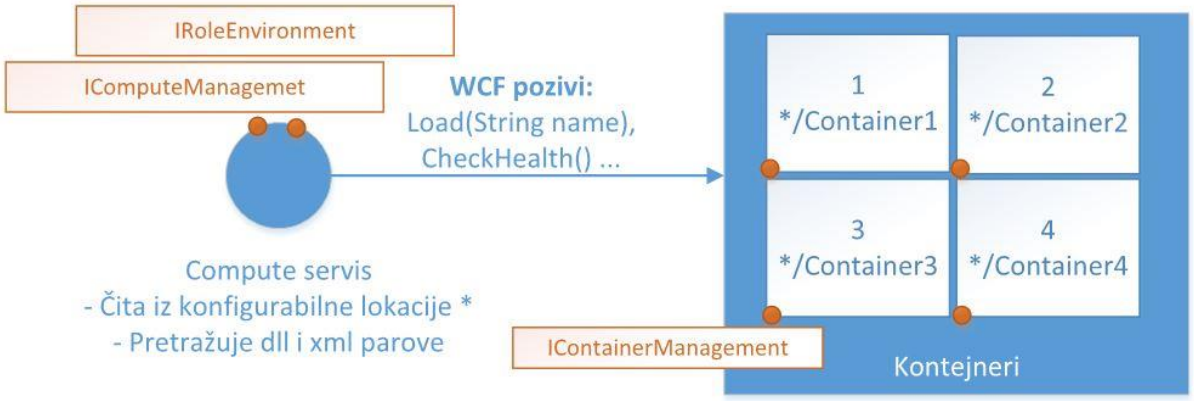
Projektni zadatak – kreiranje PaaS

Definicija zadatka:



Slika 1 - Šematski prikaz zadatka

U skladu sa šematskim prikazom (Slika 1), implementirati *Cloud compute* servis:

a) Kreirati projekat pod nazivom *Compute* koji će pokretati ukupno četiri kontejnerske konzolne aplikacije. Konzolne aplikacije kontejnera simuliraju čvorove jednog *Cloud* sistema. *Compute* servis ima za zadatak da skenira predefinisanu lokaciju, i da paket koji se sastoji od dll i xml datoteke tumači i vrši njihovo pokretanje na kontejnerima. *Compute* servis vrši sledeće akcije:
   1. Pokreće četiri procesa kontejnera koji su implementirani kao konzolne aplikacije, prosleđujući im port na kom se njihov WCF server izvršava.
   2. Periodično proverava da li se na predefinisanoj lokaciji nalazi novi paket. Predefinisana lokacija treba da bude konfigurabilna.
   3. Čita XML datoteku i proverava koji broj instanci treba da pokrene. Ukoliko je broj instanci veci od četiri, ispisuje poruku da je konfiguracija nevalidna i briše paket sa predefinisane lokacije.
   4. Kopira dll iz paketa na *n* destinacija i angažuje *n* kontejnera putem WCF servisa da učitaju dll i da ga pokrenu, pokretanjem *Start()* metode iz interfejsa *IWorker* datog u listingu 1. Broj *n* je određen u trećem koraku.
   5. Svaki kontejner ima WCF server sa metodom *Load* prikazanoj u listingu 2. Nakon učitavanja biblioteke, vrši se njegovo pokretanje i odjavljivanje da li je pokretanje proslo u redu ili je doslo do neke greške.
   6. ***Napomena***: Svi WCF serveri se izvršavaju na *localhost* adresi, ali sa različitim portovima. Zbog jednostavnosti rešenja, dozvoljeno je koristiti unapred predefinisane portove. Primer raspona može biti: 10010 – 10050.

b) Kreirati robustnost kontejnera, tako da ukoliko neki rezultuje otkazom, *Compute* servis će to ustanoviti i pokrenuti novi kontejner sa klijentskim programima koji su otkazali zajedno sa kontejnerom. *Compute* vrši oporavak na sledeći način:

1. Periodično proverava stanje kontejnera putem WCF komunikacije i *CheckHealth* metode, tako što se interfejs iz listinga 2. proširi metodom datom u listingu 3.
2. Ukoliko je neki od kontejnera rezultovao otkazom, izvršava se oporavak pokretanjem novog kontejnera i pokretanjem paketa koji su bili izvršavani u okviru tog kontejnera. Oporavak se vrši na sledeći način:
    i. Ukoliko ima slobodnih kontejnera koji već ne izvršavaju klijentski program, prvo pokrenuti novu instancu u okviru slobodnog kontejnera, a zatim krenuti u proceduru startovanja novog kontejnera.
    ii. Ukoliko su svi kontejneri zauzeti, pokrenuti prvo novi kontejner, a potom učitati klijentski program u novopokrenutom kontejneru.

c) Obezbediti da *Compute* servis može da pruža instancama informacije o bratskim instancama putem WCF servera, na sledeći način:
1. *Compute* sadrži WCF server sa intefejsom iz Listinga 4. Kada se klijentski programi pokreću u okviru kontejnera, oni zatražuju svoju adresu od Compute servisa koristeći metodu AcquireAddress(). I ovde važi napomena a.6. Iskoristiti napomenu tako da compute servis ima rezervisane portove za svaki kontejner, dovoljno je za potrebe zadatka dve adrese po klijentskom programu.
2. *Compute* servis održava interno gde se koji servis nalazi. Implementira metodu iz listinga 4, BrotherInstances() tako što na osnovu imena programa i same instance jedinstveno određene sopstvenom adresom (preporuka je da se koristi samo port kao jedinstveni identifikator), pronalazi preostale portove rezervisane za ostale instance i vraća listu portova kao odgovor.

d) Obezbediti *RoleEnvironment* klasu u vidu klijentske biblioteke.
1. Za zadatak urađen pod c, potrebno je funkcionalnost implementirati u okviru klase *RoleEnvironment* date u listingu 5. Klasa treba da dolazi u posebnoj biblioteci kako bi se mogla distribuirati korisnicima PaaS okruženja.
2. *Napomena:* zbog jednostavnosti zadatka, klijentske aplikacije neće imati više od jednog WCF servera.

e) Omogućiti horizontalnu skalabilnost kroz *Compute* servis, na sledeći način:
1. *Compute* servis ima WCF server koji implementira interfejs dat u listingu 6.
2. Potrebno je implementirati WCF klijenta, isto kao konzolnu aplikaciju koja će biti u stanju da za određenu rolu promeni broj instanci.
3. Odgovor *compute* servisa je:
    i. Ukoliko je *count* > 4: Nepodržana akcija,
    ii. Ukoliko je *count* nepromenjen u odnosu na trenutno stanje: Nema promena,
    iii. U ostalim slučajevima, vraća odgovor koliko instanci je stopirano, ili koliko je novih angažovano.

Listing 1:

```
public interface IWorker
    {
        void Start(String containerId);
        void Stop();
    }
```

Listing 2:

```
public interface IContainerManagement
    {
        String Load(String assemblyName);
    }
```

Listing 3:

```
public interface IContainerManagement
    {
        String Load(String assemblyName);

        String CheckHealth();
    }
```

Listing 4:

```
public interface IRoleEnvironment
    {
        String AcquireAddress(String myAssemblyName, String containerId);

        String[] BrotherInstances(String myAssemblyName, String myAddress);
    }
```

Listing 5:

```
public class RoleEnvironment
    {
        /// <summary>
        /// Vrednost je vrednost porta na kojoj se WCF server izvrsava
        /// Napomena: zbog jednostavnosti zadatka, moze biti samo jedan WCF server po
klijentskom projektu
        /// </summary>
        public static String CurrentRoleInstance(String myAssembly, String containerId)
        {
            throw new NotImplementedException();
        }

        /// <summary>
        /// Povratna vrednost je lista portova bratskih instanci.
        /// </summary>
        public static String[] BrotherInstances { get; }
    }
```

Listing 6:

```
public interface IComputeManagement
    {
        String Scale(String assemblyName, int count);
    }
```

Predaja konačnog rešenja:

Nakon urađenog zadatka, student šalje rešenje na sve tri mail adrese: sebastijan.stoja@uns.ac.rs, marina.stanojevic@uns.ac.rs i nikola.dalcekovic@uns.ac.rs i zakazuje se termin odbrane.

Bodovanje zadatka:

Zadatak je izdeljen u pet delova, označenih slovima: a, b, c, d, i e. Mogu se uraditi samo određeni delovi zadatka.

Tabela 1 prikazuje bodovanje, pri čemu se podrazumeva da se svaki deo zadatka odbrani usmeno.

Deo zadatka:	**Maksimalno bodova:**
A	2
B	2
C	2
D	2
E	2
Ukupno	10

Tabela 1 - Bodovanje