Требования к архитектуре и решениям разрабатываемого ПО

1. Модульность ПО.

- Разрабатываемое ПО должно состоять из отдельных модулей\приложений

Один модуль – отдельный WAR архив, с UI-формами, с имплементированными сущностями, которые обслуживает модуль.

Примеры модулей: кредиты юридическим лицам, кредиты физическим лицам, депозиты физическим лицам, ценные бумаги, банковские пластиковые карточки, межбанковские сделки, баланс НБРБ, баланс НБ Кыргыстана, баланс НБ Узбекистана, кассовое обслуживание клиентов, межбанковские и форексные сделки, нормативная и пруденциальная отчетность, и т.д.

- Для запуска приложений должен быть использован один из серверов приложений (JBoss, WildFly, Netty, Tomcat и др.) с использованием Java 8+, Postgres 12.0, AngularJS. Клиентская часть – Mozilla FireFox 72.0, Internet Explorer 11.

- На выбор банка, любой из модулей может быть вынесен на отдельный сервер приложений. К примеру, такие высоконагруженные модули, как кредиты физическим лицам, депозиты физическим лицам, банковские пластиковые карты, смс-сервер оповещений, могут быть размещены на отдельных серверах (в т.ч. и облаке), а малонагруженные модули могут быть размещены небольшой группой приложений на отдельном сервере.

При таком подходе, кратковременная остановка или перезагрузка одного из серверов приложений не вызывает паралич всей системы в целом.

- Каждый модуль должен иметь интерфейсы для взаимодействия с другими модулями и подсистемами - POST и GET сервисы импорта\экспорта требуемых данных. Например, POST-сервисы для получения сведений о выдачах\пополнениях и погашений по кредитам\депозитам совершенных в сторонних сервисах\платежных системах. Или GET сервисы для получения информации о имеющихся задолженностях\обязательствах\операциях с целью отражения информации в сторонних синтетических система учета (баланс, DWH, подсистемы отчетности и пр.)

- Каждый модуль должен разрабатываться с учетом того, что одна версия модуля будет уходить сразу на все банки, которые будут использовать комплекс модулей в целом. Кастомизация каждого конкретного модуля должна осуществляться саппортом банка путем установки локальных настроечных параметров\констант (application.properties)

1. Ядро учетной системы

Ядро учетной системы должно состоять из следующих базовых объектов и элементов:

- сущности

- действия над сущностями

- пользователи системы

- функции системы

- роли

- тарифицируемые услуги

- тарифные планы (наборы тарифицируемых услуг)

- графики выплаты или погашения

- задолженности

- обязательства

- финансовые операции

- валюты

- счета

- ставки

- сроки

- документы

- отметки на сущности

- шаблоны документов

Подсистемы, которые используют элементы и функции ядра:

- подсистема тарификации и расчета платности за оказанные услуги

- договорной (финансовый) учет

- балансовый учет

- платежная система(ы)

- генератор отчетности (pdf, rtf, xls, html)

- расписание

1. Базовые сущности

Базовой сущностью в учетной системе является условный абстрактный объект, который имеет следующие атрибуты:

* Ид сущности (счетчик)
* Тип сущности
* Статус сущности
* Дата создания сущности
* Дата закрытия сущности
* Дата последнего изменения
* Кем изменена в последний раз

Более укрупненными сущностями системы, которые создаются\редактируются\закрываются модулями системы, являются такие сущности как:

- Договор агентских отношений

- Договор аккредитива

- Договор на открытие и обслуживание импортного аккредитива

- Заявление на открытие импортного аккредитива

- Заявление на изменение импортного аккредитива

- Заявление об акцепте/отказе от акцепта документов по аккредитиву

- Заявление на аннулирование импортного аккредитива

- Банковская купюра

- Банкомат (АТМ)

- Договор покупки ветхих банкнот

- Договор проверки подлинности банковской купюры

- Договор обмена вышедшей из обращения купюры

- Договор обмена вышедшей из обращения купюры

- Покупка банкноты с зачислением безналичного покрытия

- Договор обслуживания банковской гарантии

- Гарантии и поручительства по кредитным операциям

- Требования по прочим видам гарантий

- Залог, при котором предмет залога остается у залогодателя

- Ипотека

- Залог товаров в обороте

- Залог ценных бумаг

- Залог имущественных прав (требований)

- Заклад

- Ценные бумаги, полученные в обеспечение по операциям РЕПО

- Биржевое свидетельство

- Биржевая заявка

- Договор на осуществление валютно-обменных операций

- Заявка на покупку валюты на бирже

- Заявка на свободную продажу валюты на бирже

- Поручение на обязательную продажу валюты на бирже

- Клиентская заявка на покупку валюты на внебиржевом рынке

- Клиентская заявка на продажу валюты на внебиржевом рынке

- Клиентская заявка на конверсию валюты на внебиржевом рынке

- Неттинговые расчеты по клиентским сделкам

- Банковская заявка на продажу валюты на бирже

- Банковская заявка на покупку валюты на бирже

- Реестр распределения иностранной валюты

- Генеральное соглашение

- Договор аренды банковской ячейки

- Договор аренды банковского сейфа"

- Гарантийный депозит

- Займ

- Срочныйдепозит

- Условный депозит

- Депозит до востребования"

- Розничный договор физического лица

- ДС к договору срочного банковского вклада

- ДС об изменении продукта

- ДС к договору текущего счета о предоставлении полномочий

- Заявления розничных договоров

- Поручение клиента

- Постоянно действующее поручение клиента

- Доверенности

- Договор ПБХ

- Договор поручительства

- Договор страхования

- Договор залога

- Описание предмета залога

- Договор пакетного обслуживания"

- Договор обслуживания в системе Клиент-Банк

- Заявка на проведение платежа

- Договор лизинга

- Договор неснижаемого остатка

- Договор расчетно-кассового обслуживания

- Договор банковского счета

- Договор факторинга

- Покупка драгметаллов на внешних финансовых рынках

- Покупка драгметаллов у предприятий-производителей

- Продажа драгметаллов

- Договор зарплатного обслуживания предприятия

- Договор использования Интернет-Банкинга

- Договор публичной оферты

- Карточка клиента

- Банковская пластиковая карта

- Заготовка пластиковой карты

- Кредитный договор с юридическим лицом

- График денежных обязательств по договору

- Заявка на кредит

- Кредитный договор с физическим лицом

- Договор поручительства

- Платежная инструкция

- Кредитовое авизо МТ910 (вх.)

- Дебетовое авизо МТ900 (вх.)

- Перевод (без открытия счета)

- Клиентский перевод

- Банковский перевод

- Пользователь системы

- Роль в системе

- Договор обеспечения

- Документ-основание ВЭД

- Дополнение к внешнеэкономическому договору

- Разрешение НБ по движению капитала

- Разрешение на превышение суммы авансовых платежей

- Подтверждение поставок товара

- Удостоверение о регистрации иностранной безвозмездной помощи

- Уведомление о получении иностранной безвозмездной помощи

- Паспорт сделки (до вступления в силу 178 Указа)

- Паспорт сделки

- Спецификация по кодам операций ВЭД

-Установленный лимит на банк

- Страновой лимит

- Связанный лимит

- Валютный лимит

- Лимит на группу компаний

- Персональный лимит на контрагента

- Заявление на проведение сделки СВОП

- Сделка СВОП

- Тарифный план кредитного договора с ЮЛ

- Тарифный план кредитного договора с ФЛ

- Тарифный план кредитного договора с ИП

- Тарифный план депозитного договора с ФЛ

- Тарифный план договора кассового обслуживания с ЮЛ

- Тарифный план договора договора текущего счета с ЮЛ

- Тарифный план договора кассового обслуживания с ЮЛ

- Тарифный план корпоративного договора обслуживания текущих счетов ФЛ"

- Сделка нецелевого внутрибанковского финансирования

- Межбанковский кредит размещенный

- Межбанковский кредит привлеченный

- Заявка на привлечение МБК под залог ЦБ от НБ

- Заявка на получение кредитных ресурсов

- Расчетная сделка по МБК

- Хозяйственный договор

- Доверенности (по хозяйственному договору)

- Выпуск облигаций

- Выпуск депозитных сертификатов

- Выпуск векселей

- Выпуск акций

- Сделка банка с облигациями до погашения

- Сделка обратного РЕПО с облигациями

- Сделка прямого РЕПО с облигациями

- Сделка с собственными облигациями

- Портфель ценных бумаг

- Межпортфельный перевод бумаг

- Торговая площадка

- Амортизационная ведомость

- Купонный платеж по выпуску

- Договор эквайринга

- Другие модули.

Каждая из приведенный сущностей функционально наследует свойства и атрибуты сущности более высокого\абстрактного уровня, которые, в конечном итоге, берут свое начало (унаследованы) от базовой абстрактной сущности.

Каждая из запроектированных сущностей отличается друг от друга набором атрибутов, специфичным для конкретной предметной бизнес-области.

Типовой набор статусов для большинства сущностей является следующий перечень статусов:

- создана\ действующая (0)

- аннулирована (-1)

- закрыта (1)

Такой набор статусов задается конкретно для каждого вида сущностей и может быть расширен, если типового набора недостаточно для разрабатываемой сущности.

1. Действия над сущностями (активности)

Действиями над сущностями являются пользовательские активности, которые изменяют состояние сущности. Например, увеличивают или уменьшают сумму задолженности по договору, рассчитывают сумму %, аннулируют её и т.д. Примеры действий: авторизация сделки, выдача кредита, регистрация договора поручительства по кредитному договору, аннулирование сделки, закрытие сделки и др.

Каждое действие должно уметь проверять себя - может ли оно быть выполнено в конкретный момент. Например, над кредитной сделкой запрещено выполнять действие выдачи кредита или урегулирование начисленных %, пока сделка не авторизована. При формировании списка доступных действий, каждое действие должно проверить, авторизована ли сделка. И если сделка не авторизована, то активное действие не попадает в список доступных для выполнения действий и не может быть выполнено.

Другой пример. Действие авторизации может быть выполнено только один раз. Если сделка уже авторизована, то действие не позволяет выполниться повторно и будет недоступно в списке доступных действий.

Третий пример. Авторизация сделки по принципу 4х глаз. Тот, кто завел\создал сделку, тот не может её авторизовать. Авторизовать может только другой пользователь. В таком случае, действие авторизации должно проверить ид пользователя который создал сделку. Если ид совпадает, то действие не должно быть доступно. Тот, кто завел сделку не сможет её авторизовать, т.к. действие авторизации будет для него недоступно.

Имплементация конкретного действия всегда зависит от требований, которые предъявляются к этому действию. Требования к конкретному действию могут отличаться от банка в банку. К примеру, в одном банке попросят реализовать авторизацию по принципу 4х глаз, в другом банке это не нужно. Решение – через настроечный параметр\константу, на которое ориентируется действие.

Состояние любой сущности может быть изменено только с помощью зарегистрированного действия. Действие может быть выполнено только пользователем, который имеет право выполнить это действие, которое ему назначено через роль.

Любое выполнение действие регистрируется и хранится в истории (ид, ip рабочей станции, код действия, дата\время выполнения действия, длительность выполнения действия, регистрированное сообщение об ошибке, если таковая возникла при исполнении. Должна существовать возможность просмотра истории выполненных действий над конкретной сущностью или договором.

Любая сущность, которая регистрируется в системе, должна иметь зарегистрированный набор действий, с помощью которых можно изменять сущность. От состояния «Создана» до состояния «Закрыта» сущность может быть приведена только с помощью действий. В состоянии «Закрыта» над сущностью может быть выполнено только одно действие «Открытие закрытой сделки\сущности».

Выполнять действия могут только зарегистрированные пользователи системы, которым разрешено (через роль) выполнять назначенные роли действия. Также, это может быть сервисный пользователь, который в режиме работы расписания или массового выполнения действия над набором сущностей, может выполнить действие (при условии, что действие может быть выполнено по бизнес-процессу).

Набор зарегистрированный действий для конкретной сущности позволяет избавиться от жесткой схемы, когда состояние сущности может изменяться только от одного статуса к другому статусу, и только с помощью одного перехода\маршрута.

В определенный момент времени над сущностью может быть доступно к выполнению разное количество действий (от 0 до N), и при этом не требуется обязательно менять статус самой сущности при выполнении действия. Только имплементация конкретного определяет, надо ли изменять статус самой сущности, над которой выполняется действие. Набор действий для одной и тоже сущности, но для разных банков может быть разным. Например, в своих, целях один банк может попросить доработать специальное действие, но другие банки не должны почувствовать этого изменения.

В ряде случаев целесообразно разработать прямое действие, а также обратное действие, например, установка какой-либо отметки и снятие какой-то отметки.

Только фиксированный набор действий и сущностей позволяет держать под контролем учетную модель, а также придерживаться какой- то системы при ведении разработки\доработки сущностей и действий.

1. Интерактивное выполнение действия

Действие над любой сущностью может быть выполнено интерактивным способом при непосредственном участии пользователя, при условии, что оно может быть выполнено в текущий момент над сущностью. При этом от пользователя может запроситься дополнительная информация, например, дата проведения операции, тип используемого курса, вид платежной системы, куда следует сделать зачисление и т.д. После чего действие может что-то выполнить (изменить статус сущности), а может при этом сформировать список операций на подтверждение пользователю. И после того, как пользователь соглашается со предварительным списком операций, действие совершает изменение сущности или какую-либо операцию.

Т.е. интерактивный способ выполнения действия предполагает непосредственное участие пользователя для изменения сущности. Но так же, действие может быть выполнено без участия пользователя (в silent-режиме), и при этом ориентироваться на какие-то значения по умолчанию (текущая дата операционного дня, текущий курс валюты, дефолтная настройка из констант и т.п.)

Без участия пользователя действие может быть выполнено в пакетном режиме выполнения или массовом режиме выполнения над группой сущностей.

1. Пакетное выполнение действий.

Любая сущность должна иметь специальное действие (в наборе действий), которое позволяет выполнить группу нескольких действий пакетным способом (батчем). Это означает, что группа действий может быть выполнена без участия пользователя в silent-режиме (например, при выполнении из расписания) под технологическим пользователем, но при этом в истории действий над сущностью останется отметка об выполнении одного из пакетных действий.

Пакетное действие должно уметь определить набор конкретных действий для конкретной сущности в конкретном состоянии. Выполнение этой группы действий в silent-режиме не всегда означает, что каждое действие будет выполнено. При вызове действия оно всегда само проверит, может ли оно быть выполнено. Если результат проверки = false, то действие не выполняется и пропускается, и следом идет выполнение последующего действия.

Такой подход позволяет организовать автоматизированную обработку сущностей без участия пользователей или с минимальным участием пользователей. Не для всех сущностей будет приемлем такой подход, но для очень многих сущностей подойдет такая схема обработки с минимальным вмешательством пользователей.

Для реализации такой пакетной обработки действий необходимо вести соответствующую разработку действий, которая предусматривает выполнение действия в разных режимах.

1. Массовое выполнение действия.

Для реализации массового начисления всевозможных % и комиссий по депозитам или кредитам, должна существовать возможность массового выполнения одного действия над группой однотипных сущностей. На усмотрение пользователя может выбрана группа сущностей (одна, несколько шт или все). На начальном этапе массового выполнения действия от пользователя может быть получена только да «По», по которую следует сделать начисление.

Должна существовать возможность выполнения такого массового действия из расписания или при помощи ручного запуска.

Также должна существовать возможность массового выполнения пакетного действия, как из расписания, так и в ручном режиме.

1. Какие учетные объекты не следует делать сущностями

Если у объекта системы бывает всего два состояния – создано и аннулировано – то такой объект не стоит делать выделенной сущностью с набором действий. Примеры таких объектов – балансовая проводка (её можно создать или аннулировать), лицевой счет (его можно открыть и закрыть), распоряжение на совершение балансовой операции (создать и аннулировать). Такие объекты создаются, как правило, в рамках выполнения других действий над сущностями.

Сущностями должны быть «тяжеловесные объекты», такие как договора и сделки, над которыми совершается множество мелкий действий, которые необходимо регистрировать и вести над ними историю действий.

1. Договорной учет

- Основанием для любого учета служит заключенная сделка (договор) с клиентом (физическим или юридическим лицом) или договор публичной оферты

- Основные положения учета должны опираться на Гражданский Кодекс РБ\КР\РФ, в котором даны определения договора, задолженностей, обязательств. Эти положения не должны противоречить законодательству.

- Любой договор должен содержать минимальные атрибуты, такие как:

- вид\тип договора

- дата заключения договора

- срок действия договора

- валюта договора

- сумма договора

- контрагент по договору

- статус договора

- Изменения состояния договора должно выполняться только через высокоуровневые действия (активности), которые выполняются пользователями учетных модулей.

- Примеры действий – создание\редактирование договора, авторизация договора, аннулирование договора, начисление процентных и комиссионных вознаграждений по договору, урегулирование обязательств и начисленных процентов, урегулирование сумм договоров обеспечения, завершение договора, урегулирование технического овердрафта, реструктуризация задолженности в связи с пролонгацией и т.д.

Вызов действия может быть вынесен на конкретную кнопку на форме, в список доступных действий выпадающего списка (item) или на конкретный пункт меню.

1. Финансовый или первичный учет

- Основу каждого договора должен составлять финансовый учет, который состоит из учета задолженностей, вытекающих из договора, учета обязательств по платности (погашению возникающих задолженностей), и учета финансовых операций, которые увеличивают или уменьшают суммы задолженностей

- Должна существовать возможность задания графиков погашений для некоторых видов задолженностей, от которых строятся обязательства по погашению задолженностей

- В процессе ведения договора должна существовать возможность заведения одного или нескольких типов задолженностей по договору, возможность ведения и учета договорных обязательств по погашению возникающих задолженностей, которые связаны с возникающими задолженностями (проценты, комиссии и пр.), а также создание и учет финансовых операций, которые увеличивают или уменьшают задолженности по договору

- Задолженность должна состоять из следующих атрибутов:

- вид задолженности

- тип задолженности

- валюта учета задолженности

- код договора, к которой относится задолженность

- учетная группа задолженности

- состояние задолженности

- вид базового актива

- дата начала действия задолженности

- дата завершения действия задолженности

- Обязательство по погашению задолженности должна состоять из следующий атрибутов:

- код связанной задолженности

- дата возникновения обязательства

- дата начала исполнения обязательства

- дата завершения исполнения обязательства

- дата юридического оформления обязательства

- дата регистрации обязательства в учетной системе

- дата закрытия (исполнения) обязательства

- признак аннулирования обязательства

Одна задолженность может иметь от одного до N обязательств по погашению этой самой задолженности. Простой пример – кредитный договор сроком на один год имеет одну основную задолженность по кредиту с которой связано 12 обязательств по её погашению.

- Финансовая операция по изменению суммы задолженности должна иметь следующие атрибуты:

- код связанного обязательства

- тип действия по обязательству

- код финансовой операции

- статус финансовой операции

- дата проведения в учете

- дата регистрации операции в учетной системе

- сумма финансовой операции

- статус финансовой операции

Примеры видов задолженностей:

1 Нормальная задолженность

2 Единожды пролонгированная задолженность

3 Неоднократно пролонгированная задолженность

4 Просроченная задолженность

5 Не востребованная в срок задолженность

6 Безнадежная задолженность

7 Задолженность до восстребования

8 Проблемная текущая задолженность

9 Проблемная пролонгированная задолженность

10 Проблемная просроченная задолженность

Примеры видов обязательств:

1 Требования по возврату базисного актива

2 Обязательства по возврату базисного актива

3 Требования по уплате процентов

4 Обязательства по уплате процентов

5 Специальные резервы на покрытие возможных убытков по активам, подверженным кредитному риску

6 Обязательства банка по возврату субординированных кредитов

7 Исполненные банком обязательства по обеспечению

9 Требования по обязательным резервам

10 Платежные документы, приобретенные для продажи

Примеры кодов финансовых операций:

1 Принятие к учету обязательства

2 Ввод остатков по учтенным обязательствам

3 Снятие с учета обязательства

4 Уменьшение суммы обязательства

5 Связывание обязательств

6 Открепление связи по обязательствам

7 Увеличение суммы обязательства

8 Редактирование атрибутов обязательства

9 Клирингование обязательств

1. Балансовый учет

Балансовый учет финансовых операций, является синтетической формой учета, которая является обязательной для коммерческих банков, и, фактически, является формой принудительного учета и отчетности перед финансовым регулятором (НБРБ)

Такая форма учета является недостаточной для полноценного учета всех задолженностей и обязательств, которые возникают исходя из договорного учета банка с клиентами.

Для полноценного учета договорных обязательств с клиентами, банки могут обходиться без балансового учета, но банки, фактически, обязаны отчитываться перед регулятором в такой громоздкой форме отчетности. В виде остатков по лицевым и балансовым счетам, а также в виде операций между счетами (проводками).

1. Разделение балансового учета (проводок, баланса) от первичного учета

Требования к разделению финансового учета и балансового учета

- Финансовый учет является формой первичного учета и должен существовать отдельно от балансового учета.

- Балансовый учет является вторичной формой учета

- Финансовый учет ничего не должен знать про балансовый учет, его особенности, но операции финансового учета должны быть отражены в необходимой мере в балансовом учете

- В модулях первичного учета не должно быть никаких специфичных выгрузок для других модулей

- Балансовый учет от НБРБ должен быть вынесен в отдельный модуль, который уметь «ходить» в другие учетные модули, которые осуществляют первичный и финансовый учет сделок и договорных обязательств, и отражать в себе операции финансового учета в виде проводок и остатков по лицевым и балансовым счетам

Ситуация может осложниться ещё тем, что отдельные банки могут вести несколько синтетических учетов, подобных балансовому учете от НБРБ. Например, по балансовому учету банк отчитывается перед НБРБ, а по другому виду балансового учета банк может отчитываться перед своим основным акционером, который может являться иностранным банком. В таком случае гораздо удобней иметь несколько отдельных модулей синтетического учета, которые живут автономно, но которые, при этом, отражают в себе операции первичного (финансового) учета, в том разрезе, который требуется этим модулям синтетического учета. Более того, правила балансового учета могут отличаться от страны к стране, видами классифицируемых пассивов и активом, наличием и отсутствием отдельных групп лицевых или балансовых счетов, которые специфичны для конкретной страны.

В таком случае, гораздо удобней иметь несколько отдельных модулей синтетического\балансового учета под каждую конкретную страну, при неизменной модели первичного учета (задолженность-обязательство-финансовая операция).

В самом простом случае, никакого балансового учета, может и не быть вообще. И тогда финучет может существовать сам по себе, без всякого баланса.

1. Требования к модулям балансового и синтетического учета

Для полноценной реализации требований регулятора, модули синтетического и балансового учета должны отражать в себе финансовые операции первичных учетных модулей в том разрезе, который им необходим.

Для реализации таких требований модули первичного учета должны предоставлять такие сервисы (GET), из которых можно узнать о совершенных или аннулированных финансовых операций над задолженностями, о новых сделках, международных платежах и т.п. Полученная информация о финансовой операции должна содержать сведения о типе операции, виде и типе задолженности над которой она совершалась, виде базового актива или пассива, текстовое назначение платежа или операции. Этот разрез информации должен позволить модулю синтетического учета автоматически сформировать условную проводку (возможно в составе общей клиринговой операции) по своим, заранее сделанным настройкам.

Например, если код финопреации Х, вид сделки Y, тип задолженности Z, код подразделения NNN, то автоматически создается проводка (прямая или обратная) дебет счета СЧЕТ1, кредит счета СЧЕТ2. Конкретные лицевые счета задаются в настройках на стороне банка модуля балансового учета. Возможно по одной финоперации могут быть созданы 2 и более проводок.

Настройки правил отражения финопераций в синтетическом учете, в части того, как именно отражать активные или пассивные операции первичного учета, могут отличаться от банка к банку, который может придерживаться собственной учетной политики ведения балансового учета. С использованием транзитных счетов или без, в одной части активов или пассивов используются сборные счета, в другой части – индивидуальные. И т.д.

1. Платность оказываемых услуг и тарификация сумм платы

После заключения договора банк начинает оказывать клиенту платные услуги. Платность бывает двусторонняя (клиенты платят банку % в виде комиссий по кредитам или банк сам платит клиентам (% по депозитам, по текущим счетам, кэшбэк, и д.р.).

Сумма платы за оказанную услугу должна быть рассчитана с помощью тарифа. Тариф должен позволять рассчитать сумму оказанной услуги за конкретный период с, по; по действующей ставке или с применением какой-либо нормативной величины (БВ, ставка рефинансирования НБРБ и т.п.)

Примеры тарифицируемых услуг:

- Проценты по счету учета кредитного договора

- Проценты по счету учета депозитного договора

- Комиссия за ведение счета

- За выдачу кредита

- Комиссия за выдачу наличными (нац. валюта)

- Комиссия за обслуживание договора в валюте договора

- Комиссия за обслуживание карт-счета

- Комиссия за выпуск карточки

- Комиссия за изменение договора по инициативе клиента

- Пеня за факт невозврата

- Комиссия за оформление кредитных документов

- Проценты по просроченному долгу

- Пеня за факт вынесения на счета просроченной задолженности

- Штраф за нарушение условий пользования льготным периодом кредитования

- Проценты за остатки на пассивных счетах

- Расходы за SWIFT

- Срочные платежи со счетов клиентов

- Выплата перевода физическому лицу без открытия счёта

- Абонентная плата за пользование системой "Интернет-Банк"

- Комиссия за нарушение Правил пользования карточкой

- Обслуживание СМС-оповещения (абонентская плата)

- Проверка рекламации, которая по факту оказалась необоснованно опротестованной

- Комиссионное вознаграждение за перевыпуск карты в течение срока действия

- и т.д.

Один тариф может иметь несколько видов\алгоритмов расчета. Например, тариф «Проценты по счету учета кредитного договора» может иметь несколько видов\алгоритмов расчета:

- По текущим остаткам

- По средним остаткам

- По формуле сложных процентов

- расчет % от ставки НБРБ

- расчет % от ставки либор 3М

- расчет % от суммы средних остатков

- расчет % по формуле расчета эффективной % ставки

Существующие тарифы должны объединятся в тарифные планы. Тарифные планы, в свою очередь, задаются пользователями на сделках, с указанием дат действий тарифа (с, по) внутри тарифного плана.

Тариф сам по себе должен уметь только рассчитывать предполагаемые суммы платности, за любой период, без проведения сумм по счетам финансового учета. Должна существовать возможность получить суммы рассчитанные тарифом за любой период (прошедший, будущий), без проведения сумм тарифов по счетам учета.

1. Ежедневный расчет тарифицируемых сумм

Должна существовать возможность ежедневного расчета сумм тарификации по всем типам сделок (или по заданным настройкам типов сделок), которые заведены в системе. При этом рассчитанные суммы должны сохранятся в промежуточных структурах\таблицах до момента ежемесячного отражения этих сумм на счетах финансового учета. Должна существовать возможность ручной корректировки таких сумм, или пересчета задним числом за любой период.

Должна существовать возможность регулирования и настройки перечня тарифицируемых услуг, которые должны предварительно рассчитываться на ежедневной основе, т.к. не для всех тарифов требуется такая громоздкая процедура ежедневного расчета. Для простых тарифов, например, разовых комиссий, достаточно будет рассчитать на лету сумму финансовой услуги за период с, по и отразить в финансовом учете.

1. Регламентное начисление

Должна существовать возможность регламентного начисления рассчитанных сумм платности в финансовом учете, которые были предварительно рассчитаны и сохранены. На выбор банка начисление может выполняться на любой основе (ежедневной, еженедельной, ежемесячной). Зачисление на счета финансового учета должно быть возможно только с помощи зарегистрированных в системе действий, а также при помощи расписания или пакетного выполнения действий.

Для тех тарифов, для которых не настроен ежедневный предварительный расчет, действие начисления должно получить суммы тарификации непосредственно с самого тарифа, указав период с, по. В этом случае тариф должен рассчитать на лету сумму за оказанную услугу.

1. Настройка ролей, функций и прав

Должна быть реализована стандартная модель: пользователь-роль-назначенные права роли. Под правами роли понимается право выполнения определенных действий над сущностью, а также право доступа к определенным функциям системы (как правило это UI формы).

1. Справочники и картотеки

Под справочниками в системе понимается та системная информация, которую ведет разработчик. Это справочники видов сущностей, перечень действий над сделками, кодов финансовых операций, видов задолженностей, видов тарифицируемых услуг и т.д. Без ведома разработчика эта информация не может поменяться. Такие справочники - сфера ответственности разработчика. Банк не в праве изменять эту информацию.

Под картотеками понимается тот набор информации, который ведет банк. Это картотека клиентов, картотека лицевых счетов, картотека ставок, тарифных планов и т.п. Это сфера ответственности банка.

1. Настройка наименования полей на формах и заголовков таблиц.

Должна существовать пользовательская возможность изменения наименования полей и подсказок (tooltiptext) на карточках любых сделок-сущностей. Прямо с рабочего места пользователя. Настройка полей может быть персональной для пользователя, так и для всех пользователей карточки сделки. Так же должна существовать возможность сбросить значения полей на дефолтные.

1. Кастомизация модуля под каждый конкретный банк.

Должна существовать возможность настройки каждого модуля под персональные требования конкретного банка. Например, поля А,Б,В сущности Х являются обязательными для заполнения в одном банке (так попросил банк) и НЕ являются обязательными для другого банка.

Требования разных банков к поведению одной и той же сущности могут быть разными и не должны пересекаться между собой!

Доработки сущности по требованию одного банка никак не должны повлиять на работу модуля в другом банке.

Включение\отключение того или иного функционала должно регулироваться настроечными параметрами модуля\константами.

Если банк А попросил сделать поле1 обязательным для заполнения, то других банков это не должно коснуться. Поле1 для них должно остаться необязательным для заполнения.

Если банк А попросил добавить для него новое поле на карточку сущности, то другие банки не должны увидеть это новое поле.

1. Сущность и карточка сущности.

Должна существовать возможность просмотра и редактирование полей сущности через дополнительную UI форму. Т.е. одна сущность может иметь несколько карточек для просмотра и редактирования её полей. Подразделение А имеет доступ к полям 1-10 сущности, подразделение Б имеет доступ к полям 11-20 сущности. Внутри одного банка.

1. Автоматизированное тестирование сущностей

Должна существовать возможность проведения полноценных тестов всех разработанных сущностей на тестовой базе разработчика или банка. Тест должен включать в себя программное заведение сущности, выполнение полного набора зарегистрированных действий над сущностью, и закрытие сущности.

На примере кредитной сделки это означает, что должен быть заведен краткосрочный\долгосрочный кредит, должно быть выполнено действие создание сделки, авторизация, выдача кредита, начисление %, погашение %, просрочка кредита, вынос на счета просроченной задолженности, начисление пеней и штрафов, погашение пеней и штрафов, погашение кредита, изменение графика платежей, урегулирование сумм начисленных %, закрытие кредитной сделки. Тест считается успешно пройденным если все типы задолженностей были созданы (отличны от 0) и погашены в 0, и при этом, при выполнении действий не возникло программных исключений. Для каждого вида кредитного договора должно существовать столько тестов, сколько заведено тарифицируемых видов услуг под этот вид кредитного договора.

В случае если тест сваливается с ошибкой\исключением, то лог протокола тестирования должен быть автоматически переслан по электронной почте на почтовый ящик разработчика\группы разработчиков, указанный в настройках.

Отправка версий ПО в банки без проведения автоматизированных тестов запрещена!

1. Нагрузочное тестирование заведенных сущностей.

Для тех сущностей, для которых предусмотрено заведение\изменение их свойств через REST-сервисы, должны быть созданы нагрузочные тесты. Параметры таких тестов должны быть согласованы с банками.

1. ~~Автоматизированное тестирование SQL запросов и функций~~
2. Архитектура модулей\приложений системы

Каждый прикладной модуль\приложение должен состоять из набора микросервисов. Каждый из микросервисов должен предоставлять доступ к определенному функционалу (функциям и процедурам). Примеры микросервисов:

* доступ к функциям чтения и записи используемой БД
* доступ к системным настроечным параметрам модуля\приложения через функционал log4j
* доступ к справочникам тарифов
* доступ к картотеке тарифных планов
* доступ к справочнику используемых классов сущностей
* доступ к справочнику используемых классов действий над сущностями
* доступ к коллекции блокировок редактируемых сущностей
* доступ к бизнес-правилам редактирования сущностей
* доступ к справочникам актуальным для уровня entity
* доступ к справочникам актуальным для уровня entity-contract
* доступ к справочникам актуальных объектов-обязательств
* доступ к картотеке клиентов
* доступ к сервису мониторинга состояния БД
* доступ к сервису отвечающего за отправку почтовый сообщений
* доступ к сервису отвечающего за оправку смс-сообщений
* доступ к справочникам актуальных алгоритмов построителей графиков платежей
* доступ к справочникам правил округлений сумм начисленных %
* доступ к справочникам шаблонов назначений платежей

1. Требования к программной разработке приложений

* Разработка функционала модулей должна вестись в строгом в соответствии с парадигмой ООП.
* Широкое использование функционала Java 8 – Stream API, Lamda-Expressions, Generics, Collections, Annotations, Reflections, Enterprice Beans.
* Использование формата JSON в обмене сообщений черeз REST сервисы
* Обязательная регистрация входящих\исходящих JSON сообщений в БД (отключается через настроечные параметры приложения)