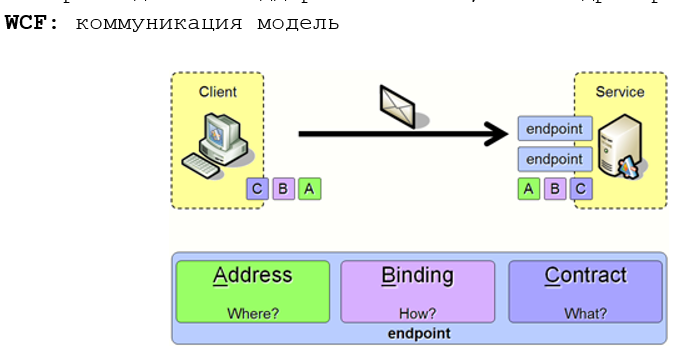
1. **WCF:** определение WCF-сервиса, коммуникационная модель, WCF-контракты, WCF-хостинг, конечные точки, стандартные привязки, основные отличия от ASMX-сервисов, поведение и безопасность WCF-сервиса, порядок разработки WCF/RPC и WCF/REST-сервисов, разработка WCF-сервиса с несколькими конечными точками.

**WCF**: Windows Communication Foundation; технология основанная на .NET FRAMEWORK для разработки приложений SOA-архитектуры, первая версия 2006 (Indigo).

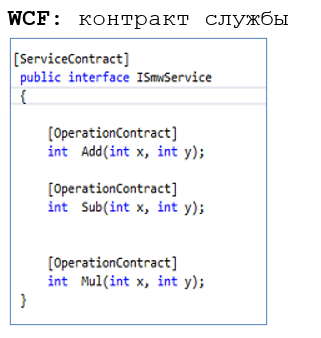
**WCF**: основные принципы

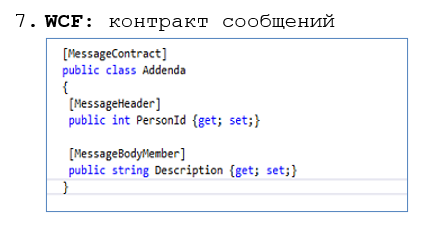
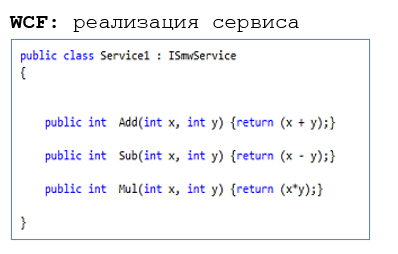
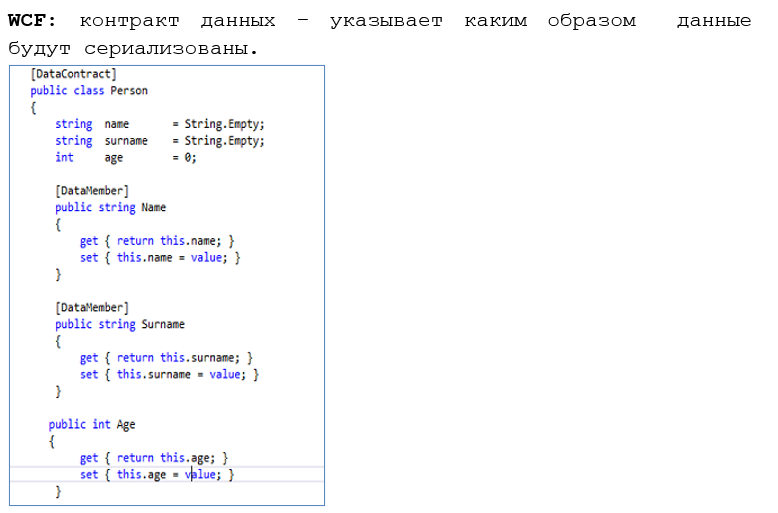
* разработка сервиса должна быть простой и иметь способность к расширению его функциональных возможностей;
* один API для всех коммуникационных протоколов;
* сервис должен быть интероперабельным (функционировать по отрытым телекоммуникационным стандартам);
* сервис должен поддерживать стандарты WS-\*;
* сервис должен поддерживать REST, RPC и др. архитектуры;

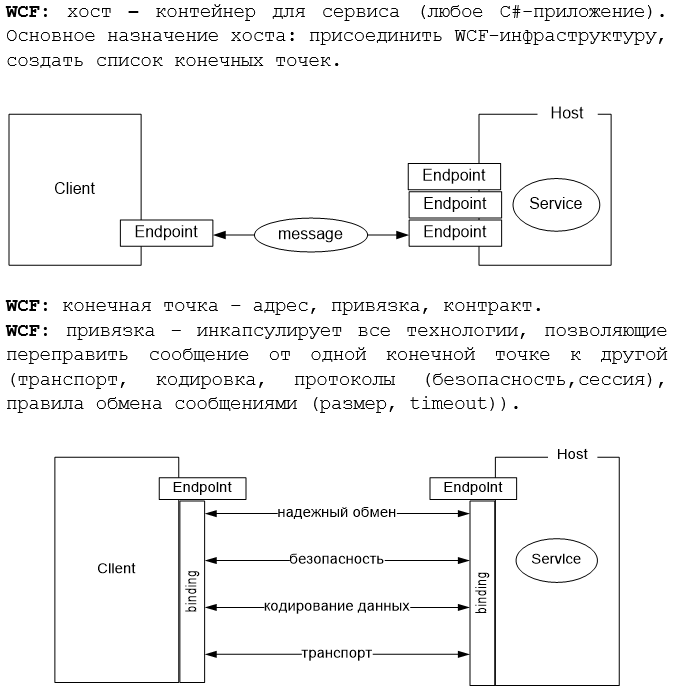


**WCF:** контракты

* контракт службы;
* контракт данных;
* контракт сообщений.



****



**WCF:** привязка: безопасность:WS-ReliableMessaging;

**WCF:** привязка: безопасность:WS-Security

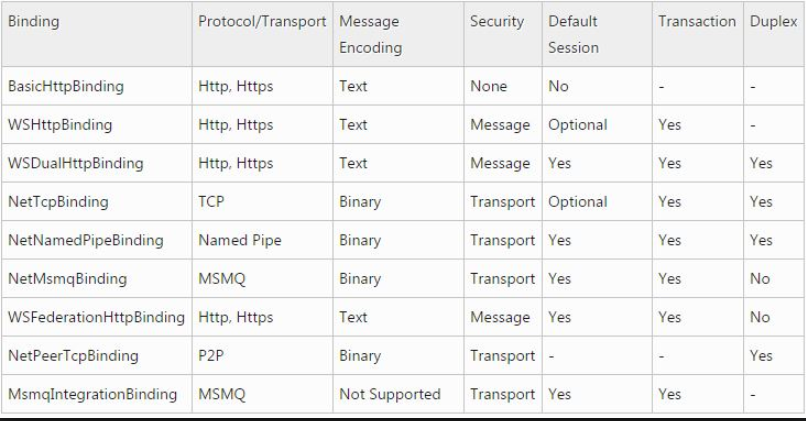
**WCF:** привязка: кодирование:

* текст (ASCII, UTF-8, UTF-16);
* двоичная (проприетарные алгоритмы);
* МТОМ.

**WCF:** привязка: транспорт:

* HTTP;
* TCP;
* IPC Named Pipes;
* MSMQ;
* Custom.

**WCF:** привязка: стандартные:



Основные отличия от ASMX-сервисов:

1. Протоколы: ASMX-сервисы используют протоколы SOAP и HTTP, в то время как WCF-сервисы могут использовать различные протоколы, такие как SOAP, HTTP, TCP, и т. д.

2. Гибкость настройки: WCF предоставляет большую гибкость настройки и конфигурирования, чем ASMX. Например, в WCF можно настроить различные протоколы, форматы сообщений, безопасность, маршрутизацию и т. д.

3. Поддержка различных стандартов: WCF поддерживает различные стандарты, такие как WS-\*, WS-Security, WS-ReliableMessaging и другие, что делает его более гибким для разработки сложных интеграционных сценариев.

Поведение и безопасность WCF-сервиса:

WCF позволяет разработчикам настроить поведение своих сервисов с использованием таких элементов, как контракты поведения (behavior contracts), расширения поведения (behavior extensions) и элементы поведения (behavior elements). Поведение может включать в себя различные настройки, такие как протоколы, сериализация, механизмы безопасности и другие аспекты работы сервиса.

В отношении безопасности, WCF предоставляет различные возможности для обеспечения безопасности взаимодействия между клиентом и сервисом. Это включает аутентификацию, авторизацию, шифрование, подпись сообщений и другие механизмы безопасности. Разработчик может выбрать подходящие механизмы безопасности и настроить их в соответствии с требованиями своего приложения.

Порядок разработки WCF/RPC и WCF/REST-сервисов:

При разработке WCF/RPC-сервисов, разработчик определяет контракт WCF-сервиса с использованием атрибутов, конфигурирует протоколы обмена сообщениями (например, HTTP или TCP) и настраивает поведения и безопасность. Затем реализуется логика операций сервиса.

При разработке WCF/REST-сервисов, разработчик определяет контракт WCF-сервиса с использованием атрибута [WebGet] или [WebInvoke], который указывает на REST-операции, и настраивает протоколы и поведения для работы с REST-запросами. Затем реализуется логика операций сервиса.

Разработка WCF-сервиса с несколькими конечными точками:

WCF позволяет определить несколько конечных точек (endpoints) для одного и того же сервиса. Каждая конечная точка может использовать разный протокол, формат сообщений, поведение и безопасность. Это позволяет сервису предоставлять различные интерфейсы и протоколы для разных клиентов. Для этого используется конфигурация WCF-сервиса, где задаются различные конечные точки и их свойства.