Факултет техничких наука

Универзитет у Новом Саду

Елементи развоја софтвера

Cache Memory

Аутори:
Бојан Кујић
Вук Огњановић
Данијел Јовановић
Дора Митић

Број индк:
PR43/2020
PR51/2020
PR55/2020
PR58/2020

15. јануар 2023.

Садржај

1 Опште информације о пројекту

Cache Memory 3.3.1 је пројекат из предмета Елементи Развоја Софтвера који се слуша у V семестру на Факултету техничких наука у Новом Саду.

Апликација прикупља податке од корисника о тренутној потрошњи топлотне енергије или о потрошњи у неком од претходних периода.

2 Техничка и контекстуална спецификација пројекта

2.1 Коришћење технологије у изради пројекта

- C# (.NET)
- Oracle XE 11G Database Server
- WPF
- CME v2.1 (IPC+)

2.2 Контекст проблема

- Свакодневница нам доноси разне проблеме везане за бележење већег скупа података. Број корисника свакодневно расте, док се неки рутински послови могу полу-аутоматизовати. Проблем настаје код бележења потрошње топлотне енергије, где је сваки власник задужен за неки број бројила која мере проток топлотне енергије.
  Тако прикупљени подаци нису од великог значаја јер фирма за дистрибуцију топлотне енергије не може направити статистички преглед и своју будућу дистрибуцију фино подесити и оптимизовати на начин да сви корисници и даље остану задовољни.

2.3 Решење проблема и техничка стратегија

- Решење проблема је направити јединствени информациони систем који ће опслуживати апликација преко које ће се уносити подаци о корисницима, потрошњи као и омогућити да постоји увид у статистику потрошње и омогућити тзв. **“fine-tuning”** будућих прегледа и статистика.

2.4 Информационо-апликациони опис

- Целокупно идејно решење састоји се од више компоненти односно **микросервиса** који могу радити заједно у склопу, али их одликује и потпуна независност од остатка информационог система.

  Common Class Library
  Компонента која није микросервис, већ искључиво моделује начин и дефиниције како и на који начин је потребно приступати бази података.

Захваљујући наведеној компоненти самој апликацији је омогућено да успостави конекцију ка бази података.

IPC Services

Компонента која моделује сервис интеракција и интеграција. Посредством дате компоненте омогућено је да корисник преко графичког интерфејса уноси податке (или шаље захтев за преглед података) другим компонентама информационог система. Компонента није чвориште података, већ само чвор кроз који подаци пролазе према компоненти за перзистенцију конекција између преосталих компоненти.

Writer Component

Микросервис који податке које прими не задржава код себе, не заузима меморију и има **zero-knowledge** принцип интегрисан у себи (не занима га и не зна које податке прима). Податке које прими од корисника преко корисничког интерфејса, прослеђује даље ка следећем активном микросервису.
Све операције је могуће вршити искључиво **посредством корисничког интерфејса** (интерактивни режим рада није подржан).

Dumping Buffer Component

Микросервис који при свом првом покретању креира **ред чекања** и омогућава операције **додавања у ред**, **брисање првог из реда**, **упис у базу података** и **периодичну проверу попуњености** реда чекања.
Све операције је могуће вршити искључиво **посредством корисничког интерфејса** (интерактивни режим рада није подржан).

Reader Component

Микросервис посредник између компоненти информационог система који је намењен да **визуелизатору података врати исчитане податке** из базе података. Налик микросервису за упис података, не познаје које податке тражи од микросервиса, већ само захтев од корисника прослеђује ка систему и враћа податке филтриране по кориснички задатом критеријуму (или врати празну листу података ако задати критеријум није испуњен).
Све операције је могуће вршити искључиво **посредством корисничког интерфејса** (интерактивни режим рада није подржан).

Historical Component

Уједно и најважнији микросервис који комуницира са базом података и врши **упис података у базу података** и **чита и филтрира податке из базе података**. Како су могућности микросервиса интеграција са базом података, одлика је да једино и он поседује **интерактивни режим рада**,

који кориснику или пак администратору омогућава да посредник постане сам микросервис и да је помоћу њега могуће уносити и читати податке директно у базу података.

Такође, микросервис може служити и за проверу ваљаности конекције према бази података.

```
HISTORICAL KOMPONENTA ZAPOCINJE SA RADOM
Za interaktivni rezim rada pritisnunti taster 'i' na tastaturi.


============ Historical Component - Interactive Menu ============
        1. Rucni Upis podataka u bazu podataka
        2. Iscitavanje svih podataka o potrosnji
        3. Iscitavanje podataka po kriterijumu
        4. Izlaz iz interaktivnog rezima rada
>> _
```

3 Циљеви пројекта и захтеви

3.1 Општи кориснички захтеви

- Захтеви које је изнео корисник су:
    1. Могућност регистрације на апликацију
    2. Могућност пријаве на апликацију
    3. Могућност уноса података о кориснику/власнику бројила
    4. Могућност прегледа унетих података
    5. Могућност интерне комуникације компоненти система
    6. Могућност прегледа статистике потрошње
    7. Могућност прегледа статистике по неком од критеријума
    8. Респонзивност упита над базом података

3.2 Технички захтеви који ће бити занемарени

- Корисник је захтевао да апликација поседује компоненте за пријаву и регистрацију на информациони систем, али с обзиром да је инфраструктура на којој се апликација покреће потпуно затвореног и изолованог типа – сам развој такве врсте компоненте у апликацији ће бити занемарен.

3.3 Технички захтеви

- Нормалан рад саме апликације је предвиђен уз искључиво поштовање одређених смерница пројектаната и развојног тима апликације. Апликацију је потребно покретати на оперативном систему **Windows 10** или новије, и да на рачунару на ком се покреће апликација буде инсталиран **.NET Framework**

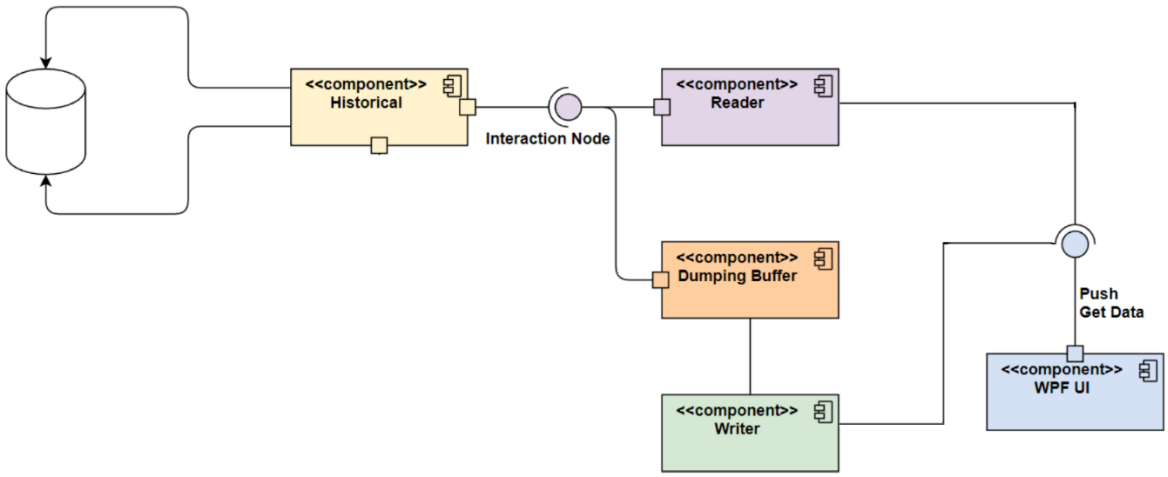
верзије 4.8 или новије. Са стране хардверских захтева потребан је минимално **двојезгарни процесор** и **2Gb** радне меморије за респонзиван рад апликације.

4 Дизајн и архиктетура система

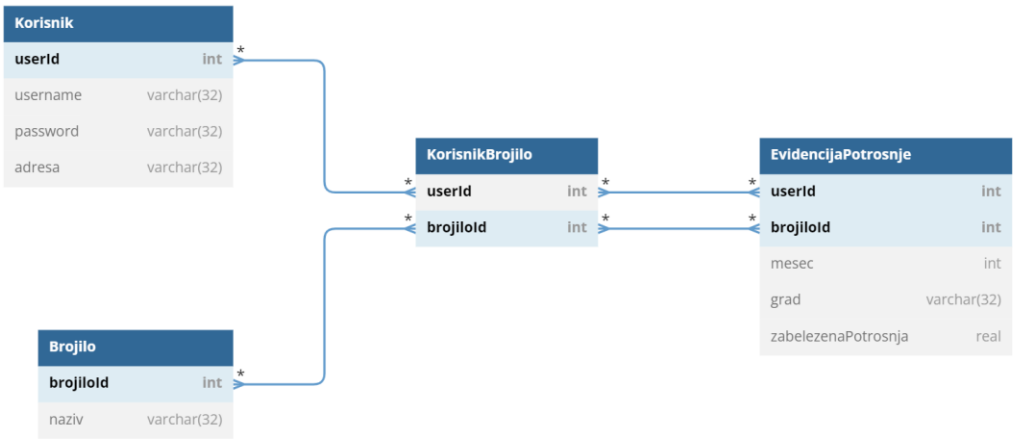
4.1 Уопштење

- Дизајн система је процес дефинисања елемената система као што су модули, архитектура, компоненте и њихови интерфејси и подаци за систем на основу специфицираних захтева.

4.2 Компоненте и интерфејси система



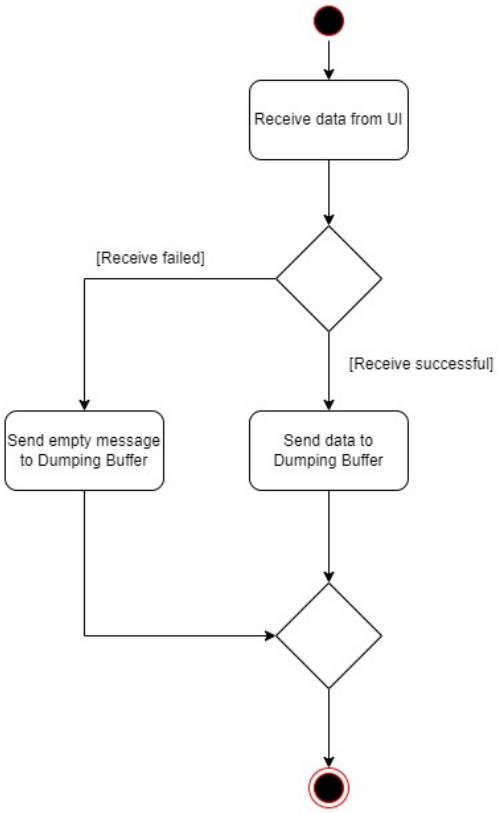
4.3 Шема базе података



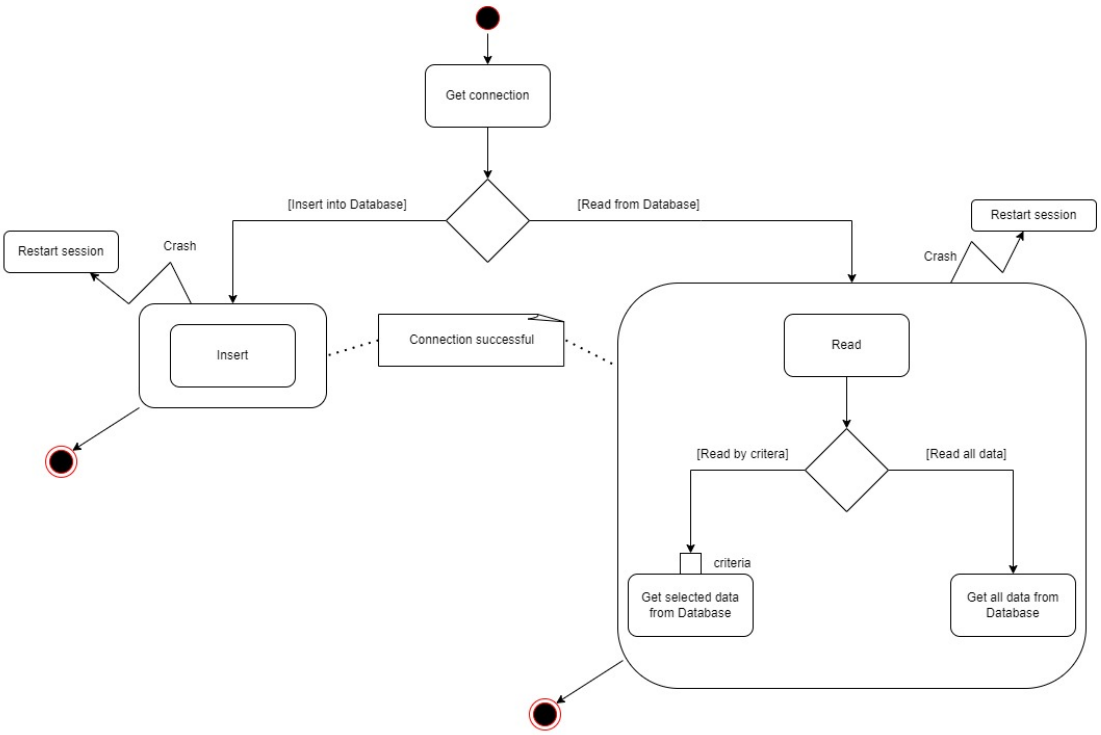
4.4 Дијаграм компоненте *Writer*

Компонента ***Writer*** добија податке из корисничког интерфејса.

У зависности од успешности примања података компонента прослеђује добијене податке или шаље празну поруку у Dumping Buffer.



4.5 Дијаграм компоненте *Historical*



Компонента ***Historical*** прво отвара конекцију са базом података, која је неопходна за функционисање апликације. Након отварања конекције могуће је писати или читати из базе. Читање из базе може вратити све податке из базе или само одабране податке у зависности од задатог критеријума. У случају грешке програм престаје са радом и потребно га је поново покренути.

5 Тест план

5.1 Тестирање испуњења корисничких тестова

- На основу критеријума прихватљивости из сваког ***Product Backlog Item-a*** потребно је осмислити и написати одговарајуће тестове били они релативно сингуларни или респективно интерактивни. Тестирање испуњења корисничких захтева одређено је на начин да сам информациони систем и апликација која се на исти ослања визуелно и контектстуално испуњава све захтеве које је корисник на почетку изложио ***Product Owner-y***, а касније идеја пренета на реализацију развојном тиму.
  Тестирање је потребно урадити из 4 фазе, од чега су приказане само 3, од чега су две предиктивно техничко-програмиране, док су преостале две интерактивно-мануелне.

5.2 Unit и Mock тестови

- Реализацију тестова поделити у две групе. Прву групу чине ***NUnit*** тестови који ће бити осмишљени, написани и покретани за компоненте које представљају ***моделе*** и/или ***посреднике*** информационог система и апликације која исти опслужује. У конкретној реализацији ***Cache Memory*** тестови ће бити направљени парцијално.
  Mock тестови ће такође, као и ***NUnit*** бити парцијално распоређени и користиће се искључиво за тестирање метода за конекцију према бази података, упис у исту.
  Преглед тестова по компоненти приказан је у табели испод.

Назив компоненте	Подсистем компоненте	NUnit	Mock	Ручни тестови
Common Class Library	**Connection**	✗	✓	✗
	ConnectionParams	✗	✗	✗
	ModelData	✓	✗	✗
Dumping Buffer Component	**AddToQueue**	✓	✓	✗
	RemoveFromQueue	✓	✓	✗
	PeriodicCheck	✗	✗	✗

	SentToHistorical	✗	✓	✗
Historical Component	GetAllDataFromDatabase	✓	✗	✗
	GetSelectedDatabyCriteria	✓	✗	✗
	WriteModelData	✗	✓	✗
	Save	✗	✗	✗
IPC Services	Interaction Node	✗	✗	✗
Reader Component	GetPodaciFromHistorical	✓	✗	✗
WPF UI	Pocetna	✗	✗	✓
	Upis Novih Podataka	✗	✗	✓
	Statistika	✗	✗	✓
Writer Component	DataPassThrough	✗	✓	✗
Збирно	17	172	6	3

Табела 1: Тестирање и верификација компоненти софтвера

5.3 Ручни тестови

- Аутоматизовани тестови су одлично решење за већинску проверу испуњености корисничких критеријума, али нису и довољни како би се производ пустио у продукцију.
  Тестови који ће бити спроведени биће интерактивни и спроводиће их **тест тим** који ће ИСКЉУЧИВО вршити интеракцију са компонентама система посредством корисничког интерфејса и на разне начине покушати да пронађе потенцијалне пропусте у самом софтверу.