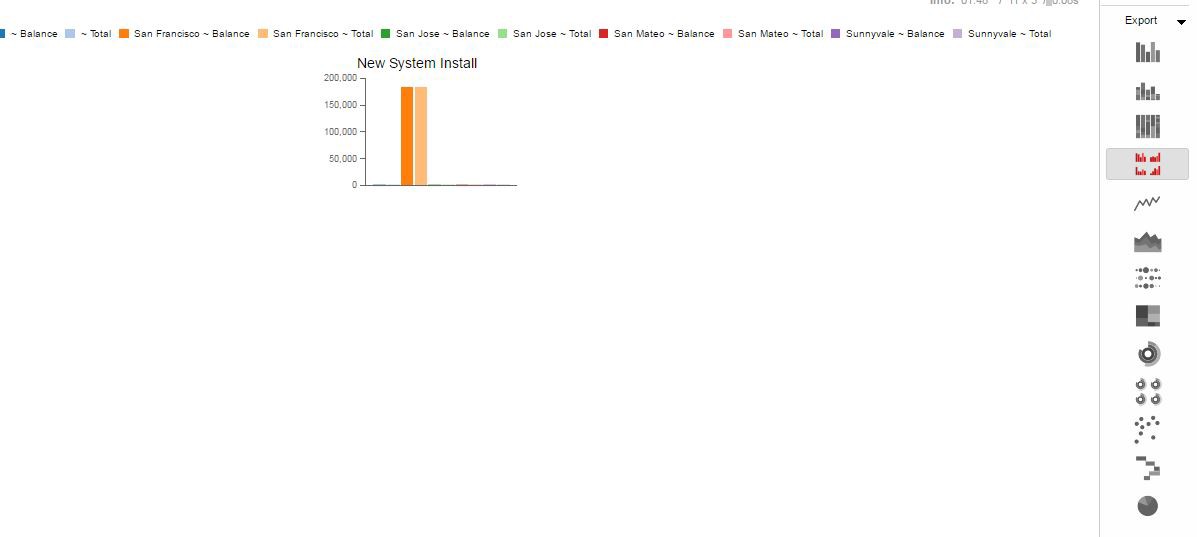


Рисунок 6.3. Различные виды отчётов Pentaho

Попробуем создать более сложный отчёт. В Vtiger отобразим ста­ тистику продаж по адресам штатов и проданным товарам. На рисунке

[6.4](#_bookmark65) представлен отчёт в виде таблицы.

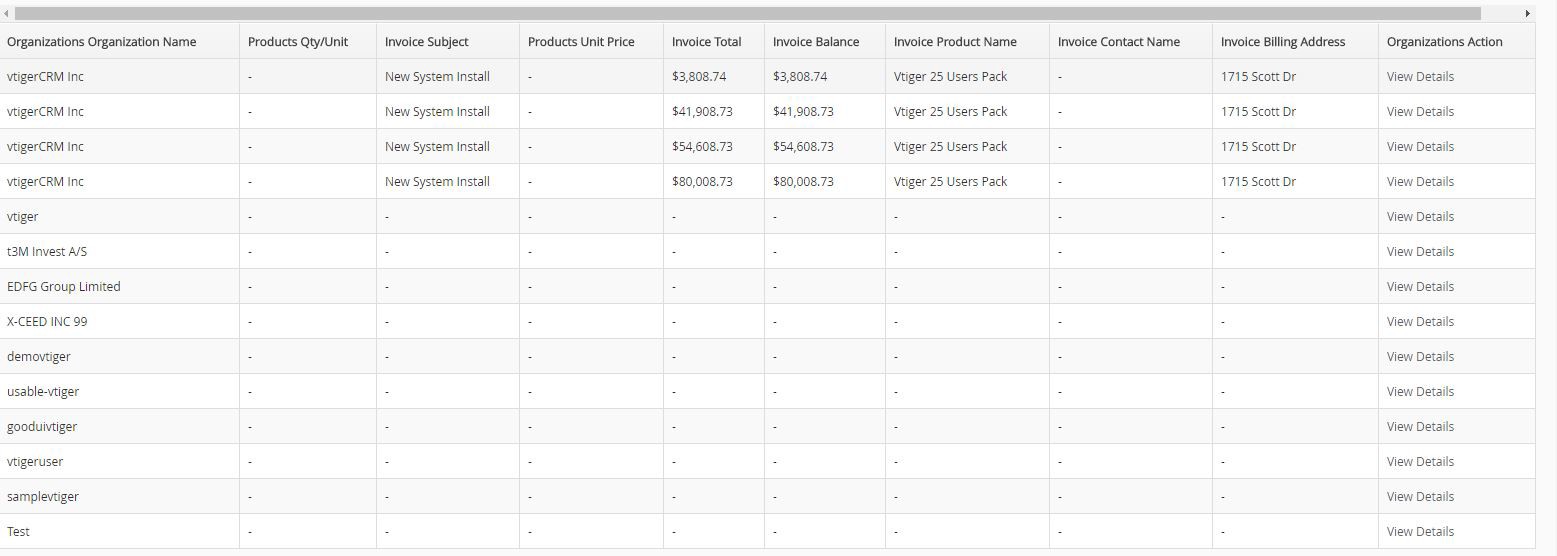


Рисунок 6.4. Статистика продаж в Vtiger

Для полноценного анализа специалистам часто приходится со­ ставлять несколько отчётов и использовать их одновременно, так как ограничение в два прикреплённых модуля к одному основному не поз­ воляет строить подробные отчёты. С помощью объединения модулей в виртуальные кубы это ограничение легко обойти. На рисунке [6.5](#_bookmark66) пред­ ставлен отчёт в Saiku инструменте, отображающий те же данные, что и в Vtiger.

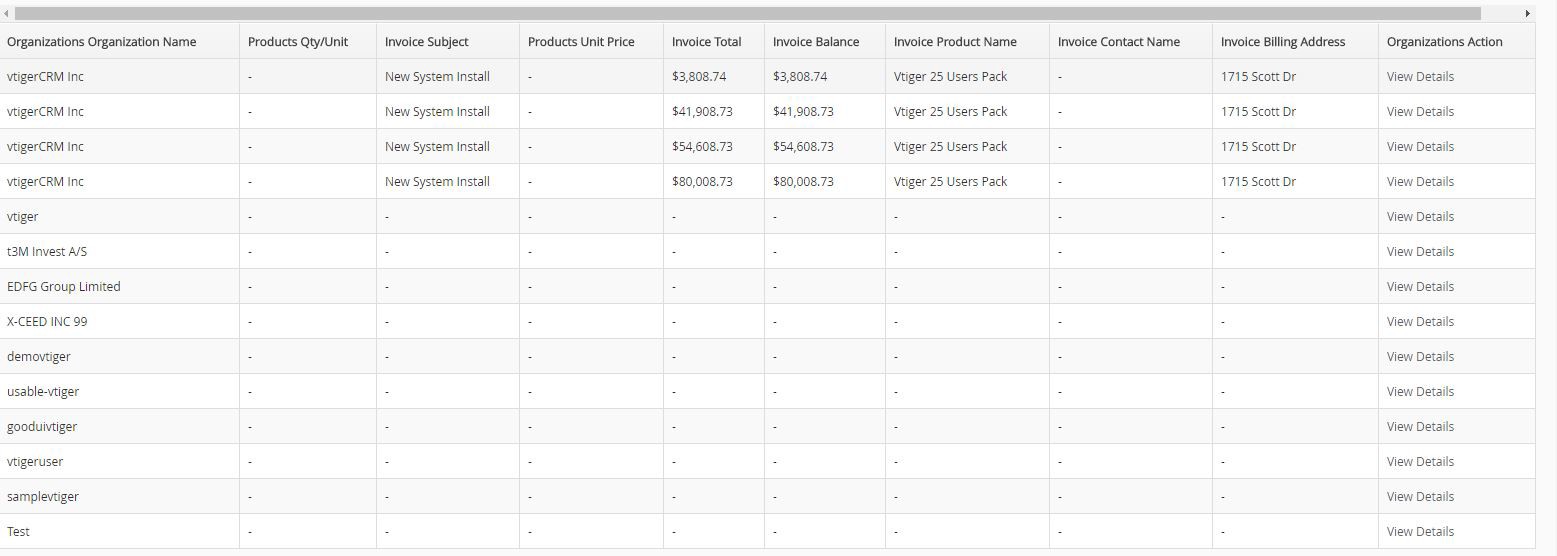


Рисунок 6.5. Подробный отчёт в Pentaho BI

Таким образом, можно сказать, что задача, поставленная в нача­ ле работы была выполнена. С помощью разработанного плагина ана­

литики могут составлять подробные отчёты без знания CRM-системы и языка запросов.

# ЗАКЛЮЧЕНИЕ

В процессе написания работы, были исследованы информацион­ ные системы, хранящие и позволяющие анализировать информацию связанную с бизнес-процессами и менеджментом.

Исследование показало, что компаниям выгодно использовать связку информационная система - платформа бизнес-аналитики. Та­ кая связка расширяет возможности анализа, предлагая мощность ин­ струментов BI платформы в обработке данных из хранилища CRM.

В данной работе исследовалась возможность интеграции Vtiger CRM - системы с открытым кодом, являющейся одним из лидеров на рынке и BI платформы Pentaho BI Suite. Данная платформа имеет возможность анализа с помощью OLAP-кубов - многомерных таблиц.

Для интеграции был написан плагин для Pentaho, преобразую­ щий конфигурацию CRM-системы в Mondrian схему. Разработанный плагин анализирует входящие в систему модули и на их основе состав­ ляет кубы, принимая за измерения данные, фигурирующие в модулях. В качестве факт-таблицы берётся главная таблица в модуле.

В процессе разработки возникли определённые трудности. Так, для правильного составления структуры хранилища данных CRM не хватало информации о связи некоторых таблиц. Для решения про­ блемы был разработан дополнительный тэг в конфигурации, прямо указывающий на колонки таблиц, через которые они имеют связь.

Разработанный плагин действительно расширяет возможности бизнес-аналитики по сравнению с встроенными в CRM-систему ин­ струментами. Также исчезает необходимость в найме специалиста, разрабатывающего и поддерживающего создание ручных отчётов, на­ пример, с помощью SQL. При присоединении платформы появляются возможности использовать гибкие инструменты аналитики и прогно­ зирования, такие как Saiku, JPivot, FusionCharts и др.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Vtiger Documentation [Электронный ресурс], Vtiger. — URL: [http:](http://community.vtiger.com/help/)

[//community.vtiger.com/help/](http://community.vtiger.com/help/) (дата обращения: 11.06.2017).

1. SalesPlatform Wiki [Электронный ресурс], SalesPlatform. — URL: <https://goo.gl/EFSpny>(дата обращения: 11.06.2017).
2. Understanding BI analytics tools and their benefits [Электронный ресурс], TechTarget. — URL: [http:](http://searchbusinessanalytics.techtarget.com/feature/Understanding-BI-analytics-tools-and-their-benefits)

[//searchbusinessanalytics.techtarget.com/feature/](http://searchbusinessanalytics.techtarget.com/feature/Understanding-BI-analytics-tools-and-their-benefits) [Understanding-BI-analytics-tools-and-their-benefits](http://searchbusinessanalytics.techtarget.com/feature/Understanding-BI-analytics-tools-and-their-benefits) (дата обращения: 11.06.2017).

1. CRM Statistics [Электронный ресурс], SmallBizCRM.com. —

URL: [http://www.smallbizcrm.com/crm-reading-lounge/](http://www.smallbizcrm.com/crm-reading-lounge/crm-market-statistics/) [crm-market-statistics/](http://www.smallbizcrm.com/crm-reading-lounge/crm-market-statistics/) (дата обращения: 11.06.2017).

1. Pentaho Reporting [Электронный ресурс], Rittan Holdings Ltd. — URL: <http://community.pentaho.com/projects/reporting/>(да­ та обращения: 11.06.2017).
2. OLAP-engine Documentation [Электронный ресурс], Pentaho Corporation. — URL: [http://mondrian.pentaho.com/](http://mondrian.pentaho.com/documentation/schema.php) [documentation/schema.php](http://mondrian.pentaho.com/documentation/schema.php) (дата обращения: 11.06.2017).
3. API Documentation for BA Platform [Электронный ресурс], Pentaho Corporation. — URL: [https://help.pentaho.com/](https://help.pentaho.com/Documentation/6.0/0R0/070/010) [Documentation/6.0/0R0/070/010](https://help.pentaho.com/Documentation/6.0/0R0/070/010) (дата обращения: 11.06.2017).
4. REST API Documentation [Электронный ресурс], WP REST API. — URL: <http://v2.wp-api.org/>(дата обращения: 11.06.2017).
5. Pentaho Presentation [Электронный ресурс], By Ricardo Pires

- Partner and BI Lead. — URL: [https://www.slideshare.net/](https://www.slideshare.net/XpandIT/real-use-cases-pentaho-big-data-ecosystem) [XpandIT/real-use-cases-pentaho-big-data-ecosystem](https://www.slideshare.net/XpandIT/real-use-cases-pentaho-big-data-ecosystem) (дата об­ ращения: 11.06.2017).

1. Интернет-ресурс [Электронный ресурс], Oracle Corporation. — URL: <https://jcp.org/en/jsr/overview>(дата обращения: 11.06.2017).
2. Интернет-ресурс [Электронный ресурс], The Internet Society (1999). — URL: [https://www.w3.org/Protocols/rfc2616/](https://www.w3.org/Protocols/rfc2616/rfc2616.txt) [rfc2616.txt](https://www.w3.org/Protocols/rfc2616/rfc2616.txt) (дата обращения: 11.06.2017).

Автогенерирование описания модели данных для интеграции системы бизнес-аналитики

Описание программы.

1 лист

## А.1. Общие сведения

Разработанное приложение-плагин реализует интеграцию CRM­ системы и платформы бизнес-аналитики Pentaho BI. Это позволяет анализировать данные CRM-системы с помощью инструментов ана­ литики, выполнять прогнозирование и строить подробные отчёты.

## А.2. Используемые технические средства

Объектно-ориентированный язык программирования Java, Плат­ форма BI с открытым исходным кодом Pentaho BI Suit, CRM-система Vtiger.

## А.3. Входные данные

Входными данными являются URL CRM-системы:

* 1. Адрес сервера;
  2. Порт сервера.

А также данные пользователя, необходимые для аутентификации:

1. Логин пользователя;
2. Пароль пользователя.

# ЛИСТИНГИ

Листинг А.1. XMLParser.java

1 package org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s ; 2

1. import org . x e r o c r y . m o n d r i a n . D a t a b a s e C o n f i g u r a t i o n ;
2. import org . j d o m 2 . D o c u m e n t ;
3. import org . j d o m 2 . E l e m e n t ;
4. import org . j d o m 2 . J D O M E x c e p t i o n ;
5. import org . j d o m 2 . i n p u t . S A X B u i l d e r ;
6. import org . xml . sax . S A X E x c e p t i o n ; 9
7. import j a v a x . xml . p a r s e r s . P a r s e r C o n f i g u r a t i o n E x c e p t i o n ;
8. import j a v a . io . I O E x c e p t i o n ;
9. import j a v a . io . I n p u t S t r e a m ;
10. import j a v a . net . H t t p U R L C o n n e c t i o n ;
11. import j a v a . net . URL ;
12. import j a v a . net . U R L C o n n e c t i o n ;
13. import j a v a . u t i l . H a s h M a p ;
14. import j a v a . u t i l . L i s t ;
15. import j a v a . u t i l . l o g g i n g . L o g g e r ; 19

20 public c l a s s X M L P a r s e r {

21

22 private s t a t i c L o g g e r log = L o g g e r . g e t L o g g e r ( D a t a b a s e C o n f i g u r a t i o n . c l a s s . g e t N a m e () ) ;

23

1. private s t a t i c f i n a l S t r i n g R O O T \_ C O N F I G U R A T I O N \_ E L E M E N T = " c o n f i g u r a t i o n s " ;
2. private s t a t i c f i n a l S t r i n g R O O T \_ M O D U L E \_ E L E M E N T = " m o d u l e s " ;
3. private s t a t i c f i n a l S t r i n g R O O T \_ B L O C K \_ E L E M E N T = " b l o c k s " ;
4. private s t a t i c f i n a l S t r i n g R O O T \_ F I E L D \_ E L E M E N T = " f i e l d s " ;
5. private s t a t i c f i n a l S t r i n g R O O T \_ R E L A T E D \_ M O D U L E \_ E L E M E N T = " r e l a t e d m o d u l e s " ;
6. private s t a t i c f i n a l S t r i n g R O O T \_ R E L A T E D \_ L I S T \_ E L E M E N T = " r e l a t e d l i s t s " ;
7. private s t a t i c f i n a l S t r i n g R O O T \_ M O D S T R I N G S \_ E L E M E N T = " m o d s t r i n g s " ;
8. private s t a t i c f i n a l S t r i n g D E S C R I P T I O N \_ E L E M E N T = " d e s c r i p t i o n " ; 32
9. private s t a t i c f i n a l S t r i n g C O N F I G U R A T I O N \_ E L E M E N T = "

c o n f i g u r a t i o n " ;

1. private s t a t i c f i n a l S t r i n g T A B L E N A M E \_ E L E M E N T = " t a b l e n a m e " ;
2. private s t a t i c f i n a l S t r i n g F I E L D N A M E \_ E L E M E N T = " f i e l d n a m e " ;
3. private s t a t i c f i n a l S t r i n g T R A N S L A T E D \_ E L E M E N T = " o r i g i n a l " ;
4. private s t a t i c f i n a l S t r i n g O R I G I N A L \_ E L E M E N T = " o r i g i n a l " ;
5. private s t a t i c f i n a l S t r i n g C O L U M N N A M E \_ E L E M E N T = " c o l u m n n a m e " ;
6. private s t a t i c f i n a l S t r i n g C O L U M T Y P E \_ E L E M E N T = " c o l u m n t y p e " ;
7. private s t a t i c f i n a l S t r i n g U I \_ T Y P E \_ E L E M E N T = " u i t y p e " ;
8. private s t a t i c f i n a l S t r i n g R E L A T E D \_ M O D U L E \_ E L E M E N T = " r e l a t e d m o d u l e " ;
9. private s t a t i c f i n a l S t r i n g M O D U L E \_ N A M E \_ E L E M E N T = " l a b e l " ;
10. private s t a t i c f i n a l S t r i n g F I E L D \_ L A B E L \_ E L E M E N T = " f i e l d l a b e l " ;
11. private s t a t i c f i n a l S t r i n g R O O T \_ E N T I T Y \_ I D \_ E L E M E N T = " e n t i t y i d e n t i f i e r " ;
12. private s t a t i c f i n a l S t r i n g E N T I T Y \_ I D \_ E L E M E N T = " e n t i t y i d c o l u m n "

;

46

1. private s t a t i c f i n a l S t r i n g C R M \_ T A B L E \_ F I E L D = " v t i g e r \_ c r m e n t i t y "

;

1. private s t a t i c f i n a l S t r i n g T R A N S L A T I O N \_ E L E M E N T = " t r a n s l a t i o n " ; 49

50 private V T i g e r X M L v T i g e r x M L = new V T i g e r X M L () ;

51

52 private S t r i n g url ; 53

1. public X M L P a r s e r ( S t r i n g url ) {
2. t h i s . url = url ; 56 }

57

58 /*\*\**

1. *\** Метод для получения конфигурации чере з HTTP запро с и формиров ания схемы
2. *\**
3. *\** @return Считанная в Java*−*объ екты конфигурация
4. *\** @throws JDOMException
5. *\** @throws IOException
6. *\** @throws ParserConfigurationException
7. *\** @throws SAXException
8. *\**/
9. public V T i g e r X M L r e a d X M L () throws J D O M E x c e p t i o n , I O E x c e p t i o n , P a r s e r C o n f i g u r a t i o n E x c e p t i o n , S A X E x c e p t i o n {
10. URL u = new URL ( url ) ;
11. log . i n f o ( " T r y i n g to c o n n e c t to url " + url ) ;
12. U R L C o n n e c t i o n uc = u . o p e n C o n n e c t i o n () ;
13. H t t p U R L C o n n e c t i o n c o n n e c t i o n = ( H t t p U R L C o n n e c t i o n ) uc ;
14. S A X B u i l d e r b u i l d e r = new S A X B u i l d e r () ;
15. I n p u t S t r e a m in = c o n n e c t i o n . g e t I n p u t S t r e a m () ;
16. log . i n f o ( " I n p u t s t r e a m get try : " + in . a v a i l a b l e () ) ;
17. D o c u m e n t d o c u m e n t = b u i l d e r . b u i l d ( in ) ;
18. in . c l o s e () ;
19. c o n n e c t i o n . d i s c o n n e c t () ; 78
20. E l e m e n t r o o t C o n f i g u r a t i o n = d o c u m e n t . g e t R o o t E l e m e n t () ;
21. a s s e r t r o o t C o n f i g u r a t i o n . g e t N a m e () . e q u a l s (

R O O T \_ C O N F I G U R A T I O N \_ E L E M E N T ) ;

1. E l e m e n t c o n f i g u r a t i o n = r o o t C o n f i g u r a t i o n . g e t C h i l d ( C O N F I G U R A T I O N \_ E L E M E N T ) ;
2. a s s e r t c o n f i g u r a t i o n != n u l l ; 83
3. E l e m e n t d e s c r i p t i o n = c o n f i g u r a t i o n . g e t C h i l d (

D E S C R I P T I O N \_ E L E M E N T ) ;

1. S t r i n g [] tmp = d e s c r i p t i o n . g e t V a l u e () . s p l i t ( " ; " ) ;
2. H a s h M a p < String , String > s o u r c e I n f o = new H a s h M a p < >() ;
3. f o r ( S t r i n g var : tmp ) {
4. S t r i n g [] k e y V a l = var . s p l i t ( " = " ) ;
5. s o u r c e I n f o . put ( k e y V a l [ 0 ] . t r i m () , k e y V a l [ 1 ] . t r i m () ) ; 90 }
6. v T i g e r x M L . g e t S o u r c e I n f o () . p u t A l l ( s o u r c e I n f o ) ;
7. p a r s e X M L ( c o n f i g u r a t i o n ) ;

93

94 return v T i g e r x M L ; 95 }

96

1. private void p a r s e X M L ( E l e m e n t c o n f i g u r a t i o n ) {
2. E l e m e n t m o d u l e s E l e m e n t = c o n f i g u r a t i o n . g e t C h i l d ( R O O T \_ M O D U L E \_ E L E M E N T ) ;
3. List < E l e m e n t > m o d u l e s = m o d u l e s E l e m e n t . g e t C h i l d r e n () ; 100
4. f o r ( E l e m e n t m o d u l e E l e m e n t : m o d u l e s ) {
5. S y s t e m . out . p r i n t l n () ;
6. M o d u l e X M L m o d u l e X M L = p a r s e M o d u l e X M L ( m o d u l e E l e m e n t ) ;
7. v T i g e r x M L . a d d M o d u l e ( m o d u l e X M L ) ;
8. }

106

107 }

108

1. /*\*\**
2. *\** Метод разбора каждо г о модуля
3. *\**
4. *\** @param module
5. *\** @return
6. *\**/
7. private M o d u l e X M L p a r s e M o d u l e X M L ( E l e m e n t m o d u l e ) {
8. M o d u l e X M L m o d u l e X M L = new M o d u l e X M L () ;
9. List < E l e m e n t > c h i l d N o d e s = m o d u l e . g e t C h i l d r e n () ;
10. E l e m e n t p1 = m o d u l e . g e t P a r e n t E l e m e n t () ;
11. E l e m e n t p2 = p1 . g e t P a r e n t E l e m e n t () ;
12. E l e m e n t p3 = p2 . g e t C h i l d ( T R A N S L A T I O N \_ E L E M E N T ) ;
13. E l e m e n t m a i n T r a n s l a t i o n s = p3 . g e t C h i l d ( R O O T \_ M O D S T R I N G S \_ E L E M E N T ) ;

122

1. /*\**
2. Получение названий для элементов
3. *\**/
4. E l e m e n t t r a n s N o d e = m o d u l e . g e t C h i l d ( T R A N S L A T I O N \_ E L E M E N T ) ;
5. E l e m e n t m o d s t r i n g s N o d e = t r a n s N o d e . g e t C h i l d (

R O O T \_ M O D S T R I N G S \_ E L E M E N T ) ;

128

1. /*\**
2. Итерация по элементам конфигурации 131 *\**/
3. f o r ( E l e m e n t c h i l d N o d e : c h i l d N o d e s ) {
4. i f ( c h i l d N o d e . g e t N a m e () . e q u a l s ( M O D U L E \_ N A M E \_ E L E M E N T ) ) {
5. S t r i n g l a b e l ;
6. i f (( l a b e l = f i n d T r a n s l a t i o n ( c h i l d N o d e . g e t V a l u e () , m a i n T r a n s l a t i o n s ) ) != n u l l ) {
7. m o d u l e X M L . s e t M o d u l e C a p t i o n ( l a b e l ) ;
8. } e l s e m o d u l e X M L . s e t M o d u l e C a p t i o n ( c h i l d N o d e . g e t V a l u e () ) ;
9. m o d u l e X M L . s e t M o d u l e N a m e ( c h i l d N o d e . g e t V a l u e () ) ;
10. continue ;
11. }
12. i f ( c h i l d N o d e . g e t N a m e () . e q u a l s ( R O O T \_ B L O C K \_ E L E M E N T ) ) {
13. List < E l e m e n t > b l o c k s = c h i l d N o d e . g e t C h i l d r e n () ;
14. f o r ( E l e m e n t b l o c k : b l o c k s ) {
15. i f (! i s C o n t a i n F i e l d s ( b l o c k ) ) {
16. continue ;
17. }
18. E l e m e n t f i e l d s E l e m e n t = b l o c k . g e t C h i l d ( R O O T \_ F I E L D \_ E L E M E N T ) ;
19. List < E l e m e n t > f i e l d E l e m e n t s = f i e l d s E l e m e n t . g e t C h i l d r e n () ;

149

1. /*\**
2. Каждо е поле *−* отд ельно е измерение .
3. *\**/
4. f o r ( E l e m e n t f i e l d : f i e l d E l e m e n t s ) {
5. F i e l d X M L f i e l d X M L = new F i e l d X M L () ; 155
6. i f (! f i e l d . g e t C h i l d ( T A B L E N A M E \_ E L E M E N T ) .

g e t V a l u e () . e q u a l s ( C R M \_ T A B L E \_ F I E L D ) ) {

1. f i e l d X M L . s e t T a b l e N a m e ( f i e l d . g e t C h i l d ( T A B L E N A M E \_ E L E M E N T ) . g e t V a l u e () ) ;
2. m o d u l e X M L . s e t F a c t T a b l e ( f i e l d . g e t C h i l d ( T A B L E N A M E \_ E L E M E N T ) . g e t V a l u e () ) ;
3. } e l s e {
4. f i e l d X M L . s e t T a b l e N a m e ( C R M \_ T A B L E \_ F I E L D ) ;
5. f i e l d X M L . s e t C r m E n t i t y ( true ) ;
6. }

163

1. i f ( f i e l d . g e t C h i l d ( R O O T \_ E N T I T Y \_ I D \_ E L E M E N T )

!= n u l l ) {

1. f i e l d X M L . s e t P r i m a r y K e y ( f i e l d . g e t C h i l d ( R O O T \_ E N T I T Y \_ I D \_ E L E M E N T )
2. . g e t C h i l d ( E N T I T Y \_ I D \_ E L E M E N T ) .

g e t V a l u e () ) ;

1. i f ( f i e l d X M L . g e t T a b l e N a m e () . e q u a l s ( m o d u l e X M L . g e t F a c t T a b l e () ) ) {
2. m o d u l e X M L . s e t K e y ( f i e l d . g e t C h i l d ( R O O T \_ E N T I T Y \_ I D \_ E L E M E N T )
3. . g e t C h i l d ( E N T I T Y \_ I D \_ E L E M E N T )

. g e t V a l u e () ) ;

1. }
2. }

172

1. f i e l d X M L . s e t C o l u m n N a m e ( f i e l d . g e t C h i l d ( C O L U M N N A M E \_ E L E M E N T ) . g e t V a l u e () ) ;
2. f i e l d X M L . s e t C o l u m n T y p e ( f i e l d . g e t C h i l d ( C O L U M T Y P E \_ E L E M E N T ) . g e t V a l u e () ) ;
3. S t r i n g f i e l d L a b e l ;
4. i f (( f i e l d L a b e l = f i n d T r a n s l a t i o n ( f i e l d . g e t C h i l d ( F I E L D \_ L A B E L \_ E L E M E N T ) . g e t V a l u e () , m o d s t r i n g s N o d e ) ) != n u l l
5. || ( f i e l d L a b e l = f i n d T r a n s l a t i o n ( f i e l d . g e t C h i l d ( F I E L D \_ L A B E L \_ E L E M E N T ) . g e t V a l u e ()

, m a i n T r a n s l a t i o n s ) ) != n u l l ) {

1. f i e l d X M L . s e t F i e l d C a p t i o n ( f i e l d L a b e l ) ;
2. } e l s e f i e l d X M L . s e t F i e l d C a p t i o n ( f i e l d . g e t C h i l d ( F I E L D N A M E \_ E L E M E N T ) . g e t V a l u e () )

;

1. f i e l d X M L . s e t F i e l d N a m e ( f i e l d . g e t C h i l d ( F I E L D N A M E \_ E L E M E N T ) . g e t V a l u e () ) ;
2. f i e l d X M L . s e t U i T y p e ( f i e l d . g e t C h i l d ( U I \_ T Y P E \_ E L E M E N T ) . g e t V a l u e () ) ;
3. /*\**
4. Если поле является с сылкой на таблицу *−* обра ботать отд ельно .
5. *\**/
6. i f ( f i e l d X M L . i s R e l a t e d () ) {
7. E l e m e n t r e l a t e d = f i e l d . g e t C h i l d ( R O O T \_ R E L A T E D \_ M O D U L E \_ E L E M E N T ) ;
8. try {
9. List < E l e m e n t > m o d u l e s = r e l a t e d . g e t C h i l d r e n ( R E L A T E D \_ M O D U L E \_ E L E M E N T ) ;
10. f o r ( E l e m e n t n o d e : m o d u l e s ) {
11. f i e l d X M L . a d d R e l a t e d M o d u l e ( n o d e .

g e t V a l u e () ) ;

1. }
2. } catch ( N u l l P o i n t e r E x c e p t i o n e ) {
3. f i e l d X M L . a d d R e l a t e d M o d u l e ( m o d u l e X M L . g e t M o d u l e N a m e () ) ;
4. }
5. }
6. m o d u l e X M L . a d d N e w F i e l d ( f i e l d X M L ) ;
7. }
8. }
9. }

200

201 /*\**

202

1. *\**/
2. i f ( c h i l d N o d e . g e t N a m e () . e q u a l s ( R O O T \_ R E L A T E D \_ L I S T \_ E L E M E N T

) ) {

1. List < E l e m e n t > r e l L i s t = c h i l d N o d e . g e t C h i l d r e n () ;
2. f o r ( E l e m e n t r e l E l e m e n t : r e l L i s t ) {
3. i f (! r e l E l e m e n t . g e t C h i l d ( R E L A T E D \_ M O D U L E \_ E L E M E N T )

. g e t V a l u e () . e q u a l s ( " " ) ) {

1. m o d u l e X M L . a d d R e l a t e d M o d u l e ( r e l E l e m e n t . g e t C h i l d ( R E L A T E D \_ M O D U L E \_ E L E M E N T ) .

g e t V a l u e () ) ;

1. }
2. }
3. }
4. }
5. return m o d u l e X M L ;
6. }

215

1. /*\*\**
2. *\** Метод проверки на наличие элементов в блоке .
3. *\**
4. *\** @param element *−* элемент для проверки
5. *\** @return True *−* е сть поля
6. *\** False *−* нет полей 222 *\**/
7. private boolean i s C o n t a i n F i e l d s ( E l e m e n t e l e m e n t ) {
8. f o r ( E l e m e n t c h i l d : e l e m e n t . g e t C h i l d r e n () ) {
9. i f ( c h i l d . g e t N a m e () . e q u a l s ( R O O T \_ F I E L D \_ E L E M E N T ) ) {
10. return true ;
11. }
12. }
13. return f a l s e ;
14. }

231

1. /*\*\**
2. *\** Меьтод поиска перевода имени для опре д ел¨eнно г о элемента
3. *\**
4. *\** @param source *−* элемент
5. *\** @param c o l l e c t i o n *−* набор переводов
6. *\** @return рус ский перевод элемента 238 *\**/
7. private S t r i n g f i n d T r a n s l a t i o n ( S t r i n g source , E l e m e n t c o l l e c t i o n

) {

1. List < E l e m e n t > m o d s t r i n g s = c o l l e c t i o n . g e t C h i l d r e n () ;
2. f o r ( E l e m e n t m o d s t r i n g : m o d s t r i n g s ) {
3. E l e m e n t o r i g i n a l = m o d s t r i n g . g e t C h i l d ( O R I G I N A L \_ E L E M E N T ) ;
4. i f ( o r i g i n a l . g e t V a l u e () . e q u a l s ( s o u r c e ) ) {
5. return m o d s t r i n g . g e t C h i l d ( T R A N S L A T E D \_ E L E M E N T ) . g e t V a l u e () ;
6. }
7. }
8. return n u l l ;
9. }
10. }

Листинг А.2. PentahoMondrianService.java

1 package org . x e r o c r y . m o n d r i a n . e n d p o i n t s ; 2

1. import org . j d o m 2 . J D O M E x c e p t i o n ;
2. import org . x e r o c r y . m o n d r i a n . D a t a b a s e C o n f i g u r a t i o n ;
3. import org . x e r o c r y . m o n d r i a n . S c h e m a ;
4. import org . x e r o c r y . m o n d r i a n . e n d p o i n t s . d t o s . r e s p o n s e s .

S t r i n g O p e r a t i o n R e s u l t D T O ;

1. import org . xml . sax . S A X E x c e p t i o n ; 8
2. import j a v a x . ws . rs . GET ;
3. import j a v a x . ws . rs . P a t h ;
4. import j a v a x . ws . rs . P r o d u c e s ;
5. import j a v a x . ws . rs . Q u e r y P a r a m ;
6. import j a v a x . ws . rs . c o r e . M e d i a T y p e ;
7. import j a v a x . xml . b i n d . J A X B C o n t e x t ;
8. import j a v a x . xml . b i n d . M a r s h a l l e r ;
9. import j a v a x . xml . p a r s e r s . P a r s e r C o n f i g u r a t i o n E x c e p t i o n ;
10. import j a v a . io . I O E x c e p t i o n ;
11. import j a v a . io . S t r i n g W r i t e r ;
12. import j a v a . u t i l . l o g g i n g . L o g g e r ; 20
13. @ P a t h ( " @ p l u g i n . j a v a . r e s t . p a t h . r o o t @ " )
14. public c l a s s P e n t a h o M o n d r i a n S e r v i c e { 23

24 private s t a t i c L o g g e r log = L o g g e r . g e t L o g g e r (

P e n t a h o M o n d r i a n S e r v i c e . c l a s s . g e t N a m e () ) ;

25

26 public P e n t a h o M o n d r i a n S e r v i c e () { } 27

28

1. @ G E T
2. @ P a t h ( " / h e l l o " )
3. @ P r o d u c e s ( M e d i a T y p e . A P P L I C A T I O N \_ J S O N )
4. public S t r i n g O p e r a t i o n R e s u l t D T O h e l l o () { 33
5. //create result DTO
6. S t r i n g O p e r a t i o n R e s u l t D T O r e s u l t = new S t r i n g O p e r a t i o n R e s u l t D T O () ;

36

1. //fill string
2. r e s u l t . s t r i n g = " H e l l o W o r l d f r o m P e n t a h o S e r v i c e ! " ; 39
3. //fill status message
4. r e s u l t . s t a t u s M e s s a g e . c o d e = " O K \_ C O D E " ;
5. r e s u l t . s t a t u s M e s s a g e . m e s s a g e = " O K \_ M E S S A G E " ; 43
6. //return result DTO
7. return r e s u l t ; 46 }

47

1. @ G E T
2. @ P a t h ( " / a d d S o u r c e " )
3. @ P r o d u c e s ( M e d i a T y p e . A P P L I C A T I O N \_ J S O N )
4. public S t r i n g O p e r a t i o n R e s u l t D T O a d d S o u r c e ( @ Q u e r y P a r a m ( " h o s t " ) S t r i n g h o s t )
5. throws J D O M E x c e p t i o n , P a r s e r C o n f i g u r a t i o n E x c e p t i o n , S A X E x c e p t i o n , I O E x c e p t i o n {

53

1. S t r i n g url = " h t t p :// " + h o s t . t r i m () ;
2. D a t a b a s e C o n f i g u r a t i o n . i n i t ( url ) ;

56

57 S t r i n g O p e r a t i o n R e s u l t D T O r e s u l t = new S t r i n g O p e r a t i o n R e s u l t D T O () ;

58

59 r e s u l t . s t r i n g = " H e l l o W o r l d f r o m P e n t a h o S e r v i c e ! " ; 60

1. r e s u l t . s t a t u s M e s s a g e . c o d e = " O K \_ C O D E " ;
2. r e s u l t . s t a t u s M e s s a g e . m e s s a g e = " O K \_ M E S S A G E " ; 63
3. return r e s u l t ;
4. }

66

1. @ G E T
2. @ P a t h ( " / m o n d r i a n " )
3. @ P r o d u c e s ( M e d i a T y p e . A P P L I C A T I O N \_ X M L )
4. public S t r i n g m o n d r i a n ( @ Q u e r y P a r a m ( " h o s t " ) S t r i n g host ,
5. @ Q u e r y P a r a m ( " u s e r " ) S t r i n g user ,
6. @ Q u e r y P a r a m ( " p a s s " ) S t r i n g p a s s )
7. throws P a r s e r C o n f i g u r a t i o n E x c e p t i o n , S A X E x c e p t i o n , I O E x c e p t i o n , J D O M E x c e p t i o n {

74

1. S t r i n g url = " h t t p :// " + h o s t . t r i m () ;
2. // String url = "http://"+ host.trim() + "?"+ user.trim() + "?"+ pass.trim();
3. D a t a b a s e C o n f i g u r a t i o n . i n i t ( url ) ;

78

79 S c h e m a s c h e m a = D a t a b a s e C o n f i g u r a t i o n . g e t S c h e m a () ; 80

1. S t r i n g W r i t e r s t r i n g W r i t e r = new S t r i n g W r i t e r () ;
2. try {
3. J A X B C o n t e x t j a x b C o n t e x t = J A X B C o n t e x t . n e w I n s t a n c e ( S c h e m a

. c l a s s ) ;

1. M a r s h a l l e r j a x b M a r s h a l l e r = j a x b C o n t e x t . c r e a t e M a r s h a l l e r () ;

85

1. // output pretty printed
2. j a x b M a r s h a l l e r . s e t P r o p e r t y ( M a r s h a l l e r .

J A X B \_ F O R M A T T E D \_ O U T P U T , true ) ;

88

1. j a x b M a r s h a l l e r . m a r s h a l ( schema , S y s t e m . out ) ;
2. j a x b M a r s h a l l e r . m a r s h a l ( schema , s t r i n g W r i t e r ) ;
3. i f ( s c h e m a . g e t C u b e s () . s i z e () == 0) {
4. throw new E x c e p t i o n ( " D u m p has 0 m o d u l e s " ) ; 93 }
5. } catch ( E x c e p t i o n e ) {
6. e . p r i n t S t a c k T r a c e () ; 96 }

97

98 return s t r i n g W r i t e r . t o S t r i n g () ; 99 }

100 }

Листинг А.3. DatabaseConfiguration.java

1 package org . x e r o c r y . m o n d r i a n ; 2

1. import org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s . V T i g e r X M L ;
2. import org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s . X M L P a r s e r ;
3. import org . j d o m 2 . J D O M E x c e p t i o n ;
4. import org . xml . sax . S A X E x c e p t i o n ; 7
5. import j a v a x . xml . p a r s e r s . P a r s e r C o n f i g u r a t i o n E x c e p t i o n ;
6. import j a v a . io . F i l e ;
7. import j a v a . io . I O E x c e p t i o n ;
8. import j a v a . io . O u t p u t S t r e a m W r i t e r ;
9. import j a v a . net . H t t p U R L C o n n e c t i o n ;
10. import j a v a . net . URI ;
11. import j a v a . net . URL ;
12. import j a v a . net . U R L C o n n e c t i o n ;
13. import j a v a . u t i l . H a s h M a p ;
14. import j a v a . u t i l . I t e r a t o r ;
15. import j a v a . u t i l . Map ;
16. import j a v a . u t i l . S c a n n e r ;
17. import j a v a . u t i l . l o g g i n g . L e v e l ;
18. import j a v a . u t i l . l o g g i n g . L o g g e r ; 22
19. public c l a s s D a t a b a s e C o n f i g u r a t i o n {
20. private s t a t i c L o g g e r log = L o g g e r . g e t L o g g e r ( D a t a b a s e C o n f i g u r a t i o n . c l a s s . g e t N a m e () ) ;
21. private s t a t i c S t r i n g v t i g e r S o u r c e ;
22. private s t a t i c D a t a b a s e C o n f i g u r a t i o n i n s t a n c e ;
23. private s t a t i c S c h e m a s c h e m a ; 28

29 /*\*\**

30 *\** Метод , инициализирующий текущую конфигурацию базы данных . 31 *\**

1. *\** @param u r l *−* адре с домена с API Vtiger
2. *\** @throws JDOMException
3. *\** @throws IOException
4. *\** @throws ParserConfigurationException
5. *\** @throws SAXException
6. *\**/
7. public s t a t i c void i n i t ( S t r i n g url ) throws J D O M E x c e p t i o n , I O E x c e p t i o n , P a r s e r C o n f i g u r a t i o n E x c e p t i o n , S A X E x c e p t i o n {
8. i f ( i n s t a n c e == n u l l ) {
9. i n s t a n c e = new D a t a b a s e C o n f i g u r a t i o n ( url ) ;
10. log . i n f o ( " url = " + url + " in C o n f i g " ) ;
11. V T i g e r X M L v T i g e r X M L = new X M L P a r s e r ( v t i g e r S o u r c e ) . r e a d X M L () ;
12. S t r i n g s o u r c e N a m e = a d d J D B C R e s o u r c e ( v T i g e r X M L . g e t S o u r c e I n f o () ) ;
13. s c h e m a = v T i g e r X M L . g e n e r a t e S c h e m a ( s o u r c e N a m e ) ; 45 }

46 }

47

48 /*\*\**

49 *\** Геттер для получения экз емпляра клас са для паттерна Синглтон . 50 *\**

51 *\** @return экз емпляр синглтона 52 *\**/

1. public s t a t i c D a t a b a s e C o n f i g u r a t i o n g e t I n s t a n c e () {
2. return i n s t a n c e ; 55 }

56

1. private D a t a b a s e C o n f i g u r a t i o n ( S t r i n g v t i g e r S o u r c e ) {
2. D a t a b a s e C o n f i g u r a t i o n . v t i g e r S o u r c e = v t i g e r S o u r c e ; 59 }

60

1. public s t a t i c S c h e m a g e t S c h e m a () {
2. return s c h e m a ; 63 }

64

65 /*\*\**

66 *\** Функция с оздания ново г о источника данных *−* БД. 67 *\**

1. *\** @return Имя подключения
2. *\**/
3. public s t a t i c S t r i n g a d d J D B C R e s o u r c e ( H a s h M a p < String , String > s o u r c e I n f o ) {
4. S t r i n g b o d y = " {\" c h a n g e d \": true , \" u s i n g C o n n e c t i o n P o o l \": true , \" c o n n e c t S q l \": \"\" , \" d a t a b a s e N a m e \": " +
5. " \" " + s o u r c e I n f o . get ( " d b \_ n a m e " ) + " \" , \" d a t a b a s e P o r t

\": \" " + s o u r c e I n f o . get ( " d b \_ p o r t " ) . s u b s t r i n g (1)

+

1. " \" , \" h o s t n a m e \": \" " + s o u r c e I n f o . get ( " d b \_ n a m e " ) + "

\" , \" n a m e \": \" " + s o u r c e I n f o . get ( " d b \_ n a m e " ) +

1. " \" , \" p a s s w o r d \": \" " + s o u r c e I n f o . get ( " d b \_ p a s s w o r d "

) + " \" , \" u s e r n a m e \": \" " +

1. s o u r c e I n f o . get ( " d b \_ u s e r n a m e " ) + " \" , \" a t t r i b u t e s \":

{} , \" c o n n e c t i o n P o o l i n g P r o p e r t i e s \": {} , " +

1. " \" e x t r a O p t i o n s \": {} , \" a c c e s s T y p e \": \" N A T I V E \" ,

\" d a t a b a s e T y p e \": {\" d e f a u l t D a t a b a s e P o r t \": 9001 , " +

1. " \" e x t r a O p t i o n s H e l p U r l \": \" h t t p :// h s q l d b . s o u r c e f o r g e . net / doc / g u i d e / c h 0 4 . h t m l # N 1 0 9 D A \" , "

+

1. " \" n a m e \": \" M y S Q L \" , \" s h o r t N a m e \": \" M Y S Q L \" , \" s u p p o r t e d A c c e s s T y p e s \": " +

79 " [\" N A T I V E \" , \" O D B C \" , \" J N D I \"] } } " ;

80

1. try {
2. S t r i n g d o m a i n n a m e = S y s t e m . g e t e n v ( " U S E R D O M A I N " ) ;
3. URL url = new URL ( " h t t p :// " + d o m a i n n a m e +
4. " / p e n t a h o / p l u g i n / data - a c c e s s / api / d a t a s o u r c e / j d b c

/ c o n n e c t i o n / T e s t D a t a s o u r c e " ) ;

1. H t t p U R L C o n n e c t i o n h t t p C o n = ( H t t p U R L C o n n e c t i o n ) url . o p e n C o n n e c t i o n () ;
2. h t t p C o n . s e t D o O u t p u t ( true ) ;
3. h t t p C o n . s e t R e q u e s t M e t h o d ( " PUT " ) ;
4. O u t p u t S t r e a m W r i t e r out = new O u t p u t S t r e a m W r i t e r (
5. h t t p C o n . g e t O u t p u t S t r e a m () ) ;
6. out . w r i t e ( b o d y ) ;
7. out . c l o s e () ;
8. h t t p C o n . g e t I n p u t S t r e a m () ;
9. } catch ( E x c e p t i o n e ) {
10. log . log ( L e v e l . SEVERE , null , e ) ; 95 }

96 return s o u r c e I n f o . get ( " d b \_ n a m e " ) ; 97 }

98 }

Листинг А.4. Schema.java

1 package org . x e r o c r y . m o n d r i a n ; 2

1. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . C u b e ;
2. import l o m b o k . G e t t e r ;
3. import l o m b o k . S e t t e r ;
4. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . D i m e n s i o n ;
5. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . V i r t u a l C u b e ; 8
6. import j a v a x . xml . b i n d . a n n o t a t i o n .\*;
7. import j a v a . u t i l . A r r a y L i s t ;
8. import j a v a . u t i l . L i s t ; 12
9. @ G e t t e r
10. @ S e t t e r
11. @ X m l R o o t E l e m e n t ( n a m e = " S c h e m a " )
12. @ X m l A c c e s s o r T y p e ( X m l A c c e s s T y p e . F I E L D )
13. public c l a s s S c h e m a {
14. @ X m l A t t r i b u t e
15. private S t r i n g n a m e = " f i r s t V e r " ; 20
16. @ X m l E l e m e n t ( n a m e = " D i m e n s i o n " , t y p e = D i m e n s i o n . c l a s s )
17. private List < D i m e n s i o n > d i m e n s i o n s = new A r r a y L i s t < >() ; 23
18. @ X m l E l e m e n t ( n a m e = " C u b e " , t y p e = C u b e . c l a s s )
19. private List < Cube > c u b e s = new A r r a y L i s t < >() ; 26
20. @ X m l E l e m e n t ( n a m e = " V i r t u a l C u b e " , t y p e = V i r t u a l C u b e . c l a s s )
21. private List < V i r t u a l C u b e > v i r t u a l C u b e s = new A r r a y L i s t < >() ; 29
22. public void a d d C u b e ( C u b e c u b e ) {
23. c u b e s . add ( c u b e ) ;
24. }

33

1. public C u b e g e t C u b e ( S t r i n g n a m e ) {
2. f o r ( C u b e c u b e : c u b e s ) {
3. i f ( c u b e . g e t N a m e () . e q u a l s ( n a m e ) ) {
4. return c u b e ;
5. }
6. }
7. return n u l l ;
8. }

42

1. public void a d d D i m e n s i o n ( D i m e n s i o n d i m e n s i o n , S t r i n g c u b e N a m e ) {
2. d i m e n s i o n s . add ( d i m e n s i o n ) ; 45 }

46

1. public void a d d V i r t u a l C u b e ( V i r t u a l C u b e c u b e ) {
2. v i r t u a l C u b e s . add ( c u b e ) ;
3. }

50

1. public S c h e m a ( S t r i n g n a m e ) {
2. t h i s . n a m e = n a m e ; 53 }

54

55 /*\*\**

1. *\** Конструктор для JAXB
2. *\**/
3. public S c h e m a () {
4. }

60

1. public D i m e n s i o n g e t D i m e n s i o n ( S t r i n g n a m e ) {
2. f o r ( D i m e n s i o n dim : d i m e n s i o n s ) {
3. i f ( dim . g e t N a m e () . e q u a l s ( n a m e ) ) {
4. return dim ;
5. }
6. }
7. return n u l l ;
8. }
9. }

Листинг А.5. VTigerXML.java

1 package org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s ; 2

1. import org . x e r o c r y . m o n d r i a n . S c h e m a ;
2. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . C u b e ;
3. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . D i m e n s i o n ;
4. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . V i r t u a l C u b e ;
5. import org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s . V i r t u a l C u b e D i m e n s i o n ; 8
6. import j a v a . u t i l . A r r a y L i s t ;
7. import j a v a . u t i l . H a s h M a p ; 11
8. public c l a s s V T i g e r X M L {
9. private A r r a y L i s t < M o d u l e X M L > x m l M o d u l e s = new A r r a y L i s t < >() ;
10. H a s h M a p < String , String > s o u r c e I n f o = new H a s h M a p < >() ; 15

16

17 /*\*\**

18 *\** Основа формирования Mondrian схемы 19 *\**

1. *\** @param name *−* имя схемы
2. *\** @return JAXB*−*объ ект с од ержащий мас сив кубов и вирт. кубов 22 *\**/
3. public S c h e m a g e n e r a t e S c h e m a ( S t r i n g n a m e ) {
4. S c h e m a s c h e m a = new S c h e m a ( n a m e ) ; 25
5. /*\**
6. Для каждо г о модуля с озда¨eтся отд ельный куб 28 *\**/
7. f o r ( M o d u l e X M L m o d u l e : x m l M o d u l e s ) {
8. C u b e c u b e = new C u b e ( m o d u l e . g e t M o d u l e N a m e () ) ;
9. c u b e . s e t T a b l e ( m o d u l e . g e t F a c t T a b l e () ) ;
10. c u b e . s e t F o r e i g n K e y ( m o d u l e . g e t K e y () ) ;
11. /*\**
12. Сначала с оздаются измерения , которые будут включены в ку б .
13. *\**/
14. f o r ( F i e l d X M L f i e l d : m o d u l e . g e t F i e l d X M L L i s t () ) {
15. D i m e n s i o n dim ;
16. i f ( s c h e m a . g e t D i m e n s i o n ( f i e l d . g e t F i e l d N a m e () ) != n u l l ) {
17. dim = new D i m e n s i o n ( c u b e . g e t N a m e () + f i e l d . g e t F i e l d N a m e () ,
18. f i e l d . g e t F i e l d C a p t i o n () ) . g e n e r a t e ( f i e l d )

;

1. } e l s e dim = new D i m e n s i o n ( f i e l d . g e t F i e l d N a m e () , f i e l d . g e t F i e l d C a p t i o n () ) . g e n e r a t e ( f i e l d ) ;
2. s c h e m a . a d d D i m e n s i o n ( dim , c u b e . g e t N a m e () ) ;
3. c u b e . a d d D i m e n s i o n ( dim ) ; 44 }
4. c u b e . s e t R e l a t e d ( m o d u l e . g e t R e l a t e d M o d u l e s () ) ;
5. c u b e . g e n e r a t e () ;
6. s c h e m a . a d d C u b e ( c u b e ) ; 48 }

49

1. /*\**
2. Затем виртуальный куб на о снове отно сящихся к модулю модулям 52 *\**/
3. f o r ( C u b e c u b e : s c h e m a . g e t C u b e s () ) {
4. V i r t u a l C u b e v i r C u b e = new V i r t u a l C u b e () ;
5. i f ( c u b e . g e t R e l a t e d () . s i z e () == 0) {
6. continue ;
7. }
8. f o r ( S t r i n g r e l a t e d : c u b e . g e t R e l a t e d () ) {
9. i f ( s c h e m a . g e t C u b e ( r e l a t e d ) == n u l l ) continue ;
10. f o r ( D i m e n s i o n d i m e n s i o n : s c h e m a . g e t C u b e ( r e l a t e d ) . g e t D i m e n s i o n s () ) {
11. v i r C u b e . a d d V i r t u a l D i m e n s i o n (new V i r t u a l C u b e D i m e n s i o n ( d i m e n s i o n , c u b e . g e t N a m e () ) ) ;
12. }
13. }
14. v i r C u b e . s e t N a m e ( v i r C u b e . g e n e r a t e N a m e () ) ;
15. s c h e m a . a d d V i r t u a l C u b e ( v i r C u b e ) ;
16. }
17. return s c h e m a ; 68 }

69

1. void a d d M o d u l e ( M o d u l e X M L m o d u l e X M L ) {
2. x m l M o d u l e s . add ( m o d u l e X M L ) ; 72 }

73

1. public H a s h M a p < String , String > g e t S o u r c e I n f o () {
2. return s o u r c e I n f o ; 76 }

77 }

Листинг А.6. ModuleXML.java

1 package org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s ; 2

1. import l o m b o k . G e t t e r ;
2. import l o m b o k . S e t t e r ; 5

6 import j a v a . u t i l . A r r a y L i s t ;

7

1. @ G e t t e r
2. @ S e t t e r
3. public c l a s s M o d u l e X M L {
4. private S t r i n g m o d u l e N a m e ;
5. private S t r i n g m o d u l e C a p t i o n ;
6. private I n t e g e r s e q u e n c e N u m ;
7. private S t r i n g key ;
8. private A r r a y L i s t < F i e l d X M L > f i e l d X M L L i s t ; 16
9. private boolean c r m E n t i t y ;
10. private S t r i n g f a c t T a b l e ; 19
11. private boolean r e l a t e d ;
12. private A r r a y L i s t < String > r e l a t e d M o d u l e s ; 22
13. public M o d u l e X M L () {
14. t h i s . f i e l d X M L L i s t = new A r r a y L i s t < >() ;
15. t h i s . r e l a t e d M o d u l e s = new A r r a y L i s t < >() ; 26 }

27

1. public void a d d N e w F i e l d ( F i e l d X M L f i e l d ) {
2. f i e l d X M L L i s t . add ( f i e l d ) ; 30 }

31

1. public void a d d R e l a t e d M o d u l e ( S t r i n g m o d u l e ) {
2. r e l a t e d M o d u l e s . add ( m o d u l e ) ; 34 }

35

36 }

Листинг А.7. FieldXML.java

1 package org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s ;

2

1. import l o m b o k . G e t t e r ;
2. import l o m b o k . S e t t e r ; 5

6 import j a v a . u t i l . A r r a y L i s t ;

7

1. @ G e t t e r
2. @ S e t t e r
3. public c l a s s F i e l d X M L {
4. private S t r i n g c o l u m n N a m e ;
5. private S t r i n g f i e l d N a m e ;
6. private S t r i n g c o l u m n T y p e ;
7. private S t r i n g t a b l e N a m e ;
8. private S t r i n g f i e l d C a p t i o n ;
9. private S t r i n g u i T y p e ;
10. private S t r i n g p r i m a r y K e y ;
11. private A r r a y L i s t < String > r e l a t e d M o d u l e s = new A r r a y L i s t < >() ; 19

20 private s t a t i c f i n a l S t r i n g R E L \_ M O D U L E \_ T Y P E = " 10 " ;

21

22 private boolean c r m E n t i t y ; 23

1. public void a d d R e l a t e d M o d u l e ( S t r i n g m o d u l e ) {
2. r e l a t e d M o d u l e s . add ( m o d u l e ) ; 26 }

27

1. public boolean i s R e l a t e d () {
2. return u i T y p e . e q u a l s ( R E L \_ M O D U L E \_ T Y P E ) ; 30 }

31

32

33 }

Листинг А.8. TableAdapter.java

1 package org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s ; 2

1. import j a v a x . xml . b i n d . a n n o t a t i o n . X m l A t t r i b u t e ;
2. import j a v a x . xml . b i n d . a n n o t a t i o n . a d a p t e r s . X m l A d a p t e r ; 5

6 public c l a s s T a b l e A d a p t e r extends X m l A d a p t e r < T a b l e A d a p t e r .

A d a p t e d T a b l e , String > {

7

1. @ O v e r r i d e
2. public S t r i n g u n m a r s h a l ( A d a p t e d T a b l e v ) throws E x c e p t i o n {
3. return v . n a m e ; 11 }

12

1. @ O v e r r i d e
2. public A d a p t e d T a b l e m a r s h a l ( S t r i n g v ) throws E x c e p t i o n {
3. A d a p t e d T a b l e amf = new A d a p t e d T a b l e () ;
4. i f ( v == n u l l ) {
5. return n u l l ;
6. }
7. amf . n a m e = v ;
8. return amf ;
9. }

|  |  |  |
| --- | --- | --- |
| 22 |  | |
| 23 |  | public s t a t i c c l a s s A d a p t e d T a b l e { |
| 24 |  |  |
| 25 |  | @ X m l A t t r i b u t e |
| 26 |  | public S t r i n g n a m e ; |
| 27 |  |  |
| 28 |  | } |
| 29 | } |  |

Листинг А.9. Dimension.java

1 package org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s ; 2

1. import org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s . F i e l d X M L ;
2. import l o m b o k . G e t t e r ;
3. import l o m b o k . S e t t e r ; 6
4. import j a v a x . xml . b i n d . a n n o t a t i o n .\*;
5. import j a v a . u t i l . A r r a y L i s t ;
6. import j a v a . u t i l . L i s t ; 10
7. @ X m l A c c e s s o r T y p e ( X m l A c c e s s T y p e . F I E L D )
8. @ G e t t e r
9. @ S e t t e r
10. @ X m l R o o t E l e m e n t ( n a m e = " D i m e n s i o n " )
11. public c l a s s D i m e n s i o n {
12. @ X m l A t t r i b u t e ( n a m e = " n a m e " )
13. private S t r i n g n a m e ;
14. @ X m l A t t r i b u t e
15. private S t r i n g c a p t i o n ;
16. @ X m l A t t r i b u t e
17. private S t r i n g t y p e = " S t a n d a r d D i m e n s i o n " ;
18. @ X m l E l e m e n t ( n a m e = " H i e r a r c h y " , t y p e = H i e r a r c h y . c l a s s )
19. private List < H i e r a r c h y > h i e r a r c h y = new A r r a y L i s t < >() ; 24

25

1. public D i m e n s i o n ( S t r i n g name , S t r i n g c a p t i o n ) {
2. t h i s . n a m e = n a m e ;
3. t h i s . c a p t i o n = c a p t i o n ; 29 }

30

1. public D i m e n s i o n () {
2. }

33

1. public D i m e n s i o n ( D i m e n s i o n o t h e r ) {
2. t h i s . n a m e = o t h e r . n a m e ;
3. t h i s . c a p t i o n = o t h e r . c a p t i o n ;
4. t h i s . t y p e = o t h e r . t y p e ;
5. t h i s . h i e r a r c h y = o t h e r . h i e r a r c h y ; 39 }

40

1. public D i m e n s i o n g e n e r a t e ( F i e l d X M L f i e l d ) {
2. H i e r a r c h y h i e r a r c h y = new H i e r a r c h y () ;
3. h i e r a r c h y . s e t N a m e ( f i e l d . g e t F i e l d N a m e () ) ; 44
4. L e v e l l e v e l = new L e v e l () ;
5. l e v e l . s e t C o l u m n ( f i e l d . g e t C o l u m n N a m e () ) ;

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 47 |  |  | l e v e l . s e t N a m e ( f i e l d . g e t F i e l d N a m e () ) ; |  |
| 48 |  |  | l e v e l . s e t C a p t i o n ( f i e l d . g e t F i e l d C a p t i o n () ) ; |
| 49 |  |  | l e v e l . s e t T y p e ( f i e l d . g e t C o l u m n T y p e () ) ; |
| 50 |  |  | h i e r a r c h y . a d d L e v e l ( l e v e l ) ; |
| 51 |  |  | i f ( f i e l d . g e t T a b l e N a m e () . e q u a l s ( " v t i g e r \_ c r m e n t i t y " ) ) | { |
| 52 |  |  | h i e r a r c h y . s e t T a b l e ( " v t i g e r \_ c r m e n t i t y " ) ; |  |
| 53 |  |  | } e l s e h i e r a r c h y . s e t T a b l e ( f i e l d . g e t T a b l e N a m e () ) ; |  |
| 54 |  |  | h i e r a r c h y . s e t C a p t i o n ( f i e l d . g e t F i e l d C a p t i o n () ) ; |  |
| 55 |  |  | h i e r a r c h y . s e t P r i m a r y K e y ( f i e l d . g e t P r i m a r y K e y () ) ; |  |
| 56 |  |  | t h i s . h i e r a r c h y . add ( h i e r a r c h y ) ; |  |
| 57 |  |  |  |  |
| 58 |  |  | return t h i s ; |  |
| 59 |  | } |  |  |
| 60 |  |  |  |  |
| 61 |  |  |  |  |
| 62 |  |  |  |  |
| 63 |  |  |  |  |
| 64 | } |  |  |  |

Листинг А.10. Cube.java

1 package org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s ; 2

3

1. import org . x e r o c r y . m o n d r i a n . S c h e m a ;
2. import org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s . F i e l d X M L ;
3. import org . x e r o c r y . m o n d r i a n . x m l \_ e l e m e n t s . M o d u l e X M L ;
4. import l o m b o k . G e t t e r ;
5. import l o m b o k . S e t t e r ; 9
6. import j a v a x . xml . b i n d . a n n o t a t i o n .\*;
7. import j a v a x . xml . b i n d . a n n o t a t i o n . a d a p t e r s . X m l J a v a T y p e A d a p t e r ;
8. import j a v a . u t i l . A r r a y L i s t ;
9. import j a v a . u t i l . L i s t ; 14
10. @ X m l A c c e s s o r T y p e ( X m l A c c e s s T y p e . F I E L D )
11. @ G e t t e r
12. @ S e t t e r
13. public c l a s s C u b e {
14. @ X m l A t t r i b u t e
15. private S t r i n g n a m e ;
16. @ X m l A t t r i b u t e
17. private S t r i n g s o u r c e ;
18. @ X m l A t t r i b u t e
19. private S t r i n g c a p t i o n ;
20. @ X m l T r a n s i e n t
21. private S t r i n g f o r e i g n K e y ;
22. @ X m l J a v a T y p e A d a p t e r ( T a b l e A d a p t e r . c l a s s )
23. private S t r i n g t a b l e ;
24. @ X m l E l e m e n t ( n a m e = " D i m e n s i o n U s a g e " , t y p e = D i m e n s i o n U s a g e . c l a s s

)

1. private List < D i m e n s i o n U s a g e > d i m e n s i o n U s a g e s ;
2. @ X m l T r a n s i e n t
3. private List < String > r e l a t e d = new A r r a y L i s t < >() ;
4. @ X m l T r a n s i e n t
5. private List < D i m e n s i o n > d i m e n s i o n s ; 35
6. public C u b e g e n e r a t e () {
7. f o r ( D i m e n s i o n d i m e n s i o n : d i m e n s i o n s ) {
8. D i m e n s i o n U s a g e d i m e n s i o n G e n = new D i m e n s i o n U s a g e () ;
9. d i m e n s i o n G e n = d i m e n s i o n G e n . g e n e r a t e ( d i m e n s i o n ) ;
10. d i m e n s i o n G e n . s e t F o r e i g n K e y ( f o r e i g n K e y ) ;
11. d i m e n s i o n U s a g e s . add ( d i m e n s i o n G e n ) ; 42 }

43 return t h i s ; 44 }

45

1. public C u b e ( S t r i n g n a m e ) {
2. d i m e n s i o n s = new A r r a y L i s t < >() ;
3. d i m e n s i o n U s a g e s = new A r r a y L i s t < >() ;
4. t h i s . n a m e = n a m e ; 50 }

51

1. public C u b e () {
2. d i m e n s i o n U s a g e s = new A r r a y L i s t < >() ;
3. d i m e n s i o n s = new A r r a y L i s t < >() ; 55 }

56

1. public void a d d D i m e n s i o n ( D i m e n s i o n d i m e n s i o n ) {
2. d i m e n s i o n s . add ( d i m e n s i o n ) ; 59 }

60 }

Листинг А.11. VirtualCube.java

1 package org . x e r o c r y . m o n d r i a n . m o n d r i a n \_ c l a s s e s ; 2

1. import l o m b o k . G e t t e r ;
2. import l o m b o k . S e t t e r ; 5
3. import j a v a x . xml . b i n d . a n n o t a t i o n . X m l A c c e s s T y p e ;
4. import j a v a x . xml . b i n d . a n n o t a t i o n . X m l A c c e s s o r T y p e ;
5. import j a v a x . xml . b i n d . a n n o t a t i o n . X m l A t t r i b u t e ;
6. import j a v a x . xml . b i n d . a n n o t a t i o n . X m l E l e m e n t ;
7. import j a v a . u t i l . A r r a y L i s t ;
8. import j a v a . u t i l . L i s t ; 12
9. @ G e t t e r
10. @ S e t t e r
11. @ X m l A c c e s s o r T y p e ( X m l A c c e s s T y p e . F I E L D )
12. public c l a s s V i r t u a l C u b e {
13. @ X m l E l e m e n t ( n a m e = " V i r t u a l C u b e D i m e n s i o n " , t y p e = V i r t u a l C u b e D i m e n s i o n . c l a s s )
14. private List < V i r t u a l C u b e D i m e n s i o n > d i m e n s i o n U s a g e s ; 19
15. @ X m l A t t r i b u t e
16. private S t r i n g n a m e ; 22

23 /*\*\**

1. *\** Конструктор для JAXB
2. *\**/
3. public V i r t u a l C u b e () {
4. d i m e n s i o n U s a g e s = new A r r a y L i s t < >() ; 28 }

|  |  |  |
| --- | --- | --- |
| 29 |  | |
| 30  31 |  | public void a d d V i r t u a l D i m e n s i o n ( V i r t u a l C u b e D i m e n s i o n d i m e n s i o n )  {  d i m e n s i o n U s a g e s . add ( d i m e n s i o n ) ; |
| 32 |  | } |
| 33 |  |  |
| 34 |  | public S t r i n g g e n e r a t e N a m e () { |
| 35 |  | V i r t u a l C u b e D i m e n s i o n dim = d i m e n s i o n U s a g e s . get (1) ; |
| 36 |  | return dim . g e t C a p t i o n () + " V i r t u a l " ; |
| 37 |  | } |
| 38 | } |  |

Листинг А.12. Код на Java

* 1. <! D O C T Y P E html >
  2. < h t m l l a n g = " en " > < head >
  3. < m e t a c h a r s e t = " utf -8 " >
  4. < title > J a v a S c r i p t F o r m V a l i d a t i o n u s i n g a s a m p l e r e g i s t r a t i o n form </ title >
  5. < l i n k rel = ’ s t y l e s h e e t ’ h r e f = ’ js - form - v a l i d a t i o n . css ’ t y p e = ’ t e x t / css ’

/ >

* 1. < s c r i p t src = " m a i n . js " > </ script >
  2. </ head >
  3. < b o d y o n l o a d = " d o c u m e n t . r e g i s t r a t i o n . url . f o c u s () ; " >
  4. < h1 > M o n d r i a n p u b l i s h Form </ h1 >
  5. <p > Use tab k e y s to m o v e f r o m one i n p u t f i e l d to the n e x t . </ p >
  6. < f o r m n a m e = ’ r e g i s t r a t i o n ’ o n S u b m i t = " r e t u r n p u b l i s h () ; " >
  7. < ul >
  8. < li > < l a b e l f o r = " url " > V T i g e r URL : </ la be l > </ li >
  9. < li > < i n p u t t y p e = " t e x t " n a m e = " url " s i z e = " 50 " / > </ li >
  10. < li > < l a b e l f o r = " u s e r n a m e " > U s e r n a m e : </ l abe l > </ li >
  11. < li > < i n p u t t y p e = " t e x t " n a m e = " u s e r n a m e " s i z e = " 12 " / > </ li >
  12. < li > < l a b e l f o r = " p a s s i d " > P a s s w o r d : </ la be l > </ li >
  13. < li > < i n p u t t y p e = " p a s s w o r d " n a m e = " p a s s i d " s i z e = " 12 " / > </ li >
  14. < li > < l a b e l f o r = " s o u r c e " > D a t a s o u r c e : </ l abe l > </ li >
  15. < li > < s e l e c t n a m e = " s o u r c e " id = " s o u r c e s " >
  16. < o p t i o n s e l e c t e d = " " v a l u e = " D e f a u l t " >( P l e a s e s e l e c t a d a t a s o u r c e ) </ option >
  17. </ select > </ li >
  18. < li > < i n p u t t y p e = " s u b m i t " n a m e = " s u b m i t " v a l u e = " S u b m i t " / > </ li > 24 </ ul >

1. </ form >
2. < s c r i p t src = " sample - r e g i s t r a t i o n - form - v a l i d a t i o n . js " > </ script >
3. < script >
4. var s e l e c t = d o c u m e n t . g e t E l e m e n t B y I d ( " s o u r c e s " ) ;
5. var m y o b j e c t = g e t S o u r c e s () ;
6. f o r ( var i n d e x in m y o b j e c t ) {
7. s e l e c t . o p t i o n s [ s e l e c t . o p t i o n s . l e n g t h ] = new O p t i o n ( m y o b j e c t [ i n d e x ]. $ , i n d e x ) ;
8. }
9. </ script >
10. </ body >
11. </ html >

Листинг А.13. main.js

1. f u n c t i o n g e t S o u r c e s () {
2. var e = new X M L H t t p R e q u e s t ;
3. e . o p e n ( " GET " , " / p e n t a h o / p l u g i n / data - a c c e s s / api / d a t a s o u r c e / j d b c / c o n n e c t i o n " , !1) , e . s e n d () ;
4. var t = J S O N . p a r s e ( e . r e s p o n s e T e x t ) ,
5. n = t . I t e m ;
6. return n } 7
7. f u n c t i o n g e t M o n d r i a n () {
8. var e = new X M L H t t p R e q u e s t ;
9. e . o p e n ( " GET " , " / p e n t a h o / p l u g i n / p e n t a h o - vtiger - p l u g i n / api / m o n d r i a n " , !1) , e . s e n d () ;
10. var t = e . r e s p o n s e X M L ;
11. return t } 13
12. f u n c t i o n p u b l i s h () {
13. var e = (new X M L S e r i a l i z e r ) . s e r i a l i z e T o S t r i n g ( g e t M o n d r i a n () ) ,
14. t = { u p l o a d I n p u t : " C o n t e n t - T y p e : t e x t / xml \ r \ n " + e , p a r a m e t e r s : " o v e r w r i t e = t r u e ; D a t a S o u r c e = " + d o c u m e n t . g e t E l e m e n t B y I d ( " s o u r c e s " ) . o p t i o n s [ d o c u m e n t . g e t E l e m e n t B y I d ( " s o u r c e s " ) . s e l e c t e d I n d e x ]. text , s c h e m a F i l e I n f o : " f i r s t V e r . xml " , x m l a E n a b l e d F l a g : " f a l s e " } ,
15. n = S t r i n g ( M a t h . r a n d o m () ) . s l i c e (2) ,

18 r = " - - " + n + " \ r \ n " ,

19 i = " - - " + n + " - -\ r \ n " ,

20 s = [ " \ r \ n " ];

1. f o r ( var o in t ) s . p u s h ( ’ C o n t e n t - D i s p o s i t i o n : form - d a t a ; n a m e =" ’

+ o + ’ "\ r \ n \ r \ n ’ + t [ o ] + " \ r \ n " ) ;

1. s = s . j o i n ( r ) + i , c o n s o l e . log ( s ) ;
2. var u = new X M L H t t p R e q u e s t ;
3. u . o p e n ( " PUT " , " / p e n t a h o / p l u g i n / data - a c c e s s / api / d a t a s o u r c e / a n a l y s i s / i m p o r t " , !0) , u . w i t h C r e d e n t i a l s = !0 , u . s e t R e q u e s t H e a d e r ( " A u t h o r i z a t i o n " , " B a s i c " + b t o a ( " a d m i n : p a s s w o r d " ) ) , u . s e t R e q u e s t H e a d e r ( " C o n t e n t - T y p e " , " m u l t i p a r t / form - d a t a ; b o u n d a r y = " + n ) , u . o n r e a d y s t a t e c h a n g e = f u n c t i o n () {
4. i f ( t h i s . r e a d y S t a t e != 4) return ;
5. a l e r t ( t h i s . r e s p o n s e T e x t ) } , u . s e n d ( s ) ;
6. var a = new X M L H t t p R e q u e s t ;
7. a . o p e n ( " GET " , " p e n t a h o / p l u g i n / data - a c c e s s / api / d a t a s o u r c e / a n a l y s i s / c a t a l o g / f i r s t V e r " , !0) , a . s e n d () ;
8. var f = a . r e s p o n s e T e x t ;
9. c o n s o l e . log ( f ) } 31
10. f u n c t i o n g e t A n a l y s i s () {
11. var e = new X M L H t t p R e q u e s t ;
12. e . o p e n ( " GET " , " p e n t a h o / p l u g i n / data - a c c e s s / api / d a t a s o u r c e / a n a l y s i s / ids " , !0) , e . s e n d () ;
13. var t = e . r e s p o n s e X M L ;
14. c o n s o l e . log ( t ) }
15. d e f i n e ( " ../ m a i n " , f u n c t i o n () {}) ;

Листинг А.14. Код на Javajs-form-validation.cs

1. h1 {
2. margin - l e f t : 70 px ;
3. }
4. f o r m li {
5. list - s t y l e : n o n e ;
6. margin - b o t t o m : 5 px ; 7 }

8

9 f o r m ul li l a b e l {

1. f l o a t : l e f t ;
2. c l e a r : l e f t ;
3. w i d t h : 100 px ;
4. text - a l i g n : r i g h t ;
5. margin - r i g h t : 10 px ;
6. font - f a m i l y : V e r d a n a , Arial , H e l v e t i c a , sans - s e r i f ;
7. font - s i z e :14 px ;
8. }

18

1. f o r m ul li input , select , s p a n {
2. f l o a t : l e f t ;
3. margin - b o t t o m : 10 px ; 22 }

23

1. f o r m t e x t a r e a {
2. f l o a t : l e f t ;
3. w i d t h : 350 px ;
4. h e i g h t : 150 px ;
5. }

29

1. [ t y p e = " s u b m i t " ] {
2. c l e a r : l e f t ;
3. m a r g i n : 20 px 0 0 230 px ;
4. font - s i z e :18 px
5. }

35

1. p {
2. margin - l e f t : 70 px ;
3. font - w e i g h t : b o l d ; 39 }