SRTTU

Journal of Electrical and Computer Engineering Innovations
(JECEI)
Journal homepage: http://www.jecei.sru.ac.ir

JECEI

Research paper

A Novel Architecture Based on Business Intelligence Approach to Exploit Big Data

M. R. Behbahani Nejad[1,*], H. Rashidi[2]

[1]*Department of Computer & IT, Qazvin Branch, Islamic Azad University, Qazvin, Iran.*

[2]*Faculty of Statistics, Mathematics and Computer Science, Allameh Tabataba'i University, Tehran, Iran.*

Article Info

Article History:
Received 13 January 2022
Reviewed 15 March 2022
Revised 02 June 2022
Accepted 06 June 2022

Keywords:
Architecture
Big data
Business intelligence
Hadoop
CPN

*Corresponding Author's Email Address: *Reza2005nejad@gmail.com*

Abstract

Background and Objectives: Big data is a combination of structured, semi-structured and unstructured data collected by organizations that must be stored and used for decision-making. Businesses that deal with the business intelligence system, as well as their data sources, have a major challenge in exploiting Big Data. The current architecture of business intelligence systems is not capable of incorporating and exploiting Big Data. In this paper, an architecture is developed to respond to this challenge.

Methods: This paper focuses on the promotion of business intelligence to create an ability to exploit Big Data in business intelligence. In this regard, a new architecture is proposed to integrate both Business Intelligence and Big Data architectures. To evaluate the proposed architecture, we investigated business intelligence architecture and Big Data architecture. Then, we developed a Unified Modeling Language diagram for the proposed architecture. In addition, using the Colored Petri-Net, the proposed architecture is evaluated in a case study.

Results: The results show that our architectural system has a higher efficiency in performing all steps, average time, and maximum time compared to business intelligence architecture.

Conclusion: The proposed architecture can help companies and organizations gain more value from their data sources and better support managers and organizations in their decision-making.

Introduction

Today, the use of Business Intelligence (BI) systems to support decision-makers at different levels of decision-making in companies and organizations is an essential requirement ([1], [2]). As the inputs to these systems are becoming more and more valuable, the results of these systems will improve the support of managers in decision-making. With the advent of a new phenomenon called Big Data, all IT issues such as intelligent business systems have been affected. Companies and organizations with business intelligence systems that also have data sources of big data type found themselves in trouble [3]. In this paper, the business intelligence architecture will be developed to solve this problem. The most important issue is the problems caused by the entry of Big Data into the business intelligence system because BI systems with their current architecture are not able to use Big Data. As a result, this paper focuses on the promotion and development of business intelligence to create the ability to exploit Big Data. This paper begins with highlighting the concepts of software architecture. We then focus on the way Big Data architecture is examined in terms of how to store, load, manage, and analyze data. Thereafter, concepts related to the BI are explored. Our new model derives from the architecture of intelligent architecture and business intelligence architecture. Based on the evaluation indicators that are extracted from the review studies [4], the business intelligence architecture and the new model are compared and weaknesses are identified to determine which architecture has the most valuable value for a company or organization. The proposed architecture is evaluated with the assistance of Unified modeling language and Colored Petri-Nets (CPNs).

This paper focuses on the lack of support for the architecture of business intelligence systems from Big Data as one of the most important information resources of companies and organizations. The collision of two categories of business intelligence systems and Big Data together causes some problems in business intelligence systems. One of these problems is the impossibility of analytic, storage, and loading Big Data in the business intelligence system. This will make IT systems unable to value Big Data, and so much of the information resources of companies and organizations that have Big Data cannot be exploited by business intelligence systems. In this paper, we have tried to provide a solution to this issue by presenting a new model for BI business architecture.

The purpose of this research is to solve the problem of business intelligence systems in dealing with Big Data. For this purpose, a new model is presented in which the architecture of business intelligence and Big Data has become integrated. The rest of this paper is organized as follows. Section 2 presents background and related works. Section 3 presents the proposed architecture. Section 4 provides an evaluation of the proposed architecture with a case study. Section 5 is considered for the summary and conclusion.

Background and Related Works

In this section, the software architecture, business intelligence architecture, and Big Data architecture are briefly examined.

Software Architecture

Garlan and Shaw [3] define the software architecture as "a collection of computational components or simply components together with a description of the interactions between these components the connectors". In 2000, IEEE defined architecture as the fundamental organization of a system embodied in its components, their relationships to each other, the environment, and the principles guiding its design and evolution [5]. Software architecture is any system where software contributes essential influences to the design, construction, deployment, and evolution of the system as a whole ([6], [7]).

Software Architecture Style

According to a definition by Clements [8], "An architectural style is a dedication of components and communications among them with each other along with a set of rules and limitations about how to use them". In another definition, according to Taylor [9] "An architectural style is a named collection of architectural design decisions that (a) are applicable in a given development context, (b) constrain architectural design decisions that are specific to a particular system within that context, and (c) elicit beneficial qualities in each resulting system."

If the category of Garlan and Shaw's [3] is considered with architectural styles, then there are generally five types of architectural style: Dataflow style, Data-driven style, styles based on the promotion, Independent-component styles, and Virtual machine styles. These architectural styles are described more in detail below.

The first architectural style is Dataflow [7]. In this style, the architecture of the system determines how its data is exchanged between different components. In other words, the way data flows in the system plays a decisive role in the behavior of the system. The flow of data in these systems is very similar to the implementation of the logic of programming languages. Usually, data-flow systems can be a good option for modeling any kind of workflow. In these systems, the presence of at least two elements that flows between them is required. The processing is mainly performed in them, meaning that the output of an input element(s) will be in the data flow direction. The main subcategories of Dataflow styles include Pipe and Filter styles. The Filter includes several elements that are responsible for processing input data and converting them to output data. The Pipe establishes communication between filters, transfer data, and information. There are some rules and restrictions. For example, the type of pipes, their capacity, how they combine filters, and so on is considered as the rules and constraints.

The second architectural style is Data centered [7]. Today, most organizations around the world have a strong dependence on their data. Maintaining a company's data is vital to the extent that large companies are willing to spend millions of dollars to secure and maintain their data. In an environment that is so important to data preservation, the emergence of software-based software architectures based on Persistent Data is not surprising. With this strong motivation, a lightweight architecture emerges as a repository that provides the basis for sharing information among the components, individuals, and organs of data sharing. The main components of the tank style are: (a) The central data repository, which is, in fact, a large data structure that is shared between processes and various departments; (b) processing elements that are potentially independent of each other. This means that with the help of the central repository, data can satisfy all their communication needs and do not need to communicate directly with each other. The third architectural style is Call/Return [7]. It is a style of software architecture that includes a variety of styles as well as the layered architecture. Systems that follow the layered style are inherently hierarchical. In a layered system, different layers provide transparency for the users.

The fourth architectural style is Service-oriented architecture [7]. It has been developed in recent years, known as service-oriented architecture (SOA). This style can somehow be expanded into a layer style or component-based style. The SOA is a model for developing software systems. Given the growth of information systems, organizations need to respond quickly to new business needs. While existing software architectures have provided some relief, evolutionary service architecture is a step-by-step service that helps organizations manage complex challenges [10].

The SOA is a matured component-based architecture, object-oriented design, and distributed systems. The SOA is a style of software design in which services are provided to the other components by application components, through a communication protocol over a network. The basic principles of the SOA are independent of vendors, products, and technologies [11]. The SOA enables application functionality to be provided as a set of services, as well as the creation of applications that make use of software services. The services are loosely coupled because they use standards-based interfaces that can be invoked, published, and discovered. In the SOA, the services are focused on providing a schema and message-based interaction with an application through interfaces that have application scope, rather than component-based or object-based. The SOA service should not be treated as a component-based service provider.

The SOA style can package business processes into interoperable services, using a range of protocols and data formats to communicate information. In the SOA, the clients and other services can access local services running on the same tier, or access remote services over a connecting network [11]. The looseness of the connection between the components of the software leads to their reusability, and the software is based on the service. In the SOA, services are divided into three categories: Service Request, Service Provider, and Registry Service.

Architectural Evaluation

For many years, analysts, engineers, and scientists have developed and used models to deal with complex systems. A model approximates the features of a real system it can also be used to evaluate systems that are not feasible in terms of method and economy before designing a system, regardless of design or initial output. As a result, existing information mismatches between different models can be overcome by reducing the high semantic distance between high-level needs and low-level architectural products.

Modeling outputs with reflecting some characteristics of the quality attributes give architects and designers of complex systems the power to visualize the entire system. Architecture is the first step in software development that can be traced to quality requirements. Qualitative attributes are considered at all stages of design and implementation and, if supported by architecture, be more easily detectable.

Models demonstrate runtime behavior by displaying architectural characteristics that can be used to evaluate many of the quality attributes, including performance and reliability. An executable model of software architecture is an implementation of the system, in which features are displayed, that includes non-narrative needs.

Applied architecture is created in the early stages of software development to reduce the risks associated with performance, operational capability, reliability, and so on. Having operational models in the initial phases provides the ability to evaluate the dynamic behavior of the system in different situations and to solve the existing problems. The real-time requirements with systematic needs and acquisition of the proper conditions for optimizing a system are the main issues that can be achieved by using executive models. Typically, modeling tools are used to create an implementation model.

There are various modeling tools available to display application architecture. The most important of these modeling tools are Petri Networks, queuing networks, simulation models, and process algebra. In addition, some of the languages that describe the architecture can also display a running architecture.

Architecture has a very important role in the software production process. Because the architect is the one who deals with all the stock-owners, they become a very influential person in the process. Designing an appropriate architecture due to the vagueness of the architectural specification is a very difficult task. With an architectural execution model, many architectural steps can be completed with high accuracy because the execution model at this level lowers the errors, recognizes Easier needs, better analyzing and evaluating of the system, and simplifies the presentation of the architecture.

Unified Modeling Language

The Unified Modeling Language is a Semi-formal and standard language for easy description of software architecture that is used to address the requirements of software engineering expertise. The main purpose of UML is to use its high descriptive power to model software architecture. The methods used in UML can handle only certain issues. Evaluation of software systems is not possible because UML is not a convenient approach for evaluation. Therefore, to evaluate software systems, it is necessary to convert the actual model to the formal model.

The main problem with UML is in determining how to evaluate and analyze the system architecture using the documentation before the production of software. Presenting an effective method to evaluate and analyze the efficiency based on the software architecture may contribute to driving a software project successfully forward ([12], [13]). Since the UML-based system is not applicable, the system's behavior verification is delayed until its implementation; hence, the Colored Petri-Net (CPN) is used as an applicable model of software architecture. It is in particular well-suited for modeling systems in which communication, synchronization, and resource sharing are important. By transforming the actual model into a formal model, the possibility of evaluating the software architecture's performance on the official model is provided.

Qualitative features are the same non-obligatory system requirements that are largely determined by an architectural style. Performance, reliability, security, availability, usability, modifiability, portability, and testing capabilities are the most qualitative features to evaluate any software architecture.

Performance is the main quality attribute of software that demonstrates how well the software works concerning time-dependent issues [14]. Software performance is the process of predicting and evaluating whether the software satisfies performance goals defined by the users. The early identification of unsatisfactory performance of Software Architecture (SA) can greatly reduce the cost of design change. This is because correcting a design flaw is more expensive the later the change is applied during the software development process [14].

Because the performance is around timing, events (interrupts, messages, requests from users, or the passage of time) occur and the system must respond to them. There are a variety of characterizations of event arrival and the response, but the performance is essentially concerned with how long it takes the system to respond when an event occurs [7]. Performance appraisal at the early levels of software development reduces costs, risks development, and so on. Therefore, performance has a very important role to play in the success of software systems and an evaluation of the efficiency of the entire software development process should be considered.

Template-based software architecture is described with three different diagrams, including class diagrams, use case diagrams, and sequence diagrams. These diagrams should describe the behavior and architecture of software architecture. CRC (Class, Responsibilities, and Collaborators) graphs show the static structure of software architecture as a software component and the relationship between them. The use case diagram also specifies the services provided by the software system and describes the order of the behavior of the system.

The Object Management Group (OMG) introduced several UML extensions [15]. This group defined the SPT profile for scheduling, performance, and time specification. The performance sub-key, similar to other profiles, uses stereotypes and labeled values to support the expansion process. Each profile contains several stereotypes. Labeled values are attributes of stereotypes in the profile and are linked to the model's key element as explanations.

Evaluation of the performance of the software architecture using the characteristic sub-heading of efficiency and time is an approach proposed to build software systems so that the qualitative objectives are visible. The proposed approach builds on the UML Performance Sub-Platform for the proper modeling and evaluation of software performance throughout the software development process.

The use case diagram models the user's use of the system and is suitable for evaluating the system's performance. Because a set of use cases is used for the performance evaluation process, it is necessary to create a performance model for each use case of the software.

The sequence diagram describes the communication pattern established by the samples. The role of the samples in the order of the diagram is to accomplish a specific objective, namely interaction. From the point of view of efficiency, certain elements and structures are used to model the system load in the sequence diagram. Any message in the graph can be attached to a condition that expresses the probability that the message will be sent. In contrast, a component diagram maintains the role of service centers and their characteristics in terms of efficiency goals.

Petri Networks

The theory of Petri Networks was introduced by Carl Petrie [16]. The use of Petri's network to evaluate the software architecture and create an executive model due to its simplicity and high availability is very much considered. Petri Networks are displayed graphically, and they provide a mathematical framework for Analysis, Validation, and Performance evaluation. The focus of Petri's networks is on synchronization, coherence, and asynchronous operation.

These networks are powered by the power of systems behavioral modeling and are used as a tool for describing architecture.

The Colored Petri-Net has been introduced as a model developed from Petri Networks [16]. The Color Petri-Net uses the capabilities of simple Petri Networks and programming languages. As shown in Fig. 1, data values in these networks are carried by the beads. Using the timestamp for a nut, it's easy to calculate the time of activation and the transfer of the nut to the destination location. Suppose that the goal is to evaluate the performance that is running from time to time – in which case it would be enough to refer to the timestamps and calculate the architectural efficiency that is associated with it.

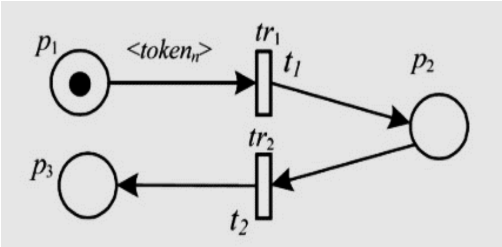

Fig. 1: A Colored Petri-Net with a timestamp for a bead [16].

Various tools support Colored Petri-Nets, including the CPN Tools software provided by the University of Aarhus, Denmark.

The first version of this software was released in October 2001 that is used for editing, simulating, and analyzing this kind of Petri-Nets.

Business Intelligence

Business Intelligence (BI) is the art of gaining a business advantage from data [17]. It is a technology-driven process for analyzing data and delivering actionable information that is used by managers, analysts, and executives to make informed business decisions. Fig. 2 shows a high-level business intelligence architecture that is used in practice.

This Fig. 3 shows that BI architecture consists of four components [18]. The first component is the data warehouse. It is a large repository of well-organized historical data. The second component is Business analytics, which are the tools that allow the transformation of data into information and knowledge. The third component is Business performance management (BPM) which allows monitoring, measuring, and comparing key performance indicators. The fourth component is the User interface (e.g., dashboards) that allows access and easy manipulation of other BI components [4].

The main business intelligence architecture is in the form of a service, as shown in Fig. 3. In this figure, there are three components for services, namely, Analytical and Reporting services, Data Management Services, and Integration Services.

The data sources can be relational Databases, Files sources, and other sources.

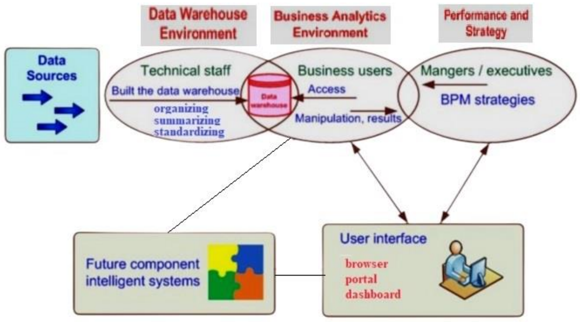

Fig. 2: A High-Level Architecture of BI [18].

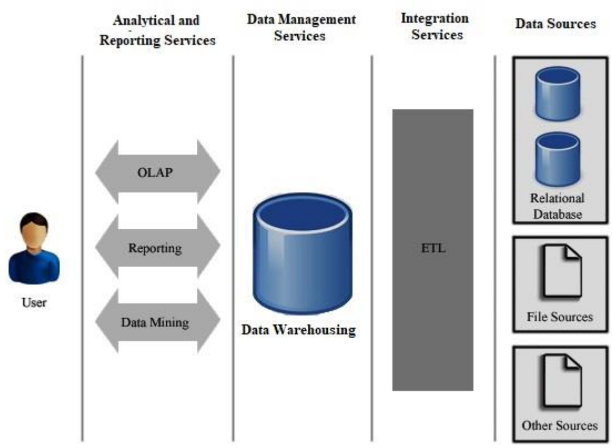

Fig. 3: Business Intelligence Architecture [4].

Big Data

Big Data is an abstract concept [19]. So far, there are many definitions of Big Data. In 2001, Doug Lenny of the Gartner Institute introduced a 3V model for defining Big Data [20]: "Data that is growing at a very high rate, has plenty of volumes so that it occupies a large amount of disk space…In addition, they are very diverse, that is, they consist of different structures of data". In other definitions, more features of metadata were provided, including data value, the complexity of data, the accuracy of data, and, based on these features, 4V, 5V, and even 7V models were also presented.

Apache most influential company in the field of Big Data. It introduced the main feature of a dataset that can be referred to as Big Data, which makes it impossible to store, manage, and process those data using common computational methods, and the rest of the Big Data Features Sub-attributes [19]. Apache defined Big Data as follows: "Big Data is referred to as a set of data that cannot be stored, managed, or processed by conventional computing methods." And in the forthcoming article, this definition is the basis of work.

Apache's definition of Big Data seems to be closer to reality [20]. According to this definition, data that has 3V attributes is Big Data, and data that does not have 3V attributes, but is not commonly analyzed, is Big Data. If

Gartner's definition of Big Data is to be accepted, it is suggested to refer to data that does not have 3V attributes but cannot be stored and analyzed in the usual way, called "Semi-Big Data" or Semi Big Data (Table 1).

Table 1: Examining the types of data in terms of 3V features, and analyzing and storing in routine ways

Data types	Ability to analyze and store commonly used	3V features
Semi-Big Data	0	0
Structured data	1	0
Big Data	0	1
Such a situation is not possible	1	1

In Table 1, routine methods are referred to as methods that can be used to analyze and store structured data. The term semi-Big Data can be defined as follows: A dataset that has one or more attributes of Big Data features (V3 or V4 or V5 or ...) but can be stored, analyzed, managed, and controlled. They do not exist in the usual way at the expected time.

It is also recommended that the Semi-Big Data and Big Data be called "NODATA". "NODATA" stands for "NOT ONLY DATA", or it could be an abbreviation for "NOT ONLY STRUCTURE DATA". "NODATA" data type is usually stored in NOSQL databases. The words NODATA and NOSQL are similar in appearance. The term "NODATA" can be defined as a set of data that cannot be stored, analyzed, managed, and controlled by commonly used computing methods. However, "NODATA Technology" refers to technologies that can store and provide analysis, management, and control of NODATA.NODATA is a term that includes Big Data and Semi-Big Data.

Database Big Data

Since over 80% of the world's data is unstructured, relational databases are unable to store and manage such data [22]. To store this data, unmatched databases should be used. In general, there are four types of non-marketable databases classified according to columns, documents, key values, and graphs.

Hadoop

Apache is the most widely used company in the field of Big Data [19]. The company has been supporting and supporting the largest Big Data project called Hadoop, which has been published in an open-source. The extension of Hadoop has made a library of Big Data-related projects that includes a large number of sub-projects. The most important projects in the library are the distributed header file system project, which is responsible for data storage, loading, and management, and the mapping/reduction project, which is responsible for data super-data analysis, as well as the company's creation and development of a number of the non-relational databases that are used to store data, are very useful. Spark is another undergraduate sub-project on the top layer of Map-Reduce that extends the Map Reduce model.

The Hadoop can be likened to an operating system designed to handle and manage a large amount of data on different machines [23]. The best example for understanding the function of Hadoop is the difference between his software, Amway and Hadoop. He transforms a physical server into multiple virtual servers and converts a remote server into a virtual server.

Advantages of using Hadoop for business analytics include scalability, inexpensiveness, swift-paced, versatile, and no failures. Its use cases include advertisements, financial services, healthcare, gaming, and the web. Many companies all over the world use Hadoop for business analysis. Some of the largest corporations include Amazon web services, Cloudera, IBM, MapR technologies, and Microsoft [24]. The most important Big Data-related technologies are cloud computing, data centers, internet objects, and Hadoops [25].

Many national governments such as the U.S. also paid great attention to big data. In March 2012, the Obama Administration announced a 200-million-dollar investment to launch the “Big Data Research and Development Plan,” which was the second major scientific and technological development initiative after the “Information Highway” initiative [26].

Big Data Architecture

A picture of the high-level Big Data architecture is shown in Fig. 4.

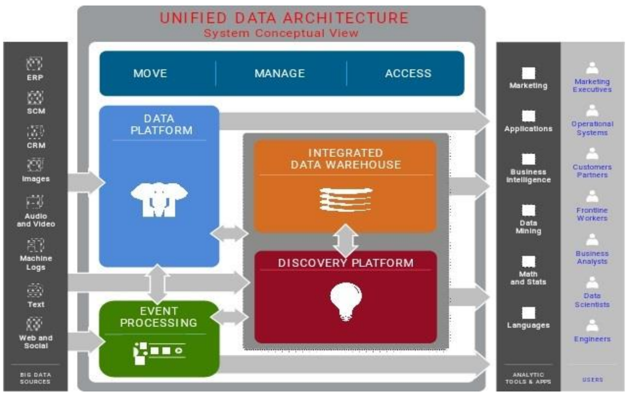

Fig. 4: High-level Big Data architecture [18].

The history of Big Data architecture projects is well represented in Table 2. In this table, the companies that worked on Business Intelligence architecture and the major activity of each company are highlighted in its right column.

Table 2: History of Projects Related to the Big Data Architecture ([27], [28])

Company (initial year)	Major Activity
Seisint (2000)	Developed a C++-based distributed file-sharing framework for data storage and query. The system stores and distributes structured, semi-structured, and unstructured data across multiple servers
Google (2003)	The genesis of Hadoop was the "Google File System" paper that was published in October 2003. This research spawned another one from Google – "Map Reduce: Simplified Data Processing on Large Clusters" in December 2004.
Yahoo (2006)	Development started on the Apache Nutch search engine project but was moved to the new Hadoop subproject in January 2006. Hadoop is born in Nutch. Hadoop 0.1.0 was released in April 2006.
Choice Point (2008)	Created parallel processing platforms
LexisNexis (2011)	Acquired Seisint Inc. in 2004. Acquired Inc. and their high-speed parallel processing platform in 2008. The two platforms were merged into HPCC (or High-Performance Computing Cluster) Systems in 2011.

Among the architectures provided for Big Data, the header has become more successful due to open source and its support by experts.

In Fig. 5, the architecture of the Big Data architecture is briefly outlined.

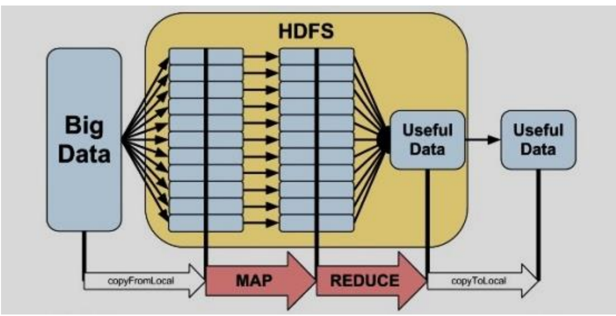

Fig. 5: General Framework for Big Data Architecture [29].

The left side of Fig. 6 depicts the mapping/reduction processes symbolically, while its right side depicts the mapping/reduction function to another type.

[1]TDWI (The Data Warehousing Institute) Research provides research and advice for business intelligence and data warehousing professionals worldwide.

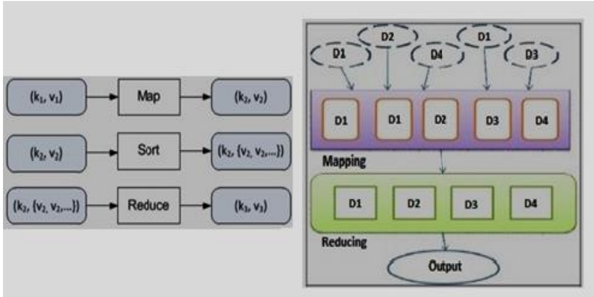

Fig. 6: Mapping and Reduction Processes [24] (Left-Side) and Mapping / Decrement Task View [28] (right-Side).

Integration of Architectures

In this part, the aim of integration, requirement of integration with examples of architectures is presented.

The Purpose of the Integration

TDWI[1] first proposed the integration of business intelligence and Big Data architectures in 2013. TDWI assumes that Hadoop usage will become mainstream in the coming years [29]. Hadoop has proved its usefulness with the toughest challenges in BI today, namely big data, advanced analytics, and multi-structured data.

The current business intelligence systems cannot support Big Data, and they have shortcomings, including the lack of performance and high costs. In a survey, the desire of companies and organizations to use the various features of the distributed file system of Hadoop was investigated. The results show that 78% of companies and organizations participating in this survey tend to use the distributed file system of Hadoop as a complement to the database management system. The data is especially useful for advanced analysis [29].

Big Data architecture often uses exploratory analysis, data mining, statistical analysis, sophisticated SQL queries, and more to analyze Big Data. The main advantage of Big Data architecture is that of intelligent business intelligence, scalability, and data diversity.

To improve the decisions process, every organization needs to use an active and integrated intelligence structure. For achieving such structure, collecting, storing for preparing data, analyzing, converting the result to useful information have a highly important role. For doing the analytical process, we should use a suitable environment that includes a warehouse, intellectual process, and a link. In the data storage, internal and external sources along with plenty of data must be stored.

This data storage can be separated based on marts and then saved in a warehouse, according to the activities of the company or organization. Recently, some big data should be added ([1], [30], [31]). An organization integrates a distributed system of header files and a database management system that can use all the data to increase business value and reduce the cost of data management. In other words, the purpose of integrating business intelligence systems and Big Data architecture is to increase the value of the business in the organization or company, to create competitive advantage, reduce costs and productivity, and also use the value of all data, not just the use of value from part of the data.

The Needs for Integration

Utilizing shared architecture can lead to more accurate decisions in the organization and can prevent future failures in the system. The integration of Big Data architecture and the BI architecture allows us to exploit more data because it creates the opportunity for more data to be analyzed and ends with better results than when it comes to Big Data architecture. The important results from the integration of Big Data and business intelligence systems are as follows [29]:

- Improving business processes and procedures as well as achieving business goals in the target organization (the target organization could be a company, industry, education, a financial system, or a global system).
- Reducing scattered data and using smaller but more valuable data.
- Enabling more accurate decisions.
- Forecasting the future to prevent system failure.

Although there are many tools for big data architecture for implementation, when we want to integrate or merge data architecture and business intelligence systems, we must select the best tools for doing so and then combine them with architecture layers of business intelligence systems. This merging can lead to improving business plans which help to meet the goals of organizations.

We can make a comparison of HDFS with DBMS.HDFS system has a distributed file system without database management, but they have several capabilities of DBMS too. These capabilities such as titling and accidental access to intelligence support of SQL language improve optimizing and searching. Of course, the performance of several capabilities of HDFS is better than DBMS capabilities such as management of a large amount of data according to file and management of unstructured data. In BI systems, we can use DBMS for administrating storing of data and big data for the distributed file system HDFS, which accompany other tools like HBase, Impala operates like DBMS.

One of the weaknesses of the database management system is the inability to store Big Data. This includes the weaknesses of the distribution system because the file header is less accurate than the database management system. Thus, the best tool for storing the Big Data of the distributed system with a header file as well as structured data is the database management system, specifically for storing data types, which are not replaced for each other. If we use the tools in the Big Data architecture as a complement to the database management system, the performance of the database management system will also be better. It is due to the database management system working hard on some parts, especially if the data is unstructured or semi-structured, but the system with the distributed header file also works well on such parts.

Obviously, if the source of database management is focused on structured data and all of the mentioned resources exempt, the performance and efficiency will be improved. Many organizations already have data that the database management system in the organization's business intelligence systems cannot process, including public relations recordings, organization XML supply documents, sensor log files, machinery, and other unstructured data that may exist in the organization. The Big Data architecture can easily store and process these data.

The integration can lead to improved business processes and improved business plans to meet the goals of the organization. Because of the cost of the source in database management, capabilities of HDFS system, and free sources present with open code and cheap, we can reduce the costs utilizing the sources of HDFS and free sources of database management. In other words, since HDFS can be used as supplementary work for the database, we can take advantage of HDFS in computing and database management. With these advantages, data in HDFS is not processed. Instead, data in database management is processed and prepared for suitable usage. HDFS system supports comparability and multi-structured database. Processing data in database management makes high precision and it can increase the organizational facilities. HDFS system can store many kinds of data, including unstructured data and structured data with ETL operations. Storing and managing structured data in database management have better performance. The management and storing unstructured data in HDFS are better and has lower costs.

HDFS with supporting tools such as Hbase, can help to store and manage the data efficiently. In order to the monitoring of intelligence and reporting, we can apply usable linkages in business intelligence systems or the other designed tools in big database architecture. The differences between visualizing information or intelligence in big data and business intelligence systems are that tools directly have relations with data, but in business intelligence systems is not so. With the integration of HDFS and database management, we can create new and joint opportunities and it leads to increase the capability of business intelligence systems, such as rating the data, Archive data source, Management unstructured data, Management of file-based, Increasing the power of processors, Management, store of data.

Integration of Big Data Architecture with other Architectures

In recent years, cloud computing has been integrated with other architectures such as BI architecture, DSS architecture, and SCM architecture. In 2013, the integrated Chan architecture was presented. Chan's integrated architecture is due to the integration of business intelligence architecture and Big Data architecture (See Fig. 6). In 2014, Samson and colleagues integrated the Big Data architecture with the BI architecture[32]. In Fig. 6, both architectures are presented without evaluation and have structural bugs. Table 3 depicts how Chan architecture and Samson architecture are compared in terms of their strengths and weaknesses.

Table 3: A comparison of architectures derived from the integration of business intelligence architecture and Big Data architecture, to provide a conceptual model

Researcher (year)	Strengths	Weaknesses
Chan (2013) [33]	- Suitable for Real-Time Systems	Failure to Structural, Lack of evaluation
Samson et al. (2014) [32]	- Scalability -Parallel processing of data warehouse with HDFS - Analysis beyond the map / reduce	Failure to Structural, Lack of evaluation
This research (2022)	-Presentation of a conceptual model at different levels and in different styles - Suitable for real-time systems and other systems - High scalability - Parallel processing of data warehouse and big data warehouse - Provide analyzes beyond Map / Reduce - Without structural forms - Architectural valuation of different methods	Evaluation is time-consuming

Comparing Architectures

We compared the architecture of Chan and Samson architecture with the proposed architecture. Table 3 shows the results of this comparison in terms of strengths and weaknesses.

Building a new architecture uses a systematic approach, that is, the architecture has an input and output. The first layer can be considered for the data source, the second layer for data storage and management, the third layer for the data analysis and intelligence, and the fourth layer for visualization tools and applications. It can be said that the first and second layers are the infrastructure layer, the third layer is the computational layer and the fourth layer is the application layer. Fig. 8 shows the proposed architecture. The performance of each layer is based on the name chosen for it.

In this architecture, each layer has several sub-layers, and each substrate can be included components. The multiplicity of components of each sub layer allows for the creation of multiple and different models based on the proposed architecture. In each model, one or more components of the sub-layer were used. Selecting one or more components of each sub-layer component is based on the need specified in the model.

A kind of architecture is proposed by Chan that applies to the business intelligence system [33]. The typical characteristics of this architecture are how to use them for real-time systems. The most problem of Chan architecture is the lack of assessment so there are some difficulties in the structure. One of these difficulties is in using tools or analyzing way (Map/Reduce) that portioned into two parts, one in the analytical sections and the other beside the HDFS. It is not structurally correct that use two parts with a name (Map / Reduce) with a fixed concept. Instead, we could name it in one place and refer it elsewhere. Samson architecture focuses on storing data through a system file with big database architecture. So it leads to facilitate comparability, whereas the traditional ways have not this capability. The most important capability mentioned for this architecture is a parallel processing warehouse with a big database so that it can provide more analyzes than those in the Map/Reduce of the Hadoop. It is made through direct accessing to data along with indirect access to data that is expected in this architecture. Samson and Chan have not presented any valuation on their architectures. From the points of view in structure or form of storing source, analyzing data, visualizing intelligence clearly has done. Big data architecture integrates with other ones in the article [34]. Moreover, big data architecture with a support system of decisions can be integrated. In this architecture, by identifying steps of the Simon model, integration of big database architecture is completed with the support of the system of design. Integration of the architecture of big databases with the architecture of the supply chain is done and is presented in the article [35].

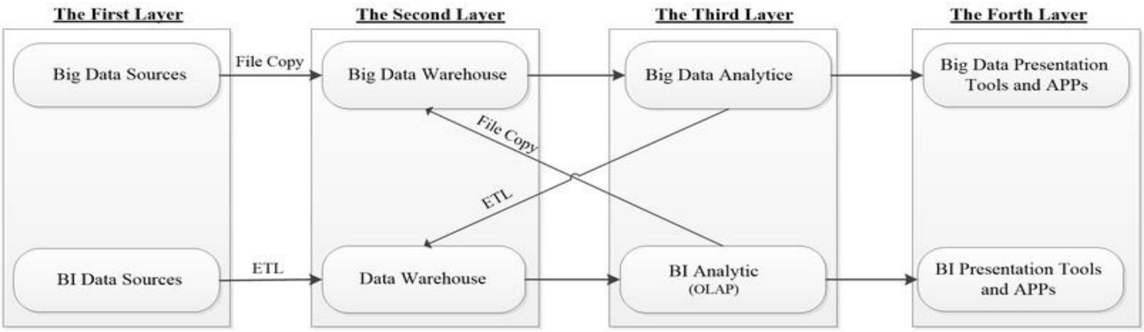

Fig. 7: The overall four-layered architecture of the proposed architecture with the system approach.

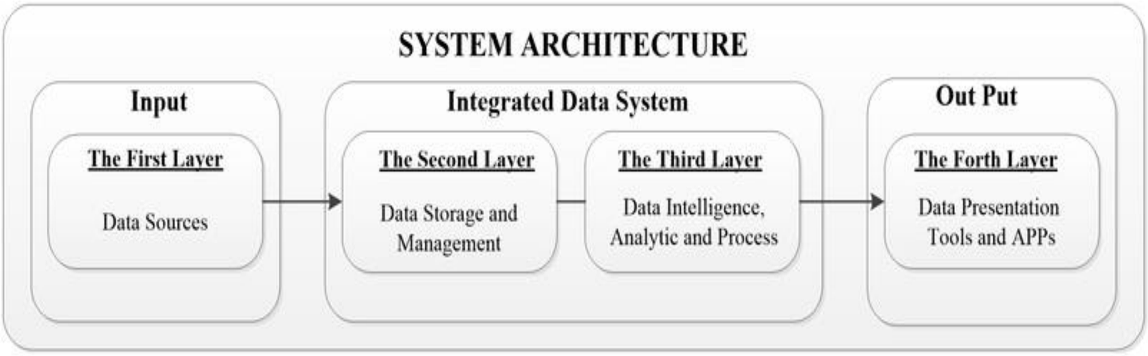

Fig. 8: A new four-layer architecture, derived from the integration of Big Data architecture and business intelligence architecture.

Proposed Architecture

The proposed architecture is a four-layered architecture that integrates Big Data and business intelligence architectures. This integration is illustrated in Fig. 7 of the template.

The first layer is a data source layer that has two substrates: The Big Data Sources sub-layer and the BI Sources sub-layer. Each of these sub-layers can be included in many data sources. The Big Data Sources sub-layer can consist of raw data streams, data flow, unstructured data, semi-structured data, multi-structured data, streaming events, images and videos, audio files, social media, text files, XML documents, webpage data, blogs, emails, Docs and PDFs, log server, marketing events, sensor data, GPS data, scientific research, machine logs, graph files, NOSQL files (Base, Mongo DB, Couch DB and sec). The BI Sources sub-layer contains structured data, operational systems data, RDBMS files (SQLServer, Oracle, Microsoft Access, MySQL and sec), software's structured data (CRM, SCM, ERP, and sec).

The second layer is the data storage and management layer**,** which can be considered as an infrastructure layer. This layer consists of two sub-layers: one is the Big Data warehouse sub-layer and the other is the Data warehouse sub-layer. Each of these substrates can include the components shown below. The Big Data Warehouse sub-layer includes HDFS, NOSQL (HBase, Mongo DB, and sec), Hive, Impala, Kudu, and RDD. The Data Warehouse sub-layer includes Data Marts, RDBMS, MPP (Massively Parallel Processing). It should be noted that the Big Data warehouse is a new concept similar to the concept of storage, with the difference that the storage warehouse is a reservoir of data in which the data are simply copied and there are no categories for the data. In some scientific sources, such a concept has been named "data lake" [36], which is referred to here as the storage bin.

The third layer is the Data Intelligence, Analytics, and Process Layer, which can be considered as the computation layer. This layer contains two sub-layers, one underlying Big Data Analytics sub-layer, and another BI Analytics sub-layer. Each of these substrates can include the following components: The Big Data Analytics sub-layer includes MAP Reduce, Real-Time Analytics, Pig, and Spark. The BI Analytics sub-layer includes OLAP.

The fourth layer is the Data Presentation Tools and Applications Layer. This layer also has two sub-layers, one underlying the Big Data Presentation Tools and Applications sub-layer, and the other underneath the BI Presentation Tools and Applications sub-layer. Each of these sub-layers has the components listed below:

The Big Data Presentation Tools and Applications sub-layer includes On-line APPs, Big Data tools (visualization, reporting, predictive, data mining, queries, machine learning), real-time APPs, and near real-time APPs. The BI Presentation Tools and Applications sub-layer includes dashboards, off-line APPs, developer environments, BI tools (visualization, reporting, predictive, data mining, queries, machine learning), custom APPs, and enterprise APPs.

Proposed Architectural Style

The proposed architectural style of different views is similar to some architectural styles but does not follow a particular style. It is the combination of several different styles and can be considered as an independent style.

The proposed architecture can be compared to data stream styles in similarity. One of a variety of data stream styles is pipe styles and filters. In the proposed architecture, the BI Analytic and Data Warehouse sections act as filters, because data is entered there, and after processing and modification, it becomes a series of information and then exits. Data changes in BI Analytic can be done using OLAP, and in the Data Warehouse, this change is done by the ETL process. On the other hand, the connection between these two parts can be considered as a pipe, which is responsible for transmitting data and information from one part to another.

In the similarity of proposed architecture with data-driven styles, one of a variety of data-driven styles is the repository style. The proposed architecture in this regard is similar to the architecture in the Data Warehouse and Big Data Warehouse sections of the data container.

As a Big Data repository, Big Data Warehouse shares its data between BI and Big Data Analytics. One of the components of data-driven styles is the processing elements that are potentially independent of each other. In the proposed architecture, BI Analytics and Big Data Analytics are process elements that are potentially independent and interconnected with the Big Data Warehouse database.

The proposed architecture can be compared to the layered style. One of a variety of styles based on overlays is a layered style. In layered style, the layers are hierarchically placed together and the lower layer provides a higher level of service, in other words, the upper layer of the client and the lower layer of the server. To display layered layout architecture, layers with headings (infrastructure layer, computing layer, and application layer) can be utilized. The proposed architecture in the form of a layered style can be shown in Fig. 9. As we can see, the first layer is an infrastructure layer that includes data sources and data management and storage. The second layer is the computation layer, which includes intelligence, data analysis, and data processing. In the third layer, the application layer includes visualization tools and application software.

The proposed architecture can also be compared with the service-oriented architectures. The proposed architecture is process-oriented, but it can be developed as a service-oriented architecture. In this case, the various components of the architecture will be linked together with a loose connection. This kind of connection will lead to the creation of a vibrant and dynamic system that can be considered as a competitive advantage for real-world businesses in today's marketplace.

In the proposed architecture, if data sources are in a cloud environment, the Data Sources layer, as well as the Data Storage and Management layers, can be provided as a service, called Infrastructure as a Service.

If we imagine the proposed four-layer architectures in the style of service-oriented architectures, it can be considered as three-layer architecture according to Fig. 10. As we can see, the infrastructure service includes data storage and management.

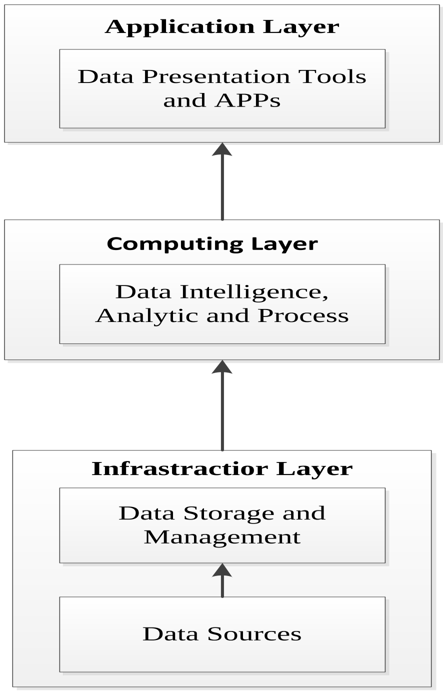

Fig. 9: Layered style architectural design.

In this figure, the computing service provides intelligence, analysis, and data processing. The application service provides visualization tools and applications. These three layers are explained below:

- The Infrastructure as a Service (Iaas) includes Big Data Warehouse as a Service and Data Warehouse as a Service. It extracts data from data sources that are commonly found in the cloud. It also has the task of storing and managing extracted data.

- The Computing as a Service (CaaS) includes Big Data Analytics as a Service and BI Analytics as a Service. It does the task of computing, analyzing data, intelligence, and processing data.
- The Application as a Service (AaaS) includes Big Data Presentation Tools and Applications as a Service and BI Presentation Tools and Applications as a Service. It does the task of visualizing and displaying information.

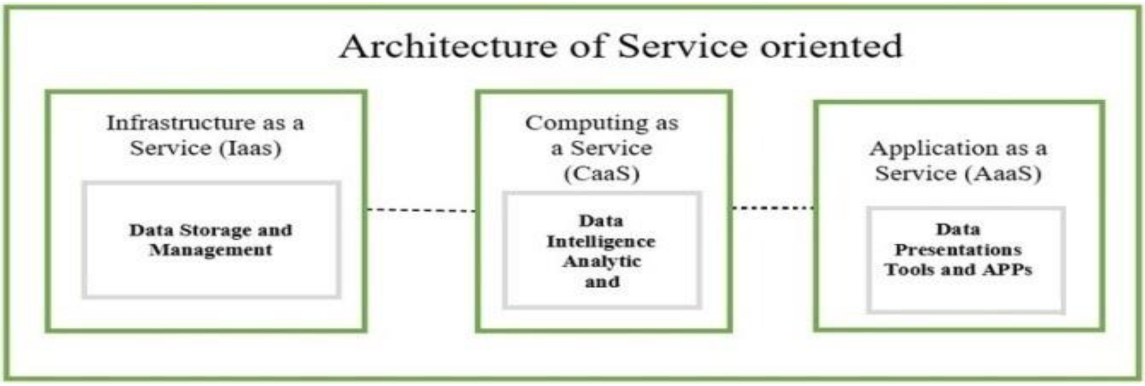

Fig. 10: Service-oriented architectural style architecture.

In the proposed architecture, the sub-layer subsystem, various components, and tools are named, all of those components and tools are developed in a service-oriented architecture, can be provided as a service. For example, in the Big Data Presentation Tools and APPS sub-layer subdirectory, the Data Mining tool is provided, this tool can be provided in the service architecture offered as Data Mining as a Service.

Evaluation with a Case Study

Before the implementation of the architecture, the colored Petri-Nets and the tools written for it could be used to evaluate the architecture and convert the actual architecture into a formal architecture, as well as provide an executable version of the architecture. Although this method has so far been less widely used in evaluating information systems, due to their high capabilities, it can be a reliable method for evaluating information systems and their architecture.

In this paper, we will focus on the proposed architecture. First, we will examine its strengths and weaknesses. Afterward, using an architecture simulation with the UML, it is then evaluated in a case study by Petri Networks.

Evaluation by Comparison of Weaknesses and Strengths

From different perspectives, one can study the weaknesses and strengths of an information system from an organizational, managerial, and Technology (technical) standpoint. Here, the strengths and weaknesses of Business intelligence, Big Data, and proposed architectures will be examined from a technological perspective.

The strengths and weaknesses of BI from the perspective of technology are as follows:

- Strengths of BI architecture include: improving decision making [37], standards support [38], generalization and adaptation [4], personalization [38], reduced costs [39], faster reporting, and more accuracy [37].
- Weaknesses of the BI Architecture include failure to support Big Data, failure to provide advanced analysis [41], lack of support for multi-structured data [29], and strong support for semi-structured and unstructured data [22].

The strengths and weaknesses of Big Data from the perspective of technology are as follow:

- Strengths of Big Data architecture include: analysis of Big Data ([29], [40]), high scalability ([24], [29]) , support for exploratory analysis [29] reducing miscellaneous data to reduce data volumes [40], providing more detailed for decisions-making by analyzing the Data, prevention of future system failure [40], acts as a good complement to the data warehouse, support for multicast data and low-cost hardware [29], low-cost Software ([21], [29]) and the storing and processing of data types (structured, semi-structured, and unstructured) ([24], [29]).
- Weaknesses of Big Data architecture include Lack of full SQL support, Low ability to query and access information in Real-Time, Evolving management tools ([29], [42]).

By integrating business intelligence and Big Data architectures into the proposed architecture, all of the weaknesses of the BI architecture are covered by the strengths of Big Data architecture, as well as the weaknesses of Big Data architecture being covered by the strengths of the business intelligence architecture. Thus, the outcomes from the integration of both business intelligence and Big Data architectures are revealed.

The strengths and weaknesses of the Proposed Architecture are as follows:

- Strengths of the proposed architecture include strengths of the Big Data architecture and strengths of the business intelligence architecture.
- Weaknesses of the proposed architecture include related to a possible problem in implementation, which can be measured after evaluation and experience collected.

It can be concluded that these two architectures can be a good complement to each other. Thus, if these two architectures can be integrated together, they will remain protected and the interaction of the two architectures would not give rise to new flaws.

In this case, with regard to the strengths of the proposed architecture, we can deduce that the strengths of the proposed architecture are almost equal to the sum of the strengths of business intelligence and Big Data architectures.

Concerning the weaknesses of the proposed architecture, we can also deduce that the proposed architectural weaknesses are equal in addressing the weaknesses of business intelligence architecture and the weaknesses of Big Data architecture. Because the weaknesses of one are the other strengths, the resulting weaknesses of the proposed architecture will be zero and there will be no weaknesses for the proposed architecture.

A weak point is that it cannot be resolved by integration, including the lack of implement the proposed architecture or the weakness created by the integration.

Evaluation by Simulation with Unified Modeling Language

A similar approach is integrated into the language of modeling.

Although this language is not able to provide an executable model of architecture with various charts, it can provide a quasi-official model of software architecture. Here, the system is supposed to be based on the architecture of the proposed system, and then work like this system will be implemented using the Unified modeling language. The schema of the simulated system is shown in Fig. 11.

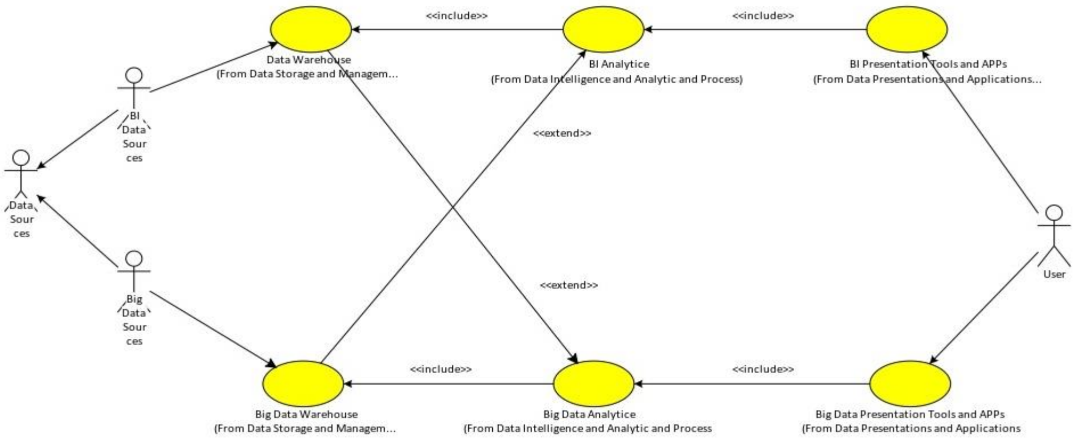

Fig. 11: The proposed architectural diagram of the proposed architecture.

Now, with the help of the use case diagrams, components, and sequences, we can the colored Petri-Net drew the architecture-based system and perform a simulation. In the simulation, we use the Colored Petri-Net in which a case study is required.

The Case Study

To evaluate the proposed architecture, we use a case study. The main goal, here, is to compare the proposed architecture with the traditional architecture used for business intelligence. With the traditional architecture of business intelligence in terms of quality performance attributes, performance is considered to be the most important quality attribute of software architecture.

The performance measurement means measuring the time needed to perform the processes of a software system. The less time this system uses, the more efficient the system. Comparing two architectures, the most important component of the analogy that shows the superiority of one architecture to another architecture is the quality attribute of efficiency. Of course, in a complete comparison of the two architectures, all the quality attributes and requirements for both architectures must be compared to fully demonstrate the superiority of the architecture to the other architecture. But one of the most important of these attributes is the efficiency attribute; the other attributes and requirements are either not measurable at this stage or their significance is less effective than the quality attribute.

In this scenario-based case study, the traditional architecture of business intelligence with the proposed architecture is compared. First, the colored Petri-net of the simulated system is drawn that based on the business intelligence architecture. Also, the colored Petri-Net of the simulated system is drawn based on the proposed architecture. Thus, hundreds of simulated simulations are executed using hundreds of simulated petty networks, and then the average runtimes of the two systems are compared to determine which system is more efficient.

Fig. 12 shows the Petri-Nets of the business intelligence system based on the typical BI architecture after implementation.

The time values shown in the Finish section are the sum of the times from the beginning of the data entered into the system to the display of the final report to the user.

Fig. 13 shows the Petri-Net system based on the proposed architecture after implementation. The time values shown in the Finish section are the sum of the times from the beginning of the data entry to the end of the final report to the user.

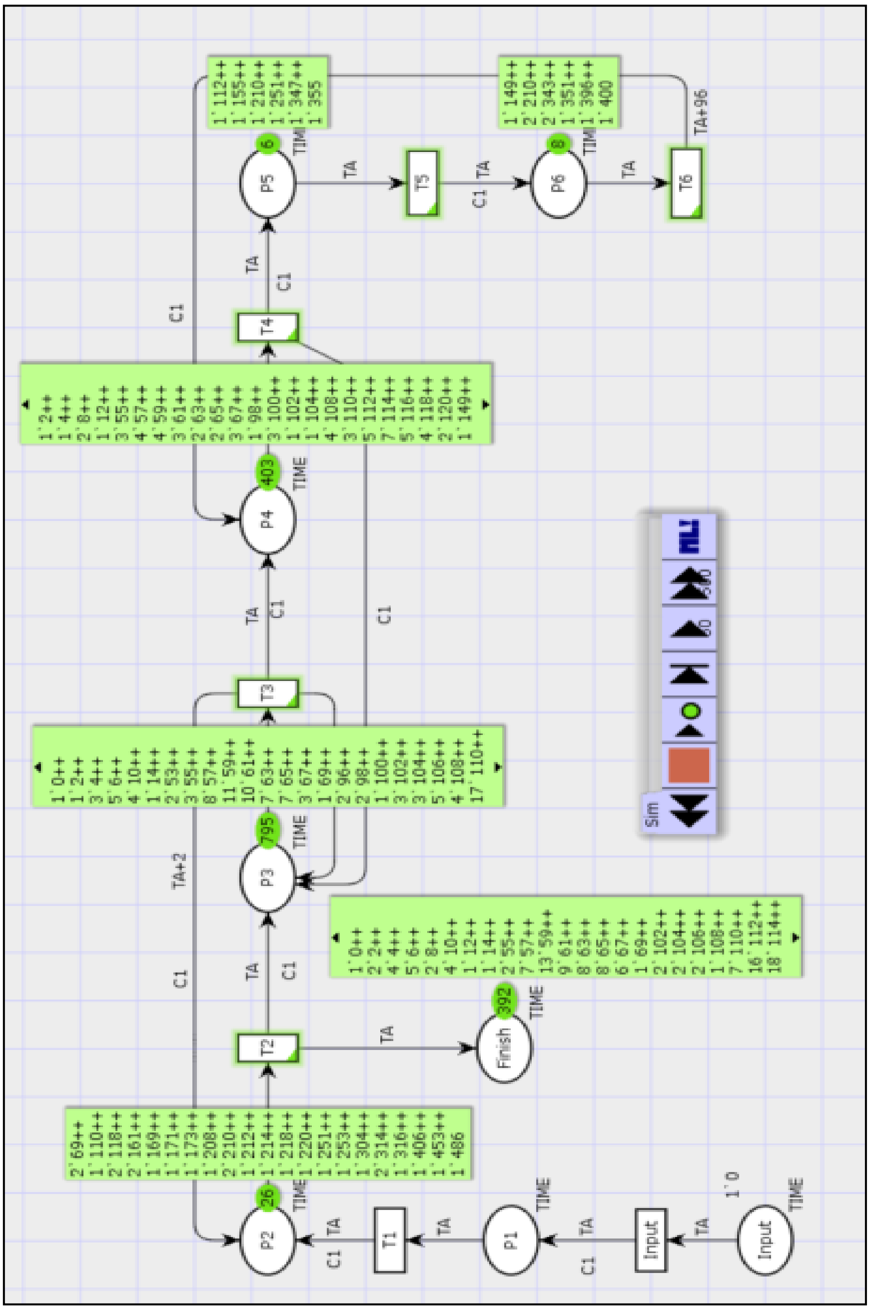

Fig. 12: Pure Color Networks of the BI System based on the architecture of the post-implementation.

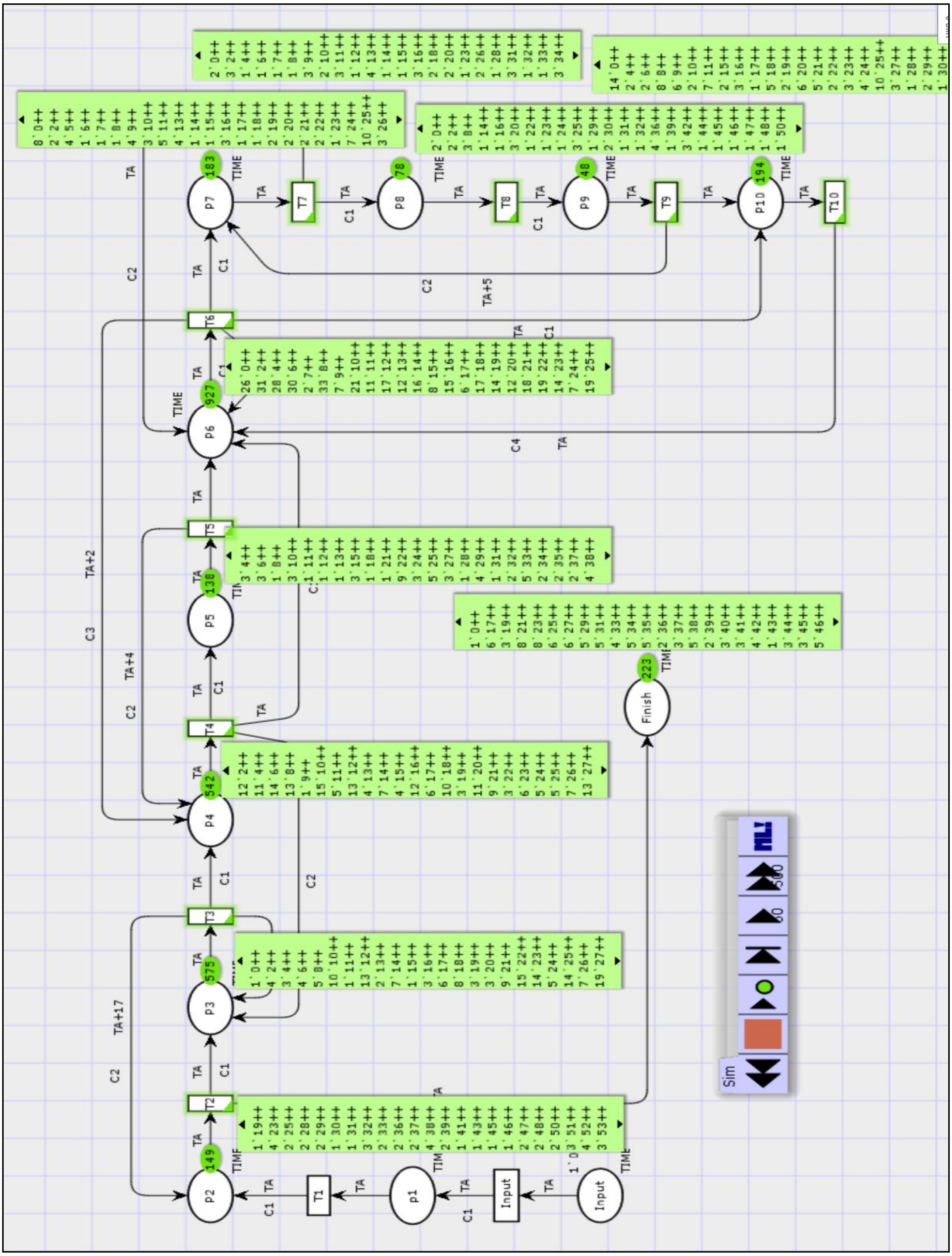

Fig. 13: Colored Petri-Net dish the proposed architecture-based architecture after implementation.

Table 4 illustrates the results obtained from the implementation of the color scheme of the Pure Color Network of the Business Intelligence Based Architecture and the proposed architecture. To compare the performance, usually, the metric of the count steps and time are used ([43], [44]). As we can see, the average runtime scenario in the business intelligence system is based on the common architecture of 378 and the average runtime scenario in the proposed architecture is 53. Also, other values, including the maximum time and total time of the proposed architecture are lower than that ofBI Architecture. Because of these lower values, the proposed system has more performance compared with the system-based system, based on the common architecture of business intelligence.

Table 4: Comparison of the results from the implementation of the Colored Petri-Nets (Time in Second)

Name	BI Architecture	Proposed rchitecture
All Count Steps	2000	1380
Count Finish Steps	268	268
Max Time	50845	138
Min Time	0	0
Total Times	101690	14331
Average Times	378.03	53.47

It is concluded that the system based on the proposed architecture is more efficient and can produce more valuable output data at a lower cost.

Results and Discussion

In this paper, we integrated the business intelligence architecture and Big Data architecture to address the problems of business intelligence systems in dealing with Big Data, resulting in a new architecture. By comparing the proposed architecture and the BI-architecture, the weaknesses of the business intelligence architecture were identified. Moreover, these weaknesses were resolved by integrating with the Big Data architecture. Also, in this paper, the evaluation of the business intelligence architecture as one of the complex architectures of information systems was carried out using the Colored Petri-Nets method. This proposed architecture allows companies and obtain more value from their data sources and receives stronger support from corporate executives and organizations in making executive decisions.

The most important result that can be drawn from this research is that we have an integrated the perspective. We promoted business intelligence systems and Big Data technologies to help managers create new opportunities in solving specific problems. This integration creates new opportunities for solving the problems of business intelligence and Big Data systems, which will greatly help to the managers and stakeholders of these systems.

Since the proposed architecture may have some side effects on the efficiency resources required, further research must be done. In addition, it is suggested that this proposed architecture be used in designing complex intelligence systems such as Business Intelligence Systems Decision support, resource management systems, supply chain management systems, and customer relationship management systems.

Author Contributions

This paper is the result of M. R. Behbahani Nejad M.Sc. project which is supervised by Mohammad Jafar Tarokh and advised by H. Rashidi and Participated by Bahman Nouriani. M. R. Behbahani Nejad proposed the main idea of the innovation and Integration of architectures, performed the simulations, carried out the data analysis, interpreted the results and wrote the first manuscript. H. Rashidi corrected the proofing the article and wrote the final version of the article. B. Nouriani Suggested petri nets for architectural evaluation.

Acknowledgment

This work is completely self-supporting, thereby no any financial agency's role is available.

Conflict of Interest

The authors declare no potential conflict of interest regarding the publication of this work. In addition, the ethical issues including plagiarism, informed consent, misconduct, data fabrication and/or falsification, double publication and/or submission, and redundancy have been completely witnessed by the authors.

Abbreviations

AaaS	Application as a Service
BI	Business Intelligence
BPM	Business Performance Management
CaaS	Computing as a Service
CPN	Colored Petri Net
CRC	Class, Responsibilities, Collaborators
Iaas	Infrastructure as a Service
NOSQL	Not Only SQL
OMG	Object Management Group
SA	Software Architecture
SOA	Service-based architecture
UML	Unified Modeling Language

References

[1] K. Batko, A. Ślęzak, “The use of big data analytics in healthcare,” J. Big Data, 9(3): 1-24, 2022.

[2] A. Heydarian, H. Rashidi, "Storage, and access system of self-adapting Health big data based on smart IoT (in Persian)," presented at the Fourth International Conference on Electrical, Computer and Mechanical Engineering, 2021.

[3] C. Dobre, F. Xhafa, "Intelligent services for big data science," Future. Gener. Comput. Syst. 37: 267–281, 2014.

[4] G. Muhammad, J. Ibrahim, Z. Bhatti, A. Waqas, "Business intelligence as a knowledge management tool in providing financial consultancy services," Am. J. Inf. Syst., 2(2): 26-32, 2014.

[5] "IEEE recommended practice for architectural description for software-intensive systems," IEEE Std 1471-2000: 1-30, 9 Oct. 2000.

[6] W. Pedryz, Architecture in Big Data, Information Granularity, Big Data, and Computational Intelligence, Studies in Big Data, 8: 275-295, 2015.

[7] L. Bass, P. Clements, R. Kazman, Software Architecture in Practice, 4th Edition, SEI Series in Software Architecture, Addison-Wesley Professional, 2021.

[8] L. Clements, B. Paul, Rick Kazman, Software Architecture in Practice, 2nd Edition, SEI Series in Software Architecture, Addison-Wesley Professional, 2003.

[9] R. N. Taylor, N. Medvidović, E. M. Dashofy, Software architecture: Foundations, Theory and Practice. Wiley, 2009.

[10] M. M. Patil. Challenges and Issues in Handling Big Data, Int. J. Innovations Adv. Comput. Sci. (IJIACS), 4(Special Issue): 620 – 625, 2015.

[11] D. S. Linthicum, Service-Oriented Architecture (SOA)", Retrieved 09-21, 2016.

[12] P. Clements, K. Klein, Evaluating Software Architecture Methods and Case Studies, Addison Wesely, 2002.

[13] B. Kumar, J. Jaspernete, "UML profiles for modeling real-time communication protocols," J. Object Tech., 9(2): 178-198, 2012.

[14] S. Balsamo, M. Marzolla, "A simulation-based approach to software performance modeling, ACM SIGSOFT software engineering notes," in Proc. the 9th European Software Engineering Conference Held Jointly with 11th ACM SIGSOFT International Symposium on Foundations of Software Engineering, 28(5): 363-366, 2003.

[15] J. Dong, "UML extensions for design pattern compositions," J. O. Tech. 1(5): 151-163, 2002.

[16] F. Liangbing, M. Obayashi, T. Kuremoto, K. Kobayashi Construction and application of learning Petri net, Manufacturing and Computer Science, 143-176, 2012.

[17] M. Muntean, Collaborative Business Environment Based on Federated Portals, Annals of T. Popoviciu Seminar, 4: 218-224, 2006.

[18] R. Sharda, D. Dursun, T. Efraim, Analytics, Data Science, & Artificial Intelligence: Systems for Decision Support, 11th edition, Pearson, Prentice-Hall, 2020.

[19] T. Erl, W. Khattak, P. Buhler, Big Data Fundamentals: Concepts, Drivers & Techniques, Pearson Service Technology Series from Thomas Erl, 1st Edition, 2017.

[20] D. Laney, "Application Delivery Strategies", Gartner (META Group), White Paper, 2001.

[21] A. N. Nandakumar, N. Yambem, "A survey on data mining algorithms on apache hadoop platform," Int. J. Emerging Technol. Adv. Eng., 4(1): 263-265, 2014.

[22] M. R. BehbahaniNejad, H. Rashidi, M. J. Tarokh, "Survey big data technologies architecture and the need to exploit it in DSS systems," presented at the 8th Iranian & 2th International Knowledge Management Conference, Tehran, 23-24 February 2016.

[23] S. R. Pakize, "A comprehensive view of Hadoop MapReduce Scheduling algorithms," Int. J. Comput. Networks Commun. Secur., 2(9): 308–317, 2014.

[24] K. Kashyap, C. Deka, S. Rakshit, "A review on big data, Hadoop and Its impact on business," Int. J. Innovative Res. Dev., 3(12): 78-82, 2014.

[25] A. Labrinidis, HV. Jagadish, Challenges, and opportunities with big data. Proc VLDB Endowment, 5(12): 2032–2033, 2012.

[26] M. Chen, S. Mao, Y. Liu, "Big data: A survey," Mob. Netw. Appl. 19: 171–209, 2014.

[27] A. F. Mohammad, H. Mcheick, E. S. Grant, "Big data architecture evolution," The fourth ACM International Symposium, 2014.

[28] Wikipedia. "Apache Hadoop" .Retrieve from https://en.wikipedia.org/wiki/Apache_Hadoop , 2015.

[29] Lockwood, Conceptual Overview of Map-Reduce and Hadoop, 2015.

[30] M. R. Behbahani Nejad, H. Rashidi, "A data architecture architecture with business intelligence approach," presented at the first Seminar on Data Science and applications: 14-15, Tehran, 2021.

[31] H. Rashidi, "Corporate planning using object-oriented rules and considering big data (in Persian)," Allameh Tabataba'i Press, 2018.

[32] S. Oluwaseun Fadiy, S. Saydamb, V. Vanyduhe, "Advancing big data for humanitarian needs," Procedia Eng., 78: 88-95, 2014.

[33] F. Chang, J. Dean, S. Ghemawat, W. C. Hsieh, D. A. Wallach, M. Burrows, "Bigtable: A distributed storage system for structured data," J. ACM Trans. Comput. Syst. (TOCS), 26(2): 1-14, 2008.

[34] T. Poleto, V. DioghoHeuer de Carvalho, A. P. Cabral Seixas Costa, "the roles of big data in the decision-support process: An empirical investigation," Decision Support Systems V – Big Data Analytics for Decision Making. ICDSST 2015. Lecture Notes in Business Information Processing, 216: 10–21, 2015.

[35] B. Sanjib, S. Jaydip, "A proposed framework of next generation supply chain management using big data analytics," presented at the National Conference on Emerging Trends in Business and Management: Issues and Challenges, , Kolkata, INDIA, March 17-18, 2016.

[36] R. Rossi, K. Hirama, "Characterizing big data management," Issues Informing Scie. Inform. Technol. 12: 165-180, 2015.

[37] H. Borut, J. Jaklič, "Assessing benefits of business intelligence systems–a case study," Manage. J. Contemp. Manage. Issues, 15(1): 87-119, 2010.

[38] S. Eybers S, J. H Strydom, Towards a classification framework of business intelligence value research. Italian Chapter of AIS (itAIS 2013), 2013.

[39] A. Carver, M. Ritacco, The Business Value of Business Intelligence. A Framework for Measuring the Benefits of Business Intelligence. Business Objects, 2006.

[40] M. Bahrami, M. Singhal, "The role of cloud computing rchitectour in big data, Chapter: Information Granularity, Big Data, and Computational Intelligence, Volume 8 of the series Studies in Big Data: 275-295, 2015.

[41] P. Russom, Integrating Hadoop into business intelligence and data warehouse, TDWI research. S. Madden, MIT, 2013.

[42] A. Abroshan, A. Harounabadi, J. Mirabedini, "Evaluation of software architecture using fuzzy colored Petri nets," Manage. Scie. Lett., 3: 665-682, 2013.

[43] K. Dwivedi, S. K. Dubey, "Analytical review on Hadoop distributed file system", presented at the 5th International Conference on Confluence the Next Generation Information Technology Summit: 25-26, Noida, India, 2014.

[44] J. Wang, X. He, Y. Deng, "Introducing software architecture specification and analyzing in SAM through an Example," Inform. Software Technol. J., 41(7): 451-467, 1999.

Biographies

Mohammad Reza Behbahani Nejad received the B.Sc. degree in Software Engineering from the Urmia University, Urmia, Iran and the M.Sc. degree in Software Engineering from the Islamic Azad University, Qazvin, Iran. He works on Business Intelligence and Big Data.

- Email: reza2005nejad@gmail.com
- ORCID: 0000-0001-6660-0699
- Web of Science Researcher ID: ADM-1867-2022
- Scopus Author ID: NA
- Homepage: NA

Hassan Rashidi is a Professor in Department of Mathematics and Computer Science of Allameh Tabataba'i University. He received the B.Sc. degree in Computer Engineering and M.Sc. degree in Systems Engineering and Planning, both from the Isfahan University of Technology, Iran. He obtained Ph.D. from Computer Science and Electronic System Engineering department of University of Essex, UK. His research interests include software engineering, software testing, and scheduling algorithms. He has published many research papers in International conferences and Journals.

- Email: Hrashi@gmail.com
- ORCID: 0000-0002-6588-5378
- Web of Science Researcher ID: AAE-2124-2022
- Scopus Author ID: NA
- Homepage: NA

How to cite this paper:
R. Behbahani Nejad, H. Rashidi, "A novel architecture based on business intelligence approach to exploit big data," J. Electr. Comput. Eng. Innovations, 11(1): 85-102, 2023.

DOI: 10.22061/JECEI.2022.8565.529

URL: https://jecei.sru.ac.ir/article_1727.html