# *Миграция автоматизированной системы кредитования Банка на новую платформу*

**Профиль пользователя:**

Банковские сотрудники

**Функциональная область:**

Миграция системы, разработка веб-приложений

**Трудозатраты:**

14 специалистов, 8 месяцев

**Технологии и средства разработки**

- Java 6
- GWT 2.1.1
- GXT (ExtJs) 2.2.1
- Spring 3.0.5
- Hibernate 3.5.1-Final
- myBatis 3.0.4
- docx4j 2.7.1
- POI 3.7
- MSSQL 2007
- WebSpere Application Server v.7

## *Задача*

Клиент обратился к Artezio с просьбой перевести внедренную в Банке автоматизированную систему кредитования на новую платформу и разработать более удобный и понятный пользовательский интерфейс.

## *Проблема*

Автоматизированная система кредитования Заказчика была разработана с использованием технологий Power Builder и MS SQL. Желанием заказчика было перевести систему на технологии Java, при этом сохранить MS SQL в качестве базы данных, а также сделать интерфейс более удобным. Веб-интерфейс должен быть создан в соответствии с корпоративными стандартами Заказчика с использованием библиотеки SBGWT .

Клиент выбрал Artezio благодаря большому опыту по выполнению проектов для банковской сферы и доступности сертифицированных специалистов с необходимыми навыками.

## *Решение*

Система проектировалась и разрабатывается с учётом следующих принципов:

- Разбивка приложения на логические слои
- Модульность приложения
- Компонентная структура с низкой связностью
- Создание интерфейсов повторно используемых модулей и компонент
- Единообразный подход к разработке компонент определённого типа

Web-интерфейсы системы разрабатываются с учётом следующих факторов:

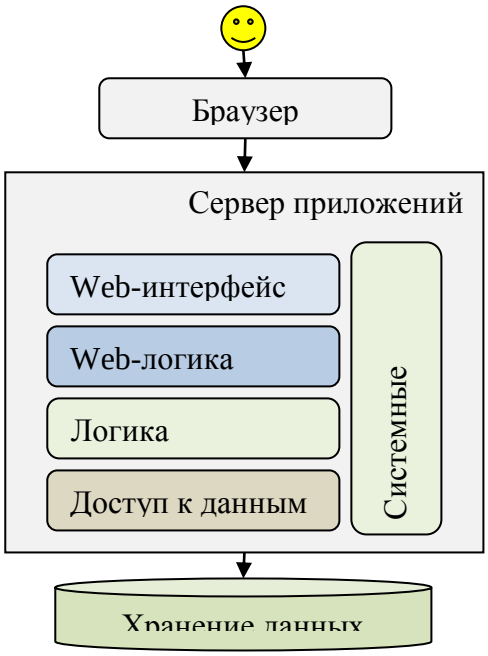
- Схожесть с интерфейсом существующего толстого клиента
- Использование стандартных компонент пользовательского интерфейса, отвечающих корпоративным стандартам Заказчика.
- Удобство использования и скорость работы интерфейса

Выбранная архитектура системы призвана обеспечивать:

- Компонентную структуру приложения
- Максимизацию повторного использования кода
- Гибкость приложения
- Расширяемость приложения

Архитектура системы построена по многоуровневому принципу. Основные логические слои:

- Слой Web-интерфейса
- Слой web-логики
- Слой логики приложения
- Слой доступа к данным
- Слой хранения данных




**Слой web-интерфейса**

Слой пользовательского web-интерфейса отвечает за отображение графического интерфейса пользователя. Данный слой обеспечивает доступ пользователя к приложению через браузер посредством протокола HTTP(S).

Данный слой представляет собой web-приложение основанное на технологии GWT. При построении web-слоя используются лучшие практики для GWT приложений, такие как:

- Улучшение структуры интерфейсного слоя с использованием шаблона Model-View-Presenter (MVP)
- Использование стандартизованного базового набора визуальных компонент
- Разделение визуальной структуры и компонентного наполнения страницы с применением UIBinder

- Группировка и кеширование ресурсов приложения в виде ResourceBundle
- Раздельная загрузка фрагментов приложения с использованием GWT Code Split
- Кеширование и повторное использование сложных объектов GWT
- Кэширование данных словарей

**Слой web-логики**

Слой web-логики отвечает за функциональность пользовательского интерфейса. Данный слой обеспечивает интерфейс пользователя данными для отображения, а так же обеспечивает обработку запросов пользователя.
Данный слой реализуется в виде сервисов GWT.

**Слой логики приложения**

Слой логики приложения содержит основную бизнес логику приложения. Данный слой позволяет отделить операции приложения от реализации интерфейса пользователя, таким образом обеспечивая модульность, инкапсуляцию и повторное использование реализованной логики приложения.
Данный слой реализуется в виде независимых бизнес-сервисов.

**Слой доступа к данным**

Слой доступа к данным позволяет приложению работать с данными в постоянном хранилище данных.
Слой доступа к данным (DAO ) реализован на базе технологии объектно-реляционного маппинга (ORM) с использованием декларативного управления транзакциями. В качестве ORM среды используется библиотека MyBatis..Для управления зависимостями между DAO объектами и бизнес компонентами, а так же для декларативного управления транзакциями используются возможности библиотеки Spring.

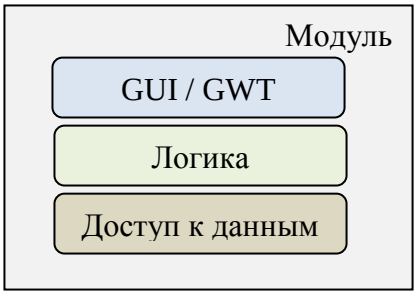
**Слой хранения данных**

Слой хранения данных отвечает за постоянное хранение данных системы.
Хранилище данных представляет собой реляционную базу данных на основе РСУБД MS SQL Server.

**Системные сервисы**

Системные сервисы обеспечивают нефункциональные кросс-компонентные аспекты приложения, такие как конфигурирование, логирование, кешировние, декларативная поддержка транзакций и т.д.
Системные сервисы основываются на библиотеке Spring и ряде специализированных вспомогательных библиотек.

Приложение структурировано в виде набора функциональных модулей. Каждый модуль реализует набор связного функционала системы и реализуется с использованием стандартного набора логических слоёв.
Типовая структура функционального модуля:



Функциональные модули независимы друг от друга.

### *Результат*

Разработанная система полностью отвечала всем требованиям заказчика. Разработанное веб-приложение позволило упростить работу пользователей с системой и сократить требования к пользовательским рабочим станциям.

ООО «Артезио»
119333, Россия, Москва
Ул. Губкина, д. 3Г, офис 504

Тел.: +7 (495) 981-0531
Факс: +7 (495) 232-2683
Email: info@artezio.com
www.artezio.ru