УНИВЕРЗИТЕТ У НОВОМ САДУ

**ФАКУЛТЕТ ТЕХНИЧКИХ НАУКА У НОВОМ САДУ**

CLOUD COMPUTING U ELEKTROENERGETSKIM SISTEMIMA

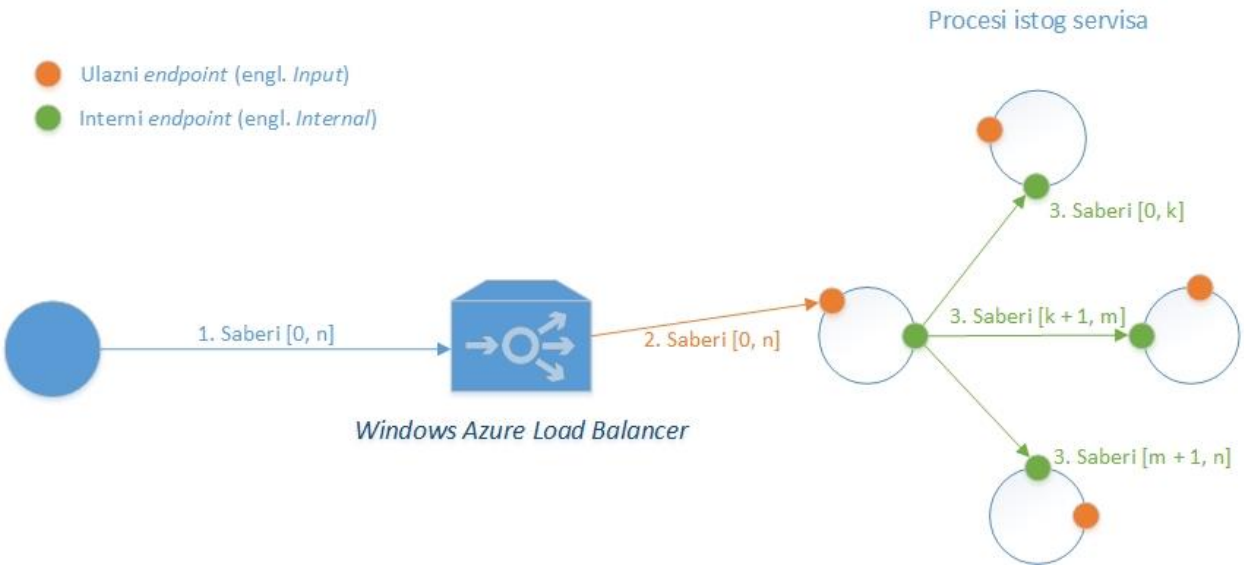
# Osnove *Microsoft WindowsAzure* servisa

**-SKRIPTA-**

Novi Sad, 2020

# Vežba 2 – *WCF* komunikacija u okviru *Windows Azure* servisa

U vežbi 2 se rešava zadatak koji koristi dve vrste *WCF* komunikacije u *Windows Azure* implementaciji međuservisne komunikacije. Jedna vrsta komunikacije podrazumeva komunikaciju između različitih procesa u okviru istog *Cloud* projekta, i naziva se *inter-role* komunikacija. Druga vrsta komunikacije predstavlja pružanje klasičnog *WCF* servisa. U klasičnom pristupu, zbog distribuiranosti rešenja, zahtev koji je upućen *WCF* servisu prvo stiže do *Windows Azure Load Balancer* komponente. Potom, zahtev se prosleđuje jednom od procesa koji pripadaju datom tipu. Zadatak je izdeljen u dve celine – A i B. Preporuka je da se zadaci rešavaju datim redosledom. Na slici 1 je prikazana arhitektura krajnjeg rešenja zadatka.




*Slika 1 Zadatak vežbe 2.*

Implementirati distribuirani servis koji sukcesivno sabira brojeve u zadatom intervalu I∈ $N_0^+$, gde je donja granica intervala uvek 0. Kreirati konzolnu aplikaciju (klijenta) koja će služiti za zadavanje gornje granice intervala. Kreirati *Worker role* tip procesa koji će vršiti sukcesivno sabiranje tako što će koristiti *inter-role* komunikaciju za distribuciju posla na sve raspoložive procese. Servis sukcesivnog sabiranja implementiraju svi procesi u okviru *Worker role* tipa procesa i nije bitno koji proces će primiti zahtev. Nakon primljenog zahteva, proces vrši podelu posla na preostale procese istog tipa procesa. Koristi *inter-role* komunikaciju za kreiranje zahteva za izračunavanje podintervala. Na kraju, vrši se zbir podintervala i zbirna vrednost predstavlja odgovor servisa za sukcesivno sabiranje brojeva.

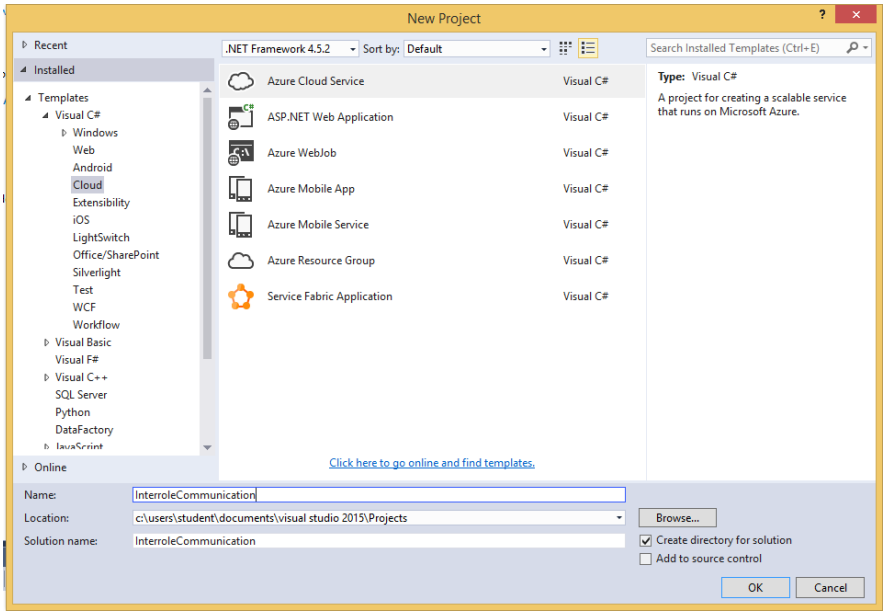
Pod sabiranjem sukcesivnih brojeva, podrazumeva se sledeći algoritam:

$$f(n) = \sum_{i=1}^{n} i$$

## Zadatak A:

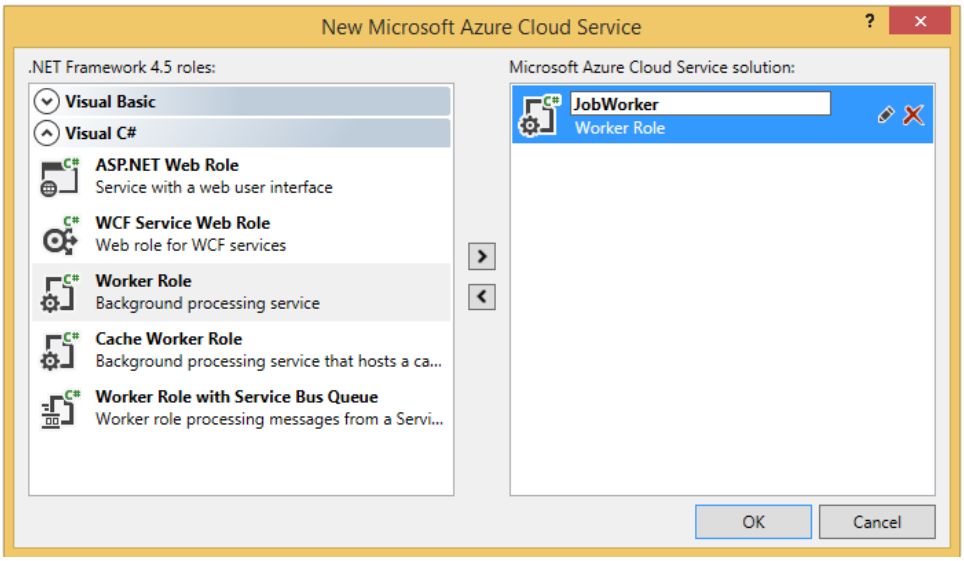
Cilj zadatka A je da se implementiraju koraci 1 i 2 sa slike 1. Prvo se treba kreirati C*loud* projekat *InterroleCommunication* koristeći šablon koji dolazi uz instalirani SDK. Postupak za kreiranje *cloud* projekta:

- Klikom na *File* meni, odabrati *New/Project*
- U *New Project* dijalogu, proširiti Visual C# u listi *Installed Templates* i selektovati *Cloud,* kao što je prikazano na slici 2. Odabrati šablon *Windows Azure Cloud Service*. Uneti ime projekta *InterroleCommunication*. Kliknuti na OK.




*Slika 2 Kreiranje cloud projekta*

U *New Windows Azure Project* dijalogu, u okviru *Roles* panela, otvoriti tab Visual C#. Selektovati *Worker Role* i dati mu naziv *JobWorker* (Slika 3).




*Slika 3 Dodavanje worker role*

Dodati *Console Application* projekat po nazivu *JobInvoker* i ovaj proces treba da omogući unos gornje granice intervala nakon čega i poziv *WCF* servisa sa vrednostima intervala [0, n] (gde je n unešena vrednost od strane korisnika).

Kreirati *Class Library* projekat koji će sadržati interfejse za *WCF* servise. Nazvati projekat "*InterroleContracts*". Dodati interfjes *IJob* u biblioteku. Definicija interfejsa je data u listingu 1.

Listing 1 – *IJob* interfejs.

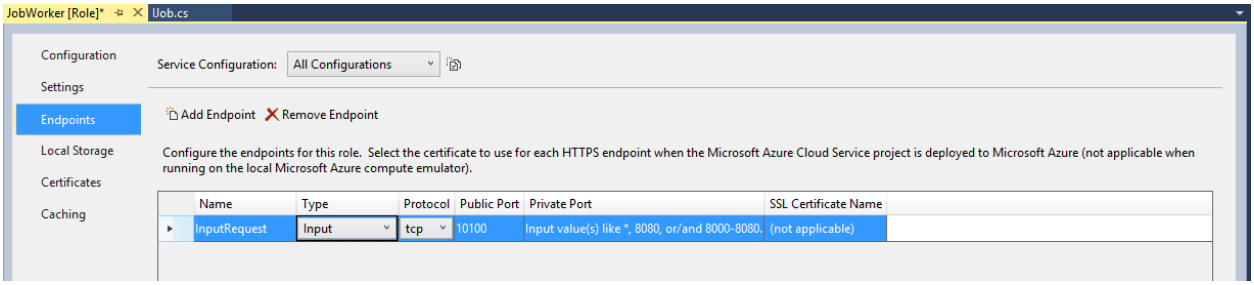
```
[ServiceContract]
public interface IJob
{
	[OperationContract]
	int DoCalculus(int to);
}
```

U okviru projekta *JobWorker* kreirati *WCF Server* kao posebnu klasu (*JobServer*) u okviru koje će biti metode za otvaranje i zatvaranje konekcije servisa. Koristiti *NetTcpBinding*, a za dobijanje krajnje *IP* adrese trenutnog procesa, iskoristiti klasu *RoleEnvironment* kao što je prikazano u listingu 2. U implementaciji servisa (*JobServerProvider* klasa koja implementira *IJob* interfejs) potrebno je izračunati sumu i ispisati poruku u *Compute* emulatoru: "DoCalculus method called - interval [1, n]". U implementaciji *OnStart*() metode *Worker role* pozvati kreiranje i otvaranje konekcije ovog servisa.

Listing 2 – kreiranje *WCF Servera*

```
private ServiceHost serviceHost;
// dodati endpoint sa ovim imenom u ServiceDefinition
private String externalEndpointName = "InputRequest";

public JobServer()
{
	RoleInstanceEndpoint inputEndPoint = RoleEnvironment.
		CurrentRoleInstance.InstanceEndpoints[externalEndpointName];
	string endpoint = String.Format("net.tcp://{0}/{1}", inputEndPoint.IPEndpoint,
		externalEndpointName);
	serviceHost = new ServiceHost(typeof(JobServerProvider));
	NetTcpBinding binding = new NetTcpBinding();

	serviceHost.AddServiceEndpoint(typeof(IJob), binding, endpoint);
}

public void Open()
{
	try
	{
		serviceHost.Open();
		Trace.TraceInformation(String.Format("Host for {0} endpoint type opened
		successfully at {1}", endPointName, DateTime.Now));
	}
	catch (Exception e)
	{
		Trace.TraceInformation("Host open error for {0} endpoint type. Error
		message is: {1}. ", endPointName, e.Message);
	}
}

public void Close()
{
	try
	{
		serviceHost.Close();
		Trace.TraceInformation(String.Format("Host for {0} endpoint type closed
		successfully at {1}", endPointName, DateTime.Now));
	}
	catch (Exception e)
	{
		Trace.TraceInformation("Host close error for {0} endpoint type. Error
		message is: {1}. ", endPointName, e.Message);
	}
}
```

Definisati *endpoint* tipa "*Input*" u *ServiceDefinition* konfiguraciji. Dvoklikom na konfiguraciju *Worker role* procesa u okviru *Cloud* projekta, pod stavkom *Endpoints* je moguće dodati novi *endpoint* (Slika 4).




*Slika 4 Definisanje input endpoint-a*

Za ispisivanje poruka u Compute emulator koristiti:

```
Trace.WriteLine("tekst_poruke", "Information");
```