UML2.ru Конференция

CUSTIS®

Опыт CUSTIS

Domain Driven Design
модель вместо требований

Докладчик:

Максим Цепков (M.Tsepkov@custis.ru)
www.CUSTIS.ru

О чем будет доклад?

Теория

DDD – mainstream проектирования

- Что это такое?
- Почему это эффективно?
- Когда его применять?

Опыт CUSTIS

DDD для корпоративных приложений

- Какие применяем модели?
- Как организуем процесс?
- Что дает заказчику?

Domain Driven Design

Что такое DDD ?

DDD – проектирование по модели

Строим модель предметной области,
вырабатываем для этого единый язык,
понимаемый всеми участниками включая заказчика

Воспроизводим модель в архитектуре программы
и коде – соответствие должно быть очевидным

Что нового в DDD ?

Единый язык для коммуникаций

Проектирование по модели было давно…

- MDA, MDD, MBRD, MBRA…

Моделей обычно несколько

- Бизнес-модель → Логическая модель → Техническая модель

DDD внес единый язык (Ubiquitous Language)

- Его понимают заказчики, аналитики, разработчики
- Его понятия можно проследить в бизнес-области
- Его элементы можно проследить в коде

Что дает Единый язык?

Заказчик владеет единым языком
и понимает модель без перевода

Разработчик может реализовать модель
в коде без дополнительного проектирования

Модель можно проследить в бизнесе и в коде

P.S. Аналитик тоже есть – он строит модель

Как работает DDD ?

Аналитик собирает требования и строит модель

– Вместе с моделью – строит единый язык

– Построение – итеративно

– Требования – короткоживущи, они лишь объясняют решения

Заказчик проверяет модель

– Устраняется основной риск разработки – ошибки в постановках

– Что-то неверно или изменилось – модель меняется

Разработчик реализует модель в программном коде

– Непосредственно, если парадигма моделирования имеется в среде

– Или применяя шаблоны для отображения

– Техническое проектирование – локально

Модель предметной области
становится моделью системы

Когда применять DDD просто?

Предметная область хорошо описывается
в известной парадигме моделирования,
имеющей средства наглядного представления
и поддержанной языками или средой разработки

Пример такой области – документооборот

– Хорошо описывается в объектной парадигме моделирования
– ООП – известный и хорошо проработанный подход
– Есть способы эффективного представления – диаграммы UML
– Объектная парадигма реализована современными языками

А если ООП не работает?

Можно использовать другие парадигмы

– надо заботится о средствах наглядного представления

– вероятно, парадигмы надо будет изучать разработчикам

– будет меньше готовых шаблонов проектирования

– надо позаботится об отражении в код

Но если такова предметная область – что делать…

Подробнее о парадигмах мой доклад на ADD-2011
http://lib.custis.ru/Non-object-models

Подведем итоги…

Почему DDD – эффективно?

Есть артефакт, описывающий систему – ее модель

Есть язык описания, понимаемый всеми

Для коммуникаций – не нужно сложного перевода

Требования на изменения

– формулирует эксперт заказчика на едином языке, часто меняя модель

– и реализует разработчик, уточняя по необходимости

Но разработчику приходится осваивать бизнес-язык?

– Это не слишком сложно – языки программирования изощреннее

– И помогает реализовывать программы, удобные пользователям

Когда применять DDD

Особенно эффективно – для длинных проектов

- Модель позволяет хорошо развивать систему много лет после внедрения

Если компания работает в одной предметной области

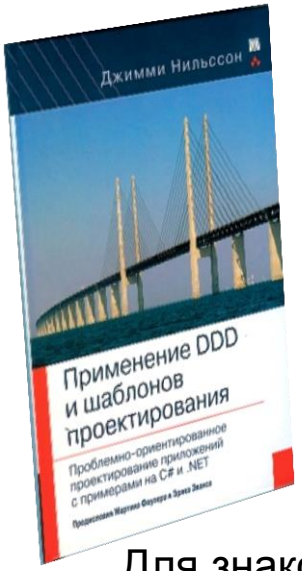
- Разработчики начинают в ней разбираться, это сильно повышает эффективность их работы

DDD: источники для знакомства

Концептуальная книга Эрика Эванса

– на английском – в 2003 г.

– на русском – только в 2010 г.



Практическая книга Джимми Нильссона

– на английском – в 2006 г.

– на русском – в 2007 г. (почти сразу!)

Для знакомства – можно смотреть материалы, ссылки, слайды и видеозапись тренинга Андрея Бибичева: http://lib.custis.ru/ddd-training

Опыт CUSTIS

DDD для корпоративных приложений

Типичное корпоративное приложение

Пользователи создают документы

По необходимости заполняют справочники

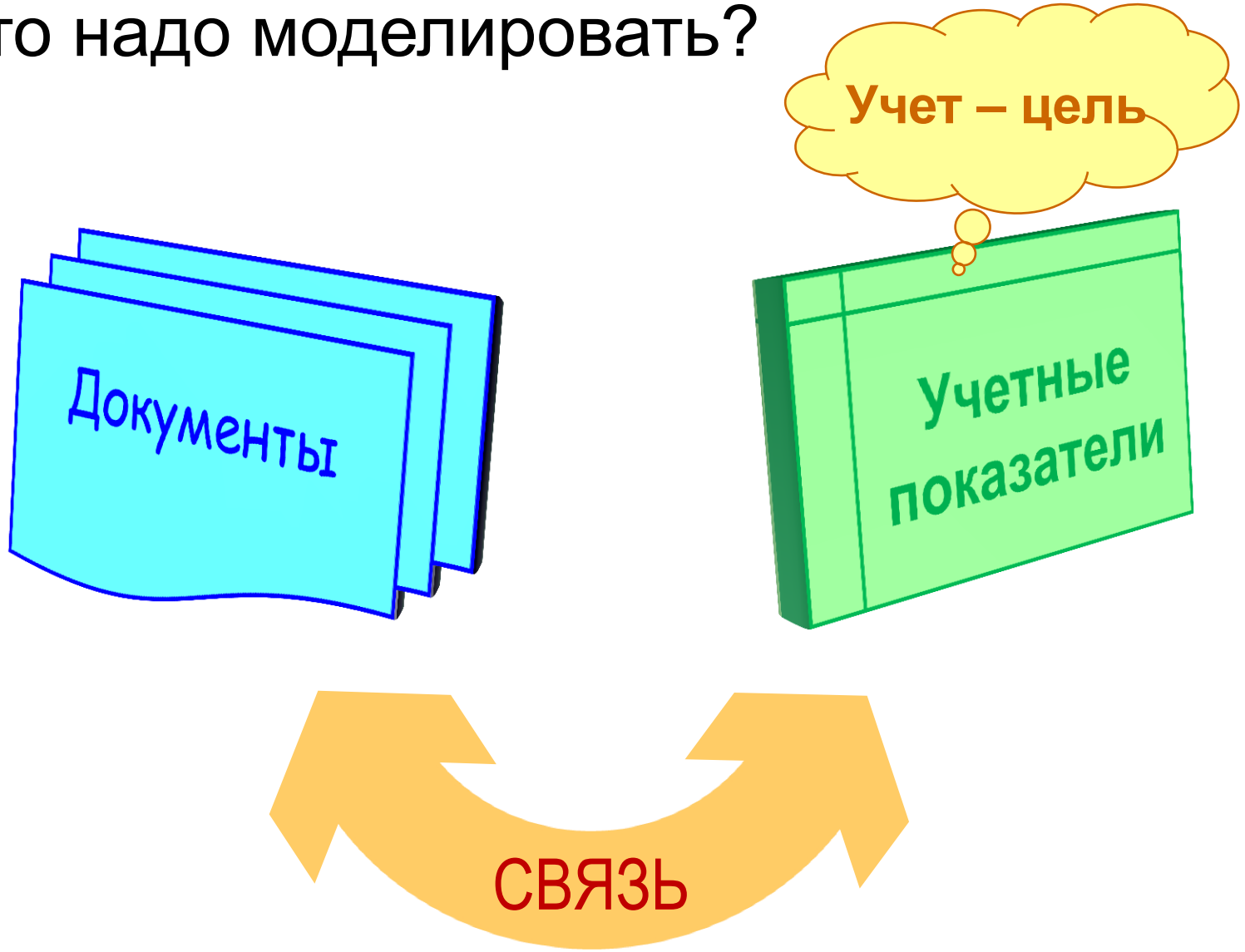
Потом документы исполняют

При этом меняются учетные данные

Которые влияют на исполнение документов

И отражаются в отчетах

Что надо моделировать?



Учетная модель

Учетная модель – не объектная

Сложность объектного представления учета

– Нет идентификации единичного объекта
– Работа идет с показателями, текущее значение которых меняется
– Изменение числового значения может менять состояние с точки зрения принятия бизнес-решения
– Часто интерес представляют агрегаты, а не отдельные значения

Представление учета оказалось за рамками UML

Однако, DDD можно применять для учетной модели

Учетная модель

Элементы учета

– Синтетические счета и их аналитика

– Проводки

– Показатели – остатки и обороты

Для представления учетной модели
мы придумали Диаграммы учета

Диаграммы показывают

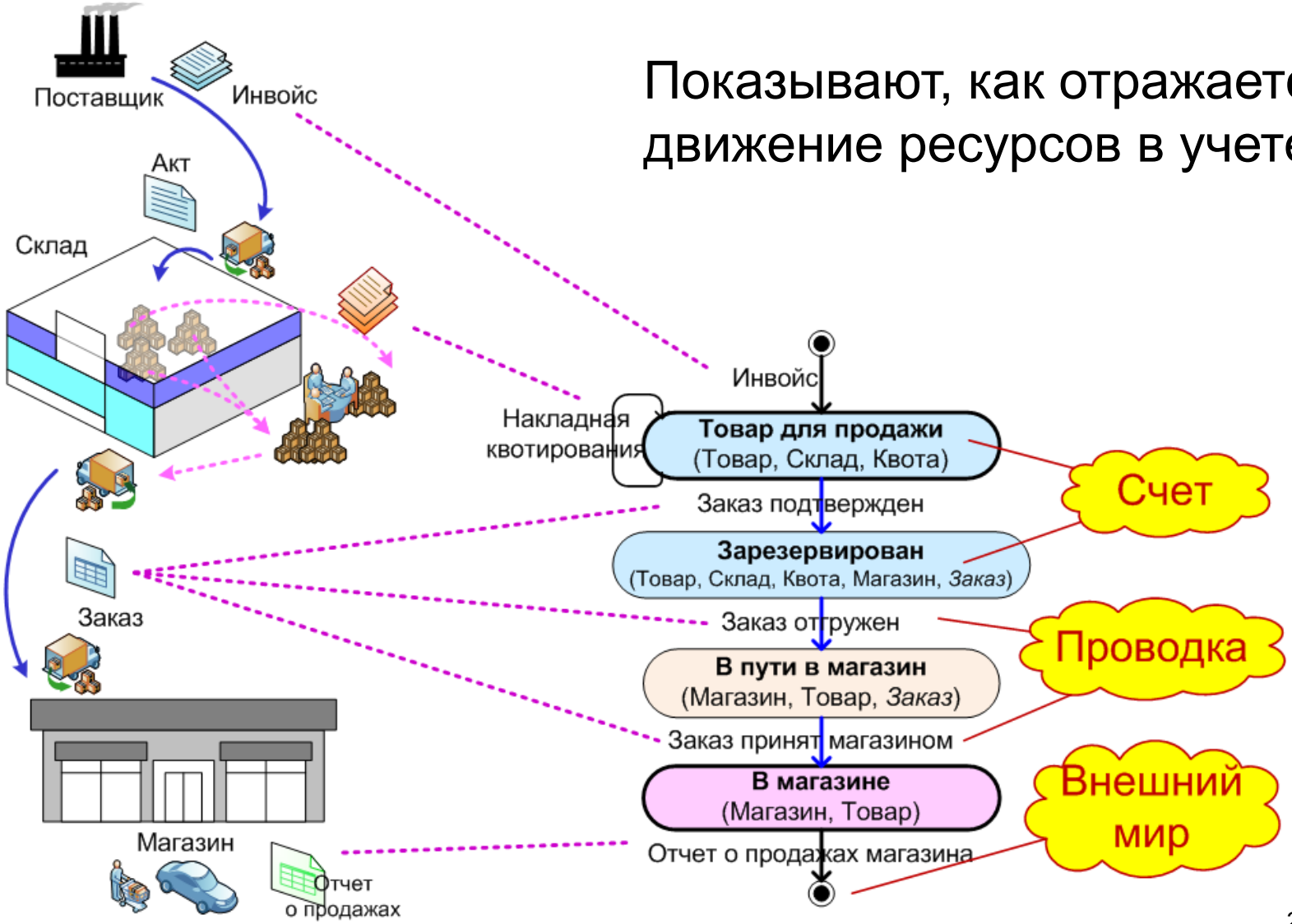
– как проводки перемещают ресурсы по синтетическим счетам

– какая аналитика счетов

– с какими событиями связано исполнение проводок

Диаграммы учета

Показывают, как отражается движение ресурсов в учете



Как живет учетная модель?

Уточнения

– определяем аналитики

– определяем источники событий учета

Изменения

– добавляем новые участки учета

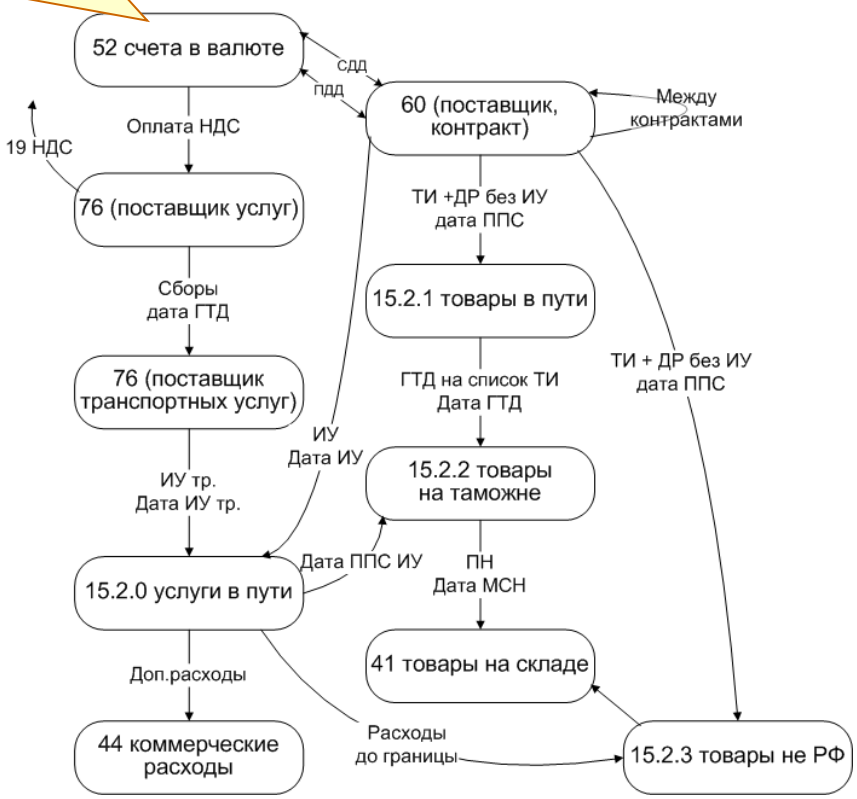
– добавляем транзитные счета

Диаграммы учета – заказчику

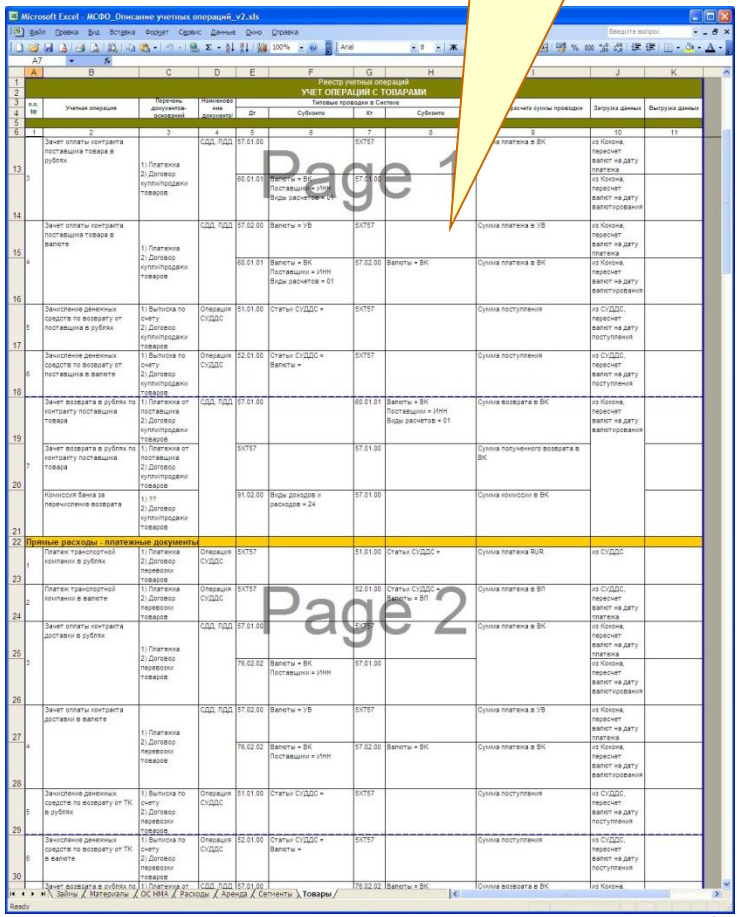
Бухгалтеры могут применять диаграммы планов счетов для разработки учетной политики, они нагляднее, чем excel

А здесь несколько рисунков

Рис. I. Расчеты с поставщиками



И так много страниц…



Подробно о диаграммах учета

Я рассказывал год назад…

ЛАФ-2010 – http://lib.custis.ru/Accounting-diagrams

Презентация и видео

«Диаграммы планов счетов –
средство моделирования и проектирования учета»

Презентация и статья

И еще осенью…

SECR-2010 – http://lib.custis.ru/Simplify-security-accounting

«Учет ценных бумаг – сделать сложное простым»

А недавно вышла статья
http://lib.custis.ru/Когда_всем_понятно

Журнал «Бухгалтер и компьютер» №5 (май) 2011

Отражение учетной модели в код

Эффективно выполняется через шаблоны

Есть Patterns for Accounting Мартина Фаулера – отражение учета в объектную реализацию

У нас – более развитая реализация и
есть собственный язык описания – GL-XML

Документы и Справочники

Хорошо работает объектная модель

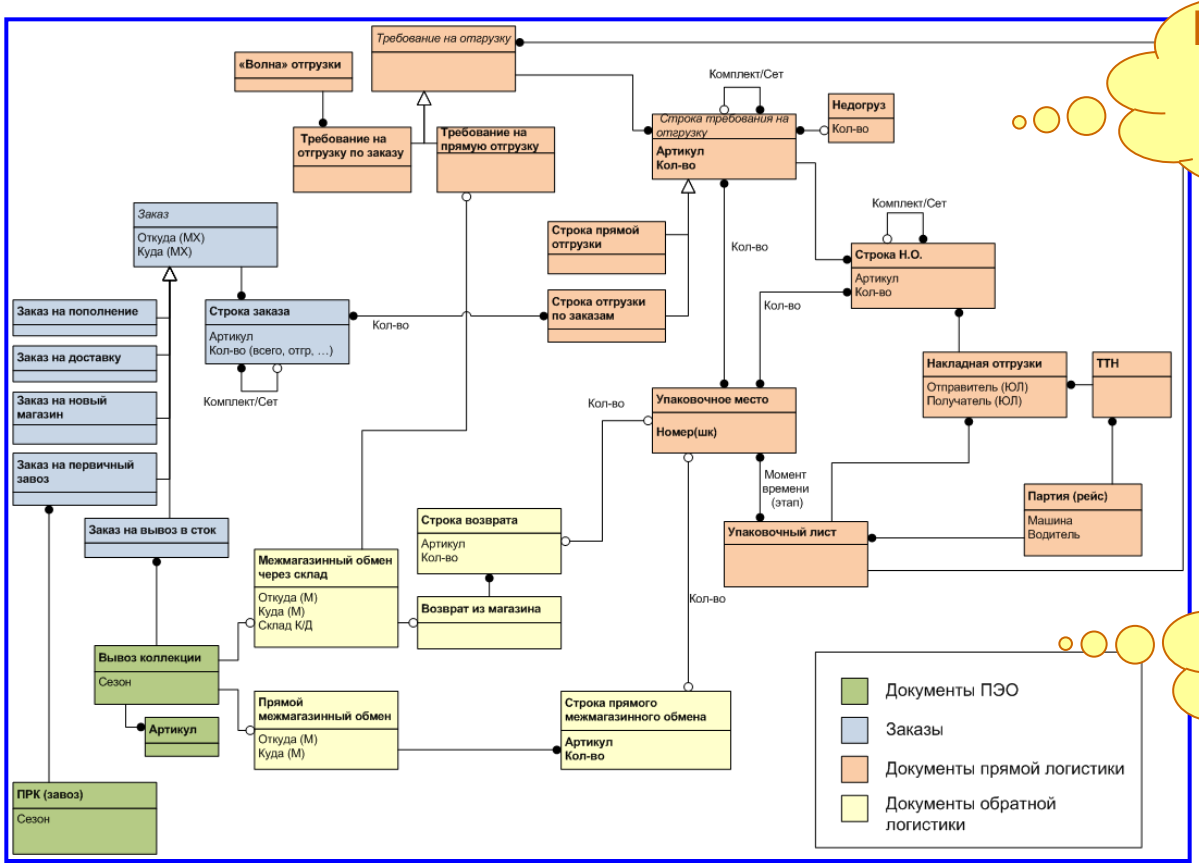
Объекты с атрибутами и методами –
элементы единого языка для предметной области
и способ их соединения в сложные конструкции

Диаграмма классов и другие диаграммы UML –
визуальный образ для наглядного представления

Объекты в программе –
способ отражения модели в реализацию

Модель должен понимать заказчик

Классы соответствуют бизнес-объектам – заказчик видит знакомые названия



Представляем диаграммой классов

Используем цветовые выделения

Модель развивается итеративно

Сначала – в крупном, основные классы без деталей

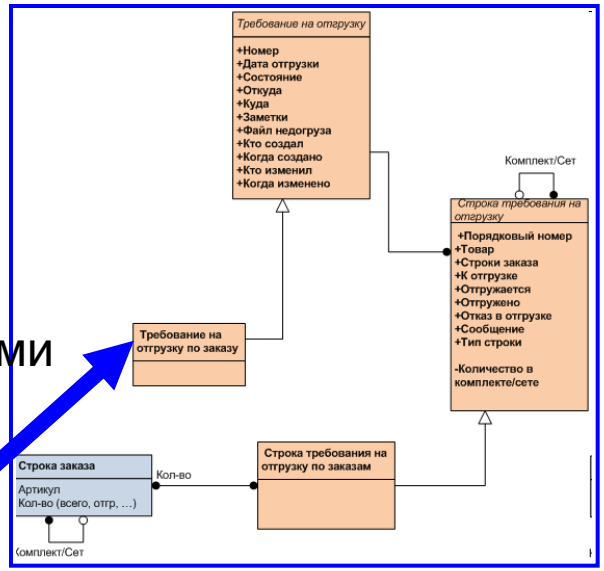
Подробности – по ходу разработки

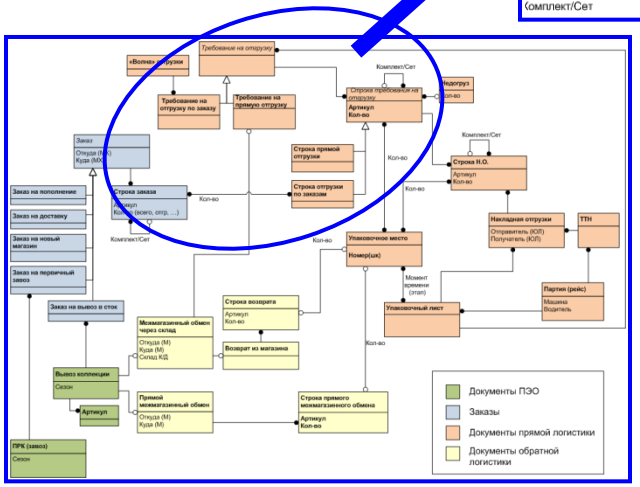
– Уточняем иерархию типов

– Определяем атрибуты и методы

– Проектируем вспомогательные классы

– Уточняем структуру ассоциаций между классами





Связь документа с учетом

Учет следует из документов

В ПРЕДМЕТНОЙ ОБЛАСТИ

Есть **журнал хозяйственных операций**, возникающих при обработке и проведении документов

ПРИ РЕАЛИЗАЦИИ

источник учета – **события** (Event Sourcing Фаулер), они возникают в методах документов

Для прозрачной модели это **должно** совпадать:
учетные события – суть хозяйственные операции

Модель – состояния документов

Шаблон State Entity

Документу приписываем **состояние**

Состояние определяет отражение в учете и другие аспекты документооборота

– текущий этап обработки документа

– какие действия можно совершать над документом

– кто отвечает за обработку документа

– кто имеет права на совершение тех или иных действий

Возможные изменения состояний документа образуют граф переходов

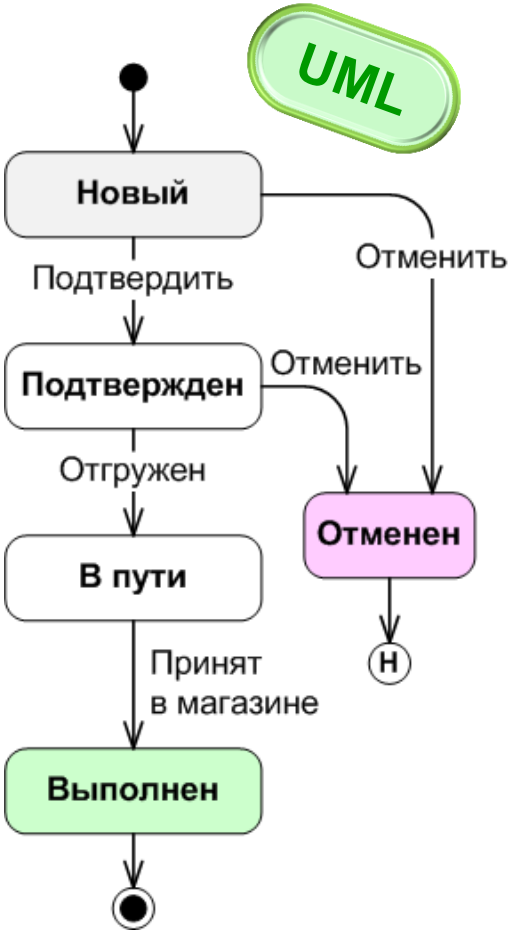
На переходах создаются хозяйственные операции

Язык модели

Состояния документа и методы – переходы между ними

Граф состояний – State machine diagram

Названия состояний и переходов – на языке бизнеса



Получилась единая модель

Единая модель

Назвали Учетной машиной

Модель представляется в проекциях

– диаграмма классов – структура объектов и их методы
– диаграмма учета – учетная модель системы
– диаграмма состояний документов – документооборот

В терминах предметной области

Связь проекций

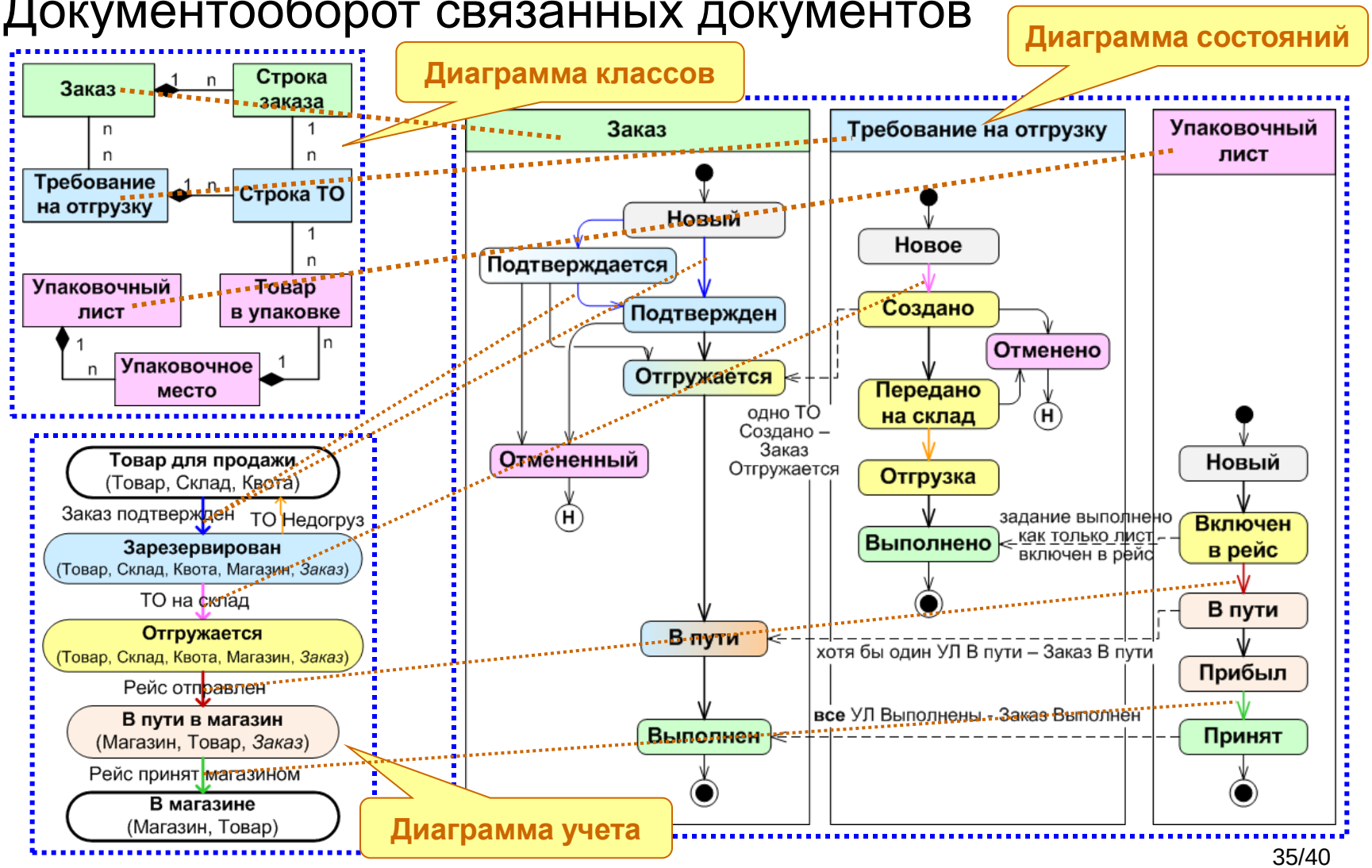
– проводки в учете связаны с переходами документов
– переходы выполняются методами документов

Разрабатывается итеративно

– сначала – основные классы и учет в крупном
– затем, **по областям** – детальная проработка классов и учета, проработка состояний документов

Пример сложной модели

Документооборот связанных документов



Эванс выделяет
диаграммы – иллюстрации,
не образующие модель

Модель или иллюстрация?

Диаграмма – не всегда модель

Когда диаграммы становятся моделью?

Входят в единый язык –
их понимают и разработчики и заказчик

Их можно сопоставить с реализацией, то есть кодом

Реализация – применение шаблона,
быть может с локальным проектированием

Иначе это не модель, а иллюстративные диаграммы
Они тоже нужны, но их стоит отличать от модели

ЗАКЛЮЧЕНИЕ
(основная мысль)

Модель + Единый язык – эффективно

Domain Driven Design – архитектура на основе модели

На единном языке предметной области,
понятном и заказчику и разработчикам,
с использованием диаграмм

Domain Driven Design – не только объектные модели

Модель позволяет эффективно развивать систему,
заменяя мелкую россыпь требований

Спасибо!

Вопросы?

Максим Цепков (M.Tsepkov@custis.ru)