Ontology-Based Workflow Generation for Intelligent Big Data Analytics

Banage T. G. S. Kumara[#], Incheon Paik[*], Jia Zhang[**], T. H. A. S. Siriweera[*], Koswatte R. C. Koswatte[*]
[#]Faculty of Applied Sciences, Sabaragamuwa University of Sri Lanka, Sri Lanka
[1]btgsk2000@gmail.com
[*]School of Computer Science and Engineering, University of Aizu, Aizu-Wakamatsu, Fukushima, Japan
[2]paikic@u-aizu.ac.jp,[4]siriweera@gmail.com,[5]ckoswatte@gmail.com
[**]Carnegie MellonUniversity, Silicon Valley, USA
[3]jia.zhang@sv.cmu.ed

***Abstract*—Big Data analytics provide support for decision making by discovering patterns and other useful information from large set of data. Organizations utilizing advanced analytics techniques to gain real value from Big Data will grow faster than their competitors and seize new opportunities. Cross-Industry Standard Process for Data Mining (CRISP-DM) is an industry-proven way to build predictive analytics models across the enterprise. However, the manual process in CRISP-DM hinders faster decision making on real-time application for efficient data analysis. In this paper, we present an approach to automate the process using Automatic Service Composition (ASC). Focusing on the planning stage of ASC, we propose an ontology-based workflow generation method to automate the CRISP-DM process. Ontology and rules are designed to infer workflow for data analytics process according to the properties of the datasets as well as user needs. Empirical study of our prototyping system has proved the efficiency of our workflow generation method.**

Keywords: **Big data analytics, Workflow, Data mining, Ontology**

I. INTRODUCTION

Data becomes Big Data when its volume, velocity, or variety exceeds the abilities of exiting IT systems to ingest, store, analyze, and process it. According to SINTEF, 2.5 quintillion bytes of data are created on the daily basis and 90% of the data in the world today were produced within the past few years[1]. Big data analytics (BDA) may yield valuable knowledge, andprovide support for decision making by discovering patterns from large set of data [1]. Thus, companies utilizing advanced BDAtechniques will grow faster than their competitors and seize new opportunities. According to the Gartner [2] survey results, the vast majority of businesses have established investment plans for BDA. Within others, data mining has been considered as an important technique for BDA.

Inthe business world, the most known standard data mining process model is called CRoss-Industry Standard Process for Data Mining (CRISP-DM) [3], which describes commonly used approaches by data mining experts. CRISP-DM breaks a data mining life cycle intosix phases:business understanding, data understanding, preparation of data, modeling, evaluation, and deployment. New knowledge is expected to be extracted in the data preparation, modeling, and evaluation steps of the model. However, our preliminary study of the process has identified several issues. First, the efficiency of the process is highly dependent on analysis parameters selected.Second, the process oftentimes demands intelligent analysis for various and dynamic environments. Third, the manual process of the steps hinders fast decision making on real-time data applications. Therefore, it is important to automate the workflow of the CRISP-DM steps.

In our research, we explore to automate the CRISP-DM process using Automatic Service Composition (ASC). ASC consists of four stages: planning, discovery, selection, and execution [4]. The core idea is to orchestrate existing services to achieve a larger task, resulting in a new composite and value-added Web service.Using the *travel planner* scenario in our previous work [4], the goal is to find a suitable composition plan according to the client's particular requirements, without checking all combinations of allservices and tasks. In the planning stage, anabstract workflow is constructed.Afterwards, the rest stages aim to identify optimal service composition for the identified abstract tasks. The performance of service composition is mainly based on the abstract workflow generated in the planning stage.In this paper, we will focus on the planning stage of our proposed automated approach.

From the literature, researchers typically use OWL-S, BPEL, or WSMO to statically describe a goal-driven abstract workflow in the planning stage. Many researchers adopt the hierarchical task network (HTN) planner [5] technique to dynamicallydevelop workflows. However, it is acknowledged that the formulation of HTN planning problem requires significant structural information.

In contrast to their work, in this paper, we present a novel ontology-based workflow generation method. Our core idea is to utilize ontology to acquire hidden domain knowledge, in order to generate more application-specific abstract workflow. Our approach also performs well with HTN planner, when complete and detailed knowledge is available on at least some hierarchically structured action execution patterns.

[1]www.sciencedaily.com/releases/2013/05/130522085217.htm

978-1-4673-7272-5/15 $31.00 © 2015 IEEE
DOI 10.1109/ICWS.2015.72

The rest of this paper is organized as follows. In Section II, we present the motivation. Section III discusses the intelligence BDA process. Section IV presents ontology based workflow generation method. Section V describes our implementation and evaluation. In section VI, we discuss related work. Finally, section VII concludes the paper.

II. MOTIVATIONS AND STRATEGIES

A. CRISP-DM Process

BDAaims to support decision making by discovering patterns and other useful information from large set of data, as shown in Figure 1. Standard data science processes, such as CRISP-DM, will conduct various data mining over data stored on certain infrastructure. Figure 1 also illustrates the six step-wise phases of operating CRISP-DM over big data. The initial phase is business understanding, aiming to understand the project objectives and requirements from a business perspective. In the data understanding phase, data scientists proceed with activities to get familiar with the data and identify data quality problems. The data preparation step covers all activities that prepress the raw data in order to yield the final dataset to be fed into modeling tool. In the modeling step, various modeling techniques are applied to analyze the dataset. Before proceeding to the final deployment of the model outcome, a thorough evaluation phase is needed to ensure to meet business requirements.

B. Motivating Example

At present, all stages of the CRISP-DM process are handled manually. Our hypothesis is that some phases may be able to be automated or semi-automated. Our idea is to model a CRISP-DM process as a workflow, and leverage ASC technique to realize some level of automation. In this section, we will describe a use case that will be used throughout the paper.

Scenario 1: *ABC airport Company plans to analyze the flight delay data to identify the factors that, led to the past delay. The company hopes to reduce airline delay through such data analytics.*

Let's apply CRISP-DM to the problem.

Business Understanding: The ultimate goal of the study is to reduce airline delay. Toward this goal, the main objective is to determine the causing factors in the past flight delay cases. To realize the objective, the researchers decide to create profiles for the air flight delay cases and identify the contributing factors.

Data Understanding: Flight delay-related data can be extracted from some trust resources, such as Research and Innovative Technology Administration (RITA) and weather history data from National Climatic Data Center, USA.

Data Preparation: First, researchers identify the core influence factors for the flight delay cases, for example weather, national aviation system delay and late aircraft delay. The identified influence factors are then treated as variables upon which cluster analysis is performed. During this phase, some data preprocessing tasks are performed including data format conversion, missing values handling, outliers handling, and Hadoop data preparation.

Modeling: Researchers feed the preprocessed data set into a file system for Big Data such as Hadoop Distribute File System (HDFS) and try to build a model. Typical candidate methods include clustering and classification algorithms. For example, clustering is a natural method for generating such a kind of segmented profiles, which contains a class of algorithms. Researchers can choose one clustering algorithm such as K-mean. By tuning the parameters (e.g., number of clusters, and maximum iteration) of the algorithm, researches can receive different clustering results. Figure 2 shows the sample output with six clusters identified, where carrier delay and late arrival delay are two parameters used.

Evaluation: Various analysis techniques such as discriminant analysis can be used to verify the "reality" of the resulted cluster categorizations.

Deployment: After the evaluation phase, the model can be deployed as a report to the client. ABC company can analyze the flight delay causes and design corresponding solutions to eliminate future airline delay.

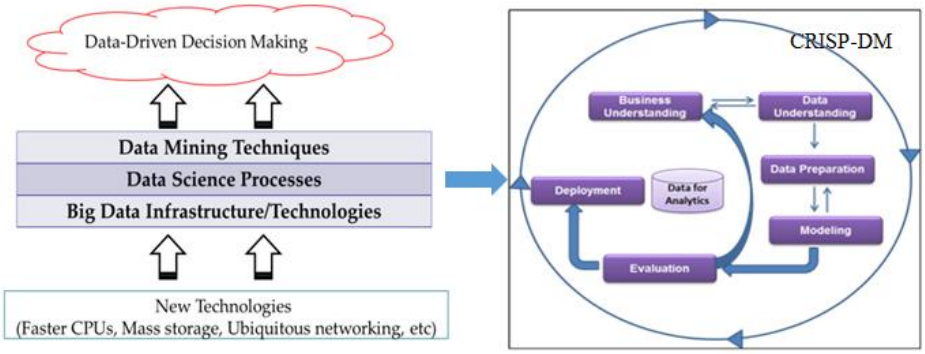

Figure 1. Whole process for BDA

For example, we can brief the profile of clusters to analyze the delay causes. As shown in Fig. 2, the members in cluster 1 have higher delay, due to higher carrier delay and late arrival delay. Since carrier delay is within the control of air carriers, the ABC airport could take some action. Late arrival delay relates with previous flight. This example illustrates how CRISP-DM can be applied to fulfill the business requirements.

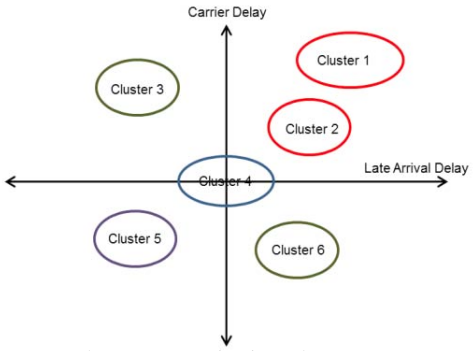

Figure 2. Sample clustering output

C. *Automatic Service Composition*

Web services technology allows users to publish software applications as universally accessible and programmable services. Users can use service composition techniques to combine existing services as components to create value-added services to solve complex problems. ASC aims to automate the process of finding proper services and compose them based on business requirements. Figure 3 shows the framework of ASC. It consists of four stages: planning, discovery, selection, and execution [4]. The planning stage identifies functional and non-functional properties, and plans a workflow whose comprising tasks are tagged with the properties. For each workflow task, the discovery stage tries to identify available related services as candidates. The selection stage tries to choose optimal services for each task. In the execution stage, the selected services are operated in the workflow. As shown in Figure 3, ASC is not a waterfall model; instead, each step may go back to the previous step

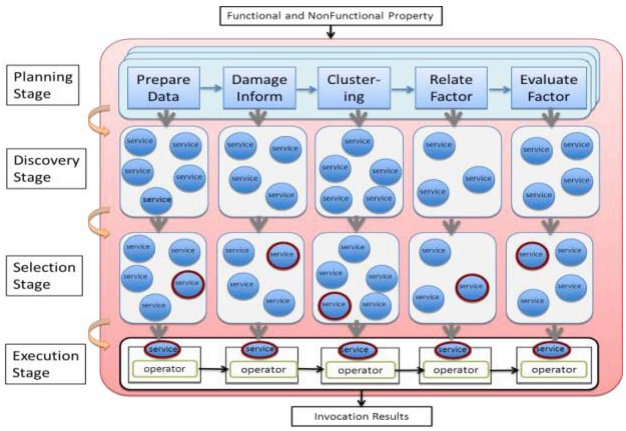

Figure 3. ASC process

for optimization. In this work, we apply ASC to help automate the CRISP-DM process.

III. ASC FOR CRISP-DM

In order to automate the process of BDA, we decompose each step of the CRISP-DM process into process services, each logically matched to a service composition process. The development of intelligent BDA process is summarized into the following steps: (1) develop service types and instances for the BDA; (2) define workflow for BDA; (3) develop service discovery algorithm for BDA; (4) develop service selection algorithm for BDA; and (5) develop composition algorithm for BDA. Figure 3 shows the whole architecture of the Intelligent BDA. In this paper, we focus onthe planning stage and workflow generation method for BDA process.

As explained in the previous section, the CRISP-DM process contains several consecutive steps. For each step, we have identified a collection of reusable tasks as shown in Table 1. For example, the data preparation step contains tasks such as check reasonable task and defects handling task. Further, each task is decomposed into atomic tasks as shown in Table 1. For example, in the defects handling task, one atomic task is to handle the missing values in data set; one another atomic task is to remove noisy data and to handle duplicated values. For the interest of space, we only list limited number of tasks for each step of the CRISP-DM process.

Table 1. Sample tasks identified in steps of CRISP-DM on Big Data infrastructure

Data Preparation Step		
Task		Sub Tasks
Format Decision Task		Convert_SQLtoCSV, Convert_XMLtoCSV
Defects Handling Task		Handle_MissingValue, Handle_Duplicates, Handle_NoisyData
Check Reasonable Task		Handle_Outlier, Handle_inconsistent, Handle_Uncertainity
Data Standardization Task		doNormalization, Handle_Synonyms
Hadoop Infrastructure Task		PrepareKV, InterfaceHDFS, setHadoopNodes, setHadoopBlockSize, setMapReduceApps
Modeling Step		
Task		Sub-Task
Similarity Measure		Jaccard, Cosine_similarity, Euclidian_distance
Decide Model	Clustering	Partitioning_clustering, Agglomerative_clustering
	Classification	Decision tree, SuportVecto Machine, Nearest Neighbor
	Recommendation	Content based, Collaborative Filtering
Evaluation Step		
Evaluation		Evaluate of model function Evaluate of result Evaluate of process
Deployment Step		
Deployment		Business deployment General deployment

Our next step is to turn the identified atomic tasks into abstract Web services. For an example, the atomic task

*Handle_NoisyData*is mapped to an abstract Web service that is capable of handling defects. Each abstract Web service is defined as its functionality and a set of inputs, outputs, preconditions and effects as follows:

WS::= <*WS_Operation*, *inputs_set*, *outputs_set*, *preconditions*, *effects*>

The *inputs_set* represents the information required to invoke the service; the *outputs_set* represents the information yielded by the service; the *preconditions* represents the additional constrains that must be satisfied in order to execute the service operation. The *effects* describe how the service execution changes the state of the world. Here, we consider the impact of preconditions and effects in generating aworkflow. It is important to note that dealing with effects is a necessary step in solving our composition problem. For an example, input and output of *Defects_Handling* service are files. If we use this service to remove missing values, it will change the state of the file (from has missing value - > No missing value). Thus, in some cases it is critical to consider the effect of the service compare to its other functionality.

IV. Ontology Based Workflow Generation

In the planning stage of ASC framework, a *WorkflowGenerator* creates an abstract workflow to satisfy the functional property of user requests. This stage typically adopts some planning methods, such as HTN planner [5] to support the workflow generation. In contrast to existing work, we have created an ontology-based method exploiting domain knowledge. We formulate the planning problem and a plan, as shown in Definition 1:

Definition 1. (Planning problem and Plan). A planning problem is a pair = <s_0, g> in the planning domain = <S, A, f>, where S is a set of states, A is set of actions, and f is state transition function. Further, s_0 is the initial state and g is the goals represented as a conjunction of logical atoms. A plan for the planning problem is represented as $\pi(a_1, a_2, \ldots a_n)$, where a_i is a classical action. The sequence of actions will form an abstract workflow guiding service composition.

We will use the use case scenario of analyzing flight delay data again to describe our proposed method. We first plan a workflow based on the features of the data set and business requirements.

Assume that the file format of the data set is XML, and has to be converted into CSV format. Thus, it is necessary to plan a *Format Decision* task. Considering that the data set may carry some missing values, a *Defects Handling* task is needed. In order to remove outliers, a Check Reasonable task needs to be planned. Assume that there is no need for data conversion or discrimination. Therefore, a *Data Standardization* task can be skipped before directly going to prepare Hadoop data. For the modeling function, assume researchers hope to use clustering, more specifically, to use similarity computing method. After the modeling phase, researches need to evaluate the results by going to the Evaluation task with the interpretation model. Finally, researchers can analyze the flight delay data using the model in the deployment task. The workflow for this use case scenario is shown in Figure 4.

Based on the workflow, an example collection of actions can be summarized as below:

Analyze the flight delay data to identify the factors

Action (Format_Decision, INPUT: file, OUTPUT: file, PRECOND:¬ CSV, EFECT: CSV)
Action (Defects_Handling, INPUT: file, OUTPUT: file, PRECOND:NULLvalue, EFECT: ¬NULLvalue)
Action (Check_Reasonable, INPUT: file, OUTPUT: file, PRECOND:Outlier, EFECT: ¬Outlier)
Action (Hadoop_Infra_Task, INPUT: file, OUTPUT: KVfile, PRECOND: (¬Outlier ∧ ¬NULLvalue∧ ……∧ ¬NoDuplicates))
Action (Similarity_Measure, INPUT: Data_with_features, OUTPUT: Similarity_valueOf _Data)
Action (Correlation, INPUT: Similarity_valueOf_Data, OUTPUT: Model)
Action(Evaluation, INPUT: Model∧TestData, OUTPUT: EvaluationResult)
Action(Check_Importance, INPUT: Model, OUTPUT: Decision, PRECOND:EvaluationResult, EFECT: Validate Model)
Action (Gen_Deployment, INPUT: Model, OUTPUT: AnalyticsResult, PRECOND: Pass_Evaluation)

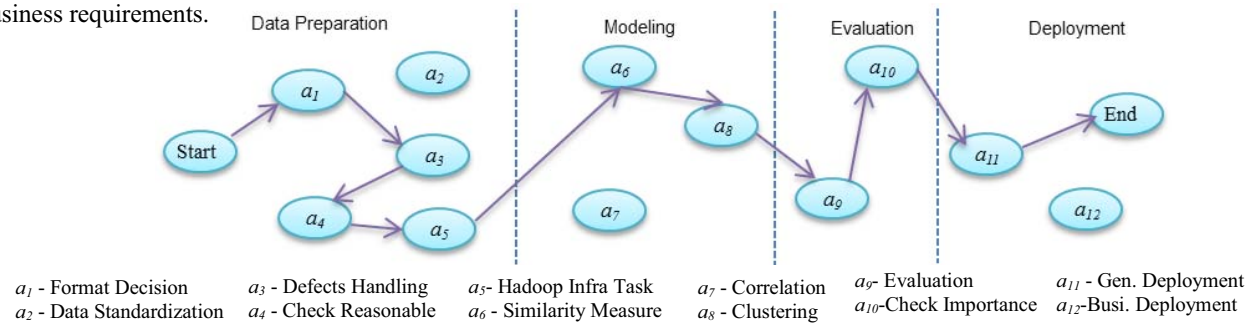

Figure 4. Workflow for scenario 1

In order to obtain such a workflow, we have developed an ontology-based workflow generation method. Our method leverages domain knowledge to guide the workflow generation. As the first step, we define domain ontology through generating ontology classes for the stages in CRISP-DM that require services and dataset. An *ontology* is a specification of *objects*, *categories*, *properties* and *relationships.* Note that ontology can carry hierarchies. We define hierarchy and ontology as follows:

Definition 2 (Hierarchy): A hierarchy H= (N, E) is defined as a directed simple graph with no cycles. H consists of a set of nodes N and a set of ordered pairs called edges $(n_p, n_c) \in E \subseteq \{N \times N\}$. The direction of an edge (n_p, n_c) is defined from the parent node n_p to the child node n_c. The ordered relationship $n_p \rightarrow n_c$ describes an *is-a* relationship between a parent node and a child node.

Example 1: Let N = {*CRISP-DM Stages, Stage1:Data_Preparation, Check_Reasonable, Outlier*}. We can define the hierarchy H according to the *is-a* relationship. For example, *Stage1:Data_Preparation is-a CRISP-DM Stages. Check_Reasonable is-a Stage1:Data_Preparation.*

Definition 3 (Ontology): A ontology is represented as O(C, R, H), where C is a set of concepts $\{c_1, c_2, ...,c_i\}$, R is a set of relationships $\{r_1, r_2, ..., r_j\}$, and H is a set of hierarchies. Here, concepts are nodes of the hierarchy.

Example 2: Ontology of CRISP-DM can be described by the concepts *Stage1:Data_Preparation, Stage2:Modeling, Stage3:Evaluation* and *Stage4:Deployment* which are related to *CRISP-DM Stages* by *is-a* relationship as shown in Fig.5.

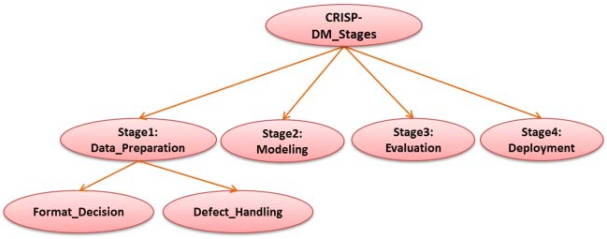

Figure 5. CRISP-DM ontology class

After generating the concepts in ontology, we assign data properties and object properties for the concepts. For example, we assign *hasAverage* and *hasValue* data properties to the concept *Data*. Next, we write Semantic Web Rule Language (SWRL) [6] rules to identify the characteristics of data set. Examples are presence of outlier and missing values using the value of data properties of the dataset. Further, we write SWRL rules to identify the services for the user requirements such as model type. Afterwards, through the rules we identify the relevant abstract services (e.g.,*Check_Reasonable* service for *Outlier*) and generate the workflow.

V. Implementation and Evaluation

We have designed a prototype of the ontology using Protégé [7]. A collection of concepts and hierarchies are generated for problem domain. The highest levels of the class hierarchy are *CRISP-DM stage*, *Data* and *Web services*. Figure 6 shows a segment of the generated ontology. Each class may have sub-classes. For example, *CRISP-DM_stage* class has four sub-classes, namely *Data_Preparation*, *Modeling*, *Evaluation* and *Deployment*.

In the ontology, we have created individuals under the *Data* class for data sets. Data properties and object properties are assigned to the concept to identify the characteristics of data set. Figure 7 shows the object properties and some data properties. For example, we have introduced *hasDataFormat* data property to the *Data* class, to identify the format of data set file. Value of the data property can be a file format such as CSV, XML or HTML.Further, to identify whether data set has any missing value, we have introduced *hasValue* data property. The value of a property can be NULL.

The *generateModel* object property is used to create the relationship between Data Class and sub-class of *Decide_Model_Service* class such as *Clustering_Service*. Workflow generation can be done in two ways: ontology inference (rule based) and query mechanism.

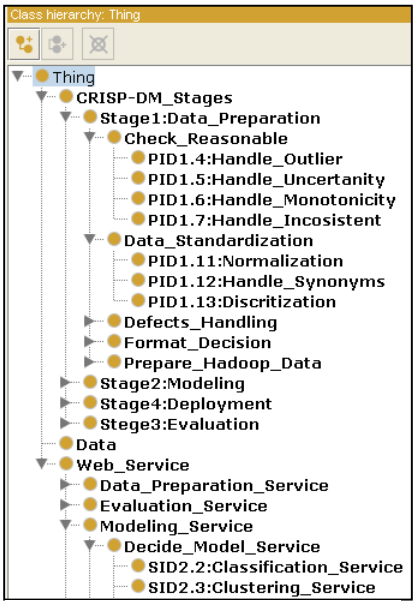

Figure 6. Part of created ontology

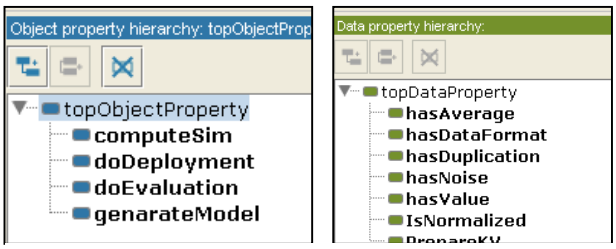

Figure 7. Object properties and data properties of ontology

A. Rule Based WorkFlow Generation

Our objective is to leverage the ontology to help to automatically generate a workflow. The query engines employ the ontology to derive additional knowledge using a deductive inference system [8]. The logical characteristics and inference system in the ontology meet the requirements of relationships among individual properties. OWL (Web Ontology Language) reasoning is mostly related to classes and classification. Thus, we have adopted SWRL that is intended to be the rule language of semantic Web to express the statements that cannot be achieved with OWL [6]. To generate the workflow, according to the user requirements and the properties of the data set, we write SWRL rules. For example, if there is an outlier in data set, we have to handle the outliers. To identify the particular abstract service for this purpose, we used SWRL queries. Table 2 shows sample SWRL queries. For the interest of space, we list limited number of rules.

According to the rule 1, if the file format is not equal to CSV format, then the workflow generator identifies the *Format_Decision_Service* as a member of workflow π. Further, if an average value of a particular field is greater than 80 or lower than 20, it implies some outliers exist. To handle outliers, workflow generator identifies *Check_Reasonable_Service* as a member of π according to the rule 3. Moreover, assume that user needs to generate clustering model. The workflow generator inserts the *Similarity_Computing_service* and *Clustering_Service* into π according to the rule 5 and rule 6. Furthermore, by rule 3 and rule 4, the workflow generator identifies whether *Defects_Handling_Service* and *Data_Standardization_Service* should become members of π. In the same manner, SWRL rules are written to identify the abstract services for the each step of the CRISP-DM process.

Take our use case scenario again as an example. File format is XML. In some data, value is NULL and the average value of a data field is 20. We thus generated an individual for the dataset with name Data_setA. Data properties and object properties are assigned to the individual as shown in Figure8. After reasoning the ontology, the workflow generator will produce an abstract workflow.

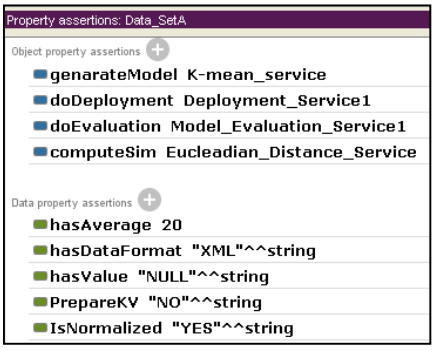

Figure 8. Object properties and data properties of Data_SetA

Figure 9 shows the output after the reasoning process. Figure 10 shows the explanation of reasoning step to identify the outliers and relevant abstract services. Figure 11 shows the explanation of the reasoning step to identify the missing value and particular abstract service.

Figure10 is explained as follows. *Data_SetA has an average value 20.* According to rules, i*f average value is equal or lower than 20, then Data_SetA has outliers. Thus, the data set has outliers. If there are outliers, then relevant abstract service is Check_Reasonable_Service.*

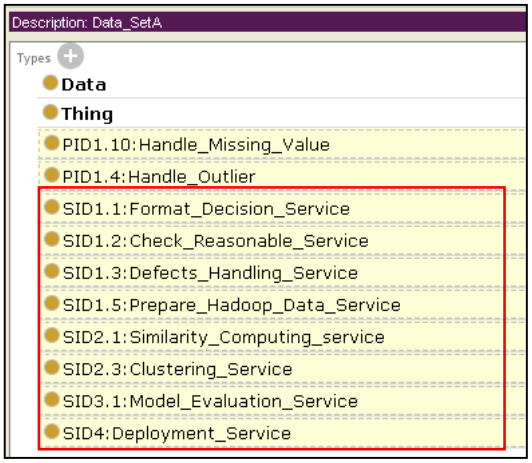

Figure 9. Output after the reasoning for scenario 1

Table 2. SWRL Rules

	Rule	Needed Task	Service
1	Data(?b) , hasDataFormat(?b, ?a) , notEqual(?a, "CSV") -> SID1.1:Format_Decision_Service(?b)	File Format Conversion	Format_Decision_Service
2	Data(?b) , hasValue(?b, ?NULL) -> PID1.10:Handle_Missing_Value(?b) PID1.10:Handle_Missing_Value(?a) -> SID1.3:Defects_Handling_Service(?a)	Handle Missing Value	Defects_Handling_Service
3	Data(?b) , integer[>= "80"^^integer](?value1) , hasAverage(?b, ?value1) -> PID1.4:Handle_Outlier(?b) Data(?b) , integer[<= "20"^^integer](?value1) , hasAverage(?b, ?value1) -> PID1.4:Handle_Outlier(?b) PID1.4:Handle_Outlier(?a) -> SID1.2:Check_Reasonable_Service(?a)	Handle Outlier	Check_Reasonable_Service
4	Data(?a) , IsNormalized(?a, ?b) , notEqual(?b, "YES") -> PID1.11:Normalization(?a) PID1.11:Normalization(?c) -> SID1.4:Data_Standardization_Service(?c)	Need Data Normalization	Data_Standardization_Service
5	Data(?b) , computeSim(?b, Eucleadian_Distance_Service) -> SID2.1:Similarity_Computing_service(?b)	Similarity Computation	Similarity_Computing_service
6	Data(?b) , genarateModel(?b, K-mean_service) -> SID2.3:Clustering_Service(?b)	Generate clustering Model	Clustering_Service

*Fig*ure 11 is explained as follows. *Data_SetA has Null value. If Data_SetA has Null value, then task is Handle_Missing_Value. Finally, if a task is Handle_Missing_Value, then abstract service is Defect_Handling_Service*.

By reasoning the ontology, we can get the result as shown in Figure 9. For the types of *Data_setA*, *Thing* and *Data* are higher level concepts of the *Data_SetA* in the ontology. Further, we can see the task that we need to do according to the property of the dataset such as *Handle_Outlier* and *Handle_Missing_Value*.

Then, generator extracts the abstract services from the list such as *Format_Decision_Service* and *Defects_Handling_Service*. The service name begins with the service ID (SID). Workflow generated by the generator shown in Figure 12.

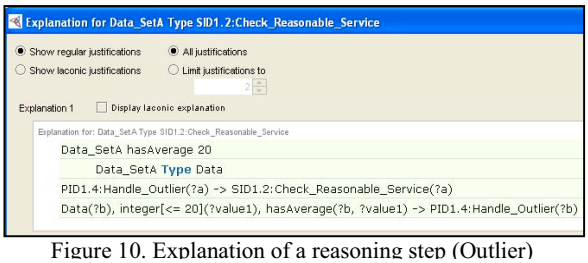

Figure 10. Explanation of a reasoning step (Outlier)

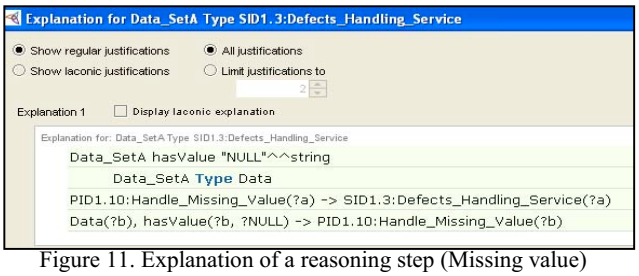

Figure 11. Explanation of a reasoning step (Missing value)

Scenario 2: *XYZ Airline Company plans to study the relationship between temperature and air traffic. Data file is in text format. Average value of a data field is 95. Further, there are some Null values in the dataset. To feed the dataset to Hadoop, they need to convert data set into Hadoop format. To analyze the relationship, they hope to use a correlation function.*

As in scenario 1, we generate an individual of type Data to represent the data file. Then, we assigned data properties and object properties to the individual according to the scenario. By reasoning the ontology, the workflow generator generates the abstract workflow and Figure 13 shows the output after the reasoning.

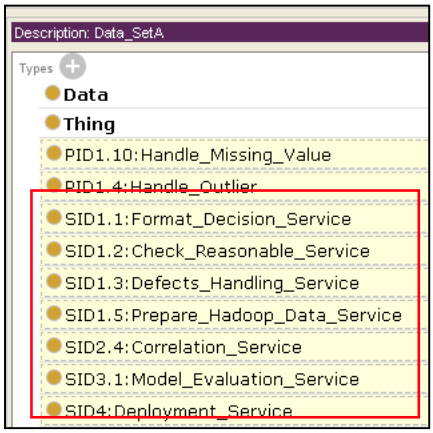

Figure 13. Output after the reasoning for scenario 2

As in the previous results, we can see the relevant abstract services with the SID. Figure 14 shows the abstract workflow.

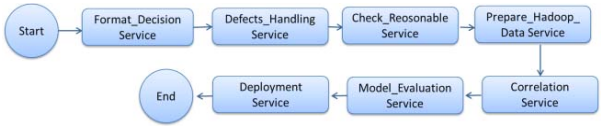

Figure 14. Abstract workflow for scenario 2

B. *Query Mechanism*

In the Protege platform, it is possible to make queries on the ontology once it has been instantiated. Here we show some basic query operations. The queries can be more or less complex. We have tested some typical queries to generate the workflow based on quarries:

Query 1: Data that hasValue NULL
Result :Handle_Missing_Value
Query 2: Service that need for Handle_Missing_Value
Result :Defect_Handling_Service
Query 3: Data that hasAverage 91
Result :Handle_Outlier
Query 4: Service that need for Handle_Outlier
Result :Check_Reasonable_Service

According to query 1, data has NULL value. Then, result is handle missing value task. From query 2, we identified the relevant task for handle missing value.

After running the query, we got *Defect_Handling_Service* as the result. Further, from queries 3 and 4, we could find the *Check_Reasonable_Service* as

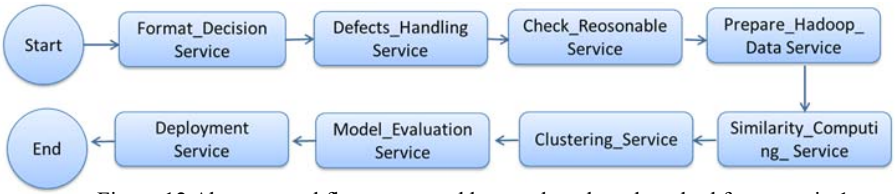

Figure 12.Abstract workflow generated by ontology based method for scenario 1

another workflow according to the property of Data. In this way, using query we can identify relevant abstract services according to the user requirements and data properties.

VI. Related Work

Researchers have reported a number of ways to analyze Big Data to find patterns and relationships, make informed predictions, and gain business insight from the steady influx of information within various business domains [9]. Among them, Wu et al. [10] presented a HACE theorem that characterizes the features of the Big Data revolution, and proposes a Big Data processing model. They have analyzed several challenges at the data, model, and system levels to explore the Big Data. Shang et al. [11] proposed an approach to assist developers of BDA Apps for cloud deployments. They have also proposed a lightweight approach for uncovering differences between pseudo and large-scale cloud deployments.

Dynamic Web composition methods are required to generate the plan automatically. Most workflow generation methods are related to AI planning and deductive theorem proving. AI planning is a well-known area but there are challenges in applying its planning techniques. Medjahedet al. [12] proposed a plan generation method based on high-level declarative description. The approach consisted of four phases: (1) specification phase that enables high-level description of the desired compositions, (2) matchmaking phase that uses rules to generate composition plans according to service requester's specifications, (3) selection phase that selects a plan from proposed collection of plans based on quality of composition parameters, and (4) generation phase. HTN planner is another type of AI planning [5]. Hierarchical planning creates plan by task decomposition. SHOP and SHOP2 (Simple Hierarchical Ordered Planner) [13] are HTN-based planners for composing Web services. Chifu et al. [14] proposed a Web composition based on domain ontology and Fluent Calculus formalism. Their work consists of an OWL to Fluent Calculus translation algorithm and a Fluent Calculus based planning strategy. They translated Web service domain ontology into a Fluent Calculus knowledge base, necessary for the composition planning phase. To find the best composition, quality-based graph planning method is proposed [15]. First, they constructed the composition planning graph and secondly by applying non-functional quality parameters the best composition schema is found by using iterative method.

VII. Conclusions

In this paper, we have presented an approach to automating the CRISP-DM process on Big Data infrastructure, using ASC toward intelligent BDA. Focusing on the planning stage of the ASC, we presented a method of generating abstract workflow of the service composition for automating BDA. We presented an ontology-based workflow generation method. The method leveraged the ontology to help to automatically generate a workflow. Based on ontology designed for CRISP-DM, we have developed two approaches for the workflow generation. First, a rule-based approach is developed for detailed inference. We have implemented SWRL rules to identify the abstract services according to the properties of dataset and user requirements. Second, a query-based approach is developed. We have used two use case scenarios to demonstrate the feasibility of our proposed method. The method able to generate more application-specific abstract workflow.

In our future work, we plan to conduct a thorough evaluation of our ontology-based workflow generation method over existing methods. We also plan to apply the proposed approach to the planning stage of the ASC to further automate entire CRISP-DM process.

References

[1] http://www.gartner.com/newsroom/id/2848718: 2015/02/02.

[2] F. Provost and T. Fawcett, "Data science and its relationship to big data and data-driven decision making", Big Data, vol. 1, no. 1, pp. 51–59, 2013.

[3] ftp://ftp.software.ibm.com/software/analytics/spss/support/Modeler/Documentation/14/UserManual/ CRISP-DM.pdf: Access 2014.09.03.

[4] I. Paik, W. Chen, and M.N. Huhns, "A scalable architecture for automatic service composition," IEEE Transactions on Services Computing, vol. 7, no. 1, pp. 82–95, Jan.–Mar. 2014.

[5] E. Sirin, B. Parsia, D. Wu, J. Hendler, and D. Nau, "HTN Planning for Web Service Composition Using SHOP2," J. Web Semantics, vol. 1, pp. 377-396, 2004.

[6] B. Motik, F. Peter, Patel-Schneider, and B.C. Grau (2009), "SWRL: A semantic web rule language combining OWL and RuleML", [Online] http://www.w3.org/Submission/SWRL/.

[7] http://protege.stanford.edu/download/protege/4.3/installanywhere/Web_Installers/

[8] I. Horrocks, "The Ontology Inference Layer OIL", Department of Computer Science University of Manchester, UK, 2000.

[9] http://www.oracle.com/technetwork/database/options/advanced-analytics/bigdataanalyticswpoaa-1930891.pdf: Access 2015. 02. 13.

[10] X. Wu, X. Zhu, G. Q. Wu, and W. Ding , "Data Mining with Big Data," IEEE Transaction Knowledge and Data Engineering, vol. 26, no. 1, pp. 97–107, 2013.

[11] W. Shang, Z. M. Jiang, H. Hemmati, B. Adams, A. Hassan, and P. Martin, "Assisting developers of big data analytics applications when deploying on hadoop clouds," in Software Engineering (ICSE), 2013 35th International Conference on, pp. 402–411, 2013.

[12] B. Medjahed, A. Bouguettaya, and A.K. Elmagarmid,"Composing Web services on the Semantic Web", The VLDB Journal, vol. 12, no. 4, November 2003.

[13] D. Nau, T. Au, O. Ilghami, U. Kuter, J. Murdock, D. Wu, and F. Yaman. "SHOP2: An HTN planning system", Journal of Artificial Intelligence Research, 20:379–404, 2003.

[14] V. R. Chifu, I. Salomie, I. Harsa, and M. Gherga, "Semantic Web service composition method based on fluent calculus", S.M. Watt, V. Negru, T. Ida, T. Jebelean, D. Petcu, (eds.) SYNASC., IEEE Computer Society, pp. 325–332, 2009.

[15] S.J. Samuel and T. Sasipraba, "An approach for graph based planning and quality driven composition of Web services," Indian Journal of Computer Science and Engineering (IJCSE), vol. 2, no. 5, p672- 679, Oct-Nov 2011.