Трехслойная архитектура приложений, основанных на использовании баз данных

Критериями качества при проектировании систем, использующих базы данных являются:

- *построение системы с **любым уровнем сложности бизнес-логики**, сохраняя при этом четкую структурированность системы;*
- *возможность существенного **расширения функциональности системы** по ходу дальнейшего жизненного цикла продукта **без существенного перепроектирования системы**, простота поддержания системы;*
- *возможность **параллельной работы целых групп разработчиков** над различными аспектами приложения и **быстрое согласование результатов** их работы*

- ***гибкое поддержание принципов распределения программных компонентов и данных*** *(например, возможности быстрого перевода системы на модель функционирования «клиент-сервер», разработка параллельно нескольких интерфейсов к системе, в том числе, и Web-интерфейса);*
- ***четкая структурированность*** *бизнес-логики;*

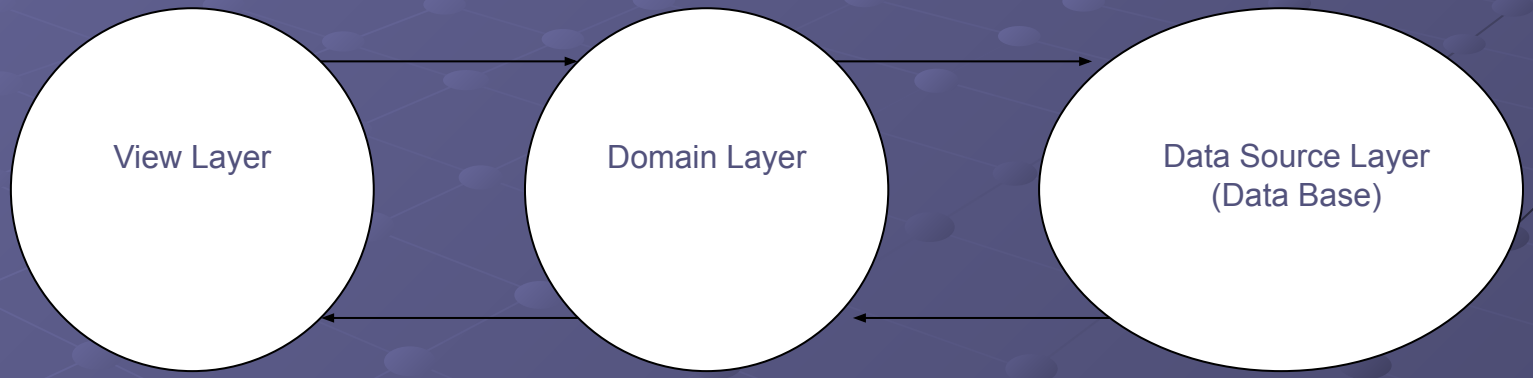
Любая большая система, основанная на использовании БД, должна отвечать этим требованиям, а если быть точнее, то уровень качества проектирования этой системы во многом зависит от того, насколько система отвечает вышеперечисленным требованиям

1. Трехуровневая архитектура. Определение.

Трехуровневая архитектура подразумевает под собой разделение всей системы на 3 слоя:

- **Слой представления (или «вид», “View Layer”)-*охватывает все, что имеет отношение к общению пользователя с системой***;
- **Слой источника данных(“Data Source Layer”)-*предоставление интерфейса необходимых операций взаимодействия внешних систем(как правило слоя бизнес логики) с источником данных ;***
- **Слой логики домена (бизнес-логика или логика предметной области, “Domain Layer”)-*описывает основные функции приложения, предназначенные для достижения поставленной перед ним цели***;

Упрощенная схема приложения при использовании 3х-слойной архитектуры показана на рисунке



Итак трехслойная архитектура приложения имеет следующие преимущества:

- *подход делает возможным параллельную разработку и тестирование каждого из слоев;*
- *позволяет изменять структуру, а иногда даже полностью заменять один из слоев, не изменяя при этом другие слои.*
- *Возможность разделения мест функционирования слоев на физическом уровне. Полная интеграция 3х-слойной архитектуры приложения с моделью клиент/сервер.*

Принципы организации Model Layer (бизнес - логики приложения)

- *сценарий транзакции (Transaction script);*
- *модель предметной области (Domain Model);*
- *модуль таблицы (Table Module).*

Сценарий транзакции (Transaction script)

Бизнес-логика в этом случае описывается набором процедур, по одной на каждую (составную) операцию, которую способно выполнять приложение

Преимущества подхода:

- *представляет собой удобную процедурную модель, легко воспринимаемую всеми разработчиками;*
- *удачно сочетается с простыми схемами организации взаимодействия со слоем источника данных(Data Source Layer);*
- *определяет четкие границы транзакции.*

Модель предметной области (Domain Model)

Типовое решение модель предметной области предусматривает создание сети взаимосвязанных объектов, каждый из которых представляет некую осмысленную сущность предметной области. каждый объект наделяется только функциями, отвечающими его природе

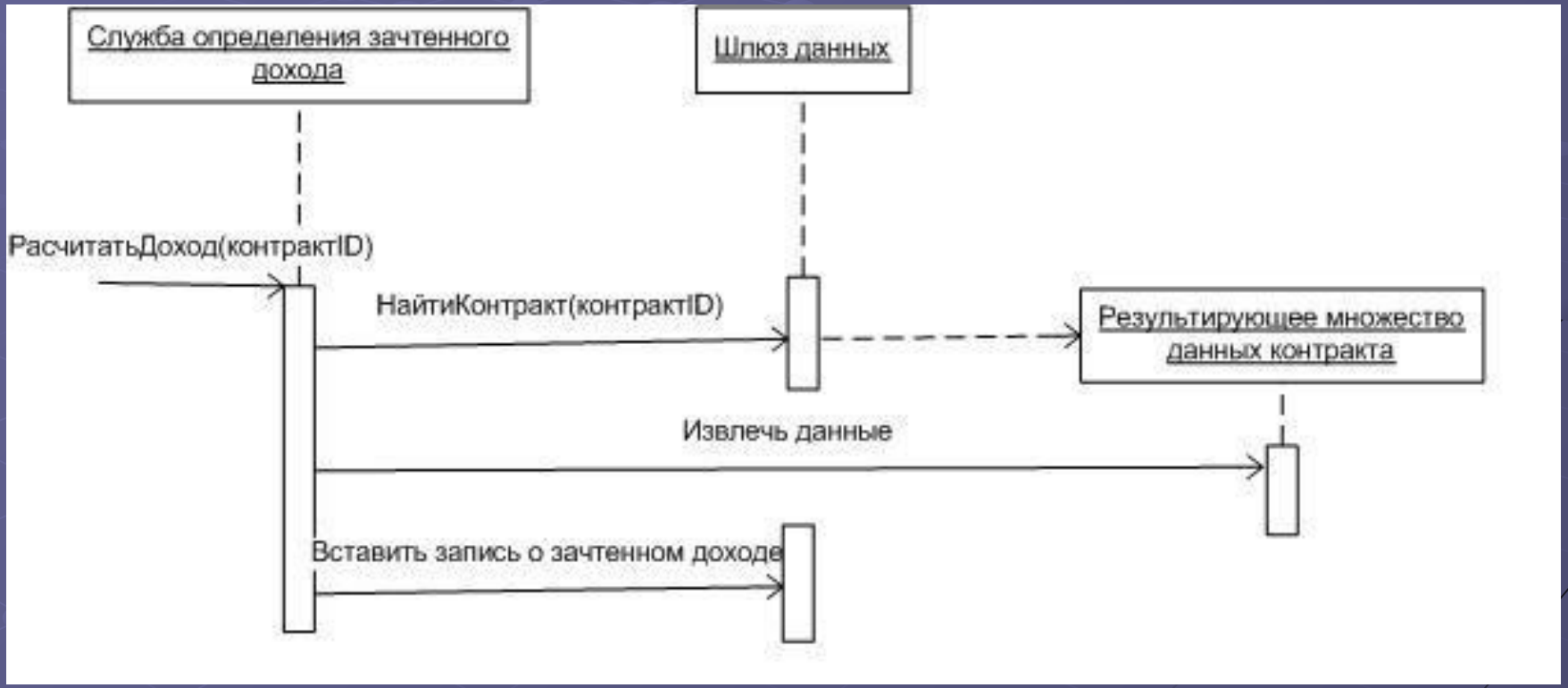
Недостаток подхода:

- *Если прежде вы не пользовались моделью предметной области, процесс обучения может принести немало трудностей, когда в поисках нужных функций вам придется метаться от одного класса к другому.*

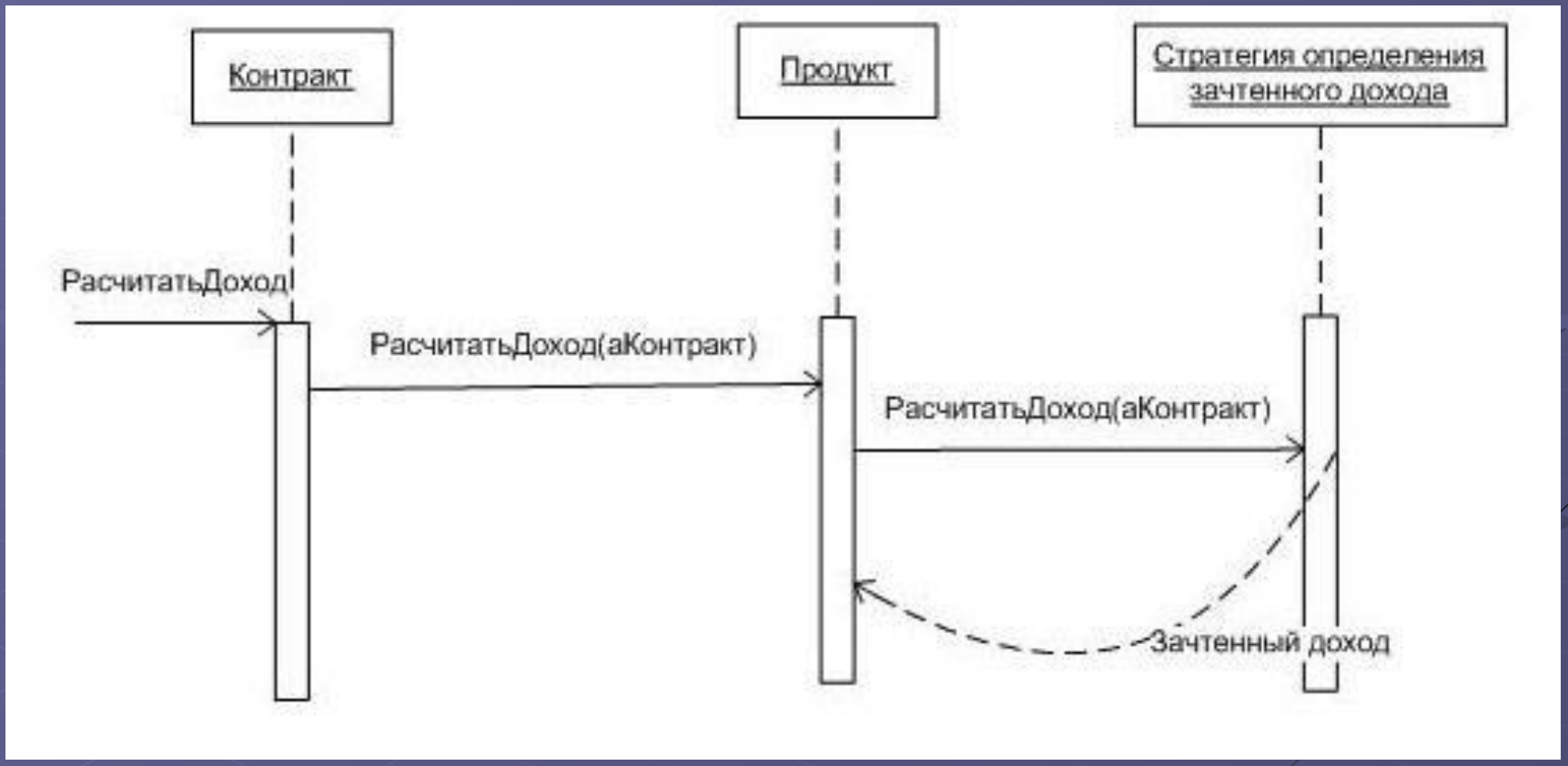
Преимущество подхода:

- *освоившись с методом, разработчик получает в свое распоряжение множество приемов,* ***позволяющих описывать возрастающую сложность бизнес-логики гибким путем с минимальными временными затратами***

Приведем схему реализации транзакции «Вычисление зачтенного договора» с помощью сценария транзакции и модели предметной области



«Вычисление зачтенного договора» с помощью сценария транзакции



«Вычисление зачтенного договора» с помощью модели предметной области

Модуль таблицы (Table Module)

Типовое решение модуль таблицы предусматривает создание по одному классу на каждую таблицу базы данных, и единственный экземпляр класса содержит всю логику обработки данных таблицы.

Преимущество подхода:

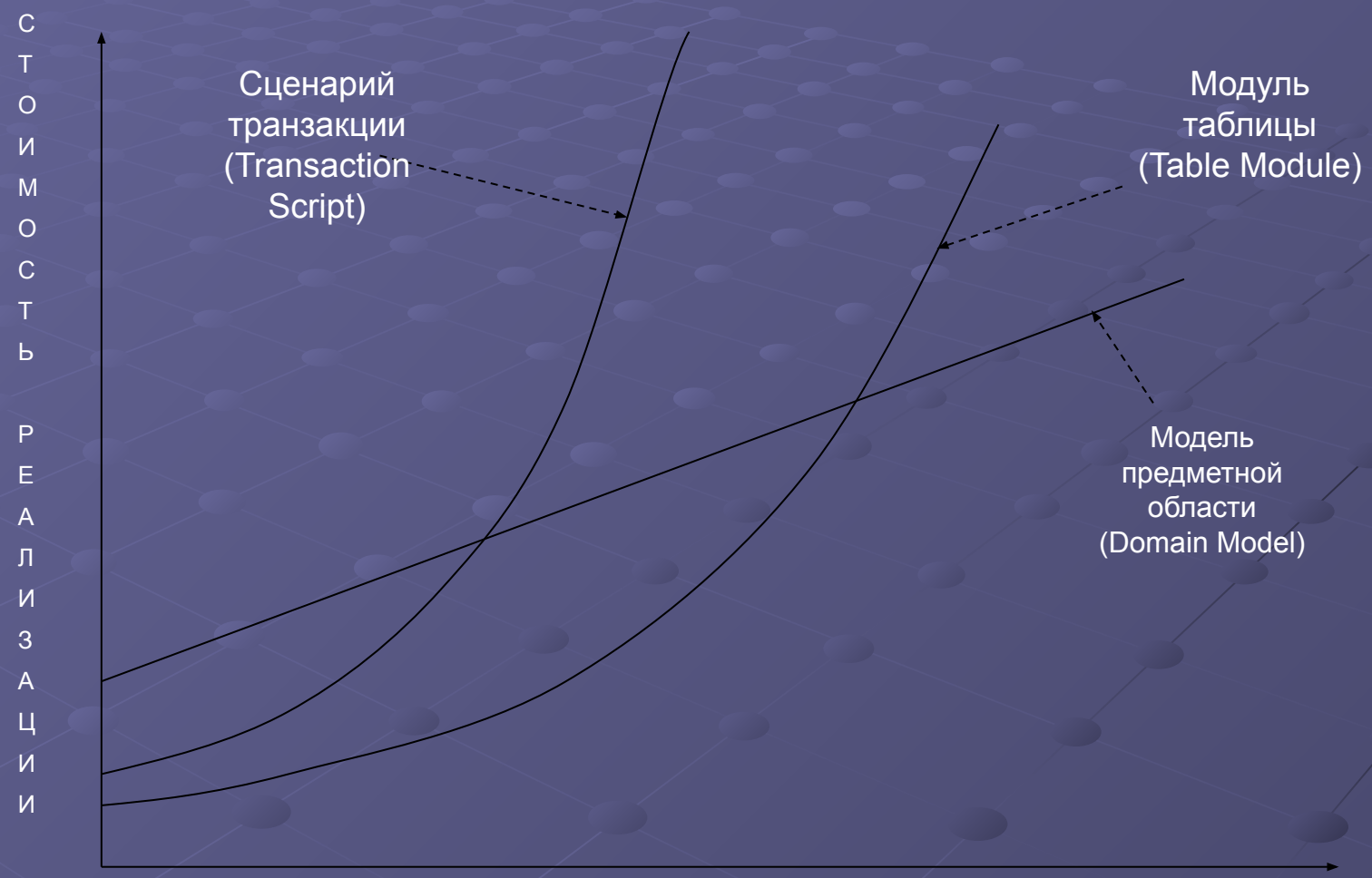
- *решение позволяет сочетать данные и функции для их обработки и в то же время эффективно использовать ресурсы реляционной базы данных.*

Недостаток подхода:

- модуль таблицы не позволяет использовать многие технологии (например, *наследование, стратегии* и другие объектно-ориентированные типовые решения), которые применяются в модели предметной области для уточнения структуры логики

Схема взаимодействия слоев кода приложения с использованием модуля таблицы

Рекомендации по выбору метода организации бизнес-логики приложения



Применение результатов

- Результаты работы будут применены при проектировании и реализации комплексной системы бухгалтерского учета с внедрением возможностей экономического моделирования и прогнозирования;
- Система будет работать в локальной сети с топологией «звезда», используя модель клиент-сервер. 1 сервер, ориентировочное число активных клиентов -70. Средняя активность клиентов – выше среднего.

Литература

- Гамма Э., Хелм Р. и др. Приемы объектно-ориентированного проектирования. Паттерны проектирования. СПб.: Питер, 2001. – 378 с.
- Фаулер М. Архитектура корпоративных программных приложений.: -М.: Издательский дом «Вильямс», 2006.-544 с.
- Г.Буч, Дж.Рамбо, А. Джекобсон Язык UML. Руководство пользователя. – М.:ДМК Пресс, 2004.-432с.

БЛАГОДАРЮ ЗА ВНИМАНИЕ!