A Comprehensive Approach for Designing Business-Intelligence Solutions with Multi-agent Systems in Distributed Environments

Karima Qayumi[1(✉)] and Alex Norta[2(✉)]

[1] School of Digital Technologies, Tallinn University, Narva Mnt 29, 10120 Tallinn, Estonia
karima.qayumi@gmail.com
[2] Large-Scale-Systems Group, Tallinn University of Technology, Akadeemia Tee 15a, 12616 Tallinn, Estonia
alex.norta.phd@ieee.org

Abstract. Multi-agent systems (MAS) are an active research area of system engineering to deal with the complexity of distributed systems. Due to the complexity of business-intelligence (BI) generation in a distributed environment, the adaptation of such system is diverse due to integrated MAS and distributed data mining (DDM) technologies. Bringing these two frameworks together in the content of BI-systems poses challenges during the analysis, design, and test in the development life-cycle. The development processes of such complex systems demand a comprehensive methodology to systematically guide and support developers through the various stages of BI-system life-cycles. In the context of agent-based system engineering, several agent-oriented methodologies exist. Deploying the most suitable methodology is another challenge for developers. In this paper, we develop an exemplar of MAS-based BI-system called BI-MAS with comprehensive designing steps as a running case. For demonstrating the new approach, first we consider an evaluation process to find suitable agent-oriented methodologies. Second, we apply the selected methodologies in analyzing and designing concepts for BI-MAS life-cycles. Finally, we demonstrate a new approach of verification and validation processes for BI-MAS life-cycles.

Keywords: Business-intelligence (BI) · Distributed data mining (DDM)
Multi-agent system (MAS) · Agent-oriented modeling (AOM)

1 Introduction

Business-intelligence (BI) is a modern management support that includes users, distributed data mining (DDM) processes, intelligence tools, information management, and analysis processes in order to improve decision-making and business performance [1–3]. Agent technologies, or multi-agent systems (MAS) [1, 4, 5] are promoted as an emerging technology that facilitates the design, implementation, and maintenance of distributed systems. DDM [6, 7] originates from the need for mining intelligence over decentralized data sources. Furthermore, DDM is known as one of the latest solutions,

A. Hameurlain and R. Wagner (Eds.): TLDKS XXXVII, LNCS 10940, pp. 113–150, 2018.
https://doi.org/10.1007/978-3-662-57932-9_4

or procedures to reduce massive data-discovery problems in highly distributed environments [1].

Researchers apply MAS together with DDM for reducing the complexities of BI-systems in distributed environments [1, 9, 10]. Additionally, literature also comprises several types of agent-based architectures and frameworks either in the content of DDM, or BI-systems in [11–16]. Our studies discover that there exists a lack in deployment of agent-oriented methodology in these mentioned architectures and frameworks. Conversely, literature presents numerous types of agent-oriented methodologies in [17–20, 29–35].

In the context of software engineering [2], each model of the designing phase must describe a specific aspect of the system under consideration. In fact, the designing processes of agent-based BI-systems require either an applicable agent-oriented methodology, or a significant approach to capture requirement specifications and translate them for the development process. The development process of BI-MAS that comprise different types of agents in a distributed environment with the ability to communicate, discover and access data from multiple sites, requires a comprehensive methodology [3]. The main challenges for developing processes of BI-MAS are the assigning of agents to perform tasks in parallel and the management of collaborations and cooperation processes in complex applications [11, 21]. In such complex systems, developers need a unified agent-oriented methodology for the entire life-cycle of agent-based BI-systems.

In this paper, we address the current gap in the state-of-the-art for developing a process of BI-MAS by answering the research question of how to develop a designing approach for MAS-based BI-systems in distributed environments? To establish a separation of concerns, we elicit the following sub-questions.

RQ1. What evaluation is required to find applicable methods out of existing agent-oriented methodologies?
RQ2. What level of support do existing agent-oriented methodologies yield for developers to define a systematic way for the conceptualization of BI-MAS models?
RQ3. What types of methods and tools are demanded to consider the verification and validation (V&V) processes for proposed BI-MAS models?

The rest of the paper is structured as follows: Sect. 2 presents the related concepts of BI-MAS, challenges of traditional BI, and new BI-MAS features in business environments. Section 3 discusses existing agent-oriented methodologies and their evaluation results. Section 4 outlines the analysis and detailed design processes of the BI-MAS architecture by deploying combined agent-oriented methodologies. Section 5 comprises the mapping processes for BI-MAS models to a formalization using Colored Petri Nets (CPN) and the results from model checking that are explored from validating and the verification processes of the BI-MAS life-cycle. Section 6 describes the evaluation results together with a critical comparative discussion that compares the results of this paper against results from other research work. Finally, in Sect. 7 we conclude our paper with a summary of our research findings and suggestions for the future development of our research.

2 Related Concepts and Challenges

In last decades, research shows that developing BI-systems changes considerably due to different views of business owners, presentation of business-concepts, and the idea behind access to business data. To resolve the current challenges of BI-systems, research studies the combination of MAS and DDM technologies as a new-generation for such systems. In this section, we briefly discuss main components, current challenges of BI, and concept of agent-based BI-systems.

2.1 Business-Intelligence Systems

BI is a term that refers to technologies, applications, and practices for the collection, integration, analysis, and presentation of business information [22, 23]. BI-systems are a set of applications, technologies and tools for the transformation of raw data into meaningful and useful information in order to improve decisions and increase business performance [1, 24]. Therefore, BI refers to broad categories of applications and technologies that are used for corporate management, optimization of costumer relations, monitoring of business activities, data mining, reporting, planning, and decision-making support on all levels of managements [4]. The value of a BI-system for business is to provide adequate and reliable up-to-date information on various aspects of enterprise activities in an organization.

BI-systems comprise an integrated set of technologies and tools that contain several modules such as Extract, Transform and Load (ETL), data warehouse, Online Analytical Processing (OLAP), and tools for data mining and reporting [24, 25]. ETL is the set of processes relevant for the transformation, organization, and integration of loading data from numerous applications and systems into target systems, e.g., data warehouses. According to [4], a data warehouse is a subject oriented, integrated, time-variant, and non-volatile collection of data that provides generalized and consolidated data in multidimensional views. OLAP techniques use data warehouses designed for sophisticated enterprise BI-systems for the interactive and effective analysis of data in multidimensional spaces. Data-mining methods such as association, clustering, classification, prediction can be integrated with OLAP operations to enhance the interactive mining of knowledge from various data sources [4].

2.2 Current Challenges of BI-Systems

Recent discussions about BI issues include OLAP techniques, data mining, and data warehouses [5]. Business users rarely have real time access to data and work on historical data that are not updated regularly. Most BI-applications need an expert/specialist to run statistical reports, or data-mining processes to generate reports for business users [6]. According to [7], the challenges of analyzing and generating information are reflected with 27.4% when business users try to collect a single version of real-time fact from multiple data sources and systems. Furthermore, the management of information challenges is reported with 35.8% for delivering, self-service reporting and analyzing of data.

On the other hand, authors of [8] discover that existing BI-systems are deficient in three points. Firstly, the current BI-systems can only provide solutions for specified situation. Secondly, current BI-systems cannot deal with data from dynamic environments. Thirdly, the update speed is slow for BI-systems where the source code must be rewritten when a new requirement is added. When the size of data highly increases for existing BI-systems, this is another challenge for BI-analysts to react immediately to events as they occur. Therefore, real-time BI-systems (RTBI) [6] emerge to provide real-time tactical support for immediate enterprise actions in reaction to events that employ classic data warehousing for deriving information. Additionally, RTBI also need a comparison between present business events and historical patterns in order to automatically detect problems in distributed environments.

2.3 Concept of Agent-Based BI-Systems

An intelligent agent (or simply an agent) is a piece of software, or a computer system that performs services and gathers information autonomously [1, 5]. Agents need to display intelligence properties in order to perceive their environment and be autonomous for performing tasks on behalf of the users in heterogeneous environments [9]. Intelligence also improves the capability of an agent, while interacting with the context to perceive changes during knowledge exploration [10]. Hence, MAS provide an effective approach for coordination and cooperation among multiple units in complex distributed systems [11] and therefore, researchers incorporate MAS technology with data-mining algorithms for developing agent/based BI-systems [12]. Recent research of literature shows a trend for developing agent-based BI-systems in different domains such as e-commerce, supply chain management, resource allocation, intelligent production and so on [1, 28]. MAS are identified as a multiple role player in BI-systems, e.g., user behavior learning, customizing interaction information, and user notification when important events occur. Consequently, recent examples of agent-based BI-systems are reported about in [1, 4, 10, 27, 28, 54–56].

3 Designing Method

In this research, we propose a new designing solution for novel BI-systems that combine MAS and DDM technologies. The term of artifact is used to describe the high-level overview of BI-MAS components and therefore, the design-science research (DSR) [26] framework is applied for understanding, executing, and evaluating our proposed artifact. According to [13], for designing a new artifact, rigor is achieved by appropriately applying existing foundations and methodologies. With respect to DSR, first, we refer into nine well-known existing agent-oriented methodologies introduced in [17–20, 29–35]. Our studies discover that each of these methodologies has its own strengths and weaknesses, and respected coverage phases that are limited by not covering the entire development life-cycles. Finding and selecting a suitable agent-oriented methodology can vary regarding the complexity of agent-based BI-systems. Hence, agent-based BI-systems are designed with different system specifications and choosing an appropriate agent-oriented methodology is a challenge for developers [14].

Literature highlights several efforts of researcher for selecting existing agent-oriented methodologies. In this regard, researchers propose multiple solutions in the format of evaluation frameworks, comparison methods, and approaches in [30–32]. Each of these evaluation processes are fulfilled based on different criteria and context to evaluate a limited number of methodologies. Still, none of these evaluation processes are known as a standard, or applicable method for comparing methodologies.

In this section for comprehending the analysis and design phases, we first briefly demonstrate our proposed new artifact BI-MAS life-cycle and applicable functional and non-functional requirements. Next, we demonstrate the selection processes of applicable agent-oriented methodology based on each phases of the standard development life-cycle (waterfall).

3.1 BI-MAS Life-Cycle

In this research, we consider those functional and non-functional requirements that are relevant for BI-systems together with DDM and MAS-technology. In general, the functional and none-functional requirements for a BI-system might be achieved from the business need within the context of organizational strategies, system structure and existing business processes. In this research, we assume to focus on the implementation of MAS technologies in the internal structure of BI-systems. Hence, agents play key roles in the BI-MAS development phases.

We explain the functional and non-functional requirements with representing a life-cycle for BI-MAS (as depicted in Fig. 1). The life-cycle starts with receiving an input from business stakeholders. The remaining life-cycle is defined as automated processes to find information about particular input-data. To implement agent-abilities for the entire BI-MS life-cycle, we assume that the workflow and data-flow are carried by MAS technologies, i.e., dispatching to data-sources, aggregation of information, data-mining processes, and so on. Table 1 illustrates all required notations used in BI-MAS life-cycle.

According to [14], system requirements must express the properties of a system and scenarios that specify the use-cases of intended systems and intended for implementation. For developing agent-based complex system, requirements and scenarios can be expressed in various degrees containing formal, semi-formal, and informal [14]. Therefore, we summarize to explain the functional-requirements of BI-MAS by considering MAS technology sequentially using sub-sections (A–F).

A. BI-MAS shall support a user interface (UI) for business users to navigate, explore and access into distributed data-sources, and receive information.
B. BI-MAS shall provide the means of aligning business intelligence, business process improvement and automation in internal logical work plan and data mining operation using MAS.
C. BI-MAS must contain those facilities, i.e., parallel ETL and OLAP processes, to speed-up data exploration processes while data collecting from different source systems into a more advanced discipline.
D. BI-MAS support parallel processes using MAS technology in data mining and knowledge discovery process that plays important roles in today's business area.

E. BI-MAS shall contain local data warehouse to store historical explored information within defined time specification.
F. BI-MAS shall emphasis on unifying data representation, cleaning, summarization, aggregation and understandable querying over transactional stores.

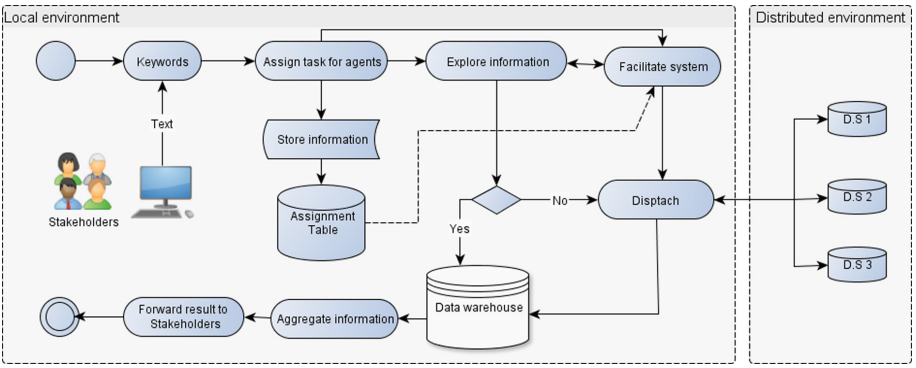

Fig. 1. BI-MAS life-cycle.

Table 1. Notation of BI-MAS life-cycle

	Starting point of the life-cycle
	Represents task that can be assigned to an agent
	Shows data storage servers
	Used to represent the condition
	Represents data warehouse
	Illustrates optional task
	Ending point of the life-cycle

Based on above mentioned assumption, we also deduce the following properties list as non-functional requirements that are applicable for BI-MAS.

Reactivity - defines ability of an agent to perceive and respond actively the environment in a timely manner.
Autonomy - represents ability of an agent to act independently without direct interaction of user.

Confidentiality - illustrates ability to protect the agents' data from other unauthorized agents, or other hosts.
Collaboration - indicates the ability of agents to interact with each other to achieve a common purpose, or objective.
Deliberately - shows activities of agents that represent an output in such a manner that output of on activity is input for next consecutive processes.
Accuracy - defines ability of agents to present the high quality of its performance during execution of several functions.
Reliability - illustrates the ability of agent during execution of several processes without any interruption whether errors occur in the system.
Trustability - represents ability in which an agent trusts another agent in same host to delegate part of their task in heterogeneous environments.
Security - indicates functions that are applied for agents to protect agent from another harmful agents, or hosts.

3.2 Agent-Oriented Methodologies

In this section, we briefly explain the nine well-known agent-oriented methodologies as follows. ***Tropos*** is an agent-oriented software engineering (AOSE) [15] methodology that covers software development processes in five phases of: early- requirements, late-requirements analysis, architectural-design, detailed design, and implementation. In early requirements analysis, Tropos focuses on the understanding of a problem by studying with organizational setting. Secondly, this methodology emphasizes on analysis phase for a deeper understanding of the environment where software must operate along with relevant functions and qualities. In the architectural-design phase, the system global architecture is defined in terms of sub-systems that are interconnected through data, control, and other dependencies. The required agents are specified at the micro-level and each agent's goals, roles and capabilities are specified in detail along with the interaction behaviors.

PASSI (Process for Agent Societies Specification and Implementation) [16] is a step-by-step requirement-to-code methodology for designing and developing multi-agent integrating system. The analysis and design phases of PASSI are determined and characterized by iterative step-by-step refinement. Therefore, producing a final stage to concrete design and implementation phases is based on the FIPA standard [16]. In addition, PASSI is composed of five models that address different design levels of abstraction such as system requirements-, agent society-, agent implementation-, code-, and deployment-models.

Prometheus [2] methodology is developed for building agent-based software systems. The main goal of Prometheus is to have a process with associated deliverables for industry practitioners and undergraduate students without a previous background in agents [17]. The Prometheus methodology consists of three phases such as system-specification, architectural-design, and detailed-design phases. The system-specification phase corresponds to the motivation layer and focuses on identifying the basic functionalities of a system. The architectural-design phase focuses on the types of functionalities that are delivered by agents. Additionally, this phase determines agent roles, agent-acquaintance diagrams, data type and protocols that are applicable in system

architecture. Finally, the detailed-design phase looks at the internal characteristics of each agent and how it can fulfil its tasks within the overall system.

ADELFE [18] methodology developed to software engineering Adaptive-MAS (AMAS). In fact, adaptive software is used in situations where either an environment is unpredictable or a system is very open. This methodology guarantees that software is developed according to the AMAS theory to cover preliminary-requirements, final-requirements, analysis, design, implementation and tests. In the analysis phase, an engineer is guided to decide to use adaptive multi-agent technology and to identify an agent through the system and environment models. In the design phase, this methodology provides cooperative agent models and helps the developer to define local agent behavior.

MOBMAS (Methodology for Ontology-Based Multi-Agent Systems) [19] is a software engineering methodology that contains activities and associated steps to conduct the system development, techniques to assist a process, and a definition of models. The development process of MOBMAS is highly iterative and incremental between activities. In total, there are five activities, each focusing on a significant area of MAS development such as analysis-activity, MAS-organization design- activity, agent-internal design-activity, agent-interaction design-activity, architecture design-activity.

MaSE (Multi-agent Systems Engineering) [20] is an established object-oriented methodology that supports a complete life-cycle to design and develop agent-based systems. MaSE has been extended to an Organization-based MAS-Engineering (O-MaSE) framework. Additionally, O-MaSE is an architecture-independent methodology [2] that consists of three steps: capturing goals, applying use-cases, and refining roles. Consequently, the design phase has four steps: creating agent classes, constructing conversations, assembling agent classes, and system design. These steps are also called models that describe a process to guide a system developer from an initial system specification to system implementation. Furthermore, this methodology proposes nine classes of models under its life-cycle such as goal-hierarchy, use-case, sequence-diagrams, roles, concurrent-task, agent-classes, conversations, agent-architecture, and development-diagrams [20].

Gaia [38, 39], is known as one of the first complete methodologies for the analysis and design of MAS. This methodology is applied after gathering requirements that cover the analysis- and design phases. In the analysis phase, the role model and interaction model are constructed. The agent model, services model, and acquaintance model are constructed during detailed design stages. Additionally, the Gaia method has many similarities with MaSE [21]. In general, both MaSE and Gaia capture much of the same type of information from requirements. In Gaia methodology, most of the proposals concentrate on the analysis phase. As a MAS concept is quite complex, this methodology provides models and guidance that is near to some anthropomorphic modelling, which is convenient for understanding the system problem [21]. In Gaia, roles and services help to organize the functionality that is associated to an agent or a group of agents.

ROADMAP (Role-Oriented Analysis and Design for Multi-agent Programming) [17, 20] methodology extends Gaia with four improvements such as formal models of knowledge, role hierarchies, explicit representation of social structures, and

incorporation of dynamic changes. In this methodology, a complex system is defined as a computational organization of interacting roles at the analysis stage optimized for quality goals, and populated with agents at the design stage. The roles in ROADMAP have runtime realization that allows runtime reasoning, social aspects modifying, and agent characterizing. In ROADMAP, the models that are constructed in analysis and design phases including use case-, environment-, knowledge-, role-(characterized by four attributes: responsibilities, permissions, activities and protocols), interaction-(contains protocol model), agent-, services-, acquaintance-models.

RAP (Radical Agent-oriented Process) [17, 40] is based on Agent-Object Relationship (AOR) modeling. The essential objective of RAP/AOR methodology is to enhance team productivity by agent-based work process management including both workflows and automatic interactions among team members in a system [2]. Unlike other mentioned agent-oriented methodologies, RAP/AOR is more concerned with distributed MAS. In AOR, several models are included, i.e., agents' actions, event perceptions, commitments, and claims. The Agent's role can be represented by AOR agent diagrams where different agent types may relate to each other through a relationship of generalization and aggregation.

3.3 The Evaluation Results

Based on the defined life-cycle for BI-MAS (discussed in Sect. 3.1), we need to find a fitting software engineering development method out of these nine well-known agent-oriented methodologies. Each of these methodologies has its own respective concept, modeling language, processes, specifications, principles, etc. It is very difficult to select one of them by chance without either understanding the development phases, or having an assessment results. Since the selection process of an agent-oriented methodology is a challenge [30–32]. finding commonalities between these proposed evaluation frameworks for performing the evolution process, must be well specified.

Table 2. Notations for evaluation process of agent-oriented methodologies.

Notations	Descriptions
F	For fully coverage
M	For mostly coverage
P	For partial coverage
N	For none or zero coverage
U	The sum of total calculation for each methodology
m	The number of agent-oriented methodologies
β	Represents the respective weights

On the basis of our study, each of these discussed methodologies has relevant development steps in their life-cycle similar to the waterfall model [22]. To evaluate these methodologies, we need criteria in our evaluation processes that fall into the

Software Requirement categories termed *Functional* and *Non-functional* (as illustrated in Table 3). Thus, the evaluation procedure with detailed description requires an equation that takes into consideration each phase of development life-cycle [23]. Thus, we define Eq. 1, in which several criteria are required as Table 2 illustrates.

As outlined in Table 3, to achieve the utility U of an agent-oriented methodology m, for computing processes, each of these defined variables {F, M, P and N} receives a score on the scale {3 | 2 | 1 | 0}. It means that 3 denotes full coverage, 2 mostly coverage, 1 partial coverage, and 0 for none coverage of each components that is used in each phases of a software development life-cycle. The respective weights (βs), for the requirements for fully F, the mostly M, and partially P, can be set in many ways. Our main concern from respective weights β in this equation is $\beta_F > \beta_M > \beta_P > \beta N$.

$$U^m = \beta_F F^m + \beta_m \sum\nolimits_i M_i^m + \beta_P \sum\nolimits_j P_j^m + \beta_N \sum\nolimits_k N_k^m \tag{1}$$

Furthermore, the evaluation results of Eq. 1 are outlined in Table 3. As a result, if we consider the second row (*Non-functional*) result that indicates very low scores of these respective nine methodologies, certain methodologies do not cover non-functional requirements at all, while others have merely low scores. Similarly, when we consider *Testing*, again the coverage is either none, or very low score. We conclude that none of these listed agent-oriented methodologies support fully the BI-MAS development life-cycle individually and none of these methodologies has high scores from initial-stage via very advanced-level of implementation to test processes.

Table 3. Evaluation result of agent-oriented methodologies.

Development life-cycle	*Tropos*	*PASSI*	*Prometheus*	*ADELFE*	*MOBMAS*	*MaSE*	*Gaia*	*ROADMAP*	*RAP/AOR*
Functional	3	2	3	2	2	2	2	3	2
Non-functional	2	0	0	0	0	1	0	2	2
Analysis	3	1	1	3	2	3	3	3	3
Design	2	2	2	2	2	3	2	3	3
Implementation	2	1	1	2	1	1	1	3	3
Testing (V&V)	1	0	0	0	1	2	1	1	1
Utility m	**12**	**6**	**7**	**9**	**8**	**12**	**9**	**15**	**14**

On the other hand, our studies discover that the Gaia methodology, the extended agent-oriented methodologies ROADMAP and RAP/AOR score best, especially during the analysis and design phases (also shown in Fig. 2). The MaSE methodology also has good scores in the analysis and design phases while the *Non-functional* and *Testing* phases have very low scores. Additionally, the ROADMAP and RAP/AOR have a comparable foundation for their development life-cycle and support each other. It means both offer promising options in the analysis- and design-phases during deployment [2].

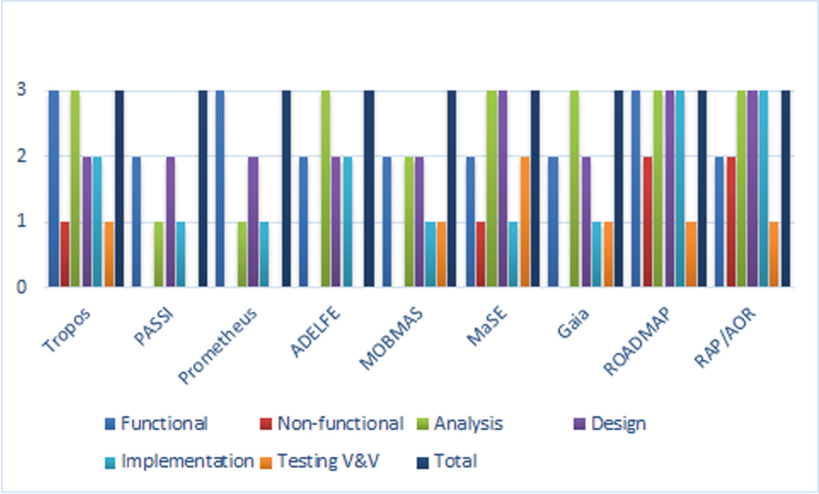

Fig. 2. Statistical analysis of agent-oriented methodologies

According to [17, 43], ROADMAP and RAP/AOR provide a common framework of system features for specifying, designing, developing, and implementing intelligent agent systems. At the level of computational design and implementation of these methodologies the focus rests on different kinds of models from various aspects. For instance, a goal-model to define actors in the intended system, a domain-model to identify related objects in the system domain, a knowledge-model to indicate the properties of objects in their respective contexts, an interaction-model to represent MAS realistic interactions, and behavior-model to address the decision making and performing activities of each agent. These contexts support us to select these two methodologies to cover the analysis- and design-phases for BI-MAS life-cycle.

4 Detailed Design Phase of BI-MAS Architecture

To pursue the ROADMAP and RAP/AOR methodologies along with AOM techniques [2] for fulfilling the analysis- and design-phases, our objective is to develop several required models that transfer the defined functional and none-functional requirements along with the terms of agent functions, roles, and behaviors. To provide a clearer understanding from the analysis- and design phases, we consider the graphical notation that is sufficiently expressed to handle the complexity of BI-MAS in following subsections.

4.1 The Goal Model

In the goal model of BI-MAS that is depicted in Fig. 3, we first present the root-functional goal of *Run BI-system* with the attached role of *Stakeholder*. According to ROADMAP and RAP/AOR methodologies [2], the root-functional goal is called the

value proposition that is too complex and therefore must be further refined into manageable functional sub-goals. During characterizing the BI-MAS goal model, simple tree-hierarchy schema diagram is generated by strict top-down decomposition. This schema leads us to connect one particular agent per branch that contains relative sub-tasks relevant to one requirement. To achieve a goal, the system requires a specific role for each agent and also sub-goals with quality goals to represent functional and non-functional requirements. In the first case, the quality goals *Autonomy* and *Collaboration* mean that the agents of a BI-system are capable of performing their tasks autonomously and support each other during knowledge exploration. The main goal includes roles and sub-goals that define capacities, or positions with functionalities that are needed for the BI-MAS.

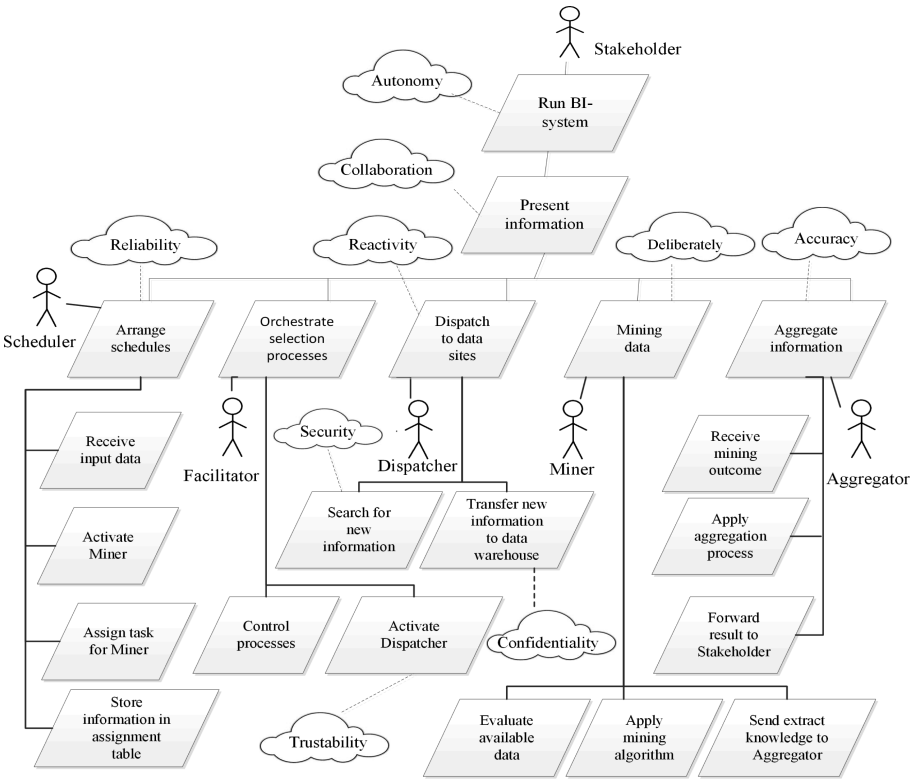

Fig. 3. The goal model of the BI-MAS.

We decompose the main goal that is associated with *Present information* into smaller related sub-goals such as *Arrange schedules, Orchestrate selection processes, Dispatch to data-sites, Mining data,* and *Aggregate information*. The *Arrange schedules* goal is decomposed into four sub-goals of *Receive input data, Activate Miner*,

Assign task for Miner, and *Store information in assignment table*. Additionally, this goal is attached to the role of *Scheduler* and the quality goal of *Reliability* that represents the responsibility of an agent for setting an assignment to other single, or a group of agents based on the received input data. The *Orchestrate selection processes* goal includes two sub-goals *Control processes* and *Activate Dispatcher* with a quality goal of *Trustability*.

Furthermore, this goal is attached to the role of *Facilitator* that is responsible for activation and termination of *Dispatcher* agents. The *Dispatch to data sites* goal comprises also two sub-goals *Search for new information* with the quality goal of *Security*, and *Transfer new information to data warehouse* with an attached quality goal of *Confidentiality*. We attach the role of *Dispatcher* to this goal that is responsible to explore new information from different data-sites for transferal to a data warehouse. The *Mining data* goal contains three sub-goals *Evaluate available data, Apply mining algorithm*, and *Send extract knowledge to Aggregator*. This goal also attaches to the role *Miner* together with the quality goal of *Deliberately* that represents the performance of agents during knowledge exploration for sharing with other mining processes. The *Aggregate information* goal includes three sub-goals *Receive mining outcome, Apply aggregation process*, and *Forward result to Stakeholder.* We describe these goals with the attached role of *Aggregator* with the quality goal of *Accuracy* that is responsible to obtain knowledge from other miner agent/agents separately and after the modification and collection processes, it submits the result to the *Stakeholder*.

4.2 The Domain Model

With respect to ROADMAP and RAP/AOR methodologies [2], for each defined role there must be an agent mapped in. The model that shows knowledge about the environments and illustrates relationships of agents is called domain model [24]. In this section, we discuss the domain model that represents the entities of the problem domain that are relevant for BI-MAS environments (shown in Fig. 4). This model describes the main domain entities, the agents' roles, and their relationships with each other within two environments. In fact, an agent environment produces and stores objects that can be modeled as resources, which are accessed by agents [2]. In this regard, we consider two types of environments in which the agents either exist or migrate to. The local environment where all the activities of agents perform between each other, are also called agent host [6]. The distributed environment where the *Dispatcher* agents can migrate to is related to *Data sites of system*. The agent that plays the role of *Stakeholder* can interact with real BI users in a local environment via *User interface.*

All other remaining agents are software agents identifiable based on their activities between these two environments. For instance, the *Scheduler* agent is responsible for activating and assigning tasks to the *Miner* that is situated in the local environment. The *System assignment table* comprises domain entities where all information about agents and their responsibilities are stored that belong to this environment. The *Miner* agent is responsible for discovering knowledge from a *Data warehouse* that is modeled as a

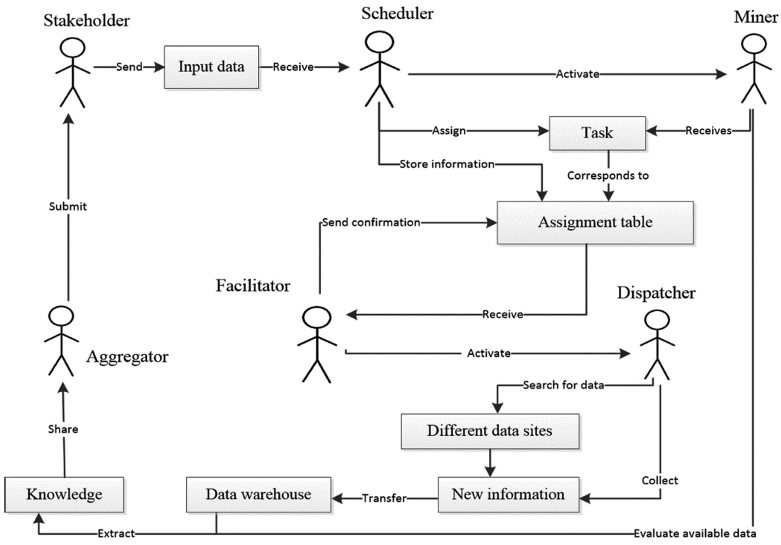

Fig. 4. The domain model of the BI-MAS.

domain entity of the local environment. In addition, the *Aggregator* has a responsibility to summarize the collected information from the *Miner* agent and submit to *Stakeholder*. Furthermore, the data sites of a system is modeled as a domain entity (*Different data sites*) in the domain model that belongs to a distributed environment. Here, the *Dispatcher* agent is responsible for transferring new information from a distributed to local environment.

In the next section, we derive a knowledge model from the domain entities that is relevant for the agent knowledge base.

4.3 The Knowledge Model

A knowledge model can be viewed as an ontology that provides a framework of knowledge for agents of problem domains [2]. The agents' knowledge model demonstrates the agent role, internal knowledge and the relationship with objects in the environment. Agents represent information about itself via knowledge attributes that are intrinsic properties of an agent. There are two kinds of attributes: numeric- and non-numeric attributes. A non-numeric attribute of an agent represents one or more quality dimensions. The knowledge-attribute types are string, integer, real, boolean, date and enumeration. In the knowledge model depicted in Fig. 5, each agent is modeled by representing a type, name, id, and relationships within an environment. For instance, the *Miner* agent knows about the task that is assigned by the *Scheduler* agent, and knows about the *DataWareHouse* where it can search to explore new knowledge.

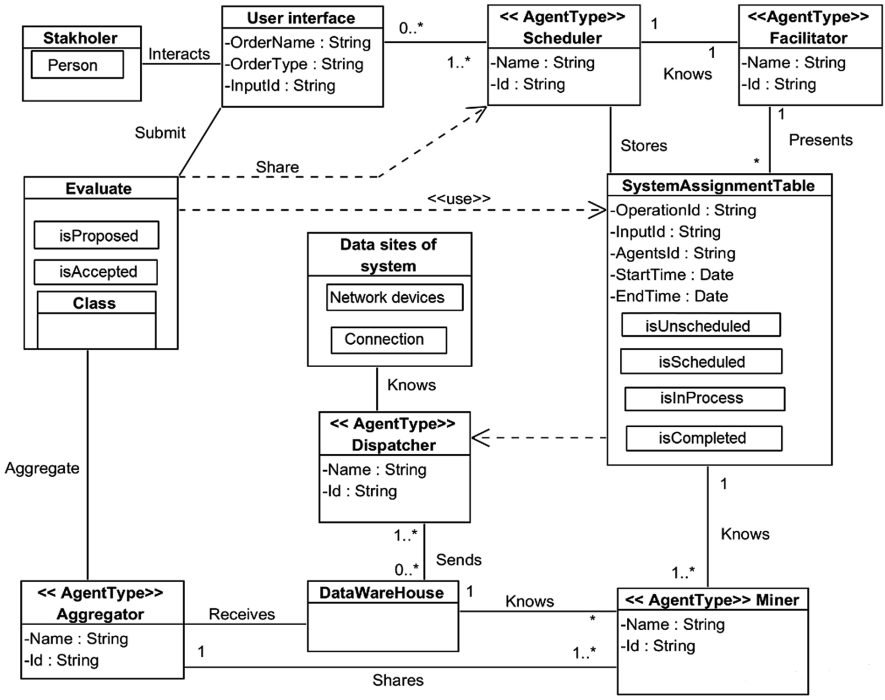

Fig. 5. The knowledge model of the BI-MAS.

Several objects of various types such as *SystemAssignmentTable*, *DataWareHouse*, *Data sites of system*, and *User interface,* are defined in the knowledge model. These objects are shared between all agents playing different roles in the BI-MAS component. For instance, the *SystemAssignmentTable* in which all information about the agents activities are stored by the *Scheduler* agent is shared between the *Facilitator*, *Miner*, and *Dispatcher* agents. Moreover, in each object of the knowledge model, several related attributes and predicates are defined. For example, the object of *SystemAssignmentTable* in Fig. 5 describes the attributes of *OperationId*, *InputId*, *AgentId*, *StartTime*, and *EndTime* that represent the information relevant to each agent. Furthermore, *SystemAssignmentTable* illustrates several status predicates of agents such as *isUnscheduled*, *isScheduled*, *isInProcess*, and *isCompleted* that are demonstrates the status of agents. Next, we describe more about what message flow and interaction occur between agents involved in BI-MAS.

4.4 The Interaction Model

According to ROADMAP and RAP/AOR methodologies [2], the interaction modeling must represents the interaction links between multiple agents of a system. Through interaction modeling, we exhibit a clear concept for an observer to understand what message flow and interaction occurs between agents and how the sequence of actions are performed by each agent. Additionally, the interaction model can be captured by

any of interaction-diagrams, or interaction-sequence diagrams, or interaction-frame diagrams. Below, we depict the interaction-diagram between *Scheduler*, *Miner*, *Facilitator*, *Dispatcher*, and *Aggregator* agents with a depicted scenario of the mining processes.

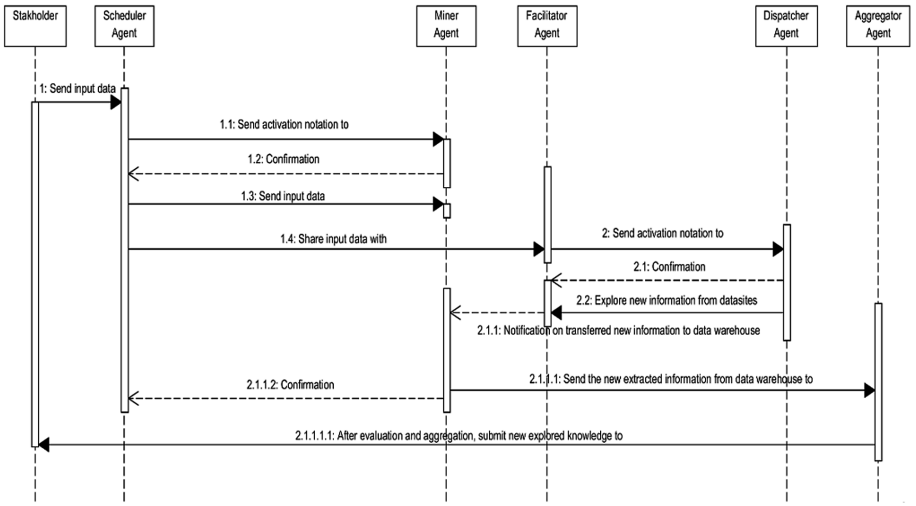

Fig. 6. The interaction model of the BI-MAS.

In interaction model depicted in Fig. 6, the arrows between agents demonstrate the existence of an interaction link that allowing an agent to initiate interaction with another agent. In general, an interaction can occur either by sending a message to another agent or performing a physical action affecting to it. Information about each interaction is extracted from responsibility, or role of each agent. In this diagram, each action event is characterized by a sequence number. These numbers constitute an interaction sequence between agents that are involved in this diagram.

Next, we elaborate the system scenario and behavior model that identify the sequence of various activities in which each agent plays a specific role in BI-MAS concept.

4.5 The Behavior Model

A behavior model is a scenario that is described to achieve the system goal by system agents [2]. A scenario can be defined also as collective activities that involve either a single, or multiple agents. Similarly, a scenario is illustrated with sub-scenarios that are corresponding to sub-goals of the system. In this section, we present a motivational scenario for each agent that has a specific role in BI-MAS. This scenario is based on the format of a goal-based use-case that is originally used in the RAP/AOR methodology [25]. For instance, the scenario corresponding to the goal "*Present Information*"

(as illustrated in Table 4) has five sub-scenarios with respective sub-goals such as "*Arrange schedules*", "*Orchestrate selection processes*", "*Dispatch to data sites*", "*Mining data*", and "*Aggregate information*". In addition, during the behavior modeling, a scenario can be triggered by a situation that involves the agent initiating related scenarios. For example, a scenario to reach for system goal "*Present Information*" is triggered by an action "*Send input data*" that is performed by a *Stakeholder* (shown in Table 4).

Table 4. A Scenario for achieving the goal "Present Information".

SCENARIO 1					
Goal	Present information				
Initiator	Stakeholder				
Trigger	Send input data by Stakeholder				
Description					
Condition	Step	Activity	Agent type/roles	Resources	Quality goals
	1	Arrange Schedules (Scenario 2)	Scheduler	Input data	Reliability
If data warehouse is empty	2	Orchestrate selection processes (Scenario 3)	Facilitator		Trustability
	3	Dispatch to data-sites (Scenario 4)	Dispatcher	Input data	Reactivity, security, and confidentiality
	4	Mining data (Scenario 5)	Miner	Data warehouse	Deliberately
If new explored information is not same	5	Aggregate information (Scenario 6)	Aggregator	New knowledge	Accuracy

Table 5. A Scenario for achieving the goal "Arrange schedules".

SCENARIO 2					
Goal	Arrange schedules				
Initiator	Scheduler				
Trigger	Input data received by Scheduler				
Description					
Condition	Step	Activity	Agent type/roles	Resources	Reliability
	1	Receive input data	Scheduler	Input data	
If more than one input data arrive	2	Activate Miner agent	Scheduler	Input data	
	3	Assign task for Miner	Scheduler	Input data	
	4	Store information in assignment table	Scheduler	Input data	

Table 6. A Scenario for achieving the goal "Orchestrate selection processes".

SCENARIO 3					
Goal	Orchestrate selection processes				
Initiator	Facilitator				
Trigger	Assign task for Miner by Scheduler				
Description					
Condition	Step	Activity	Agent type/roles	Resources	Quality goals
	1	Control processes	Facilitator	Assignment table	Trustability
If the requested data more than one	2	Activate Dispatcher	Facilitator		

Table 7. A Scenario for achieving the goal "Dispatch to data sites".

SCENARIO 4					
Goal	Dispatch to data sites				
Initiator	Dispatcher				
Trigger	Agent function activates by Facilitator				
Description					
Condition	Step	Activity	Agent type/roles	Resources	Quality goals
If data belong to different data sites	1	Dispatch to data-sites	Dispatcher	Different data sites	Reactivity
If the new information is not already in data warehouse	2	Search for new information	Dispatcher	Data warehouse	Security
	3	Transfer new information to data warehouse	Dispatcher	Data warehouse	Confidentiality

Table 8. A Scenario for achieving the goal "Mining data".

SCENARIO 5					
Goal	Mining data				
Initiator	Miner				
Trigger	New information transferred to data warehouse by Dispatcher				
Description					
Condition	Step	Activity	Agent type/roles	Resources	Quality goals
	1	Evaluate available data	Miner	Data warehouse	Deliberately
If data transferred more than one data site	2	Apply mining algorithm	Miner		
	3	Send extract knowledge to Aggregator	Miner		

Table 9. A Scenario for achieving the goal "Aggregate information".

SCENARIO 6					
Goal	Aggregate information				
Initiator	Aggregator				
Trigger	New knowledge shared by Aggregator				
Description					
Condition	Step	Activity	Agent type/roles	Resources	Quality goals
	1	Receive mining outcome	Aggregator	New knowledge	Accuracy
	2	Apply aggregation process	Aggregator	New knowledge	
	3	Forward result to Stakeholders	Aggregator	New knowledge	

Agent behavior models are platform-independent [2], and can be expressed only in terms of an abstract agent architecture. According to the abstract agent, an agent behavior is determined by a controller based on agent perceptions and knowledge. In this model (shown in Fig. 7), the controller is modeled in the terms of roles and behavior that is relevant for BI-MAS requirements. In fact, the agent's role starts a sequence of activities comprising various actions. During the execution of each cycle of an abstract agent, each agent role can be recognized and triggered by a perception. Each defined agent can percept a message, or action that is performed by other agents. In addition, some types of roles are not triggered in a system and occur as a start event once per each execution cycle of an abstract agent. For instance, Fig. 7 represents the behavior of an agent called *Scheduler* who initiates the execution life-cycle process by perceiving input data.

The sequences of roles that are marked with R mean the following. R1 shows activities "*Receive input data*", "*Activate Miner agent*", "*Assign task for Miner*", and "*Store information in assignment table*" that are listed in Table 5. The condition attached to role R3 means it is triggered only if the requested input data does not exist in the *data warehouse*. According to R3, the *Facilitator* performs activities that are outlined in Table 6. Consequently, role R4 follows the *Dispatcher* with performing activity types "*Search for data*" and "*Transfer new information to data warehouse*" (as illustrated in Table 7). When the data is transferred in the *data warehouse*, the role R2 starts the activities of type "*Evaluate available data*", "*Apply mining algorithm*", and "*Send extract knowledge to Aggregator*" (as illustrated in Table 8). Finally, after receiving a confirmation message about new explored information, the role R5 is started with activities types "*Receive mining outcome*", "*Apply aggregation process*", and "*Forward result to Stakeholders*" by *Aggregator* (as illustrated in Table 9). To know more about the notations that are used in agent behavior model demonstrated in Fig. 7, we refer the reader to reference [2].

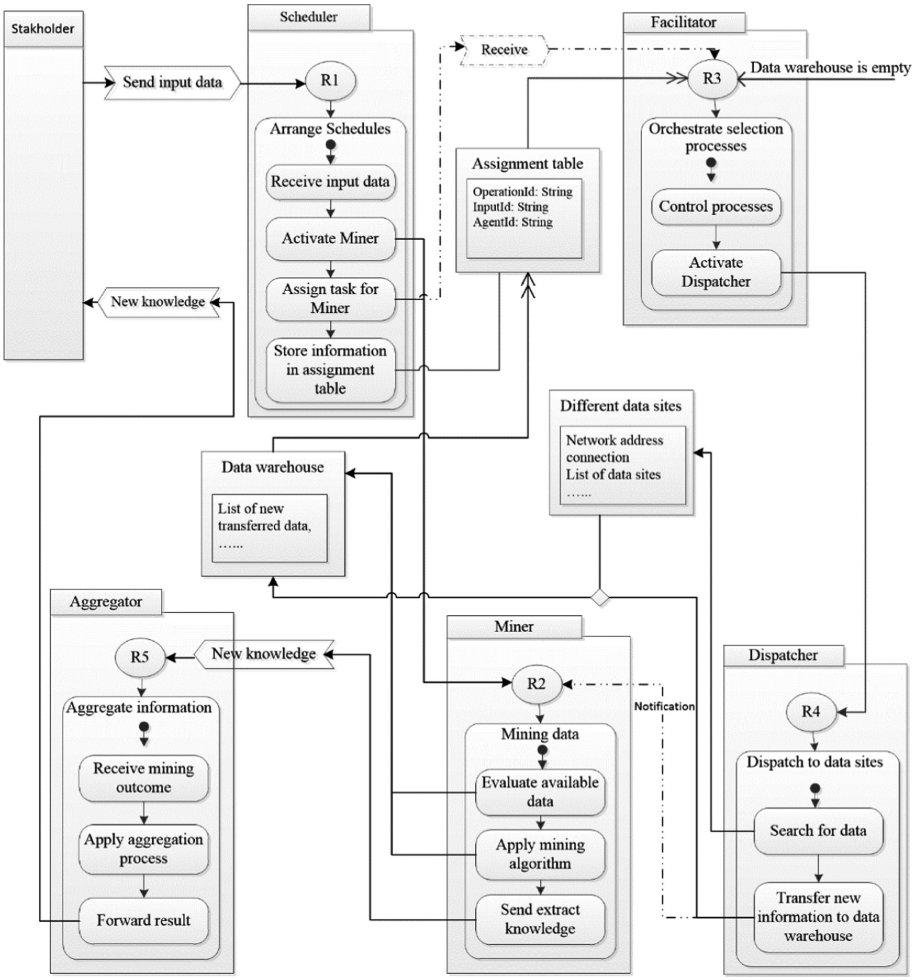

Fig. 7. The behavior model of the BI-MAS.

4.6 Overview of BI-MAS

As illustrated in Fig. 8, the baseline system requirements are transferred into a high-level overview of BI-MAS by performing the analysis- and design-phases with the support of the ROADMAP and RAP/AOR methodologies.

In Fig. 8, we present a three layered view for the BI-MAS framework. In this research, the main focus is on the agent-level and therefore, the important components of the BI-MAS comprise agents with different roles defined into an integrated layer-based structure. Each layer comprises a single, or multiple agents that have key roles to perform specific functional requirement (discussed in Sect. 3.1). To determine the correlation among each layer, we assume to have one agent in the interface level and the remaining must be defined on the operating level. For instance, the *Stakeholder*

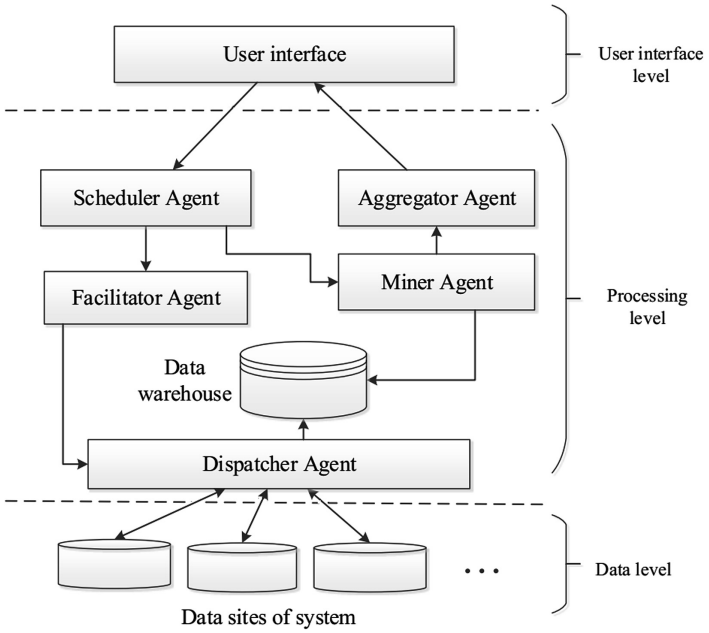

Fig. 8. The general overview of the BI-MAS.

agent is defined as a human agent that interacts with the BI-MAS stakeholders through *User Interface*. For simplicity, the *User Interface* module comprises functionality to capture the input data (keywords) and reports back the research result (new explored information) to stakeholders via BI-MAS system interface. It means that the operation of BI-MAS follows sequences, i.e., an operation starts from the top (*User Interface*) to bottom, and from the bottom to the top, while the research results are found and transferred to stakeholders. The remaining agents (as shown in Fig. 8) are defined in the *processing level* as follows:

Scheduler Agent - agent is responsible to This agent is responsible to receive the requested keywords and determine the types of operations defined under the BI-system and creates a work plan for other agents accordingly. After assigning tasks to a single-, or group of agents, updated information is stored into the *System Assignment Table* (as shown in Fig. 4).

Facilitator Agent - This agent is responsible to facilitate the mining process due to activation and termination of the *Dispatcher* agents. Moreover, this agent comprises a knowledge module that stores the history of requested keywords and previously retrieved information in the data warehouse that helps the *Miner* agent to explore new information without -waiting for the *Dispatcher* agent.

Miner Agent - The *Miner* agent plays an important role in the mining of data from the local environment (*Data warehouse*) by deploying mining algorithms. Additionally, this agent comprises a module to share the explored information automatically with other *Aggregator* agent.

Dispatcher Agent - To dispatch the agent into different data-sites, we use mobile agents [26] that have the capability to travel into different network locations via Internet connections. This agent is responsible to determine the computational resources at different domains and store the retrieved information into data warehouse.

Aggregator Agent - This agent is responsible to aggregate the collected new information received from either single, or multiple *Miner* agents. In order to present a compact and meaningful knowledge, the *Aggregator* agent plays a transformation role by resolving the conflicts and contradictions of newly mined information. Finally, this agent is responsible to report back the obtained knowledge to the *Stakeholder* agent.

Moreover, in *Data level*, we assume to represent distributed *data sites* that might have different types of databases with different datasets. With respect to the analysis and design phases, the developed life-cycle of BI-MAS requires to verify and validate with effective and technical methods.

In the next section, we describe the mapping processes of BI-MAS into several formalization format.

5 Mapping the BI-MAS Models to a Formalization

In general, the verification and validation (V&V) [27] processes are conducted to assure the quality of product-, or development life-cycle based on system requirements. According to [28], the construction of V&V processes of self-adaptive software systems such as agent-based distributed systems have remained a very challenging task for developers. There is a need for novel V&V methods to provide assurance of the result for the entire life-cycle of complex-systems. As illustrated in Table 3, even the V&V processes are not supported fully by any of existing agent-oriented methodologies. In this section, we intend to represent a new approach of V&V processes for BI-MAS with new methods and tools.

In this regard, our studies show that CPN-tool receive interest of researchers for designing V&V processes of distributed systems [29]. With respect to [47, 48], Colored Petri Nets (CPN) is a notation for the modeling and validating of systems in which concurrency, communication, and synchronization are the foci. In order to formulate the BI-MAS life-cycle, it is important to map BI-system models to a formal and deterministic notation that allow us to fulfill the V&V processes. To accomplish the V&V processes, we consider CPN-tools that supports extensions with time, color, and hierarchy for modeling and analysis of distributed systems by a graphical simulation tool [30]. The CPN language allows to organize a model as a set of modules, and it includes a time concept for representing the time token to execute events in the modelled system. The modules connect with each other through a set of well-defined interfaces in a similar way as known from many modern programming languages.

Table 10. Acronyms, names and descriptions of token colors.

Level	CPN module	Data property	Description	Type
1	BI-MAS	sc	Scheduler unique id based on that receives input data from stakeholder/stakeholders	Integer
		sh	Based on this numbers, several stakeholders can request multiple keywords with specific id	
		mid	Activating numbers for Miner agents	
		key	Stakeholder keywords searching for new knowledge	String
		s	The final explored new knowledge	
		t	The time sequences in which the data is arrived in a place from different data-sites	Time
2	Activate Agents	sh1	The stakeholder id that is stored in temporary place to fulfill the aggregation processes	Integer
		key1	The stakeholder keyword that is stored in temporary place to fulfill the aggregation processes	String
		s1	The final explored knowledge that is stored in temporary place to fulfill the aggregation processes	
		t1	The time sequences that are used to compare two results arrived from different data-sites based on one keyword	Time
3	Search for data-sites	dsid	Activated numbers for Dispatcher agents	Integer
		t	The time that is generated for each sequence of data, which is explored from different data-sites	Time
4	Data-site1	sid1	Unique id that is related to data-site1	String
		f1_key	Finding key based on input keyword on data-site1	
		s_r1	Present the result for searching keywords on data-site1	
	Data-site2	sid2	Unique id that is related to data-site2	String
		f2_key	Finding key based on input keyword on data-site2	
		s_r2	Present the result for searching keywords on data-site2	
	Data-site3	sid3	Unique id that is related to data-site3	String
		f3_key	Finding key based on input keyword on data-site3	
		s_r3	Present the result for searching keywords on data-site3	

The CPN model contains places, drawn as ellipses or circles, transitions drawn as rectangular boxes, a number of directed arrows connecting places and transitions, and finally some textual inscriptions. For instance, places and transitions are called nodes that are connected with directed arrows. An arrow always connects a place to a transition, or a transition to a place. In a CPN model, places may hold multiple tokens that carry color (i.e., attributes with value). In addition, transitions are ready to fire when all input places hold the required sets of tokens and produce condition-adhering tokens into output places. In this section, we present step by step processes of mapping and formal transformation of BI-MAS life-cycle that is prerequisite for V&V processes.

5.1 Related BI-MAS Data

For the formalization processes, the data elements of the BI-MAS model and sub-modules are declared for 4 refinement hierarchical-levels. Table 10 lists all the relevant token colors with their hierarchic refinement availability that is used for all lower- but not for any higher hierarchy levels. In the left column of Table 10, number 1–4 represents sequentially from level 1 to level 4 the lowest refinement levels of the BI-MAS components. In the second- and third columns showing are the name of nested modules and token colors that are used during the formalization of BI-MAS models. The fourth column textually explains the data-flow properties of the BI-MAS life-cycle. Finally, the fifth column presents the token colors properties while their types are defined either integer, string, or time. The integer-type of tokens is used as identification number and string-type tokens can be either stakeholder input keywords, or a matching result for corresponding search key. Time-type tokens also called time stamps that can be used for different purposes in CPN models. In this paper, we use time as a sequence number associated with objects to specify the first and last arrival of tokens from other places into targeted places.

5.2 Formalized BI-MAS with Nested-Modules

With respect to [31], to resolve the complexity of a distributed system, the design processes must be produced by modularity, regularity, and hierarchy characteristics. As CPN-tools support hierarchy nested-modules, we use this property of CPN for applying nested modules to cover the entire BI-MAS life-cycle. The top-level module of BI-MAS depicted on Fig. 9 formalizes the cooperative environment of BI-MAS that are used for data exploration, or data mining in distributed systems. Here, we assume that each token represents several unique ids associated with search keys and matching search results. This associated id helps the *Aggregator* agent to prevent conflicts and contradictions at the end of life-cycle. The searching- and mining processes in this complex system are well formed based on discrete business-process specifications that start with a unique state, in which the tasks are processed in a parallel structure by agents that lead to a unique end state. In this figure the life-cycle starts the processes either by receiving a single *input data* or multiple *input data* (as discussed in Table 11) simultaneously as requested by *Stakeholders*. The mining processes ends while agents find new information based on the *input data* from different *data-sites* (shown in Fig. 12).

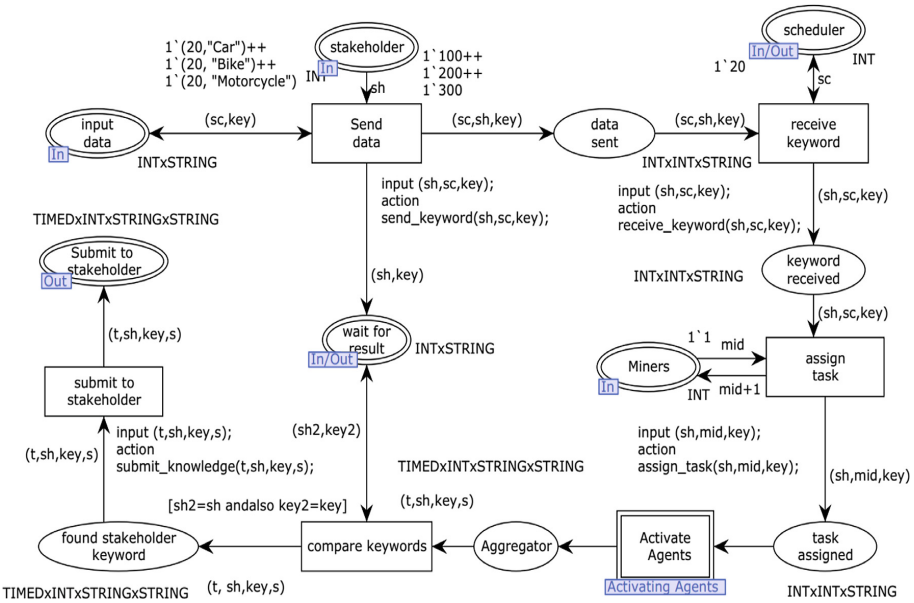

Fig. 9. The BI-MAS top level.

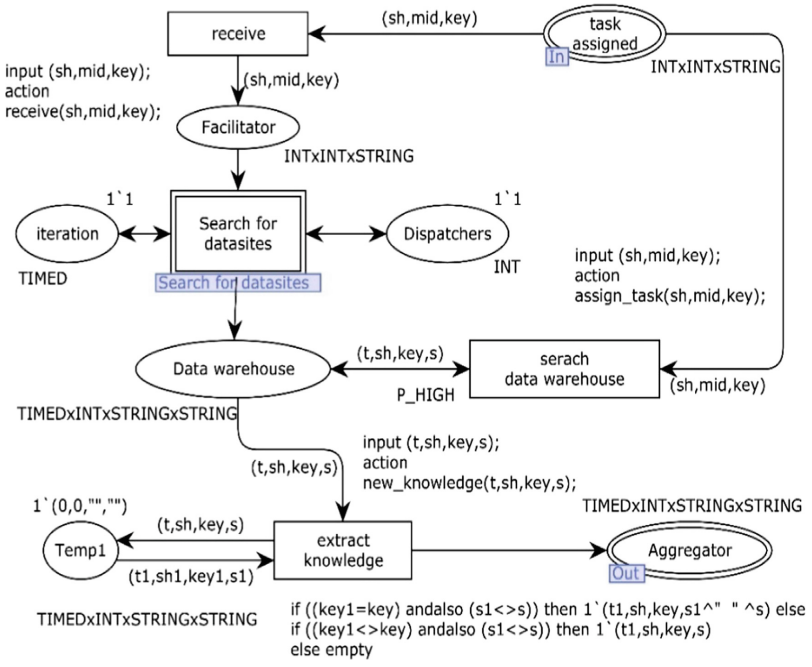

Fig. 10. Activity of *Facilitator* for generating *Dispatcher* agents.

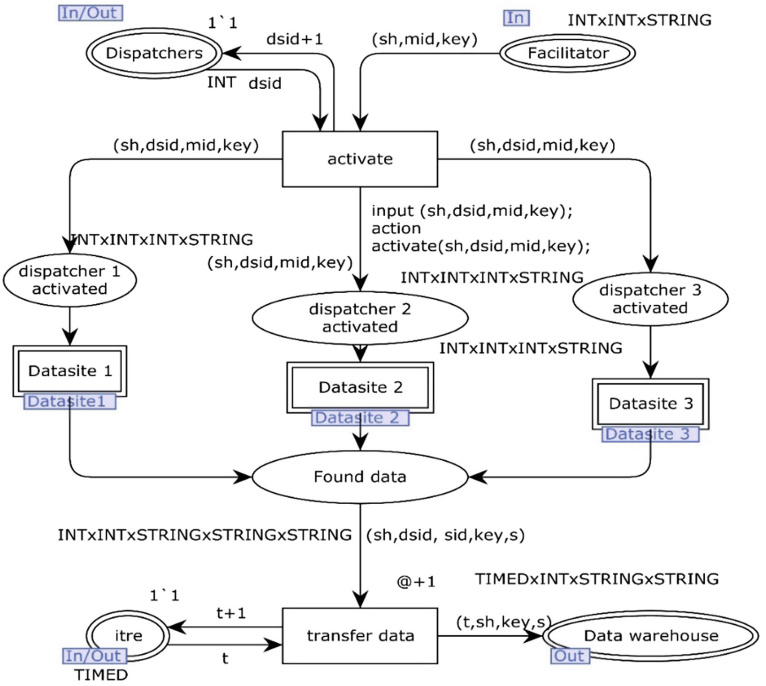

Fig. 11. Cooperative behavior of agents in different data-sites.

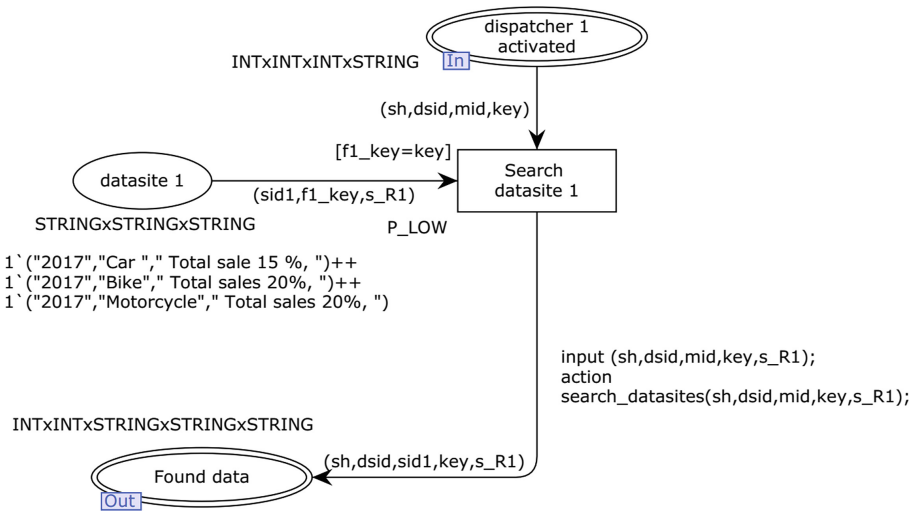

Fig. 12. One sample of the data-sites with two data sets in CPN model.

The second model covers the formalization of the two main agents' life-cycles relevant to *Facilitator* and *Aggregator*, as shown with double lined rectangles (*Activate Agents*) in Fig. 9. As demonstrated in Fig. 10, the life-cycles perform sub-module looking to *data-sites* in automated structure, while the *Facilitator* receives at least one keyword, or multiple keywords simultaneously. Here, for each search-key multiple *Dispatchers* can be activated based on the number of *data-sites*. In this part, the agents perform parallel computing that are used for quick access and manipulation of such distributed *data-sites*. In addition, the activity of *Aggregator* proceeds by a normalization function that is necessary for avoiding conflicts and contradicting data sets. For instance, here we define an evaluation function that is applied for outcome results of matching search-keys to normalize the consecutive search results depending on one keyword (explained in Table 11).

The construction of the third model is related to double lined rectangles (*Search for data-sites*) in Fig. 10. As illustrated in Fig. 11, the activities of an agent depend on the cooperative behavior of agents to exploit such computing environments for scaling up the data mining process. Here, the module shows that the data-mining processes can fulfill without loading all data sets into a single site. Instead, the resulting mining process transfers data into the warehouse. For V&V processes of BI-MAS, we assume to have three data-sites with different data-sets. Due to page limitation, we present here one sample, i.e., data-site 1 in Fig. 12. The remaining two other data-sites have the same structure while only the contents are different.

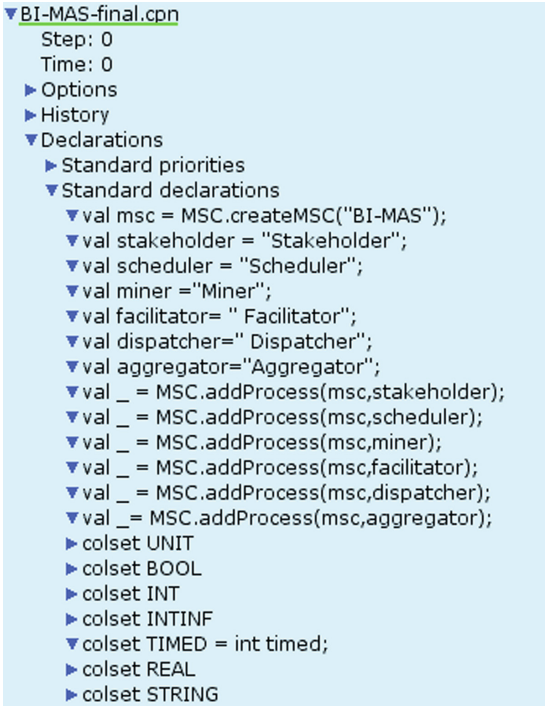

Fig. 13. Agent-interaction model with built-in functions in CPN-Tools

5.3 Transformations of BI-MAS Models to CPN Tools

The transformation of the corresponding types of conceptual models (explained in Sects. 4.1, 4.2, 4.3, 4.4, 4.5, 4.6), require syntactically procedures to represent the automated simulation results using CPN-tools. The mapping constructs of knowledge-, behavior- and interaction attributes can be transferred either by built-in functions, or user defined properties of CPN-tools. Message Sequence Charts (MSC) is well-defined functions for system engineering are used to present the communication messages of sender and receiver in complex systems [32]. In addition, MSC functions are used for adding new processes, presenting events between processes, and adding internal events into single process, or external events between two objects.

Figure 13 shows the defined functions of transforming such as knowledge-, interaction- and behavior models of BI-MAS using CPN-tools. To setup the MSC function, it is required to add declarations to CPN. We create one process for each agent as shown in Fig. 13, i.e. *Stakeholder*, *Scheduler*, *Miner*, *Facilitator*, *Dispatcher* and *Aggregator*. Based on scenarios of behavior models in Sect. 4.5, it is important to declare pockets for a sender who can send data from one place to receivers. In List.1, we explain the compound data types that are transmitted between agents by using MSC function.

List1. ***MSC functions to describe the pockets contain***

```
1  fun send_keyword(sh,sc,key)=MSC.addEvent(msc,stakeholder,scheduler, "SEARCH
   ["^key^"]");
2  fun receive_keyword(sh,sc,key)=MSC.addInternalEvent(msc,scheduler, "RECEIVE
   KEYWORD [ "^INT.mkstr(sh)^","^INT.mkstr(sc)^","^key^" ]");
3  fun assign_task(sh,mid,key)=MSC.addEvent(msc,scheduler,miner,"ASSIGN TASK [
   "^INT.mkstr(sh)^","^INT.mkstr(mid)^","^key^" ]");
4  fun receive(sh,mid,key)=MSC.addEvent(msc,scheduler,facilitator,"REQUEST DISPATCHER
   ACTIVATION [ "^INT.mkstr(sh)^","^INT.mkstr(mid)^","^key^" ]");
5  fun activate(sh,dsid,mid,key)=MSC.addEvent(msc,facilitator,dispatcher,"ACTIVATE
   DISPATCHER [ "^INT.mkstr(sh)^","^INT.mkstr(dsid)^","^INT.mkstr(mid)^","^key^" ]");
6  fun search_data-sites(sh,dsid,mid,key,s)=MSC.addInternalEvent(msc,dispatcher, " FOUND
   KEYWORD[ "^INT.mkstr(sh)^","^INT.mkstr(dsid)^","^INT.mkstr(mid)^","^key^","^s^" ]");
7  fun transfer_data(sh,dsid,mid,ndid,key)=MSC.addInternalEvent(msc,dispatcher, "
   TRANSFER TO DATA WAREHOUSE[
   "^INT.mkstr(sh)^","^INT.mkstr(dsid)^","^INT.mkstr(mid)^","^INT.mkstr(ndid)^","^key^"
   ]");
8  fun new_knowledge(t,sh,key,s)=MSC.addEvent(msc,miner,aggregator, " EXTRACT
   KNOWLEDGE["^TIMED.mkstr(t)^","^INT.mkstr(sh)^","^key^" ,"^s^" ]");
9  fun submit_knowledge(t,sh,key,s)=MSC.addEvent(msc,aggregator, stakeholder, " SUBMIT
   KNOWLEDGE["^TIMED.mkstr(t)^","^INT.mkstr(sh)^","^key^" ,"^s^" ]");
```

6 Evaluation and Discussion

With respect to [33], the BI-MAS life-cycle can be evaluated by applying different types of methods. In this paper, we consider in three types of empirical and non-empirical methods that are applicable for designed modules of BI-MAS as follows.

6.1 Validation and Verification of BI-MAS

As CPN models are executable and are used to model and specify the behavior of agents in BI-MAS, this section presents the visualization and simulation result in CPN models. For V&V processes, we present the simulation results that are generated automatically. Each module of CPN can be simulated interactively, or automatically. For interactive simulation, we use Message Sequence Chart (MSC) [32] that generates automated results similar to single-step debugging. It also provides a way to 'walk through' into CPN models that investigate different scenarios in detail and check whether the model performs as expected. For simulating processes, we use two different scenarios as illustrated in Table 11 comprising testing and performance analysis.

Table 11. Scenario for simulation process of BI-MAS.

NO.	Scenario description	Initial input	Final output	Test-goal
I	Single input scenario- In this scenario, we assume that an organization has business for three products such as *Car, Bike*, and *Motorcycle*. Based on the number of products, three data-sites located in different physical locations containing different information. In this scenario, the stakeholder requests to receive particular information of type (Car) product	Searching for ("Car")	As shown in Fig. 14, the output is aggregated information from different data-sites only regarding Car	To test the workflow and data-flow for single- input as a token for the entire life-cycle of BI-MAS
II	Multiple-input scenario- The stored information is the same as in the previous scenario. Based on the number of products, three data-sites contain different information. In this scenario, the takeholder requests information regarding multiple inputs simultaneously	Searching for ("Car", "Bike", "Motor cycle") at same time	Due to page limitation, the outputs figures can not be demonstrated	To test the workflow and data-flow for parallel processing the life-cycle of BI-MAS

Figure 14 shows an example of an MSC result during the execution of Scenario-I. The MSC has six columns for this behavior- and interaction scenario, i.e. explained in Sect. 4.5. The leftmost column represent the senders and the rightmost columns represent the receivers. The MSC captures a scenario where the first data packet sent by the *Stakeholder* and the four middle columns represent the sender and receiver of the BI-MAS life-cycle. Finally, this process is ended where the *Aggregator* transmits the data packet (e.g., contains explored new information) to *Stakeholder*.

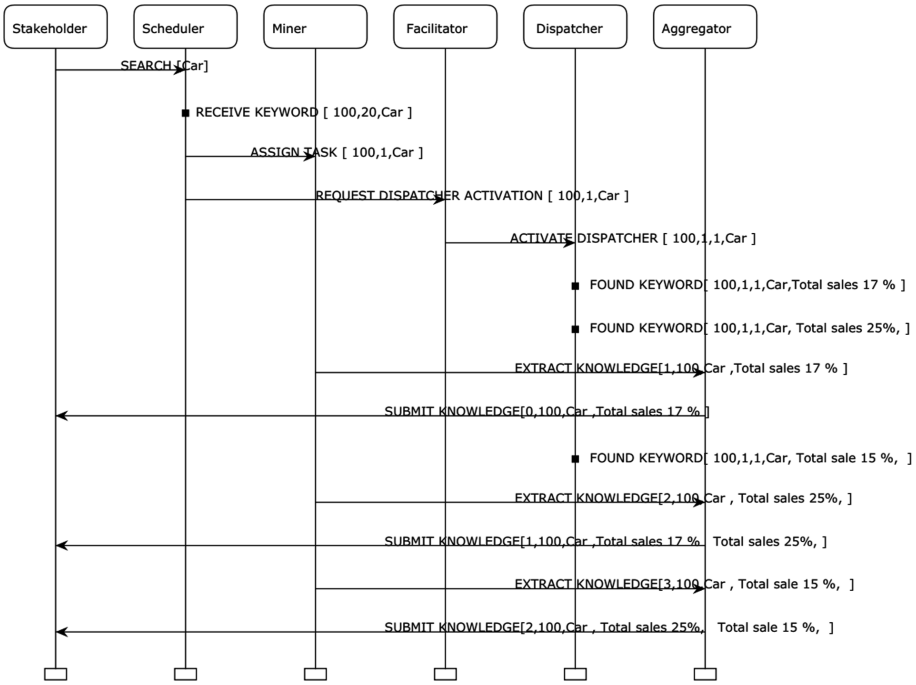

Fig. 14. MSC generated result for the BI-MAS life-cycle using CPN-tools

6.2 BI-MAS Models Properties

Colored Petri Nets is a formal modeling language that is well suited for modeling, validating and analyzing larger and complex systems. CPN Tools supports state spaces for hierarchical networks and offers facilities for collecting data during simulations and for generating different kinds of performance analysis reports. With respect to [48–53], the state space calculation and analysis considers each node that is involved in graphical representation of CPN models. Therefore, for testing and performance analysis of the BI-MAS life-cycle, we select the standard state space analysis instruments to collect data about the system performance.

In the first step, we consider the state space generation report for the BI-MAS life-cycle. The statistic result of state space report outlined in Table 12 provides basic information about size of behavioral properties and involved nodes. The first part of this result is generated based on single input Scenario-I (i.e., listed in Table 11) and the remaining parts of table are related to Scenario-II where we assume that 3 *Stakeholders* searching for 3 different keywords. In the fourth column of the table, one sample of dead transition is caused by an intentional separation of the BI-MAS in two parts for generating the state-space report. *Dead* and *live* define two properties of CPN to check the connection between entire nodes of graph.

Table 12. State space report of BI-MAS sub-models.

BI-MAS sub-models	State space	Scc graph	Dead Transition Instances	Live Transition Instances
Single key searching - I	nodes: 37 arcs: 53 sec: 0 status: full	nodes: 37 arcs: 53 sec: 0	None	None
Single key searching-II	nodes: 562 arcs: 1024 sec: 0 status: full	nodes: 562 arcs: 1024 sec: 0	search_data_warehouse 1	None
Multi-key searching- I	nodes: 3905 arcs: 6272 sec: 2 status: full	nodes: 3905 arcs: 6272 sec: 1	None	None
Multi-key searching- II	nodes: 343 arcs: 548 sec: 0 status: full	nodes: 343 arcs: 548 sec: 0	search_data_warehouse 1	None

The second V&V method is model checking, for which Table 13 shows results. To apply this method, we use several checking properties such as reachability, detection of loops, performance peaks during run time, full system utilization, and consistent termination. These results are generated automatically based on two test cases (i.e., either single-, or multi-set scenarios of Table 11) are used as input data for the BI-MAS life-cycle.

Table 13. Model checking results for BI-MAS lifecycle.

		Model Property				
	Modules	**Loops**	**Performance Peaks**	**Utilization**	**Home marking**	**Dead marking**
	BI-MAS	No	compare keywords	yes	no	no
	Activate Agents	No	activating agents	yes	no	no
	Search for data-sites	No	transfer data	yes	no	no
Data-sites	Data-site1	No	search data-site 1	yes	no	no
	Data-site2	No	search data-site 2	yes	no	no
	Data-site3	No	search data-site 3	yes	no	no

For model checking, we have three separate data-sites with different names and content as listed in Table 13, to test the parallelisms in the BI-MAS. The model-checking outcome is outlined in the second column that shows no loop exists in the entire life-cycle of the BI-MAS. We prevent the loops by implementing parallel methods as depicted in Fig. 11 where three *Dispatcher* agents re activated simultaneously to search for different data-sites.

Performance peaks in Table 13 represent places in the system that are bottlenecks. Each peak requires computing power and time during execution time. Peaks exist in all sub-modules of the BI-MAS with disparate levels but their potentials are very low. For the first module of BI-MAS, a peak occurs for *compare keywords* and *ids* whether there are multi-stakeholders waiting for multi-key results. In the second sub-module *activating agents*, a peak arises while the *Facilitator* agent generates several *Dispatcher* agents based on a number of data-sites for each keyword received. For the third sub-module search for data-sites, a peak is visible during *transfer data* that is associated with a new token (time) based on a particular time sequence when the result arrives. For all three data-sites, peaks occur due to searching and comparing processes for matching several keywords within divers data sets.

The home marking that represents an initial making [34], is considered to find all initial reachable nodes relevant to the BI-MAS life-cycle. Referring to [30], the home marking can be reached from any marking state. As outlined in Table 13, no home marking exists in the defined nodes of the BI-MAS The results for checking dead markings is similar to the *Dead Transition Instances* demonstrated in Table 12. Finally, the *Utilization* test represents all the subsets of the BI-MAS are used. It means that all modules are used during the execution processes of the BI-MAS lifecycle.

6.3 Related Work Discussion

Our agent-based solution for the comprehensively designed next-generation of BI-systems is not comparable with other proposed agent-based architectures, or frameworks from literature due to the extension as stressed in the following steps.

(a) Defining a proper agent-based life-cycle for a BI-MAS with applicable functional- and non-functional requirements that are known as essential for the development of any types of agent-based BI-systems.
(b) Selection and implementation of proper agent-oriented methodologies for the development procedure of BI-MAS models.
(c) Performance of step by step analysis- and design-phases with very detailed and overall concepts for each model of a BI-MAS.
(d) Fulfillment of V&V processes including formal mapping, modeling, transformation and with the analysis results, employing different accepted checking methods.

By referring to related work of agent-based BI-systems, we reflect several consideration points that are outlined in Table 14. The first column of this table presents the list of proposed agent-based BI-systems in literature. Furthermore, in the scope of agent-oriented methodologies, a gap exists for developed agent-based BI-systems. As represented in the second column, all of these proposed BI-solutions have developed without the implementation of any specific agent-oriented methodology. The systematic analysis- and design-phases listed in the third column are equally not given for all methods. Only two of the methods are covered, while the overall phases are not define on a sufficiently detailed-level. The fourth column represents the V&V processes that are applicable only for two proposed BI-systems. As demonstrated, one of these solutions covers only simulation processes, while others cover partially experimental results related to the developed system. Overall, neither authors demonstrate the proper transformation from the analysis and design phases to implementation, nor do the authors present proper V&V results with standard tools, or methods.

Table 14. Comparison results of BI-MAS with related agent-based BI-systems

Types of agent-based BI-system	Agent-oriented methodology	Analysis & design	V&V	References
BI fusion of agent network	No	Partially covered	No	[8]
Combination Framework of BI solution Multi-agent platform (CFBM)	No	No	No	[35]
Multi Agent Based Business Intelligence (MABBI)	No	No	Simulation	[36]
Agent-based architecture of BI system	No	No	No	[37]
MAS for managing supply chains	No	No	No	[6]
Stock Trading Multi-Agent System (STMAS)	No	Partially covered	Partially covered	[11]
Self-Organized Multi-agent Technology based BI Framework	No	No	No	[38]
Model for using Agent Based Systems (ABS) in BI	No	No	No	[12]

7 Conclusion

As we represent in the content of this paper, an agent-based BI-system requires a comprehensive road-map and highly systematic development approach along with an extensive verification- and validation processes. We discover that a complex system development life-cycle comprises important phases such as analysis and design that are essential for developers. Beside the implementation of a BI-MAS life-cycle, the analysis- and design phases must ensure developers that adopted concepts such as MAS and DDM are sufficiently. To achieve this enhancement, we define the BI-MAS life-cycle along with functional and non-functional requirements for several agent-levels. To transfer each of these requirements on a system-level within the structure of MAS technology, we emphasize the need to find a comprehensive methodology. Finding an applicable agent-oriented methodology is a considerable challenge, according to literature. To tackle this challenge, we present a novel approach to evaluate nine well-known agent-oriented methodologies that shows all methodologies are incomplete.

Moreover, we also demonstrate the conceptualization processes for BI-MAS models using agent-oriented methodologies such as ROADMAP and RAP/AOR along with AOM techniques to generate a holistic development methodology. During the enactment of ROADMAP and RAP/AOR, we discover are complementary for constructing the agent-level models of a BI-MAS in the analysis- and design-phases. Additionally, several challenges and limitation occur too during these two phases. For instance, a subset of BI-MAS agents must act in a static environment and other agents are part of a dynamically changing-, or distributed environments. Representing such a distinction in models is also a challenge for other existing agent-oriented methodologies. Moreover, none-functional requirements cannot be addressed with these two methodologies, e.g., agent security is a key component in distributed environments. Adding the security concept only in a model as a quality goal in goal-model without projection into the domain model, knowledge model, etc., is not sufficient in a BI-system development life-cycle. During the implementation of a BI-MAS, a need arises for integrating complementary models to represent the processes of authenticating agents, threat detection, and so on. It is necessary to use modified methods and tools to achieve such diverse model integration.

On the other hand, our studies also discover that the CPN-tool is a good candidate with its mathematical properties to perform V&V processes of agent-based BI-systems. As V&V is the core part of development processes, we assume to perform these processes with three different methods. Besides the intended processes for V&V, CPN shows limitations while mapping and transforming BI-MAS models, e.g., the inner action of BI-MAS agents cannot be modeled using CPN-tool. On the other hand, the usage of state-space methods of CPN is generates analytical statistics about state spaces, boundedness-, home- and live-markings, and fairness properties. Consequently, a diagnostic understanding about dependability and concurrency conflicts emerges for a BI-MAS.

The current BI-MAS high-level components require exclusive tools and platforms during the implementation-phase in real life organizations. In future work, to obtain the full view of BI-MAS in enterprise-level concepts, we consider more research and extensive models, i.e. data-warehouse architecture, types of servers, types of required services, etc. In the enterprise-level deployments of BI-MAS again, there exists a need for further research including two parts. Firstly, for the definition of the entire contextual organizational structure on a system-level, the integration of data warehouses and analytics tools requires additional research work. Secondly, describing and developing user-interfaces, middleware applications, and secure protocols are the second part that needs research and development work. Unclear is also the projection of important non-functional requirements such as security into other model types, e.g., for agent behavior and –interaction. Finally, the verification- and simulation capabilities of CPN do not cover all aspects of a BI-MAS to address dependability issues and concurrency conflicts. Thus, we plan to explore additional formal checking techniques for the goal of highly relyable system development.

References

1. Bologa, A.-R., Bologa, R.: Business intelligence using software agents. Database Syst. J. **2** (4), 31–42 (2011)
2. Draheim, D.: Smart business process management. In: 2011 BPM and Workflow Handbook, Digital Edition. Future Strategies, Workflow Management Coalition, pp. 207–223 (2012)
3. Atkinson, C., Draheim, D.: Cloud-aided software engineering: evolving viable software systems through a web of views. In: Mahmood, Z., Saeed, S. (eds.) Software Engineering Frameworks for the Cloud Computing Paradigm. Springer, London (2013). https://doi.org/10.1007/978-1-4471-5031-2_12
4. Mabrouk, T.F., El-Sherbiny, M.M., Guirguis, S.K., Shawky, A.Y.: A multi-agent role-based system for business intelligence. In: Sobh, T. (ed.) Innovations and Advances in Computer Sciences and Engineering. Springer, Dordrecht (2010). https://doi.org/10.1007/978-90-481-3658-2_35
5. Carbonell, J.G., Siekmann, J.: Multi-Agent Systems and Applications III. Springer, Heidelberg (2005). https://doi.org/10.1007/3-540-45023-8
6. Hemamalini, R., Mary, L.J.: An analysis on multi - agent based distributed data mining system. Int. J. Sci. Res. Publ. **4**(6), 1–6 (2014)
7. Devi, S.: A survey on distributed data mining and its trends. Int. J. Res. Eng. Technol. **2**(3), 107–120 (2014)
8. Zeng, L., et al.: Distributed data mining: a survey. Inf. Technol. Manage. **13**(4), 403–409 (2012)
9. Salih, N.K., Zang, T., Viju, G.K., Mohamed, A.A.: Autonomic management for multi-agent systems. IJCSI Int. J. Comput. Sci. Issues **8**(5), 338–341 (2011)
10. Lee, C., Lau, H., Ho, G., Ho, W.: Design and development of agent-based procurement system to enhance business intelligence Expert syst. Appl. **36**, 877–884 (2009)
11. Khozium, M.O.: Multi-agent system overview: architectural designing using practical approach. Int. J. Comput. Technol. **5**(2), 85–93 (2013)
12. Kargupta, H., Hamzaoglu, I., Stafford, B.: Scalable, distributed data mining-an agent architecture. In: Proceedings Third International Conference on Knowledge Discovery and Data Mining, pp. 211–214 (1997)

13. Krishnaswamy, S., Zaslavsky, A., Loke, S.W.: An architecture to support distributed data mining services in e-commerce environments. In: Advanced Issues of E-Commerce and Web-Based Information Systems, WECWIS 2000, pp. 239–246 (2000)
14. Sen, S.K., Dash, S., Pattanayak, S.P.: Agent based meta learning in distributed data mining system. Int. J. Eng. Res. Appl. (IJERA) **2**(3), 342–348 (2012)
15. Loebbert, A., Finnie, G.: A multi-agent framework for distributed business intelligence systems. In: 45th Hawaii International Conference on System Sciences (2012)
16. Ghandehari, E., Saadatjoo, F., Chahooki, M.A.Z.: Method integration: an approach to develop agent oriented methodologies. J. Artif. Intell. Data Min. **3**(1), 59–76 (2015)
17. Sterling, L.S., Taveter, K.: The Art of Agent-Oriented Modeling. MIT Press Ebooks, Cambridge (2009)
18. Bresciani, P., Perini, A., Giogini, P., Giunchiglia, F., Mylopoulos, J.: Tropos: an agent-oriented software development methodology. Auton. Agent. Multi-Agent Syst. **8**, 203–236 (2004)
19. Cossentino, M., Gaglio, S., Sabatucci, L., Seidita, V.: The PASSI and agile PASSI MAS meta-models compared with a unifying proposal. In: Pěchouček, M., Petta, P., Varga, L.Z. (eds.) CEEMAS 2005. LNCS (LNAI), vol. 3690, pp. 183–192. Springer, Heidelberg (2005). https://doi.org/10.1007/11559221_19
20. Juan, T., Pearce, A., Sterling, L.: ROADMAP: extending the Gaia methodology for complex open system. In: The first International Joint Conference on Autonomous Agents and Multiagent System: Part 1, pp. 3–10. ACM (2002)
21. Desouza, K.C.: Intelligent agents for competitive intelligence: survey of applications. Compet. Intell. Rev. **12**(4), 57–63 (2001)
22. Chaudhuri, S., Dayal, U., Narasayya, V.: An overview of business intelligence technology. Commun. ACM **54**(8), 88–98 (2011)
23. Expert OLAP.com. http://olap.com/learn-bi-olap/olap-bi-definitions/business-intelligence/. Accessed 21 Dec 2017
24. Ozlszak, C.M., Ziemba, E.: Approach to building and implementing business intelligence system. Interdiscip. Jo. Inf. Knowl. Manag. **2**, 135–148 (2007)
25. Ranjan, J.: Business intelligence: concept, components, techniques and benefits. J. Theor. Appl. Inf. Technol. **9**(1), 60–70 (2009)
26. Matillion: What businesses really want from business intelligence and Analytics (2017). www.matillion.com
27. Liu, S.: Business Intelligence Fusion Based on Multi-agent and Complex Network. J. Softw. **9**(11), 2804–2812 (2014)
28. Bobek, S., Perko, I.: Intelligent agent based business intelligence. In: Current Developments in Technology-Assisted Education, pp. 1047–1051 (2006)
29. Hevner, A.R., March, S.T., Park, J., Ram, S.: Design science in information systems research. MIS Q. **28**(1), 75–105 (2014)
30. Dam, K.H., Winikoff, M.: Comparing agent-oriented methodologies. In: Giorgini, P., Henderson-Sellers, B., Winikoff, M. (eds.) AOIS -2003. LNCS (LNAI), vol. 3030, pp. 78–93. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-25943-5_6
31. A. Sturm and O. Shehory, "A framework for evaluating agent-oriented methodologies," Agent-Oriented Information Systems, Springer, pp. 94–109, 2004
32. Sturm, A., Shehory, O.: A framework for evaluating agent-oriented methodologies. In: Giorgini, P., Henderson-Sellers, B., Winikoff, M. (eds.) AOIS -2003. LNCS (LNAI), vol. 3030, pp. 94–109. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-25943-5_7

33. Herlea, D.E., Jonker, C.M., Treur, J., Wijngaards, N.J.E.: Specification of bahavioural requirements within compositional multi-agent system design. In: Garijo, Francisco J., Boman, M. (eds.) MAAMAW 1999. LNCS (LNAI), vol. 1647, pp. 8–27. Springer, Heidelberg (1999). https://doi.org/10.1007/3-540-48437-X_2
34. Padgham, L., Winikoff, M.: Prometheus: a methodology for developing intelligent agents. In: Agent-Oriented Software Engineering III, pp. 174–185 (2003)
35. Bernon, C., Gleizes, M.-P., Picard, G., Glize, P.: The ADELFE methodology for an intranet system design. In: Equipe SMAC; Systèmes Multi-Agents Coopératifs (2002)
36. Tran, Q.-N.N., Low, G.: MOBMAS: a methodology for ontology-based multi-agent systems development. Inf. Softw. Technol. **50**(7–8), 697–722 (2008)
37. Soleimanian, F., Zabardast, B., Amini, E.: Analysis and design by agent based MaSE methodology: a case study. Int. J. Comput. Appl. **63**(4), 10–15 (2013)
38. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia mthodology for agent-oriented analysis and design. In: JAAMAS, pp. 1–27 (2000)
39. Zamboell, F., Jennings, N.R., Wooldridge, M.: Developing multiagent systems: the Gaia methodology. ACM Trans. Softw. Eng. Methodol. (TOSEM) **12**(3), 317–370 (2003)
40. Taveter, K., Wagner, G.: Agent-oriented modeling and simulation of distributed manufacturing. Idea Group Inc., pp. 1–14 (2007)
41. Domann, J., Hartmann, S., Burkhardt, M., Barge, A., Albayrak, S.: An agile method for multiagent software engineering. In: The 1st International Workshop on Developing and Applying Agent Framework (DAAF), pp. 928–934. ELSEVIER (2014)
42. Norta, A., Yangarber, R., Carlson, L.: Utility evaluation of tools for collaborative development and maintenance of ontologies. In: 2010 14th IEEE International on Enterprise Distributed Object Computing Conference Workshops (EDOCW), pp. 207–214 (2010)
43. Taveter, K.: Towards radical agent-oriented software engineering processes based on AOR modelling. In: Agent-oriented methodologies, Idea Group Inc., pp. 277–316 (2005)
44. Jennings, N.R., Norman, T.J., Faratin, P., O'Brien, P., Odgers, B.: Autonomous agents for business process management. Appl. Artif. Intell. **14**(2), 145–189 (2000)
45. Al-Neaimi, A., Qatawneh, S., Saiyd, N.A.: Conducting verification and validation of multi-agent systems. arXiv preprint arXiv:1210.3640 (2012)
46. Cheng, Betty H.C., et al.: Software engineering for self-adaptive systems: a research roadmap. In: Cheng, B.H.C., de Lemos, R., Giese, H., Inverardi, P., Magee, J. (eds.) Software Engineering for Self-Adaptive Systems. LNCS, vol. 5525, pp. 1–26. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-02161-9_1
47. Jensen, K., Kristensen, L.M.: Coloured Petri Nets: Modelling and Validation of Concurrent Systems. Springer, Heidelderg (2009). https://doi.org/10.1007/b95112
48. Honby, G.S.: Measuring, enabling and comparing modularity, regularity and hierarchy in evolutionary design. In: Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation, pp. 1729–1736. ACM (2005)
49. Donatelli, S.: Petri Nets and Other Models of Concurrency. Springer, Turku (2006). https://doi.org/10.1007/978-3-662-55862-1
50. Venable, J., Pries-Heje, J., Baskerville, R.: FEDS: a framework for evaluation in design science research. Eur. J. Inf. Syst. **25**, 77–89 (2016)
51. Norta, A.: Creation of smart-contracting collaborations for decentralized autonomous organizations. In: Matulevičius, R., Dumas, M. (eds.) BIR 2015. LNBIP, vol. 229, pp. 3–17. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-21915-8_1

52. Kutvonen, L., Norta, A., Ruohomaa, S.: Inter-enterprise business transaction management in open service ecosystems. In: 2012 IEEE 16th International Enterprise Distributed Object Computing Conference (EDOC), pp. 31–40 (2012)
53. Thai, T.M., Amblard, F., Gaudou, B.: Combination framework of BI solution \& multi-agent platform (CFBM) for multi-agent based simulations. In: 3EME Conference francophone sur le Gestion et l'Extraction de Connaissances: Journée Atelier aide à la Décision à tous les, Etages (AIDE@ EGC 2013), pp. 35–42 (2013)
54. Sperka, R.: Agent-based design of business intelligence system architecture. J. Appl. Econ. Sci. **VII**((3(21)) (2012)
55. Venkatadri, M., Sastry, M.G., Manjunath, G.: A novel business intelligence system framework. Univers. J. Comput. Sci. Eng. Technol. **1**, 112–116 (2010)
56. Rao, V.S.: Multi agent-based distributed data mining : an overview. Int. J. Rev. Comput. (2076–3328), 82–92 (2010)