

Пројектовање и имплементација архитектуре великих скупова података за прикупљање, обраду и анализу *GitHub* догађаја

Design and Implementation of a Big Data Architecture for Collecting, Processing, and Analyzing GitHub Events

Бојан Радовић, Факултет техничких наука, Нови Сад

Студијски програм – РАЧУНАРСТВО И АУТОМАТИКА

Кратак садржај – У овом раду представљено је софтверско рјешење за прикупљање, обраду и анализу *GitHub* догађаја које интегрише пакетну обраду и обраду токова података у реалном времену. Уведен је приступ за оцјењивање и вредновање ентитета у оквиру платформе *GitHub* кроз метрику скор *Gazer*. Рад описује архитектуру система и примјере његове употребе. Резултати показују да предложено рјешење омогућава поређење ентитета на основу више димензија, као и анализу *GitHub* података прилагођену различитим типовима корисника.

Кључне речи: *GitHub* догађаји, анализа података, пакетна обрада података, обрада токова података у реалном времену, метрика скор *Gazer*

Abstract – *This paper presents a software solution for collecting, processing, and analyzing GitHub events that integrates batch processing and real-time stream processing. An approach for evaluating and ranking entities within the GitHub platform is introduced through the Gazer Score metric. The paper describes the system architecture and provides examples of its use. The results demonstrate that the proposed solution enables the comparison of entities across multiple dimensions, as well as the analysis of GitHub data tailored to different types of users.*

Keywords: *GitHub events, data analysis, batch data processing, real-time stream processing, Gazer Score*

1. УВОД

Савремени развој софтвера у великој мјери заснива се на употреби рјешења отвореног кода (енгл. *open source*), која представљају основу значајног броја комерцијалних и некомерцијалних софтверских система. Велики дио ових пројеката развија се и одржава у оквиру платформе *GitHub*, која омогућава сарадњу великог броја корисника, организација и заједница широм свијета.

НАПОМЕНА: Овај рад проистекао је из мастер рада чији ментор је био др Владимир Димитријески, ванр. проф.

Као резултат таквог начина развоја, платформа *GitHub* генерише огромне количине података о активностима корисника, развоју репозиторијума и интеракцијама унутар софтверских пројеката. Ови подаци представљају значајан извор информација који се могу користити за анализу развојних процеса, праћење трендова, као и процјену утицаја појединачних ентитета у оквиру екосистема.

1.1. Мотивација

Мотивација произилази из потребе за дубљим и свеобухватнијим разумијевањем активности у оквиру рјешења отвореног кода и платформе за управљање изворним кодом. Иако постоје различити алати за аналитику *GitHub* података, већина њих ограничена је у погледу детаљности, флексибилности и могућности прилагођавања специфичним потребама корисника. Постојећа рјешења пажњу усмјеравају на појединачне метрике, као што су број звјездица или активности комитовања, док недостаје интегрисан приступ који обухвата више аспеката понашања корисника, динамику развоја репозиторијума и карактеристика на нивоу организације. Такође, не постоје јасно дефинисане метрике за вредновање ентитета у оквиру платформе *GitHub* на основу њихове активности, утицаја и релевантности. Постојећи приступи најчешће се заснивају на изолованим метрикама активности, што не омогућава свеобухватну процјену утицаја и значаја појединачних ентитета.

1.2. Циљеви и задаци

Циљ овог рада је омогућавање анализе великих скупова података које генерише платформа *GitHub*, са пажњом усмјереном на могућност прикупљања, обраде, складиштења и визуализације података о активностима корисника, репозиторијума и организација. Битно је омогућити досљедност и упоредивост аналитичких приказа за различите типове корисника, уз прилагођавање начина визуализације њиховим специфичним потребама и нивоима детаљности. Такође, један од циљева је дефинисање начина вредновања ентитета у оквиру платформе *GitHub* на основу њихове активности, утицаја и релевантности. Остваривање наведених циљева захтијева пројектовање софтверског рјешења које комбинује различите модуле: пакетну обраду

података, обраду токова података у реалном времену, оркестрацију процеса обраде података, као и визуализацију резултата.

1.3. Структура рада

Рад је организован у 8 поглавља. У првом поглављу представљени су мотивација, циљеви и задаци рада. Друго поглавље даје преглед постојећих рјешења из области анализе *GitHub* догађаја. У трећем и четвртном поглављу дефинисане су битне изведене метрике. Пето поглавље представља архитектуру софтверског рјешења. Шесто поглавље даје примјер употребе софтверског рјешења. Седмо поглавље даје закључак и правце даљег развоја. Осмо поглавље садржи списак коришћене литературе.

2. Преглед постојећег стања у области

У наставку представљене су најзначајније платформе и алати који омогућавају аналитику над *GitHub* догађајима [1].

2.1. *GitLume*

GitLume [2] представља платформу намијењену анализи активности корисника и репозиторијума на платформи *GitHub*. Основна намјена система је праћење развојних активности и продуктивности корисника кроз анализу историје измјена кода и развоја пројеката током времена. Платформа омогућава визуализацију активности на нивоу корисника и репозиторијума, пружајући увид у број активности комитовања, учесталост доприноса и динамику развоја софтверских пројеката. Посебна пажња посвећена је индивидуалној активности програмера, укључујући анализу континуитета рада, обима доприноса и промјена у репозиторијуму. Захваљујући једноставном корисничком интерфејсу и пажњи усмјереној на визуелизацију активности, *GitLume* омогућава брз увид у стање и развој репозиторијума, као и праћење ангажованости појединачних учесника у софтверским пројектима.

2.2. *Augur*

Augur [3] представља платформу за анализу рјешења отвореног кода која је развијена у оквиру иницијативе *CHAOSS* (енгл. *Community Health Analytics for Open-Source Software*). Основна сврха платформе је прикупљање и анализа података о активностима корисника, репозиторијума и заједница које учествују у развоју софтвера. Платформа прикупља податке из различитих извора, укључујући *GitHub* репозиторијуме, системе за евиденцију и праћење проблема (енгл. *issue trackers*) и системе за контролу верзија, након чега их обрађује и трансформише у аналитичке показатеље. На основу прикупљених података могуће је анализирати активности сарадника, динамику развоја пројеката, брзину одговора на пријављене проблеме и степен укључености заједнице. На овај начин платформа омогућава организацијама и истраживачима да стекну увид у карактеристике и развојне трендове заједнице која учествује на рјешењима отвореног кода.

2.3. *GrimoireLab*

GrimoireLab [4] представља платформу за анализу софтверских пројеката која је развијена у оквиру иницијативе *CHAOSS*. Основна сврха платформе је прикупљање, обрада и визуализација података из различитих извора, укључујући *GitHub*, *GitLab*, листе е-поште, системе за евиденцију и праћење проблема и друге системе који се користе у развоју софтвера. Платформа омогућава изградњу свеобухватне слике о „здрављу“ софтверских пројеката, кроз анализу активности заједнице, брзине развоја, одзива на проблеме и доприноса спољних сарадника.

2.4. Идентификовани проблеми и могућа унапређења

Анализиране платформе пружају значајне могућности за прикупљање, обраду и визуализацију података из платформе *GitHub*. *GitLume* усмјерен је на анализу активности корисника и репозиторијума, *Augur* на праћење стања и одрживости рјешења отвореног кода, док *GrimoireLab* омогућава интеграцију и анализу података из већег броја извора. Ипак, уочава се простор за додатно унапређење постојећих приступа. Постојећа рјешења најчешће се ослањају на приказ појединачних статистичких показатеља и метрика, док је могућност интегрисаног вредновања и рангирања различитих типова ентитета ограничена. Корисници, репозиторијуми и организације посматрају се кроз засебне скупове показатеља, без јединственог модела који би омогућио њихово поређење на основу више аспеката активности и утицаја. Додатно, постојеће платформе углавном нуде униформне аналитичке приказе, независно од типа корисника и његових потреба. Због тога постоји потреба за рјешењем које омогућава прилагођене аналитичке приказе за различите типове корисника сходно њиховим случајевима коришћења.

3. ДЕФИНИЦИЈА СЕСИЈЕ РАЗВОЈНИХ АКТИВНОСТИ

Сесија развојних активности представља временски континуиран период активности корисника, у оквиру којег корисник интерагује са осталим ентитетима кроз различите активности, при чему временски размак између сусједних догађаја не прелази унапријед дефинисани праг неактивности. Формално, нека је дат уређен низ догађаја једног корисника:

$$E = \{e_1, e_2, \dots, e_n\}, t(e_i) \leq t(e_{i+1}) \quad (1)$$

гдје $t(e_i)$ представља временску ознаку догађаја. Сесија развојних активности дефинише се као максималан подниз узастопних догађаја $S \subseteq E$ такав да важи:

$$t(e_i) - t(e_{i-1}) \leq \Delta \quad (2)$$

гдје је Δ праг неактивности (енгл. *session gap*). У случају да је:

$$t(e_i) - t(e_{i-1}) > \Delta \quad (3)$$

сматра се да започиње нова сесија развојних активности.

Сесија развојних активности представља битну метрику за израчунавање скорa *Gazer*, јер омогућава идентификацију континуиране активности корисника.

На овај начин добија се бољи увид у стварну динамику рада, будући да се појединачни догађаји не посматрају изоловано, већ у оквиру логички повезаних цјелина.

4. ДЕФИНИЦИЈА МЕТРИКЕ СКОР *GAZER*

Посматрање појединачних метрика отежава објективно вредновање и међусобно поређење ентитета, јер свака од њих описује само један аспект њихових развојних активности. Због тога уведена је метрика скор *Gazer*, која обједињавањем више различитих метрика омогућава објективније вредновање и поређење ентитета. Метрика скор *Gazer* рачуна се различито за сваки тип ентитета али сви модели заснивају се на јединственом концептуалном оквиру који обухвата четири кључне димензије:

1. **Активност** (интензитет развојних активности),
2. **Колаборацију** (степен укључености више корисника и организација у процес развоја),
3. **Квалитет** (одржавање и стабилност система) и
4. **Популарност** (интересовање заједнице за пројекат).

Сваку од наведених димензија карактерише једна или више метрика које описују одговарајући аспект ентитета. У зависности од типа ентитета, поједине димензије имају различиту релевантност и сходно томе различите тежине у коначној метрици.

Нека је:

- E скуп свих типова ентитета система, $E = \{\text{организација, репозиторијум, корисник}\}$
- $e \in E$ посматрани тип ентитета система и
- T временски интервал анализе.

Тада се скор *Gazer* дефинише као:

$$\text{Gazer Score}(e, T) = \sum_{i=1}^n w_i * (e, T) \quad (4)$$

гдје је:

- $f_i(e, T)$ функција која мјери вриједност метрике типа i и
- w_i тежински коефицијент.

Тежински коефицијенти у оквиру метрике скор *Gazer* дефинисани су хеуристички, на основу процјене аутора рада о релативном значају појединих активности.

4.1. СКОР *GAZER* ПОЈЕДИНАЧНОГ КОРИСНИКА

Појединачни корисник представља ентитет платформе *Github* и типа корисника пројектованог система који учествује у развоју и доприноси пројектима отвореног кода. У наставку приказана је формула за рачунање скор *Gazer* за појединачног корисника.

$$\begin{aligned} \text{Gazer Score}(e, T) &= 2 * f_{grane}(e, T) \\ &+ 2 * f_{tagovi}(e, T) \\ &+ 2 * f_{sesije}(e, T) \\ &+ 4 * f_{pr}(e, T) \\ &+ 5 * f_{zvjezdice}(e, T) \\ &+ 5 * f_{pratioci}(e, T) \end{aligned} \quad (5)$$

гдје су: $f_{grane}(e, T)$ број направљених грана; $f_{tagovi}(e, T)$ број направљених тагова; $f_{sesije}(e, T)$ број сесија

развојних активности; $f_{pr}(e, T)$ број захтјева за усвајање измјена; $f_{zvjezdice}(e, T)$ број добијених звјездица и $f_{pratioci}(e, T)$ број пратилаца.

5. АРХИТЕКТУРА СОФТВЕРСКОГ РЈЕШЕЊА

Софтверско рјешење организовано је као скуп међусобно повезаних, али функционално раздвојених модула, који заједно омогућавају прикупљање, обраду, складиштење, оркестрацију и визуализацију података. Процесирање података заснива се на јасној подјели између модула за пакетну обраду и модула за обраду токова података. Пакетна обрада намијењена је периодичној обради већих количина историјских података, док је модул за обраду токова података задужен за континуирано прикупљање и обраду догађаја у реалном времену. Изнад ова два слоја налазе се модули оркестрације и визуализације, чија је улога координација извршавања процеса и презентација резултата крајњим корисницима. Битно је напоменути да су подаци организовани по принципима архитектуре Медаљон.

5.1. Модул за пакетну обраду података

Пакетна обрада података у систему реализована је кроз низ процеса који се извршавају над унапријед дефинисаним временским интервалом. У оквиру овог модула имплементирани су три процесне компоненте. Компонента за прикупљање података имплементирана је као *Python* скрипта која је задужена за преузимање података из екстерних извора, укључујући *GH Archive*, *GitHub REST* и *GitHub GraphQL API*. Прикупљени подаци уписују се у бронзани слој складишта података, Компонента за трансформацију имплементирана је као *Spark* посао који обрађује податке из бронзаног и уписује у сребрни слој складишта података. Компонента за читавање такође је имплементирана као *Spark* посао, чија је улога да на основу обрађених података рачуна агрегиране статистике за дефинисани временски период и уписује их у базу података *PostgreSQL* ради даље анализе и визуализације.

5.2. Модул за обраду токова података

Слој за обраду токова података обухвата механизме за континуирано прикупљање и обраду података који пристижу из екстерних извора у реалном времену. Овај слој имплементиран је коришћењем технологија из екосистема *Apache Kafka*, које омогућавају поуздан пренос, обраду и складиштење токова података. Циљ обраде је израчунавање и сервирање статистичких показатеља над догађајима груписаним у временске прозоре фиксне дужине. Модул за обраду токова података састоји се од четири компоненте. Генератор је компонента задужена за прикупљање података из екстерних извора, првенствено *GitHub REST API*-ја, и њихово објављивање на *Кафка* теме. Аналајзер је компонента која чита податке са *Кафка* тема и врши израчунавање статистичких показатеља у реалном времену. Конектор је компонента задужена за упис израчунатих статистика, припремљених за сервирање у базу података *PostgreSQL*. Регистар шема је компонента која омогућава централизовано

управљање шемама токова података који се размјењују преко Кафка тема.

5.3. Модул за оркестрацију

Оркестрациони модул задужен је за управљање извршавањем процесних компоненти пакетне обраде и координацију њиховог међусобног рада. Као оркестратор у оквиру система користи се алат *Apache Airflow* који омогућава дефинисање и аутоматизацију сложених токова обраде података кроз модел усмјереног ацикличног графа (енгл. *Directed Acyclic Graph, DAG*). Примјена алата *Airflow* омогућава јасно раздвајање одговорности између компоненти задужених за обраду података и компоненти задужених за управљање њиховим извршавањем. На овај начин логика обраде остаје независна од механизма планирања и оркестрације, што доприноси већој модуларности и лакшем одржавању система. Поред управљања редослиједом извршавања, *Airflow* омогућава дефинисање зависности између процеса, периодично покретање задатака, праћење статуса извршавања, као и аутоматско руковање грешкама и поновно покретање неуспјешних процеса. Централизовано управљање токовима обраде омогућава већу поузданост и контролу над процесима пакетне обраде [5].

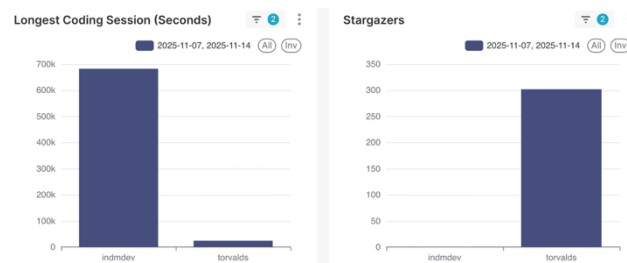
5.4. Модул за визуализацију

Визуализациони модул представља кориснички оријентисан дио архитектуре чија је основна улога презентација резултата обраде података кроз интерактивне аналитичке приказе. Овај модул обједињује резултате пакетне обраде и обраде токова података у реалном времену и омогућава њихово јединствено приказивање крајњим корисницима. За визуализацију података и приказ аналитичких резултата користи се *Apache Superset*, алат намијењен изградњи аналитичких приказа и истраживању података. *Superset* је повезан са базом података *PostgreSQL*, у коју се уписују резултати добијени из пакетне обраде и обраде у реалном времену. На основу података у бази креирани су интерактивни аналитички прикази, који омогућавају анализу статистичких показатеља и праћење активности корисника, организација и репозиторијума кроз различите временске интервале. Кориснички интерфејс омогућава филтрирање, поређење и истраживање података кроз више перспектива, чиме се корисницима омогућава брже доношење закључака и ефикасније коришћење аналитичких резултата [6].

6. ПРИМЈЕР УПОТРЕБЕ СОФТВЕРСКОГ РЈЕШЕЊА

У наставку представљен је примјер употребе софтверског рјешења из перспективе крајњег корисника. Начин коришћења система и приступ контролним таблама зависе од типа корисника и његових аналитичких потреба. Свака контролна табла садржи филтере *period* и *login*, који омогућавају филтрирање података приказаних на графиконима према изабраним варијаблностима.

На слици 1 приказан је дио контролне табле намијењене појединачном кориснику. Представљено је поређење 2 корисника: *torvalds* и *indmdev* за временски период који почиње 2025.11.07. и завршава се 2025.11.14. На графицима приказано је поређење трајања најдуже сесије развојних активности и броја звјездица добијених на репозиторијумима за одабрани период.



Слика 1. Примјер поређења трајања најдуже сесије развојних активности и броја звјездица добијених на репозиторијумима

7. ЗАКЉУЧАК

Анализирана су постојећа рјешења из области аналитике *GitHub* података и идентификована њихова ограничења. На основу уочених недостатака пројектована је и имплементирана архитектура система великих скупова података за прикупљање, обраду и анализу *GitHub* догађаја. Предложено рјешење омогућава интеграцију података из више извора, вредновање и међусобно поређење ентитета као и визуализацију добијених аналитичких резултата кроз контролне табле прилагођене различитим типовима корисника.

Могући правци даљег развоја обухватају разматрање имплементације архитектуре Капа, увођење додатних метрика и модела за рангирање, као и интеграцију са другим платформама за управљање удаљеним *Git* репозиторијумима.

8. ЛИТЕРАТУРА

- [1] “*GitHub event types*”, Приступљено: 23. април 2026. [Online]. Доступно на: <https://docs.github.com/en/rest/using-the-rest-api/github-event-types?apiVersion=2026-03-10>
- [2] “*GitLume*”, Приступљено: 23. април 2026. [Online]. Доступно на: <https://gitlume.com>
- [3] “*Augur Project, GitHub repository*”, Приступљено: 23. април 2026. [Online]. Доступно на: <https://github.com/augurlabs/augur>
- [4] “*GrimoireLab*”, Приступљено: 23. април 2026. [Online]. Доступно на: <https://chaoss.github.io/grimoirelab>

Кратка биографија:

Бојан Радовић завршио је основне академске студије 2023. године на Факултету техничких наука у Новом Саду, студијски програм Рачунарство и аутоматика.

Контакт: rbojan2000@gmail.com