

Description of the Distributed Architecture for Processing Streaming Social Network Data

Binazir Ganji¹, Ali Rezaee^{*,*}, Sahar Adabi³, Ali Movaghar⁴

1. Ph.D. Student, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. binazir.ganji@srbiau.ac.ir
2. Assistant Professor, Department of Computer Engineering, Science and Research Branch, Islamic Azad University, Tehran, Iran. * *Corresponding Author*, alirezaee@srbiau.ac.ir
3. Assistant Professor, Department of Computer Engineering, North Tehran Branch, Islamic Azad University, Tehran, Iran. sahar_adabi@iau-tmb.ac.ir
4. Professor, Department of Computer Engineering, Sharif University of Technology, Tehran, Iran. movaghar@sharif.edu

Abstract

Introduction: To analyze big data, especially streaming social network data, we require real-time and distributed systems to process streaming data with high speed and efficiency. In this paper, a distributed architecture for collecting, ingesting, processing, storing and visualizing streaming social network data based on Kappa architecture is introduced. Also, the proposed architecture includes a component for detecting anomalous data.

Method: We utilize the 4+1 architectural view models to visually illustrate the various architectural layers, components, and their interactions.

Results: The proposed architecture serves as a distributed solution designed for processing streaming social network data. We utilized the 4+1 architectural view model and UML diagrams to outline proposed architecture. This documentation clearly outlines the data processing pipeline and specifies both functional and non-functional system requirements.

Discussion: The proposed architecture is designed to process streaming social network data, leveraging distributed and parallel solutions for improved efficiency. Anomaly detection is a pivotal component integrated within the architecture to identify outlier data, enhancing processing precision and quality. By utilizing the 4+1 architectural view model and UML diagrams, the proposed architecture is effectively outlined, ensuring a well-defined structure that aids in organizing information. This structured approach provides stakeholders with tailored architectural views that cater to their individual needs and priorities. Notable functional requirements include real-time processing, while non-functional requirements encompass scalability, interoperability, portability, usability, and efficiency.

Keywords: streaming social networks, stream data processing, distributed architecture, 4+1 architectural view model, UML diagrams.

توصیف معماری توزیع شده برای پردازش داده‌های شبکه‌های اجتماعی جریانی

دوره چهارم، تابستان ۱۴۰۲
شماره دوم، صص: ۴۳-۵۳

تاریخ دریافت: ۱۴۰۲/۰۱/۲۱
تاریخ پذیرش: ۱۴۰۲/۰۳/۰۲

بی نظیر گنجی^۱، علی رضایی^{۲*}، سحر آدابی^۳، علی موقر^۴

۱. دانشجوی دکترا، گروه مهندسی کامپیوتر، واحد علوم و تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران. binazir.ganji@srbiau.ac.ir
۲. استادیار، گروه مهندسی کامپیوتر، واحد علوم و تحقیقات، دانشگاه آزاد اسلامی، تهران، ایران. (نویسنده مسئول)، alirezae@srbiau.ac.ir
۳. استادیار، گروه مهندسی کامپیوتر، واحد تهران شمال، دانشگاه آزاد اسلامی، تهران، ایران. sahar_adabi@iau-tnb.ac.ir
۴. استاد، گروه مهندسی کامپیوتر، دانشگاه صنعتی شریف، تهران، ایران. movaghar@sharif.edu

چکیده: برای تجزیه و تحلیل کلان داده‌ها از جمله داده‌های شبکه‌های اجتماعی جریانی، به سیستم‌های بلادرنگ و توزیع شده برای دریافت، پردازش و نمایش نتایج تجزیه و تحلیل جریان داده با سرعت و کارایی بالا، ضمن حفظ توالی داده‌های ورودی نیاز داریم. در این مقاله، یک معماری توزیع شده برای جمع‌آوری، جذب، پردازش، ذخیره و نمایش نتایج حاصل از پردازش داده‌های شبکه اجتماعی جریانی بر پایه معماری کاپا، پیشنهاد شده است. باتوجه به اینکه داده‌های ناهنجار، بر کیفیت و دقت نتایج حاصل از پردازش داده‌ها، تأثیر منفی می‌گذارند، لذا در معماری پیشنهادی، مؤلفه تشخیص داده‌های ناهنجار نیز در نظر گرفته شده است. ما لایه‌های مختلف معماری، اجزا و تعاملات آن‌ها را با استفاده از مدل نمای معماری ۴+۱ و نمودارهای UML مربوط به هر نما، توصیف می‌کنیم. براساس مستندات حاصل، جنبه‌های مختلف این معماری بررسی و مهم‌ترین آن‌ها استخراج می‌شود. یک ساختار واضح از معماری ارائه شده که به سازماندهی اطلاعات کمک می‌کند، دید لازم از معماری، به ذی‌نفعان داده می‌شود و آن‌ها این فرصت را دارند که روی دیدگاهی که بیشترین ارتباط را با آن‌ها دارد، تمرکز کنند و نیازهای عملکردی و غیرعملکردی سیستم، مشخص می‌شود.

واژه‌های کلیدی: شبکه‌های اجتماعی جریانی، پردازش داده‌های جریانی، معماری توزیع شده، مدل نمای معماری ۴+۱، نمودارهای UML.

۱. مقدمه

شبکه‌های اجتماعی، متشکل از انواع ابزارهای تحت وب هستند که کاربران خود را قادر می‌سازند تا ایده‌ها و اطلاعات جدید را در یک محیط تعاملی و مجازی به اشتراک بگذارند [۱]. تجزیه و تحلیل داده‌های بزرگ، از جمله شبکه‌های اجتماعی، به یک زیرساخت سخت-افزاری مقیاس‌پذیر با قابلیت‌های پردازش موازی نیاز دارد. این سیستم باید حافظه، پهنای باند، توان عملیاتی کافی داشته‌باشد و بتواند چندین کار را همزمان انجام دهد و در عرض چند ثانیه، پردازش موازی الگوریتم‌های تحلیلی پیشرفته را انجام دهد. تکنولوژی‌های سنتی، به دلیل عدم مقیاس‌پذیری، داشتن ساختاری متمرکز و ساختار داده‌های یک ریخت، کارایی لازم برای پردازش کلان داده‌ها را ندارند. فضای ذخیره‌سازی مقیاس بزرگ و منابع محاسباتی قدرتمند دو چالش اصلی در پردازش این نوع کلان داده‌ها هستند [۲]. محاسبات کلان داده، از جمله تجزیه و تحلیل شبکه‌های اجتماعی، باید به صورت پردازش توزیع‌شده و بر روی خوشه‌ای از رایانه‌ها پیاده‌سازی شود. ایجاد یک سیستم ذخیره‌سازی داده‌های بزرگ که بتواند حجم زیادی از داده‌های متنوع (متن، صدا، تصویر، ویدئو...) را جذب کند، پرهزینه و زمان‌بر است. در این سیستم‌ها، داده‌ها را می‌توان به صورت جریانی و دسته‌ای وارد کرد. سرعت، یک چالش برای فرآیند خط لوله داده است. با پیچیده‌تر شدن داده‌ها در سیستم پردازش کلان داده، توسعه آن سیستم نیز دشوارتر می‌شود. به خصوص برای پردازش بلادرنگ داده‌ها، در شبکه‌های اجتماعی، جذب داده‌ها باید متناسب با سرعت ورود داده‌ها به سیستم باشد. در غیر این صورت ممکن است، بخشی از داده‌ها وارد خط لوله داده نشوند.

هنگامی که ما با منابع داده‌های مختلف در شبکه‌های اجتماعی کار می‌کنیم، کیفیت داده‌ها یک چالش است. فرمت‌های داده ناسازگار، داده‌های تکراری و مقادیر ازدست‌رفته و نیز داده‌های غیرعادی، تجزیه و تحلیل را غیرقابل اعتماد می‌کند. اطلاعات تولید شده توسط سیستم‌هایی مانند شبکه‌های اجتماعی، دارای عظیمی برای تبدیل داده‌ها به ارزش است، اما به دلیل عدم وجود فناوری‌های مناسب برای مدیریت این داده‌ها، نمی‌توان از پتانسیل این داده‌ها، به طور کامل استفاده کرد. چارچوب‌های کلان داده، فرصت‌هایی برای مدیریت و تجزیه و تحلیل حجم زیادی از داده‌ها ارائه می‌دهند. اما قابلیت‌های آن‌ها بیشتر بر روی سرعت، تحمل خطا و موازی‌سازی متمرکز است که در مرحله دریافت داده‌ها و پردازش آن‌ها، اهمیت بیشتری دارد. اما نکته مهم این است که اگر داده‌های بی‌کیفیت و غیرعادی به این سیستم پردازشی داده‌شود، نتایج تحلیل از دقت کافی برخوردار نبوده و باعث می‌شود تا کاربران نسبت به اطلاعات استخراج شده از عملیات پردازشی، اعتماد خود را از دست‌دهند. بنابراین، تجزیه و تحلیل داده‌ها باید پس از آماده‌سازی مناسب و حذف داده‌های ناهنجار، انجام شود.

نوشتن کد برای دریافت، استخراج، پاک کردن و بارگذاری داده‌ها می‌تواند دشوار باشد. زیرا حجم داده‌های شبکه‌های اجتماعی، افزایش یافته و بسیار متنوع شده‌است. بنابراین باید حرکت به سمت پردازش خودکار داده‌ها انجام شود. در تحلیل شبکه‌های اجتماعی با حجم زیادی از داده‌ها مواجهیم و نیاز به معماری مقیاس‌پذیر است که در برابر تغییرات پایدار بوده و به راحتی قابل توسعه باشد. افزایش سرعت این محاسبات، نیازمند راه‌حل‌های توزیع‌شده و موازی است [3, 4].

بنابراین، برای حل بسیاری از چالش‌های فوق، در این مقاله، یک معماری توزیع‌شده برای پردازش داده‌های شبکه اجتماعی جریانی با در نظر گرفتن حذف داده‌های ناهنجار، پیشنهاد شده‌است.

قسمت‌های اصلی این مقاله، به شرح زیر است:

- ارائه یک معماری برای جمع‌آوری، جذب، پردازش، ذخیره و نمایش داده‌های شبکه‌های اجتماعی جریانی که قابلیت تشخیص داده‌های غیرعادی را دارد.

- برای بهبود در دسترس بودن^۲ و عملکرد^۳، این معماری، توزیع شده‌است و قابلیت استفاده از ابزارهایی با کارکرد توزیع‌شده و الگوریتم-های موازی را دارد. همچنین، یک مؤلفه تشخیص ناهنجاری، برای بهبود دقت و کیفیت نتایج پردازش، در نظر گرفته شده‌است.

- معماری پیشنهادی با استفاده از مدل نمای معماری ۴+۱ مبتنی بر نمودارهای UML، مدل‌سازی و توصیف شده‌است.

ساختار مقاله به شرح زیر است: بخش ۲ نمای کلی از کارهای مرتبط با ارائه می‌دهد، بخش ۳ معماری پیشنهادی و لایه‌های مختلف آن نشان می‌دهد. در بخش ۴ نماهای ۴+۱ مبتنی بر UML، ارائه شده است. در نهایت، بخش ۵ نتیجه‌گیری و کار آینده را تشریح می‌کند.

۲. کارهای مرتبط

در این بخش خلاصه‌ای از تحقیقات مرتبط ارائه شده‌است. در این تحقیقات از دو معماری برای پردازش کلان داده‌های جریانی استفاده شده‌است: معماری لامبدا [5] و معماری کاپا [6]. معماری لامبدا از تکنیک‌های پردازش دسته‌ای و جریانی، برای تجزیه و تحلیل کلان-داده‌ها استفاده می‌کند. در مقابل، معماری کاپا تنها بر روی جریان و پردازش بلادرنگ بدون لایه دسته‌ای، تمرکز دارد.

در [7]، یک معماری پردازش داده‌های بزرگ پنج لایه، به نام BDPA معرفی شده‌است. این معماری شامل یک لایه جمع‌آوری، لایه ذخیره‌سازی، لایه پردازش، لایه تحلیل و لایه کاربردی است تا یک رویکرد استاندارد برای جمع‌آوری، مدیریت، پردازش و تجزیه و تحلیل حجم زیادی از داده‌های ایستا یا پویا را ایجاد کند. این راهکار بر پایه معماری Lambda است. محدودیت‌هایی مثل در نظر نگرفتن سرعت داده‌ها، فرکانس و توالی داده‌ها پس از دریافت، ناتوانی در حذف داده‌های پرت در آن وجود دارد. چارچوبی بلادرنگ توسط Laska و همکاران [8] برای پردازش داده‌های جریان مکانی-زمانی دستگاه‌های

IoT پیشنهاد شده است. در این چارچوب از Apache Kafka برای دریافت داده و از Apache Storm برای پردازش داده استفاده می‌شود. مزایای کلیدی این چارچوب شامل پردازش بلادرنگ، توزیع، تحمل خطا و استفاده از سیستم کارگزار پیام است.

در پایان نامه [9] Fotiadis، راه‌حلی برای پردازش بلادرنگ داده‌ها، بر اساس معماری کاپا و پلت فرم ابری Microsoft Azure ارائه شده است. این راه‌حل، شامل دریافت داده‌های جریانی از چندین منبع مستقل و مسیریابی آن به عنوان رویداد به زیرسیستم صف پیام است. یک جزء تجزیه و تحلیل جریان، به عنوان یک زیرسیستم مصرف کننده، عمل می‌کند و رویدادها را از صف پیام می‌خواند. رابط کاربری توپیتر برای دریافت داده‌ها، استفاده می‌شود. در مقابل، Azure Event Hub برای دریافت داده‌ها، Azure Stream Analytics برای تجزیه و تحلیل داده‌ها و Power BI برای نمایش نتایج استفاده می‌شود. استفاده از پلتفرم ابری، در این راه‌حل، مزایای متعددی مانند قابلیت اجرای پردازش توزیع شده، مدیریت پردازش از طریق یک لایه پردازش واحد، مقرون به صرفه بودن، مقیاس پذیری و تحمل خطا را ارائه می‌دهد.

در مقاله [10]، نویسندگان یک چارچوب بلادرنگ برای پردازش، تبدیل، ذخیره و نمایش داده‌های جریانی توپیتر پیشنهاد کردند. این چارچوب از فناوری‌های منبع باز مانند آپاچی کافکا برای دریافت توپیت‌ها، Spark Streaming برای پردازش جریان‌های داده و پایگاه داده Cassandra برای ذخیره توپیت‌ها استفاده می‌کند. هدف این کار، استفاده از چارچوب یکپارچه پیشنهادی، برای انجام تحلیل و پردازش بلادرنگ توپیت‌ها است. معماری ارائه شده در این مقاله، می‌تواند برای انواع داده‌های جریانی مختلف استفاده شود و دارای تحمل خطای بالا، قابلیت‌های پردازش بلادرنگ، مقیاس پذیری و استفاده از پایگاه داده برای ذخیره توپیت‌ها برای پردازش مجدد است. چارچوب جدیدی برای پیش‌بینی کلان‌داده‌های بزرگ، در شبکه‌های IoT بر اساس Apache Spark در [۱۱] معرفی شده است. این چارچوب از چهار جزء اصلی تشکیل شده است: معماری شبکه اینترنت اشیا، معماری جریان داده‌های بزرگ، مدل پیش‌بینی سری زمانی برای جریان داده‌ها و یک روش پیش‌بینی جریان داده‌های بزرگ. معماری پیشنهادی، از معماری کاپا برای پردازش داده‌های جریانی استفاده می‌کند. با کمک Apache Spark و کتابخانه Mlib آن، الگوریتم‌های مختلف یادگیری ماشین بر روی داده‌ها به صورت توزیع شده و موازی با سرعت بالا اجرا می‌شوند. الگوریتم رگرسیون افزایشی خودکار، برای مدل‌های پیش‌بینی، استفاده شده است. معماری پیشنهادی بلادرنگ، قابل انطباق و با کارایی بالا است و قادر به مدیریت طیف وسیعی از الگوریتم‌های طبقه‌بندی و خوشه‌بندی است. شایان ذکر است که جذب داده در این معماری انجام نمی‌شود و عدم وجود کارگزار پیام، منجر به از دست رفتن داده‌ها می‌شود. علاوه بر این، تشخیص نقاط پرت در نظر گرفته نمی‌شود.

Khrijji و همکاران [12] یک معماری شبکه بی‌سیم متصل به ابر، به نام REDA را ارائه کرده‌اند که برای پردازش بلادرنگ داده‌های شبکه حسگر بی‌سیم جهت کنترل رطوبت خاک استفاده می‌شود. معماری شامل چندین مؤلفه است که با واحد سنسجش شروع می‌شود و داده‌ها را جمع‌آوری کرده و از طریق واحد دروازه به سرویس ابری AWS ارسال می‌کند. پردازش جریان رویداد، با استفاده از آپاچی کافکا انجام می‌شود و نتایج با استفاده از پایگاه داده MongoDB ذخیره می‌شوند. معماری پیشنهادی، انعطاف پذیر است و می‌تواند حداکثر ۸۰۰۰ پیام در ثانیه را با تأخیر کم، ذخیره کند. این مطالعه اثربخشی معماری REDA را برای پردازش بی‌درنگ داده‌های شبکه حسگر بی‌سیم، در کنترل رطوبت خاک نشان می‌دهد. Folino و همکاران [13]، چارچوبی را برای پردازش بلادرنگ و تشخیص موارد پرت در داده‌های گزارش کاربر، بر اساس پشته ELK و Kubernetes ارائه می‌کنند. پشته ELK از ابزارهای منبع باز مانند Elasticsearch، Logstash و Kibana برای جستجو، تجزیه و تحلیل و نمایش داده‌ها استفاده می‌کند. در چارچوب Kubernetes، داده‌ها باید در سیستم فایل توزیع شده، ذخیره شوند. برای تشخیص ناهنجاری‌ها در رفتار کاربران، یک الگوریتم طبقه‌بندی بر اساس سناریوی تشخیص بالماسکه پیشنهاد شده است. این چارچوب از چندین ابزار پردازش داده‌های بزرگ برای پخش داده‌ها و تشخیص ناهنجاری استفاده می‌کند. با این حال، هیچ راه‌حلی برای پردازش توزیع شده و دریافت جریان داده ارائه نشده است. نویسندگان مقاله [14]، یک سیستم توزیع شده برای پردازش داده‌های جریان مربوط به Covid-19 با استفاده از داده‌های شبکه اجتماعی توپیتر پیشنهاد کرده‌اند. روش پیشنهادی از Apache Pulsar به عنوان یک سیستم پیام‌رسانی توزیع شده، استفاده می‌کند. راه‌حل ارائه شده در این مقاله شامل تجزیه و تحلیل فضایی مجموعه داده است و برای پردازش داده‌های بزرگ جریانی، توزیع شده و مقیاس پذیر است. در جدول ۱، مقایسه بین ویژگی‌های معماری کارهای مرتبط با معماری پیشنهادی انجام شده است.

۳. معماری پیشنهادی

در این بخش، معماری مرجع سیستم پیشنهادی، معرفی می‌شود و به دنبال آن یک تصویر دقیق از لایه‌ها، اجزاء و تعاملات معماری پیشنهادی با استفاده از نمودارهای UML ارائه می‌شود. ما از مدل نمای معماری 4+1 برای به تصویر کشیدن معماری پیشنهادی خود استفاده کرده‌ایم.

جدول ۱: مقایسه ویژگی‌های معماری کارهای مرتبط با معماری پیشنهادی

مقاله	معماری مرجع	توزیع شده	کارگزار پیام	پردازش موازی	تشخیص ناهنجاری	توصیف معماری
[7] (2019)	Lambda	✓	-	✓	-	-
[9] (2021)	Kappa	✓	✓	✓	-	-
[8] (2018)	Kappa	✓	✓	✓	-	-
[10] (2021)	Kappa	✓	✓	✓	-	-
[15] (2022)	Lambda Kappa	✓	✓	✓	-	-
[12] (2022)	Kappa	✓	✓	✓	-	-
[13] (2023)	Kappa	✓	-	✓	✓	-
[14] (2023)	Kappa	✓	✓	✓	-	-
معماری پیشنهادی	Kappa	✓	✓	✓	✓	✓

داده‌ها را در زمان واقعی پردازش می‌کند. معماری کاپا ساده، انعطاف-پذیر و مناسب برای پردازش آنلاین جریان‌های داده است [6, 16]. از معماری کاپا، به‌عنوان معماری مرجع برای معماری پیشنهادی این تحقیق، استفاده شده است. شکل ۱، با الهام از معماری کلان داده [17] Microsoft Azure قسمت‌های مختلف معماری پیشنهادی را نشان می‌دهد. در این شکل، پردازش دسته‌ای از معماری کلان داده Azure در نظر گرفته نشده است و تمرکز بر پردازش جریانی است. از طرفی مؤلف‌های برای تشخیص داده‌های ناهنجار در نظر گرفته شده است.

لایه‌های مختلف معماری پیشنهادی شامل موارد زیر است:

• **لایه جمع‌آوری داده:**

لایه جمع‌آوری داده‌ها از API‌های مرتبط برای جمع‌آوری داده‌های ساختاریافته، بدون ساختار و نیمه ساختاریافته از شبکه‌های اجتماعی استفاده می‌کند.

• **لایه جذب داده:**

دریافت داده، شامل انتقال انواع داده‌های مختلف به اجزای سیستم برای پردازش است. سیستم‌های پردازش جریان توزیع شده در زمان واقعی، معمولاً از سیستم‌های واسطه پیام برای ذخیره داده‌ها قبل از پردازش استفاده می‌کنند. مکانیسم‌های دریافت پیام، می‌توانند تعادل بار و تحمل خطا را بر اساس اندازه، فرمت، فرکانس و سرعت داده‌ها بهبود بخشند [18]. یکی از ابزارهایی که می‌توان در این لایه استفاده کرد، آپاچی کافکا [19] است که به‌طور گسترده‌ای به‌عنوان یک سیستم کارگزار پیام با توان عملیاتی بالا، شناخته می‌شود.

• **لایه پردازش داده:**

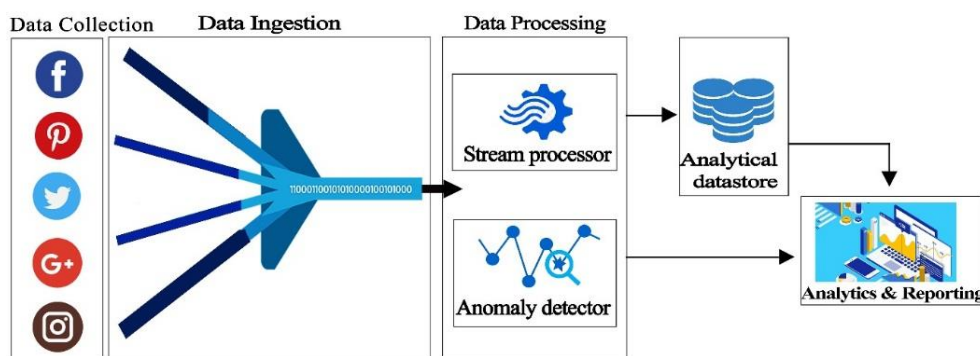
در این لایه، عملیات‌های متعددی از جمله پردازش، پیش پردازش، ادغام، نمایه‌سازی و تشخیص ناهنجاری انجام می‌شود [۲۰]. در معماری پیشنهادی، این لایه شامل دو جزء است:

۱. مؤلفه پردازش: این مؤلفه، می‌تواند عملیات پردازشی را به صورت توزیع شده و موازی با سرعت بالا، انجام دهد. یکی از ابزارهای قابل استفاده در این لایه، Apache Spark است که یک موتور پردازش توزیع شده است و با استفاده از روش نگاهشت-کاهش^۸ الگوریتم‌های پردازشی را به صورت بلادرنگ، انجام می‌دهد.

۱.۳. نمای کلی معماری پیشنهادی

معماری [3] Lambda یا معماری [6] Kappa را می‌توان برای پردازش جریان داده‌های بزرگ استفاده کرد. معماری لامبدا، یک چارچوب قوی برای پردازش مقادیر زیادی داده با استفاده از روش‌های پردازش دسته‌ای و جریانی است. این معماری از سه لایه مجزا تشکیل شده است: لایه دسته‌ای، لایه سرعت و لایه سرویس. لایه دسته‌ای، مسئول ذخیره مجموعه داده‌های اصلی و محاسبه منظم توابع دلخواه در آن مجموعه داده است.

لایه سرعت، عملکردهای محاسباتی را بر روی داده‌ها در زمان واقعی انجام می‌دهد. در نهایت، لایه سرویس نتایج پردازش در دو لایه قبلی را در قالب گزارش‌های گرافیکی در اختیار کاربران قرار می‌دهد. این ابزارها درک مجموعه داده‌های پیچیده و استخراج بینش‌های معنی‌دار را آسان می‌کنند [5]. معماری کاپا، شبیه معماری لامبدا است. با این تفاوت که شامل لایه دسته‌ای نیست، بنابراین

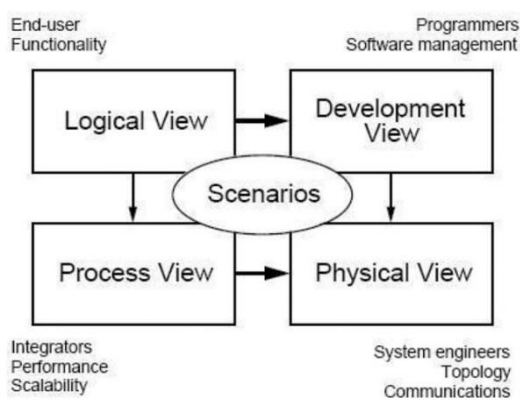


شکل ۱: معماری پیشنهادی برای پردازش داده‌های شبکه‌های اجتماعی جریانی [۱۷]

تحمل خطا و توزیع وظایف پردازش، استفاده کرد. نتایج حاصل از پردازش، کاربران را قادر می‌سازد تا تصمیمات سریع و آگاهانه اتخاذ کنند. ۶. در معماری پیشنهادی، یک مؤلفه برای تشخیص داده‌های ناهنجار در نظر گرفته شده است. با حذف داده‌های پرت، سیستم می‌تواند نتایج دقیق و قابل اعتمادتری تولید کند و کیفیت کلی فرآیند تجزیه و تحلیل داده‌ها، افزایش می‌یابد.

۴. نماهای معماری پیشنهادی

در این بخش، نماهای معماری پیشنهادی با استفاده از نمودارهای UML و مدل نمای معماری ۴+۱ ارائه شده است. ۴+۱ یک مدل نما، برای توصیف معماری سیستم‌های نرم‌افزاری، با استفاده از نماهای متعدد و هم‌رند است. نماها برای توصیف سیستم، از دیدگاه ذینفعان مختلف، مانند کاربران نهایی، توسعه‌دهندگان، مهندسين سیستم و مدیران پروژه استفاده می‌شوند. در مدل ۴+۱ نماهای مورد نظر شامل نمای منطقی، نمای توسعه، نمای سناریو، نمای فرآیند و نمای فیزیکی می‌باشند [22] [23]. در شکل ۲، مدل نمای معماری ۴+۱ آمده است.



شکل ۲: مدل نمای معماری ۴+۱ [22]

۱.۴. نمای منطقی

دیدگاه منطقی، مربوط به عملکردی است که سیستم در اختیار کاربران نهایی قرار می‌دهد. برای نمایش نمای منطقی، از نمودارهای UML استفاده می‌شود و شامل نمودارهای کلاس و نمودارهای توالی است [۲۴]. نمودارهای کلاس، یک نمای کلی ساختاری از سیستم را ارائه می‌دهند و هر کلاس، روابط مربوطه را برجسته می‌کند. نمودارهای توالی، تعاملات بین مؤلفه‌ها را در طول زمان و جریان فعالیت را در هر مرحله نشان می‌دهند. این نمودارها، به درک عملکرد سیستم و نحوه پردازش داده‌ها کمک می‌کنند. نمودار کلاس در شکل ۳ و نمودار توالی در شکل ۴ مربوط به معماری پیشنهادی آمده است.

۲. مؤلفه تشخیص ناهنجاری: نقاط داده‌ای که به‌طور قابل توجهی با سایر داده‌های مجموعه داده متفاوت هستند، داده‌های ناهنجار می‌باشند [20]. این ناهنجاری‌ها، بر کیفیت و دقت پردازش تأثیر می‌گذارند. مؤلفه تشخیص ناهنجاری، داده‌های ناهنجار را با کمک الگوریتم‌های یادگیری ماشین تشخیص می‌دهد و دقت و کیفیت تحلیل را افزایش می‌دهد.

• لایه پایگاه داده تحلیلی:

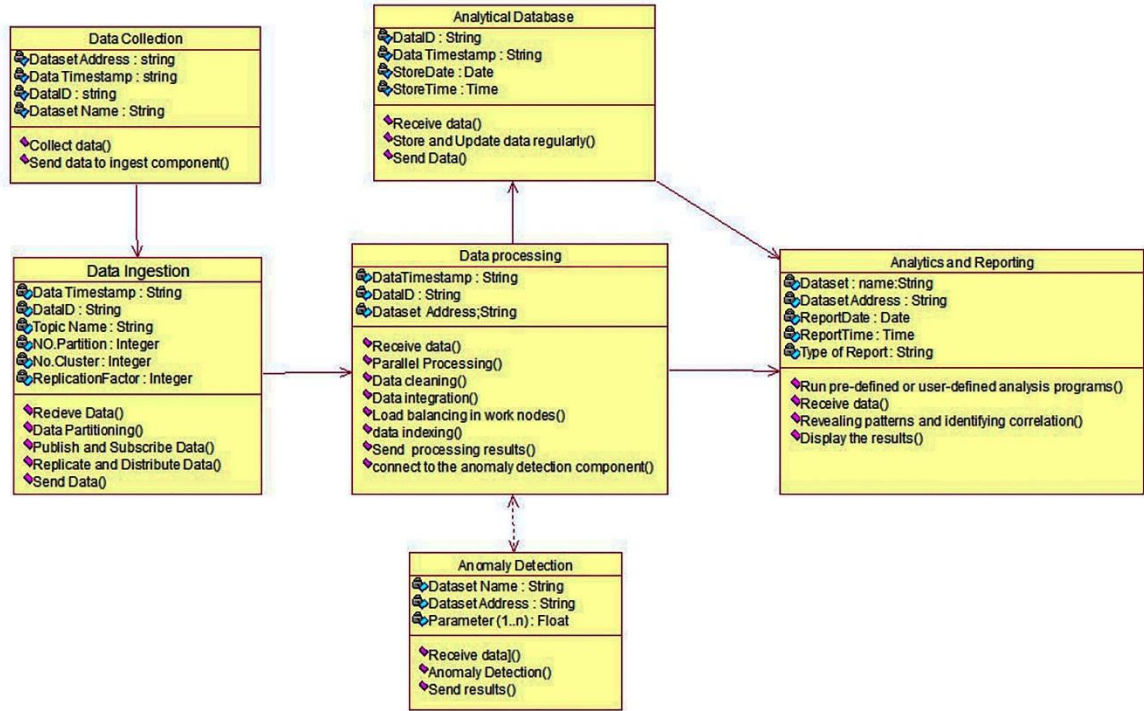
پایگاه‌های داده تحلیلی، مقادیر زیادی از داده‌های یک سازمان را ذخیره می‌کنند که برای به‌دست آوردن بینش در مورد تجارت، مشتریان و موارد دیگر استفاده می‌شود. در این پایگاه‌ها، اطلاعات مرتباً به‌روز می‌شوند. انواع پرس‌وجوها، کاربران را قادر می‌سازد تا داده‌ها را از پایگاه داده بازیابی کنند، بدون اینکه ساختار داخلی پایگاه داده را بدانند [21].

• لایه تجزیه، تحلیل و گزارش:

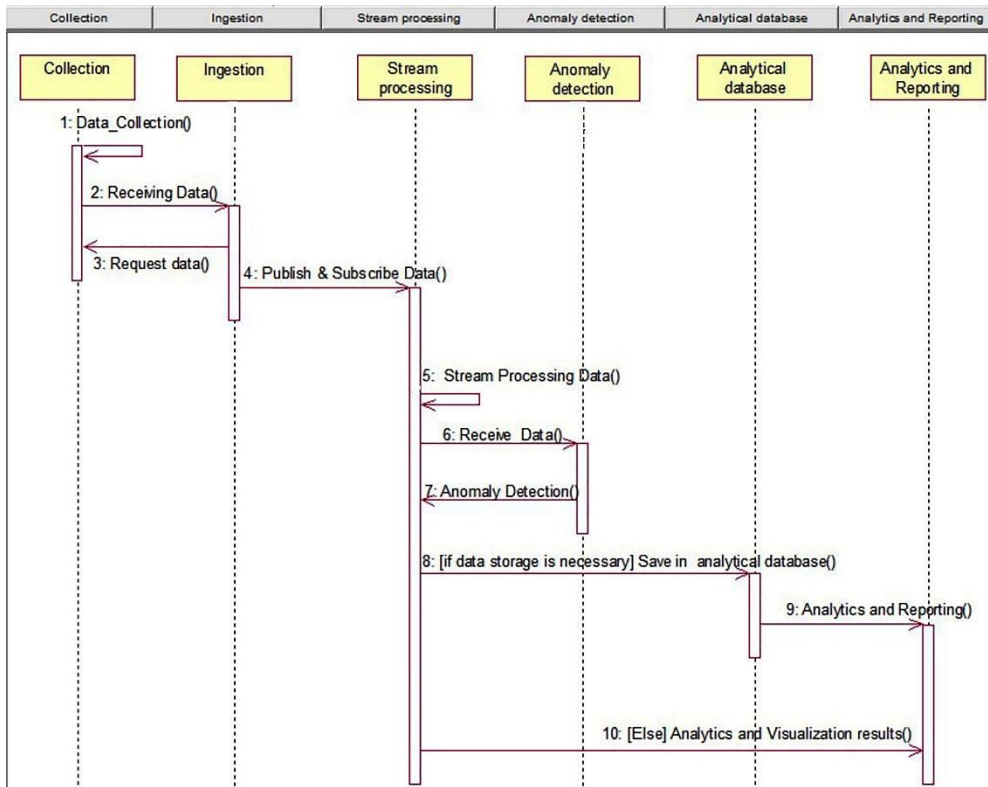
در این لایه، نتایج تحلیل داده‌ها در اختیار کاربران قرار می‌گیرد. برای این کار، ابزارهای مربوط باید از برنامه‌های تحلیلی استاندارد و تعریف شده توسط کاربر و مدل‌های داده‌کاوی تطبیقی با قابلیت اتصال به زبان‌های سطح بالا پشتیبانی کنند [7]. معماری پیشنهادی، امکان جمع‌آوری، جذب، پردازش، ذخیره‌سازی و نمایش داده‌های شبکه اجتماعی را برای تجزیه و تحلیل فراهم می‌کند. این امر شامل، استفاده از تکنیک‌های نمایش داده‌های بزرگ مانند نمودارها، نقشه‌ها، نمودارهای جعبه و ... است. با این تکنیک‌ها، کاربران می‌توانند بینش‌های ارزشمندی را در مورد فرآیندهای اساسی به‌دست آورند.

از ویژگی‌های اساسی معماری پیشنهادی، می‌توان به موارد زیر اشاره کرد:

- این معماری، براساس معماری کاپا برای پردازش داده‌های جریانی از جمله داده‌های شبکه‌های اجتماعی جریانی، پیشنهاد شده است.
- در معماری پیشنهادی، حجم زیادی از جریان‌های داده، بدون توجه به سرعت یا فرمت آن‌ها، به‌طور مؤثر پردازش می‌شوند. این معماری، می‌تواند داده‌های ساختاریافته، نیمه ساختاریافته و بدون ساختار را در اشکال و ساختارهای مختلف، مدیریت کند.
- پردازش را می‌توان بدون نیاز به ذخیره کل داده‌ها انجام داد.
- در این معماری، با استفاده از مؤلفه جذب داده‌ها، سرعت ورود داده‌ها با سرعت پردازش سیستم متناسب است. این امر، تضمین می‌کند که هیچ داده‌ای در طول فرآیند پردازش از بین نمی‌رود و تمام داده‌های دریافتی به‌طور مؤثر، پردازش و ذخیره می‌شوند.
- این معماری بر روش‌های توزیع شده و الگوریتم‌های موازی، برای پردازش کارآمد داده‌ها متکی است. ابزارهای محبوب مانند Apache Spark و Apache Kafka را می‌توان به راحتی در سیستم، برای بهبود



شکل ۳. نمودار کلاس معماری پیشنهادی



شکل ۴. نمودار توالی معماری پیشنهادی

۲.۴. نمای توسعه

این دیدگاه، به توسعه‌دهندگان و مدیران نرم‌افزار یک دید کلی از وضعیت سیستم از منظر پیاده‌سازی ارائه می‌دهد که معمولاً نمای پیاده‌سازی نامیده می‌شود. برای توصیف اجزای سیستم، UML از یک نمودار مؤلفه استفاده می‌کند. این سیستم، شامل چندین مؤلفه ضروری

است. این سیستم، شامل چندین مؤلفه ضروری است. این سیستم، شامل چندین مؤلفه ضروری است.

سیستم در زمان اجرا تمرکز دارد [۲۴]. شکل ۷، نمودار فعالیت مربوط به نمای مذکور، را نشان می‌دهد.

۵.۴. نمای فیزیکی

نمای فیزیکی، آرایش ساختاری لایه فیزیکی نرم افزار را نشان می‌دهد که شامل فعل و انفعالات فیزیکی بین اجزای آن می‌شود. نمودار استقرار، برای نشان دادن این دیدگاه استفاده می‌شود [۲۴]. در شکل ۸، نمودار استقرار معماری پیشنهادی، نشان داده شده است.

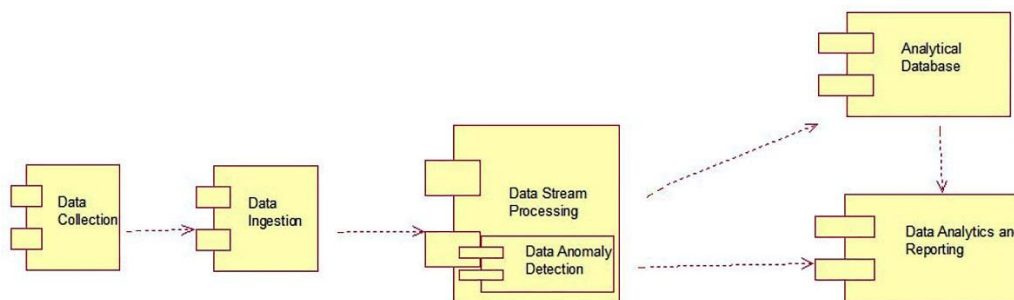
است که هر کدام ممکن است در برخی موارد دارای مؤلفه‌های فرعی باشند. انواع مؤلفه‌های اصلی مورد استفاده در معماری پیشنهادی، در جدول ۲ و نمودار مؤلفه‌ها در شکل ۵، ارائه شده است.

۳.۴. نمای سناریو

این دیدگاه، معماری را از طریق مجموعه‌ای از سناریوها توصیف می‌کند که هر کدام دنباله‌ای از تراکنش‌های وابسته به یکدیگر را نشان می‌دهد که هم توسط سیستم و هم توسط کنشگران، انجام می‌شوند و یک دیدگاه سیستمی سطح بالا، را از دیدگاه کنشگران ارائه می‌دهد و در تجزیه و تحلیل سیستم، برای استخراج نیازمندی‌ها و نشان دادن عملکرد سیستم مفید هستند [۲۴]. نمودار use case مربوط به نمای سناریو، در شکل ۶ نشان داده شده است.

۴.۴. نمای فرآیند

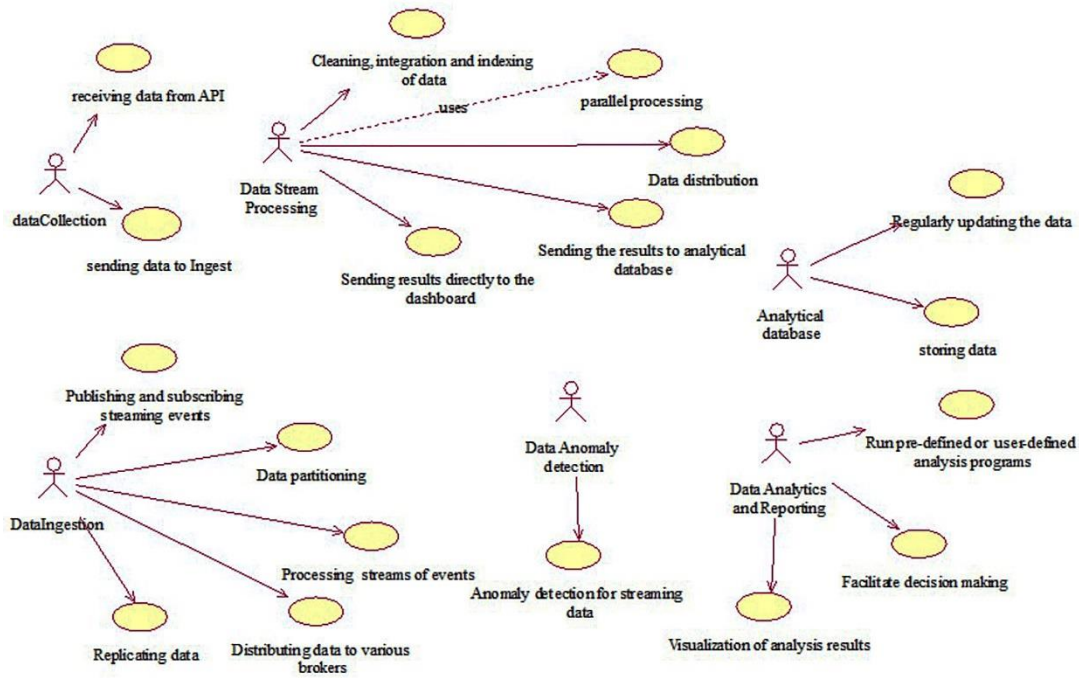
نمای فرآیند با جنبه‌های دینامیکی سیستم، سروکار دارد. فرآیندهای سیستم و ارتباطات بین آن‌ها را توصیف می‌کند. همچنین بر رفتار



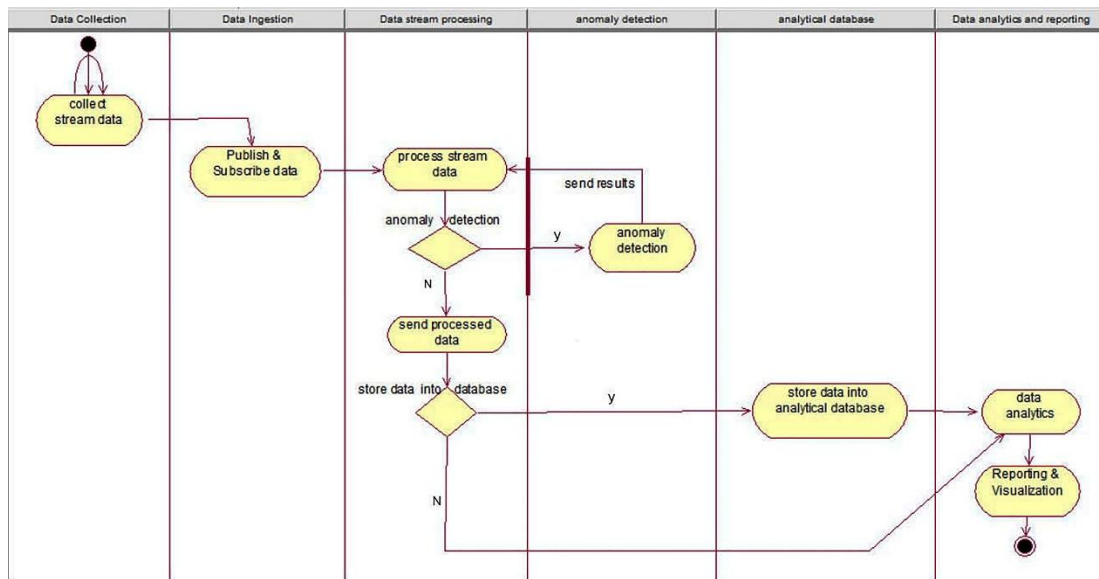
شکل ۵. نمودار مؤلفه سیستم پیشنهادی

جدول ۲: مؤلفه‌های معماری پیشنهادی

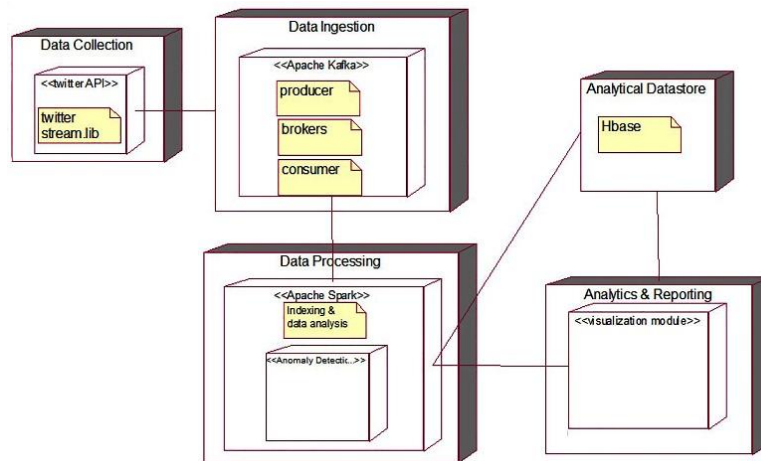
نام مؤلفه	وظیفه
جمع آوری داده	دریافت داده‌ها از API، ارسال داده به مؤلفه جذب داده
جذب داده	تکرار داده‌ها برای تحمل خطا و در دسترس بودن، انتشار و اشتراک رویدادهای جریان، پارتیشن‌بندی داده‌ها بر اساس مهر زمانی، تامین در دسترس بودن داده، پردازش رویدادهای جریان توزیع داده‌ها به کارگزاران مختلف، تعادل بار در خوشه‌های مختلف
پردازش داده‌های جریانی	پردازش موازی داده‌های جریانی، پاکسازی، یکپارچه‌سازی، و نمایه‌سازی داده‌ها، توزیع داده‌ها، افزایش تحمل خطا، تامین در دسترس بودن داده‌ها، تعادل بار، ارسال نتایج پردازش به پایگاه داده تحلیلی یا مستقیماً به مؤلفه تحلیل و گزارش‌گیری
تشخیص ناهنجاری	تشخیص ناهنجاری با سرعت بالا، افزایش دقت و کیفیت نتایج تجزیه و تحلیل
پایگاه داده تحلیلی	ذخیره حجم عظیمی از داده‌ها، به‌روزرسانی منظم داده‌ها، تامین در دسترس بودن داده‌ها، ارسال داده‌ها به مؤلفه تحلیل و گزارش‌گیری
تحلیل و گزارش‌گیری	مصورسازی نتایج تجزیه و تحلیل داده‌های ذخیره‌شده یا نمایش خروجی به‌صورت بلادرنگ، تسهیل و تسريع در تصمیم‌گیری کاربران، امکان استفاده از برنامه‌های تحلیل از پیش تعریف‌شده و همچنین برنامه‌های تحلیل تعریف‌شده توسط کاربر.



شکل ۶. نمودار use case در معماری پیشنهادی



شکل ۷. نمودار فعالیت در سیستم پیشنهادی



شکل ۸. نمودار استقرار سیستم پیشنهادی

۵. نتیجه‌گیری و کارهای آتی

در این مقاله، معماری پیشنهادی می‌تواند برای پردازش توزیع‌شده داده‌های شبکه‌های اجتماعی جریانی، استفاده شود. معماری پیشنهادی، بر اساس معماری کاپا است و می‌تواند داده‌های جریانی از جمله داده‌های شبکه‌های اجتماعی را جمع‌آوری، جذب، پردازش، ذخیره و نمایش دهد. مؤلفه تشخیص ناهنجاری، برای شناسایی داده‌های پرت در معماری پیشنهادی در نظر گرفته شده است که عملکرد آن دقت و کیفیت نتایج پردازش را افزایش می‌دهد. ما از مدل نمای ۴+۱ و نمودارهای UML مربوط به آن نما، برای به‌تصویر کشیدن معماری پیشنهادی خود، استفاده کرده‌ایم. معماری از نماهای مختلف مثل منطقی، توسعه، فرایند، سناریو و فیزیکی به کمک نمودارهای UML مربوطه، به صورت دقیق و از جنبه‌های مختلف توصیف شده است. بنابراین، ذی‌نفعان شناخت کافی از سیستم را به دست می‌آورند و ارتباط و همکاری بین ذی‌نفعان تسهیل می‌شود، چراکه آن‌ها برای بحث درباره سیستم، به مستندات یکسانی دسترسی دارند. بر اساس مستندات حاصل، خط لوله پردازش داده، مشخص شده است. مؤلفه‌ها و نحوه تعامل آن‌ها در نمودارهای مختلف ارائه شده‌اند. در برخی نمودارها، جزئیات عملکرد مؤلفه‌ها مورد توجه نیست، صرفاً چیدمان و نحوه پیاده‌سازی آن‌ها مدنظر است، مثل نمودار مؤلفه یا استقرار. از طرفی برخی نمودارها، جزئیات مؤلفه‌ها و نحوه تعامل را نمایش می‌دهند، مثل نمودار کلاس یا فعالیت. با توجه به این مستندات و ویژگی‌های معماری پیشنهادی، می‌توان نیازهای عملکردی و غیرعملکردی سیستم را مشخص نمود. از جمله نیازهای عملکردی این سیستم، پردازش بلادرنگ است. چراکه در این سیستم، با توجه به استفاده از داده‌های جریانی و ماهیت متغیر آن‌ها و عدم امکان پردازش دسته‌ای، پردازش بلادرنگ داده‌ها ضروری است. که مؤلفه‌های جذب،

پردازش و تحلیل و گزارش‌گیری، با استفاده از راهکارهای توزیع‌شده و الگوریتم‌های موازی، این نیاز را می‌توانند به خوبی فراهم کنند.

نیازهای غیرعملکردی می‌توانند شامل مقیاس‌پذیری، قابلیت همکاری بین مؤلفه‌ها، دسترسی پذیری، قابل حمل بودن، سهولت در استفاده و کارایی باشند. این سیستم، با حجم متغیر و معمولاً رو به افزایش داده، روبرو است، که تغییر در حجم داده‌ها، نباید تأثیری در پیکربندی سیستم داشته باشد. این امر لزوم مقیاس‌پذیر بودن معماری را تأکید می‌کند. به دلیل حجم بالای داده‌های ورودی و نیاز به سرعت بالای پردازش، این معماری باید کارایی بالایی داشته باشد که قابلیت استفاده از راهکارهای توزیع‌شده و موازی، افزایش کارایی و نیز دسترسی پذیری را میسر می‌کند. از طرفی به دلیل پردازش خودکار داده‌ها و عدم دخالت کاربر در خط لوله پردازش، مؤلفه‌ها باید قابلیت همکاری بالایی داشته باشند و به سکوی نرم‌افزاری خاصی وابسته نباشد و به‌سادگی قابل استفاده و قابل حمل باشند.

در تحقیقات آتی، می‌توان معماری پیشنهادی را به کمک آزمون‌های رسمی بدون درگیر شدن با پیچیدگی پیاده‌سازی معماری، بررسی کرده و آن را از دیدگاه‌های مختلف، ارزیابی کرد. همچنین می‌توان یک الگوریتم مؤثر و بلادرنگ، برای تشخیص ناهنجاری داده‌های جریانی پیشنهاد داد و یا معماری را برای پردازش بلادرنگ داده‌ها و روش‌های کاهش زمان اجرا، مورد ارزیابی قرارداد.

- processing in wireless sensor networks". The Journal of supercomputing, **78**(3): p.3374-3401 ,2022, DOI: 10.1007/s.6-03955-021-11227
- [13] G. Folino, C. Otranto Godano, and F. S. Pisani, "An ensemble-based framework for user behaviour anomaly detection and classification for cybersecurity". The Journal of Supercomputing, 2023, DOI: 10.1007/s-023-11227-05049x.
- [14] Y. M. Özgüven and S. Eken, "Distributed messaging and light streaming system for combating pandemics: A case study on spatial analysis of COVID-19 Geo-tagged Twitter dataset". Journal of Ambient Intelligence and Humanized Computing., **14**(2), p. 773-778, 2023, DOI:10.1007/s.0-03328-021-12652
- [15] J. Rosandic, "Real-Time Streaming Data Management, Processing, Analysis and Visualization", 2022.
- [16] A. Barradas, A. Tejada-Gil, and R.-M. Cantón-Croda, "Real-Time Big Data Architecture for Processing Cryptocurrency and Social Media Data: A Clustering Approach Based on k-Means". Algorithms. **15**(5): p. 140, 2022, DOI: <https://doi.org/10.3390/a.15.5.140>.
- [17] *Big data architecture style*, [cited 2024, Available from: <https://learn.microsoft.com/en-us/azure/architecture/guide/architecture-styles/big-data>.
- [18] J., Alwidian, et al., "Big Data Ingestion and Preparation Tools". Modern Applied Science. **14**(9), 2020, DOI: 10.5539/mas.v14n9p.12
- [19] Sax, M. J., *Apache Kafka*. Encyclopedia of Big Data Technologies, ed. S. Sakr and A. Zomaya. Cham: Springer International Publishing, 1-8, 2018.
- [20] M. R. Basirati, et al., "Understanding changes in use cases: A case study". IEEE 23rd International Requirements Engineering Conference (RE): p. 352-361, 2015.
- [21] D. Hawkins, "Identification of Outliers, Springer Netherlands", 1980.
- [22] T. Contributor, *analytic-database*. 2024 [cited 2024, Available from: <https://www.techtarget.com/searchbusinessanalytics/definition/analytic-database>.
- [23] P. B. Kruchten, "The 4+1 view model of architecture". IEEE software. **12**(6): p. 42-50, 1995.
- [24] M. Kontio, "Architectural Manifesto: Designing Software Architectures. Part 5. Introducing the 4+1 View Model". IBM developerWorks, 2005.
- [1] Bakeshlu, M. and M. Tahghighi Sharbyan (2023), A New Approach Based on Deep Learning Algorithms to Study Effective Factors of Using Social Networks on Students' Performance, Intelligent Multimedia Processing and Communication Systems (IMPCS), 3(4)[Persian].
- [2] Jafari, M. and A. Kalbasi (2021), A Comparative Study of Open-Source Software for Deployment and Management of Cloud Computing Utilizing a Big Data Processing Quality Mode, Intelligent Multimedia Processing and Communication Systems (IMPCS), 2(4)[Persian].
- [3] B. Yadranjiaghdam, S. Yasrobi, and N. Tabrizi, "Developing a real-time data analytics framework for twitter streaming data". IEEE International Congress on Big Data (BigData Congress): p. 329-336, 2017.
- [4] M. Y. Amare and S. Simonova, "Learning analytics for higher education: proposal of big data ingestion architecture". SHS Web of Conferences. 92, 2021, DOI: <https://doi.org/10.1051/shsconf/20219202002>
- [5] J. Warren and N. Marz, *Big Data: Principles and best practices of scalable realtime data systems*. Simon and Schuster, 2015.
- [6] J. Kreps, *Questioning the Lambda Architecture*. 2014 [cited 2024, Available from: <https://www.oreilly.com/radar/questioning-the-lambda-architecture/>.
- [7] J. Y. Zhu, B. Tang, and V. O. Li, "A five-layer architecture for big data processing and analytics". International Journal of Big Data Intelligence. **6**(1): p. 38-49, 2019, DOI: 10.1504/IJBDI.2019.97399
- [8] M. Laska, et al., "A scalable architecture for real-time stream processing of spatiotemporal IoT stream data—Performance analysis on the example of map matching". ISPRS International Journal of Geo-Information. **7**(7): p.238, 2018.
- [9] O. Fotiadis, *Distributed stream and event processing pipeline in serverless architecture*. University of Piraeus (Greece), 2021.
- [10] S. Arora and R. Rani, "A Novel Framework for Distributed Stream Processing and Analysis of Twitter Data". International Conference on Innovative Computing and Communications: p. 147-161, 2021.
- [11] J. Rosandic, *Real-Time Streaming Data Management, Processing, Analysis and Visualisation*. University of Zagreb, 2022.
- [12] S. Khriji, et al., "Design and implementation of a cloud-based event-driven architecture for real-time data

پی‌نوشت

⁶ Data ingestion layer

⁷ Data processing layer

⁸ Map-reduce

⁹ Analytical data store layer

¹⁰ Analytics and reporting layer

¹ Pipeline

² Availability

³ Performance

⁴ 4+1 architectural view model

⁵ Data collection layer