

# Управление распределенным складом

Архитектор 1С



**Меня хорошо видно  
& слышно?**



# Защита проекта

## Тема: Управление распределенным складом



**Гыдилика Александр**

Ведущий программист-разработчик  
1С

17 лет работы в сфере 1С

# План защиты

Цель и задачи проекта

Какие технологии использовались

Что получилось

Выводы

Вопросы и рекомендации



# Цель и задачи проекта

Цель проекта: проект предусматривает ведение складского учета в распределенных узлах и управляемого из центрального офиса

1. Реализация складского учета товара (приемка, перемещение, списание)
2. Фактическое выполнение складских операций на распределенных узлах
3. Возможность оперативного увеличения количества узлов (складов)
4. Отражение результатов работы со складскими документами в центральной базе
5. Применение механизмов архитектурного проектирования, разработки, тестирования



# Какие технологии использовались

1. 1С предприятие 8, 1С: EDT
2. Github + gitExtension
3. Docker+Kafka
4. VS Code + плагины
5. Vanessa Automation + YAxUnit
6. прочее



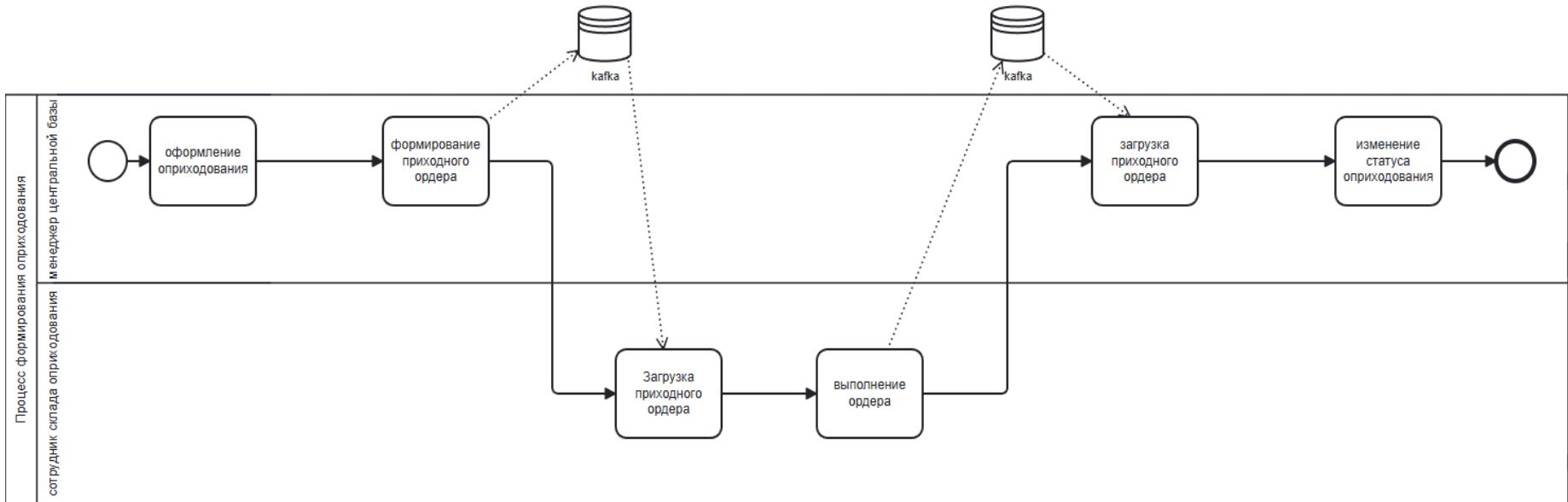
# Что получилось

При помощь инструмента **VS Code**  
+ **plant UML** разработана  
диаграмма обмена между  
системами

Диаграмма обмена систем

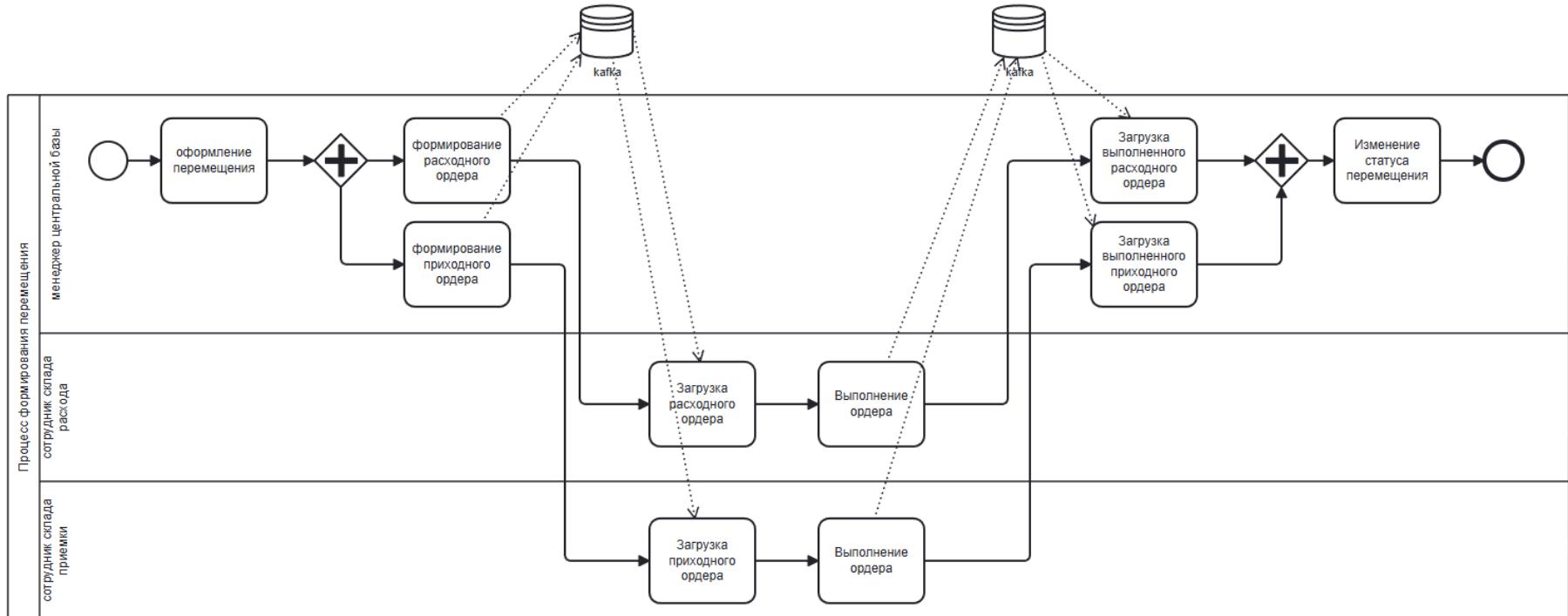


# Процесс обмена документа "Оприходование"

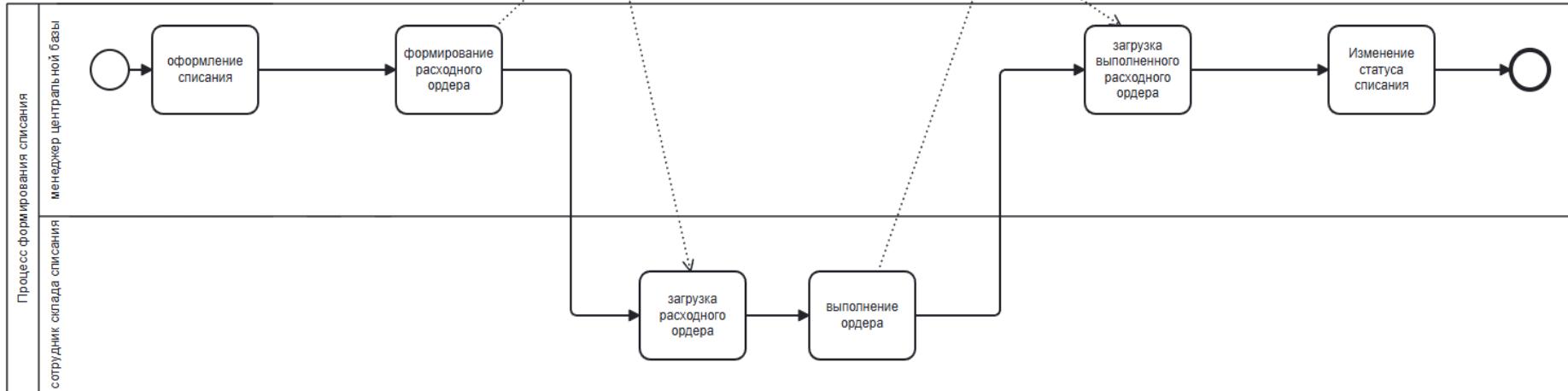


Применение системы BPMN позволяет реализовать схему процесса

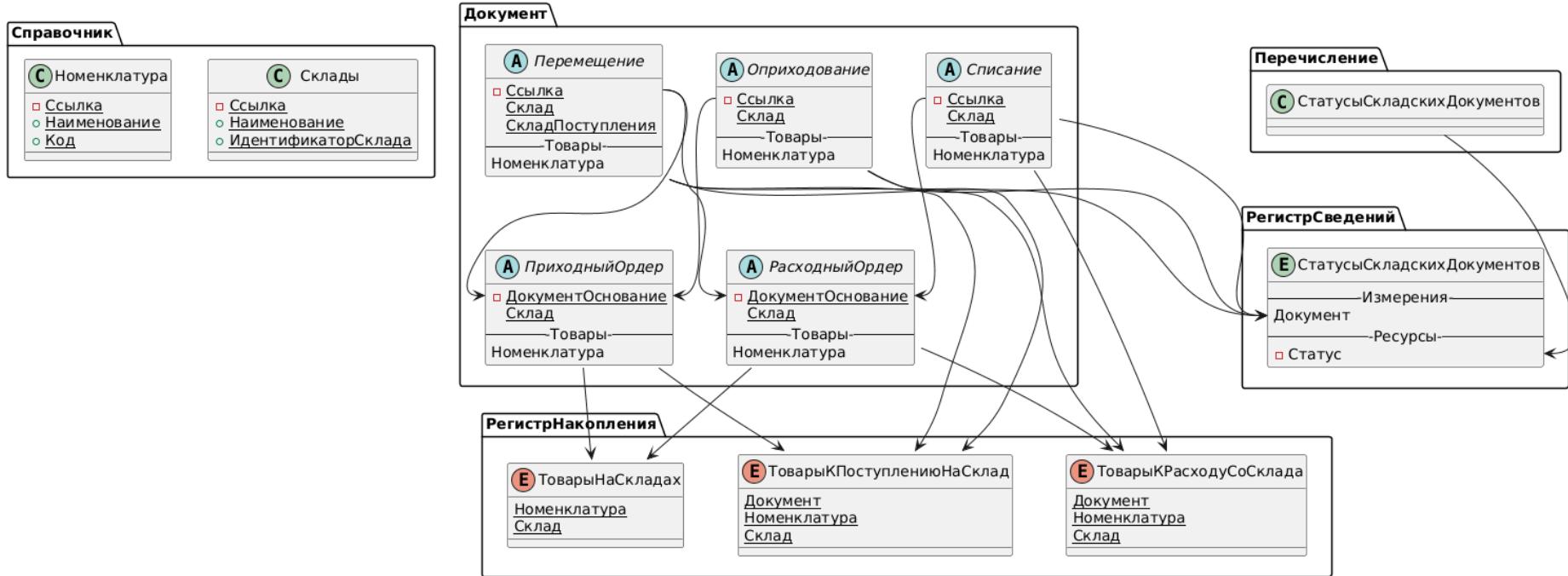
# Процесс обмена документа "Перемещение"



# Процесс обмена документа "Списание"



# ER-диаграмма



ER-схема реализована с помощью внешней обработки построения схем



# Общий вид форм

Внешний вид форм  
максимально упрощен ввиду  
специфики операций и  
невысоким навыкам людей,  
которые будут с ними работать

Перемещение товаров

Провести и закрыть   Записать   Провести

Номер: 1 от: 30.11.2025

Организация: Наша фирма

Склад отправитель: Центральный склад

Склад получатель: Филиал1

Добавить   Изменить   Открыть серийные номера

Номенклатура	Характеристика	Серия	Ед. изм.	Количество
Наш товар 1	белый		шт.	1
Наш товар 1	черный		шт.	1

Ответственный: Вася Пупкин

Комментарий:



# Сценарное тестирование

Инструмент Vanessa-automation дает возможность протестировать пользовательские сценарии взаимодействия в разных базах

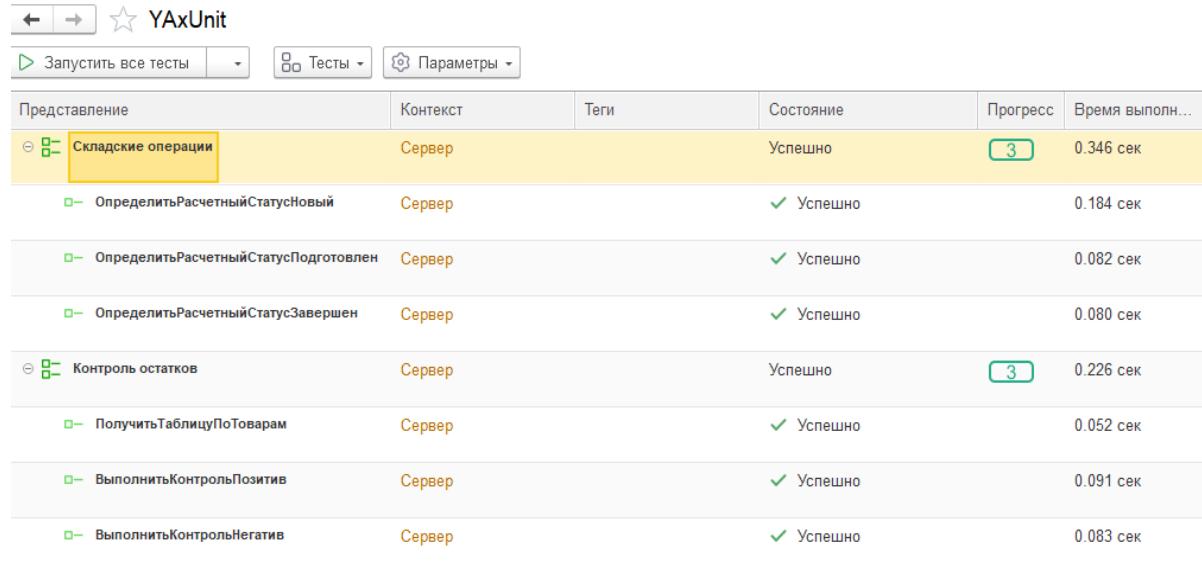
```
Добро пожаловать! × 1_0_СозданиеИОбменДокумента ×

1 #language: ru
2
3 @Синхронизация
4
5 Функционал: Обмен документа
6
7 Контекст:
8   Дано я подключаю TestClient "Этот клиент" логин "Admin" пароль ""
9   И я закрываю все окна клиентского приложения
10
11 Сценарий: Создание документа и его выгрузка
12   *Создаем документ
13     И В командном интерфейсе я выбираю "Склад" "Оприходование"
14     Тогда открывается окно "Оприходование"
15     И я нажимаю на кнопку с именем 'ФормаСоздать'
16     Тогда открывается окно "Оприходование (создание)"
17     И из выпадающего списка с именем 'Склад' я выбираю точное значение "Филиал 1"
18     И в таблице 'Товары' я нажимаю на кнопку с именем 'ТоварыДобавить'
19     И в таблице 'Товары' из выпадающего списка с именем 'ТоварыНоменклатура' я выбираю точное значение "Элемент 105"
20     И я перехожу к следующему реквизиту
21     И в таблице 'Товары' в поле с именем 'ТоварыКоличество' я ввожу текст "7,000"
22     И в таблице 'Товары' я завершаю редактирование строки
23     И я нажимаю на кнопку с именем 'ФормаПровести'
24     Тогда открывается окно "Оприходование * от *"
25     И я нажимаю на кнопку с именем 'ФормаДокументПриходныйОрдерСоздатьНаОсновании'
26     Тогда открывается окно "Приходный ордер (создание)"
27     И я нажимаю на кнопку с именем 'ФормаЗаписать'
28     И я запоминаю значение поля "Номер" как "НомерДокумента" {"НомерДокумента": "000000033"}
29     И я нажимаю на кнопку с именем 'ФормаПровестиИЗакрыть'
30   *Выгружаем документ
31     И В командном интерфейсе я выбираю "Синхронизация" "Синхронизация кафа"
32     Тогда открывается окно "Синхронизация кафа"
33     И в таблице 'Список' я перехожу к строке:
34       | "Адрес сервера" | "Код" | "Назначение" | "Наименование" | "Склад" | "Топик" | "Центральный офис" |
35       | "kafka:9092" | "00000002" | "Документы" | "Документы" | "Центральный склад" | "docs" | "да" |
36     И в таблице 'Список' я выбираю текущую строку
37     Тогда открывается окно "Документы (Синхронизация кафа)"
38     И я нажимаю на кнопку с именем 'ВыгрузитьДанные'
39   *Переходим в подчиненную базу
40     И я подключаю TestClient "Филиал" логин "Admin" пароль ""
```



# Unit-тесты

Благодаря Unit тестам всегда можем выявить ошибки как при разработке нового, так и при изменении текущего функционала. Своевременная разработка таких тестов способствует написанию сложных алгоритмов путем разбиения на множество более простых



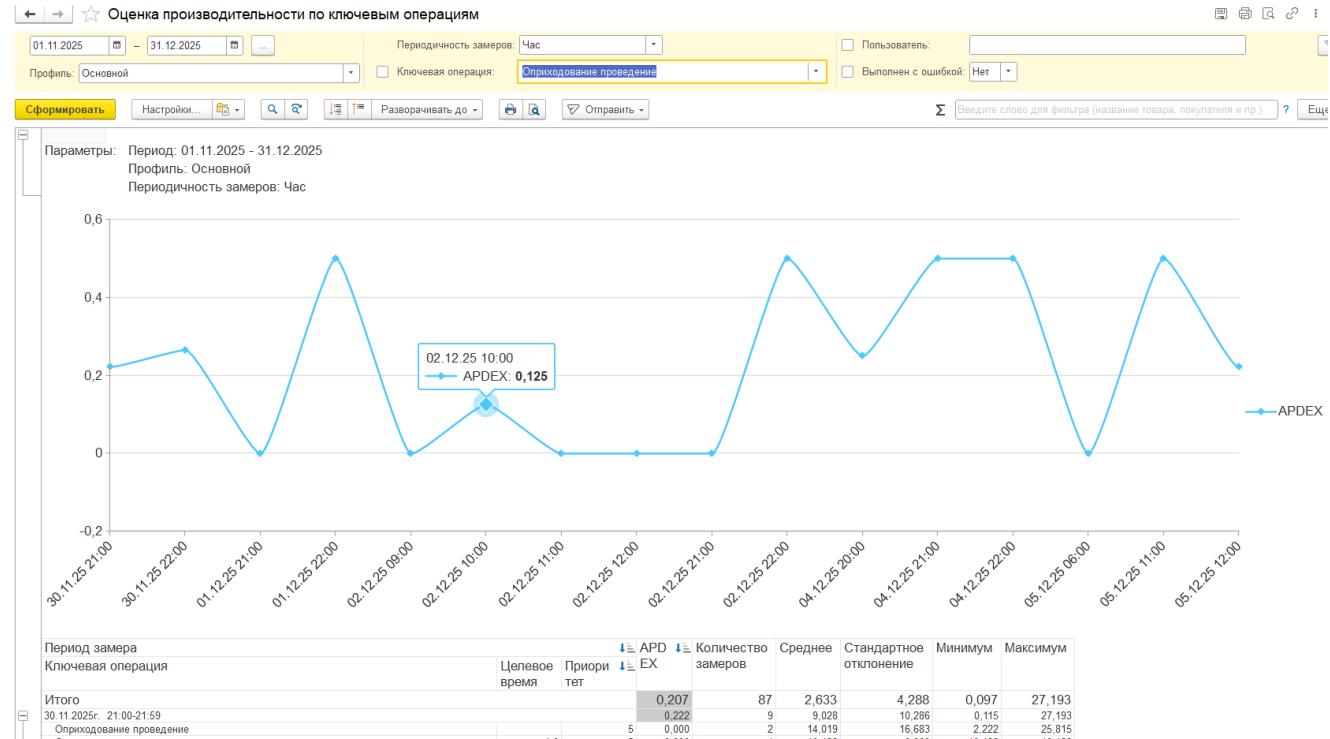
The screenshot shows the YAxUnit test runner interface. At the top, there are navigation buttons (back, forward, search), a title bar with the logo and name, and dropdown menus for 'Запустить все тесты' (Run all tests), 'Тесты' (Tests), and 'Параметры' (Parameters). The main area is a table with the following columns: Представление (Representation), Контекст (Context), Теги (Tags), Состояние (Status), Прогресс (Progress), and Время выполн... (Execution time). The table lists two test suites: 'Складские операции' and 'Контроль остатков'. Under 'Складские операции', three tests are listed: 'ОпределитьРасчетныйСтатусНовый', 'ОпределитьРасчетныйСтатусПодготовлен', and 'ОпределитьРасчетныйСтатусЗавершен'. All three tests are marked as 'Успешно' (Successful) with execution times of 0.184 sec, 0.082 sec, and 0.080 sec respectively. Under 'Контроль остатков', three more tests are listed: 'ПолучитьТаблицуПоТоварам', 'ВыполнитьКонтрольПозитив', and 'ВыполнитьКонтрольНегатив'. All three tests are also marked as 'Успешно' with execution times of 0.052 sec, 0.091 sec, and 0.083 sec respectively.

Представление	Контекст	Теги	Состояние	Прогресс	Время выполн...
Складские операции	Сервер		Успешно	3	0.346 сек
└ ОпределитьРасчетныйСтатусНовый	Сервер		Успешно		0.184 сек
└ ОпределитьРасчетныйСтатусПодготовлен	Сервер		Успешно		0.082 сек
└ ОпределитьРасчетныйСтатусЗавершен	Сервер		Успешно		0.080 сек
Контроль остатков	Сервер		Успешно	3	0.226 сек
└ ПолучитьТаблицуПоТоварам	Сервер		Успешно		0.052 сек
└ ВыполнитьКонтрольПозитив	Сервер		Успешно		0.091 сек
└ ВыполнитьКонтрольНегатив	Сервер		Успешно		0.083 сек



# APDEX

Встроенные модули  
БСП позволяют  
анализировать  
производительность  
механизмов и  
своевременно  
принимать меры по  
оптимизации  
функционала



# Выводы

1. Комплексное использование современных механизмов проектирования и разработки упорядочивает процесс и позволяет избегать ошибок построения бизнес-логики, программирования, конфликтов взаимодействия
2. Применение технологии Git позволяет избегать потерь доработок и других полезных файлов. Применение шаблона упорядочивает хранение данных проекта
3. Автоматизированное тестирование ускоряет процесс разработки на дистанции
4. Построение схем и процессов до начала разработки упрощает сам процесс



# Вопросы и рекомендации



если есть вопросы



если вопросов нет

**Спасибо за внимание!**

