Część naszego projektu zawierająca WCFowy serwer umożliwiający wymianę danych między poszczególnymi aplikacjami składa się z trzech projektów C#: biblioteki definiującej usługę WCF, serwera self-hosted WCF, stanowiącego nasz skład danych oraz pokazowo-debugowego klienta WCF.

Projekt „TripiWCFService” (biblioteka) zawiera definicję klas przechowujących wszystkie dane istotne dla utworzonych przez nas aplikacji, to jest PositionNode i Trip. Znajduje się tutaj również deklaracja interfejsu ITripService, który definiuje kontrakt obowiązujący przy komunikacji z wykorzystaniem WCF – klasa utworzona na jego podstawie potrafi zalogować użytkownika, utworzyć nową wycieczkę i dodać do niej punkty, a także edytować te struktury danych. Na podstawie tej biblioteki generowaliśmy klasy proxy dla aplikacji mobilnej oraz Silverlighta. Dodatkowo dla spełnienia wymagań dotyczących zabezpieczeń Silverlighta powinniśmy jednocześnie implementować w określony sposób zapisany tu interfejs ICrossDomainPolicyResponder – taką implementację dostarcza abstrakcyjna klasa TripService, z której dziedziczą potem dwie przykładowe, w pełni funkcjonalne implementacje serwera wycieczek – TripServiceVolatile i TripServiceXml.

W projekcie „TripiWCFPreprodServer” znajduje się dość prosta aplikacja windowsowa, uruchamiająca w standardowy sposób self-hosted usługę WCF (w tym wypadku TripServiceXml) oraz tworząca okno, w którym wyświetlane są niektóre informacje dotyczące działania usługi (na przykład login użytkownika aktualnie łączącego się z serwerem lub szczegóły właśnie dodawanej wycieczki). Wszystkie istotne parametry serwera są przechowywane w pliku konfiguracyjnym .xml zgodnie z zasadami dotyczącymi usług WCF.

Aby ułatwić nam pracę przy tworzeniu całości projektu stworzyliśmy aplikację „TripiWCFClientMockup”, która wyświetla okno zawierające grupy przycisków umożliwiających nawiązanie interakcji z serwerem poprzez WCF – można stąd wykorzystać pełną paletę metod oferowanych przez ITripService. Używaliśmy tej aplikacji do wypełnienia bazy danych serwera przykładowymi danymi testowymi oraz do wyświetlenia zgromadzonych tam danych w celu e.g. weryfikacji poprawności nadsyłania danych przez aplikację mobilną. Poza celami testowymi klient ten nie ma zastosowania w gotowym projekcie.