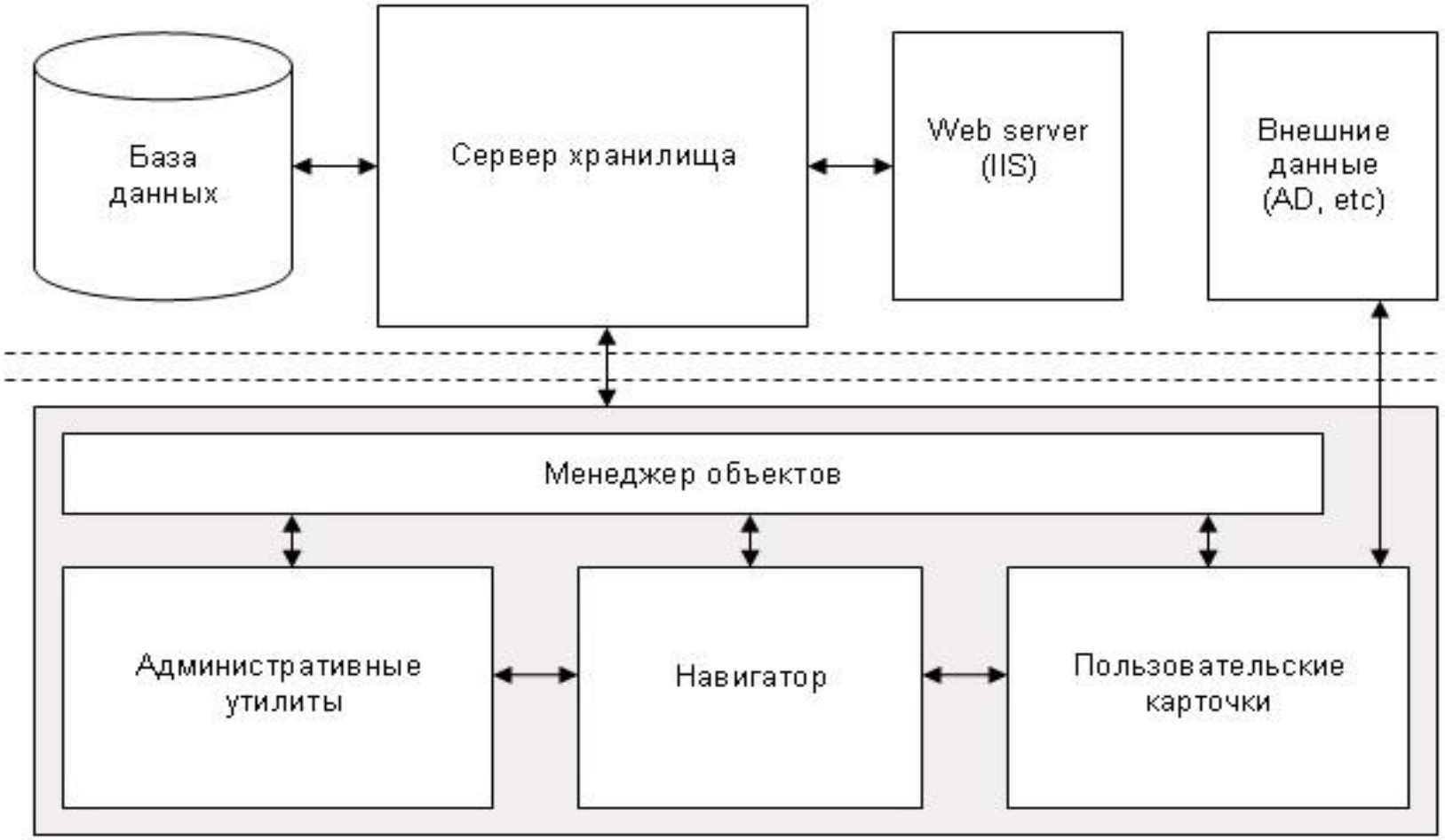
Генерация объектной модели для DocsVision и использование ее при синхронизации сервисов

Астащенко Александр, 445 группа
Научный руководитель:
В.Г.Шистеров

DocsVision

- **DocsVision** – корпоративная система электронного документооборота, позволяющая, автоматизировать бизнес-процессы, ведение делопроизводства и электронный документооборот в организации.

Архитектура DocsVision



Работа с DocsVision.Platform

- Создаем сессию:
  - var sessionManager = SessionManager.CreateInstance();
  - var session = sessionManager.CreateSession();

- Для доступа к данным используется CardManager
  - session.CardManager.GetCardData(Guid);
  - session.CardManager.GetCardDictionaryData(Guid);

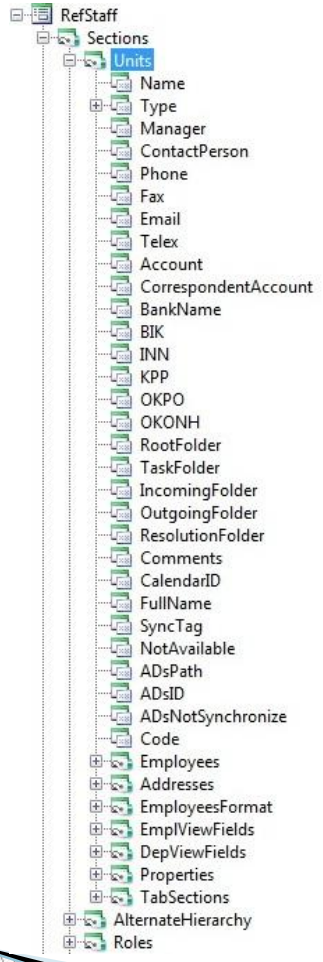
Пример работы со справочником сотрудников

```
var cardData =
  session.CardManager.GetDictionaryData(staffId);
var rowDataUnit=
  cardData.Sections[unitSectionId].CreateRow();
rowDataUnit["Name"] = "NewOrganization";
var rowDataEmployee =
  rowDataUnit.ChildSections[employeeSectionId]
  .Rows.AddNew();
rowDataEmployee["LastName"] = "Ivanov";
```

Цель работы

- Написание объектной модели для разработки на платформе DocsVision
- Написание объектной модели для промежуточного хранений данных при синхронизации
- Контролируемое обновление данных на всех уровнях
- Управление репликациями при синхронизации DocsVision

Model First



В DocsVision, как и в Entity Framework, используется подход model first:

- Мы описываем схемы карточек
- По этим схемам создается SSDL

Было решено для генерации объектной модели использовать те же схемы карточек

Технологии для генерации кода

- Custom Tools
- T4
- Отдельно сгенерировать код для нескольких схем и подложить в проект
- MetaCreator

MetaCreator

http://code.google.com/p/metacreator/

```
public class Sample
{
    /*!
    for(int i=0;i<10;i++)
    {
        WriteLine("public string Pro" + i + "{ get; set; }")
    }
    */
    public string MySample;

    public override string ToString()
    {
        return null;
    }
}
```

```
[TestMethod]
public void Should_generate_properties()
{
    var obj = new Sample();
    obj.
    A                                   usive();
}

[Te
pu                                      ld_add_notification_imple
{
    v                                   Sample();
    v                                   as INotifyPropertyChang
    A                                   ull(notify);
    v
```

- Equals
- GetHashCode
- GetType
- MySample
- Pro0
- Pro1
- Pro2
- Pro3
- Pro4
- Pro5
- Pro6
- Pro7
- Pro8
- Pro9
- ToString

Что было сделано?

- Написан парсер для схем карточек
  - Собирает всю информацию о полях
  - Типизация ссылочных полей
- Выявлен отдельный интерфейс
  - Написано 4 различных генератора для различных подсистем сервиса синхронизации

Статистика

- В парсере и генераторах около 2000 строк кода
- Паттерны, заполняемые при генерации, занимают около 700 строк кода
- Из схем 12 схем карточек получилось около 100 000 строк кода

Результаты работы

- Написана объектная модель, которая может применяться не только в этом проекте, либо может быть доработана
- Достигнута улучшенная управляемость кода в проекте
- Запущен в тестовую эксплуатацию сервис синхронизации DocsVision

Дальнейшее развитие

- Ознакомиться с бизнес-процессами, действующими в DocsVision, и разработка для их создания-редактирования отдельной или встроенной утилиты
- Самообновляемость объектной модели
- Вынести транзакционность операций на модельный уровень