Fakultet tehničkih nauka
Univerzitet u Novom Sadu

Razvoj elektroenergetskog softvera

PROJEKAT: RESIDENT EXECUTOR

Mentor:
Zorana Babić

Studenti:
Nataša Šćekić, PR 10/2018
Đurđica Marić, PR 4/2018

1. Simulacija rada sistema

Client preko *Data Access* komponente upisuje podatke o potrošnji električne energije i vremenu upisa u bazu podataka za određeni grad. Upis se može izvršavati automatski, na svake 2 sekunde, ili ručno. Za to vreme, *Resident Executor*, na određeni vremenski interval koji je podesiv, poziva *Calculation Function* funkcije čije identifikacije čita iz XML fajla. Pre samog izvršavanja, vrši se provera da li je proračun sa poslednjim vremenom merenja već izvršen. Ukoliko jeste, proračun se ne ponavlja. Funkcije pomoću *Data Access* komponente upisuju u bazu podataka podatke o vremenu proračuna, poslednjem vremenu merenja i vrednosti koju su dobile izvršavanjem proračuna. Pristup proračunu imaju *Client*, koji ima mogućnost njegovog ispisa, i *Resident Executor*, koji ih pamti.

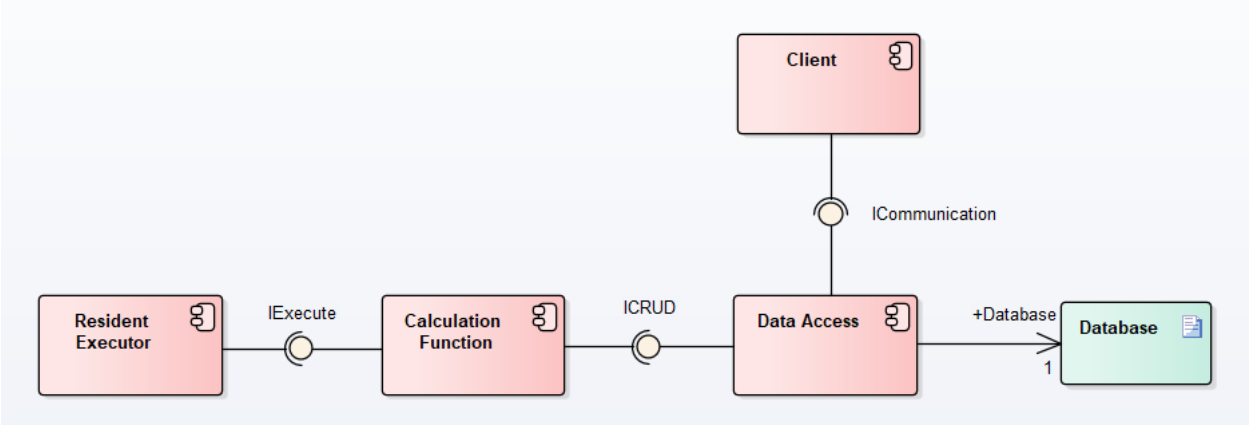
2. Dizajn i arhitektura sistema

Sistem se sastoji od sledećih komponenti:

- *Resident Executor*
- *Calculation Function*
- *Data Access*
- *Client*

2.1. ***Component Diagram***

Na sledećem *Component Diagram*-u su prikazane sve komponente sistema i njihove međusobne zavisnosti *(Slika 1.)*:



Slika 1.: Component Diagram

- *Resident Executor* komponenta je egzistencijalno zavisna od *Calculation Function* komponente koja implementira *IExecute* interfejs,
- *Calculation Function* komponenta je egzistencijalno zavisna od *Data Access* komponente koja implementira *ICRUD* interfejs,
- *Client* komponenta je egzistencijalno zavisna od *Data Access* komponente koja implementira *ICommunication* interfejs.

2.1.1. ***Resident Executor***

Osnovni zadatak *Resident Executor* komponente je da poziva odgovarajuću *Calculation Function* funkciju (na osnovu identifikacije koju učitava iz XML fajla), ne znajući logiku koja se krije iza bilo koje od njih. Nakon učitavanja identifikacije, funkcije se smeštaju u red za izvršavanje odakle se jedna po jedna poziva u određenom vremenskom razmaku, koji je moguće podesiti.

2.1.2. ***Calculation Function***

Calculation Function komponentu čine tri funkcije: *Calculation Funcion 1*, *Calculation Function 2* i *Calculation Funciton 3*. Ono što je zajedničko za svaku od njih je da vrše proračunavanja, čije rezultate zajedno sa vremenom proračuna i poslednjim vremenom merenja upisuje u bazu podataka. Razlika se ogleda u vrsti proračuna koji se izvršavaju i to na sledeći način:

Calculation Function 1 izračunava prosečnu potrošnju električne energije u tekućem danu,
Calculation Function 2 izračunava minimalnu potrošnju u tekućem danu,
Calculation Function 3 izračunava maksimalnu potrošnju u tekućem danu.

Ukoliko se neka od funkcija pozove više puta za isto poslednje vreme merenja, izvršiće se samo jednom.

2.1.3. ***Data Access***

Pomoću *Data Access* komponente vrši se komunikacija sa bazom podataka i omogućava se izvršavanje *CRUD* operacija (*Create, Read, Update, Delete*) nad njom.

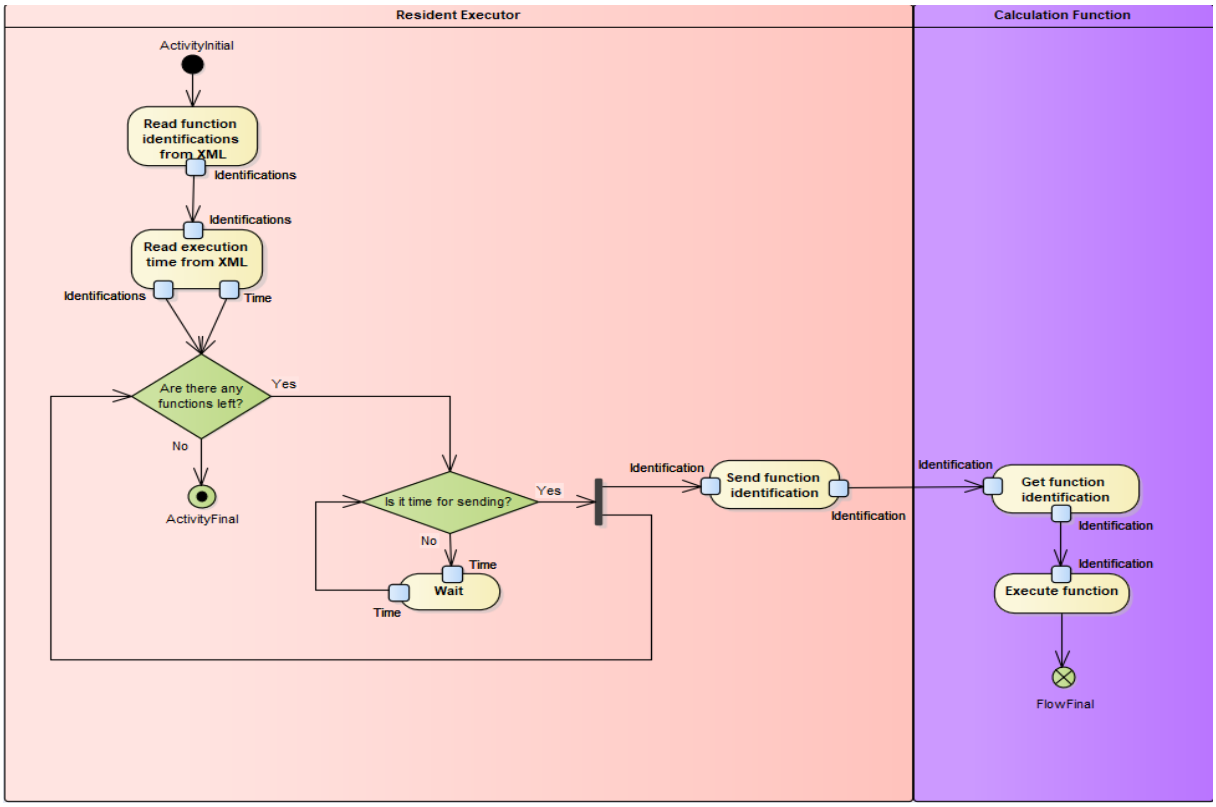
2.1.4. ***Client***

Client predstavlja deo sistema koji ima mogućnost upisa datuma, vremena i potrošnje električne energije (mW/h) za određeni grad, kao i mogućnost ispisa proračuna koji su izvršile funkcije iz *Calculation Function* komponente.

2.2. ***Activity Diagram 1***

Na sledećem *Activity Diagram*-u je prikazano učitavanje identifikacija funkcija iz XML fajla u *Resident Executor* komponentu, kao i njihovo prosleđivanje komponenti *Calculation Function*. *(Slika 2.)*

Prvi korak ovog procesa je čitanje identifikacija funkcija iz XML fajla. Zatim se takođe čita i vreme koje će proći između slanja zasebnih identifikacija *Calculation Function* komponenti. Ukoliko nema identifikacija za slanje, aktivnost se završava. U suprotnom slučaju, proverava se da li je vreme za slanje sledeće identifikacije funkcije. Ukoliko nije, čekamo da dođe vreme slanja. Ukoliko jeste, šalje se prva sledeća identifikacija funkcije i proverava se da li je ostalo još identifikacija za slanje. Nakon slanja identifikacije, komponenta *Calculation Function* je prima, a potom izvršava funkciju određenu pristiglom identifikacijom.

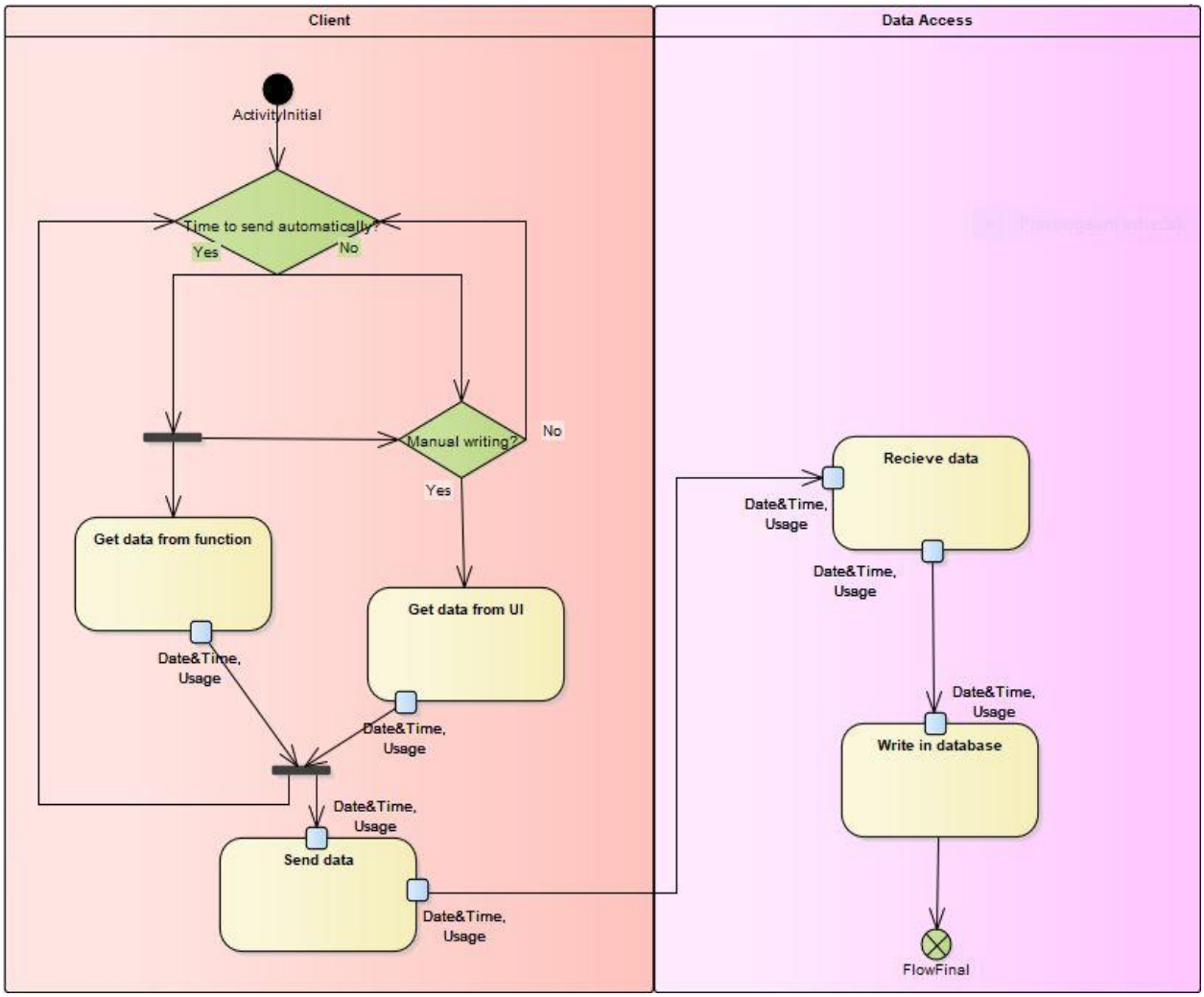


Slika 2.: Učitavanje identifikacija funkcija iz XML fajla i njihovo prosleđivanje

2.3. ***Activity Diagram 2***

Aktivnost opisana na drugom *Acitvity Diagram*-u je upisivanje podataka u bazu podataka koje može da se vrši automatski i od strane korisnika. (*Slika 3.*)

Automatsko upisivanje podataka u bazu podataka se vrši na svake 2 sekunde na način da se prosleđuju podaci koji se generišu u beskonačnoj petlji, nezavisno od korisnika, koji u svakom trenutku ima mogućnost da izabere opciju ručnog unosa podataka. U tom slučaju korisnik unosi željene podatke putem *User Interface*-a. Ukoliko nije vreme za upis i/ili korisnik ne želi da ručno unese podatke, proces se vraća na početak. Nakon što se prikupe podaci, bilo da su generisani u petlji ili uneseni, proces se deli na dve grane. Jedna od njih se vraća na početak i proverava da li je vreme za novi upis, dok druga nastavlja sa slanjem podataka prema *Data Access* komponenti. *Data Access* prima pristigle podatke i upisuje ih u bazu podataka.



Slika 3.: Upis podataka u bazu podataka od strane Client komponente.