Урок 8

[Обзор сервис-ориентированной архитектуры приложений (SOA). Web-сервис asmx. WCF-сервис. Web-api сервис. 1](#_Toc487541978)

[1. Обзор сервис-ориентированной архитектуры приложений (SOA) 1](#_Toc487541979)

[2. Web-сервис asmx 1](#_Toc487541980)

[3. WCF-сервис 1](#_Toc487541981)

[4. Web-api сервис 1](#_Toc487541982)

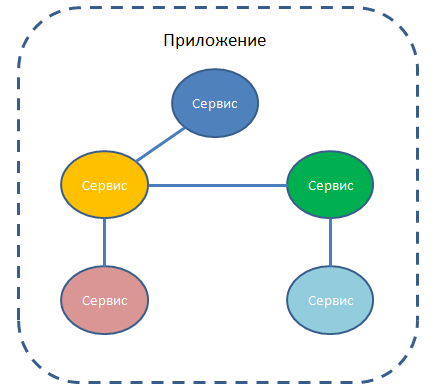
[5. ДЗ 1](#_Toc487541983)

# Обзор сервис-ориентированной архитектуры приложений (SOA). Web-сервис asmx. WCF-сервис. Web-api сервис.

1. Обзор сервис-ориентированной архитектуры приложений (SOA)

<http://masters.donntu.org/2014/fknt/belyy/ind/index.htm>

Сервис – это функциональная единица доступная внешнему миру с помощью стандартного механизма обмена сообщениями. Сервис-ориентированное приложение представляет собой агрегацию сервисов в логически целостное приложение, также как объектно-ориентированное приложение является агрегацией отдельных объектов.



Приложение, также, может выступать как еще один сервис, аналогично тому, как крупный объект может быть композицией более мелких объектов.

Внутри сервиса, разработчики могут по-прежнему использовать специфические языки программирования, технологии и фреймворки. Однако, между сервисами сохраняются стандартные протоколы и сообщения, контракты и обмен метаданными.

Сервисы, входящие в состав отдельного приложения, могут располагаться в одном и том же месте или могут быть распределены по разным узлам локальной сети или Интернет. Отдельные сервисы могут разрабатываться разными компаниями, размещаться на разных технологических платформах и находиться в разных часовых поясах. Все эти аспекты скрыты от клиентских приложений, взаимодействующих с сервисами. Клиентские приложения и сервисы могут взаимодействовать друг с другом путем отправки и получения сообщений. Механизм обмена сообщениями между клиентскими приложениями и сервисами сводит на нет разницу между клиентскими приложениями и сервисами, преобразуя входящие и исходящие сообщения к стандартным протоколам сетевого обмена.

Принципы сервис-ориентированной архитектуры:

* Сервисы должны иметь четкие границы. Любой сервис ограничен возможностями технологии, с помощью которой он реализован, и своим расположением. Эти ограничения не должны проявляться в контрактах сервисов или типах данных, используемых для обмена.
* Сервисы автономны. Сервис, для своей работы, не должен нуждаться в своих клиентах или других сервисах. Он должен функционировать и обновляться независимо от клиентских приложений. Также, сервис должен иметь свой механизм авторизации, не зависящий от уровня доступа пользователя клиентского приложения.
* Сервис предоставляет только контракты взаимодействия с ним и схему данных. Детали реализации и работы веб-сервиса должны быть недоступны извне.
* Совместимость сервисов основана на правилах(policy). Сервисы должны публиковать правила, описывающие, что они могут делать и как клиентские приложения взаимодействуют с ними.

Наиболее распространенные протоколы реализации веб-сервисов:

* SOAP(Simple Object Access Protocol). По сути это три стандарта: SOAP/WSDL/UDDI
* REST (Representational State Transfer)
* XML-RPC (XML Remote Procedure Call)

1. Web-сервис asmx
2. WCF-сервис
3. Web-api сервис
4. ДЗ

Измените WPF приложение для ведения списка сотрудников компании, из урока №5, **используя связывание данных, DataGrid и ADO.NET**.  
1. Создайте таблицы Employee и Department в БД MSSQL Server и заполните списки сущностей начальными данными.  
2. Для списка сотрудников и списка департаментов предусмотрите визуализацию (отображение). Это можно сделать, например, с использованием ComboBox или ListView.  
3. Предусмотрите возможность редактирования сотрудников и департаментов. Должна быть возможность изменить департамент у сотрудника. Список департаментов для выбора, можно выводить в ComboBox, это все можно выводить на дополнительной форме.  
4. Предусмотрите возможность создания новых сотрудников и департаментов. Реализуйте данную возможность либо на форме редактирования, либо сделайте новую форму.