Towards a Requirements Engineering Artefact Model in the context of Big Data Software Development Projects

Research in Progress

Darlan Arruda and Nazim H. Madhavji
Department of Computer Science
University of Western Ontario
London, Canada
darruda3@uwo.ca, madhavji@gmail.com

Abstract — **There is ample literature that suggests that the field of Big Data is growing rapidly. Also, there is emerging literature on the need to create end-user Big Data applications, as distinct from "data analytics" that typically employs machine learning algorithms to find value in large datasets for the stakeholder. A solid foundation for creating sound applications is a thorough understanding of domain and artefact models that embody artefact types and activities involved in a software project. This paper focuses on the Requirements Engineering (RE) aspect of a Big Data software project. Currently, there are no known RE artefact models to support RE process design and project understanding. To fill this void, this paper proposes a RE artefact model for Big Data end-user applications (BD-REAM). The paper also describes a method for creating the artefact model, including the basic elements and inter-relationships involved in the model.**

Keywords - Big Data; Requirements Engineering; Artefact Model; Big Data Requirements Engineering Artfect Model.

I. INTRODUCTION

A 2016 IDC [9] study predicts that "in aggregate, the Big Data technology and services market is estimated to grow at a CAGR (compound annual growth rate) of 22.6% from 2015 to 2020 and reach $58.9 billion in 2020. Revenue for Big Data software is estimated to grow at a CAGR of 25.7% from 2015 to 2020 and reach $15.9 billion in 2020. Revenue for Big Data services, which consists of professional and support services, is estimated to grow at a CAGR of 23.9% from 2015 to 2020 and reach $15.2 billion in 2020."

While literature analysis and economic outlook suggests that the field of Big Data is growing rapidly, as yet there is no recognisable *body of knowledge* concerning the development of Big Data end-user software applications. This void is reflected in the field of Requirements Engineering (RE) as well. This situation is also reported in the literature on Big Data software engineering [6], [29], which highlights the gaps and challenges in the development of Big Data software applications and the need for processes and methodologies for Big Data software projects.

There is thus a need to better understand the activities, work products, constraints and methods involved in developing Big Data end-user software applications, for example, services operating on Big Data and with which users interact [6]. These applications are quite distinct from analytics software that employs machine learning algorithms [36]. While the idea of engineering end-user Big Data applications first appeared in [6], it is a vision begging for concrete details such as software artefact types and inter-relationships that illuminate that vision for practical use.

In this paper, we attempt to throw some light on different types of artefacts and inter-relationships involved in end-user Big Data projects, with particular focus on Requirements Engineering (RE). This is a foundational phase for every large software project. It deals with what the customer wants, and how the system should behave during usage [11], [18].

The detailed artefacts and inter-relationships are embodied in a model, called *a Requirements Engineering Artefact Model (REAM) in the c*ontext of Big Data software projects (BD-REAM). This type of model can be used as a reference for the design of project-specific processes [1], [12], software maintenance [12], and for supporting project decisions throughout the entire product life-cycle [1]. The research contribution in this paper is that of BD-REAM, which adds to the current, meagre body of knowledge on creating Big Data end-user applications.

The rest of the paper is organised as follows: section 2 presents an overview of the artefact and process modelling. Section 3 describes the development process used to create a BD-REAM. In section 4, we present an illustrative example. Section 5 discusses the limitations of this work. Section 6 summarises this paper and provides brief directions for future work.

II. ARTEFACT AND PROCESS MODELLING

Modelling is a recognised contributory factor in the success of a software organisation [14]. For a given target system, models are an abstract representation where key artefacts are depicted in the model along with inter-relationships with other artefacts [15]. Fernandez et. al., [13] note that modelling software systems during development has become an accepted practice in the software engineering (e.g., Model-driven development [33]). Complementing product modelling, processes can also be modelled where activities are represented in the models along with inputs/outputs of the activities [32]. In RE also modelling is promoted for both

artefacts and activities [1] and benefits of such models include: understanding the application domain, aid to product and process design, process enactment, software maintenance, and others [1], [12], [32].

III. RE Artefact Model Creation

This section describes the 4-step modelling process [12] followed to create the BD-REAM:

(i) Identification of elements and concepts. A Systematic Literature Review (SLR) (*not reported in this paper*) was conducted on Requirements Engineering involving Big Data Applications. In total, 311 papers were identified and, after methodical selection, 13 papers were deemed relevant to be used in our review. In addition to the results of the SLR, we also selected traditional software and RE literature [11], [12], [18]. The selected papers and traditional software and requirements engineering literature were then analysed and the model elements (artefacts) identified, from which a glossary of terms was created (see Table I). The definition of each element was extracted either from the results of our SLR or traditional Requirements and Software Engineering literature.

(ii) Definition of the artefact relationships. Using the glossary of terms and interpreting the domain knowledge from the scientific literature, we created a table of artefacts and their inter-relationships (see Table II).

(iii) Definition of cardinalities. Using the basic table of artefacts and inter-relationships and interpreting the domain knowledge from the scientific literature, a relationship cardinality document was created.

(iv) *Synthesising the artefact model (Figure 1).* Using fragments of artefacts, their inter-relationships, and cardinality information, they were inter-connected iteratively, respecting RE domain knowledge, eventually resulting in the artefact model shown in Figure 1. The proposed Requirements Engineering Artefact Model (BD-REAM) is composed of three basic elements: *Artefact:* a rectangular shape (UML Class) identified with the name of the artefact it represents; *Association*: a line connecting two artefacts. Each association is labelled to indicate the type of relationship between the artefacts; and *Cardinality:* it indicates quantity. If the cardinality is not expressed in the association line, it means that it is has a value of 1.

The following relationships are represented in the model: (i) *Is-derived-from* represents the relationship between the artefacts when from one artefact (e.g., Big Data scenarios) one or more artefacts can be derived and specified (e.g., quality requirements are derived from Big Data scenarios); (ii) *Is-identified-from* represents the relationships when from one artefact (*e.g., organisational goals)* one or more artefacts (e.g., Big Data Scenarios, Constraints and Concerns, etc.) are identified; (iii) *Is-part-of* relationship represents aggregation and it is illustrated when one or more artefacts are part of one or more major artefacts (e.g., functional requirement is part of software requirements); (iv) *Contains* relationship is used when one or more artefacts have or hold information from another artefact within (e.g., software requirements contains analysed requirements); and (v) *Used in* relationship means that one artefact can be used to guide in the definition of other artefacts (e.g., project constraints are used in Big Data scenarios).

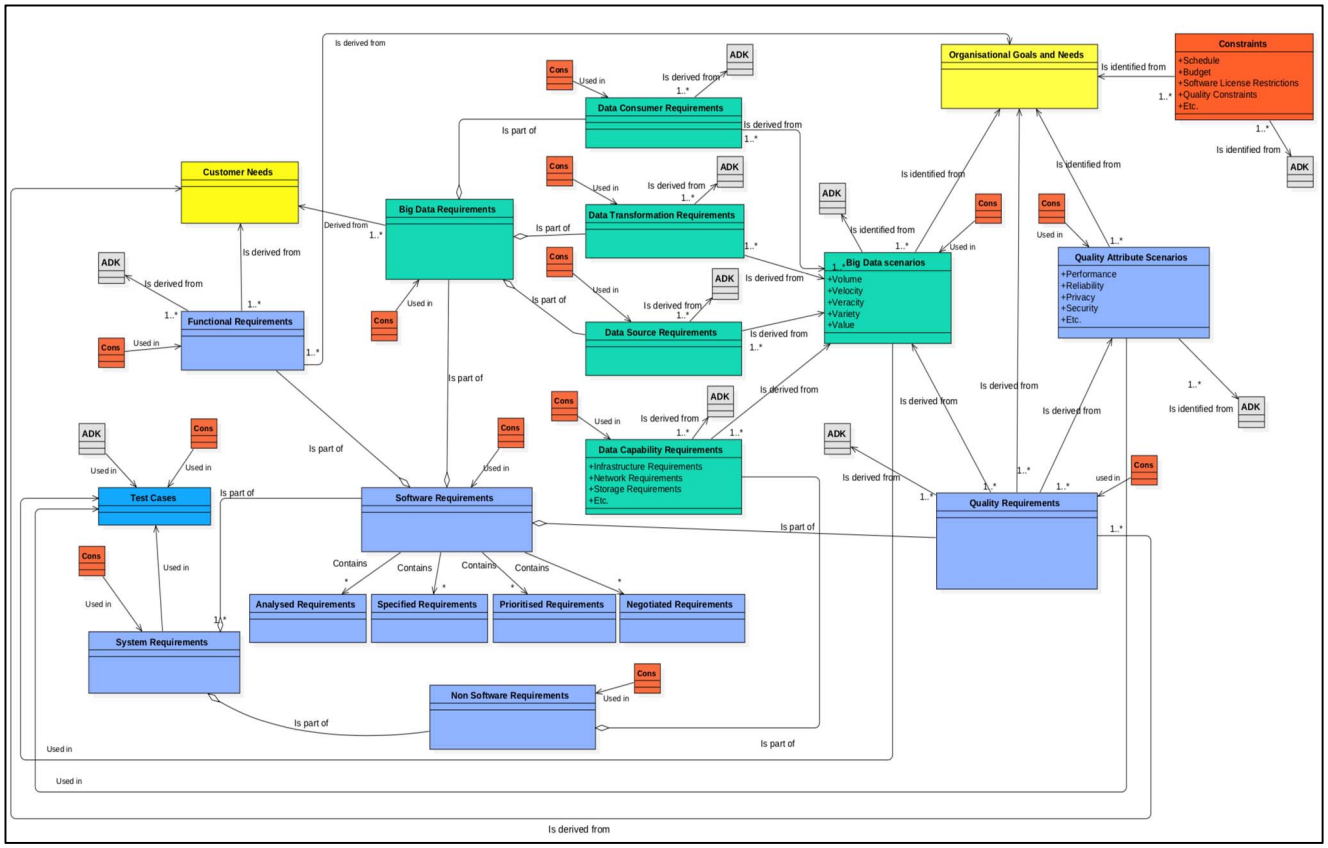
TABLE I. Identified Elements and their Descriptions

Element	Description
Big Data Scenarios	"A scenario is a tool used during requirements analysis to describe a specific use of a proposed system. Scenarios capture the system, as viewed from the outside (e.g., by a user, business, using specific examples)" [24]. Big Data Scenarios are scenarios that incorporate Big Data characteristics in their descriptions (e.g., volume, velocity, variety, veracity, etc.)
Organisational Goals and needs	Describe what a company expects to accomplish over a specific period of time. In software engineering, organisational goals drive the conception, creation, and evolution of software systems [21]. They are associated with the needs of the organization rather than the needs of the customers [12].
Constraints	Are anything that restricts or dictates the actions of the project team [25] (e.g., Scope, schedule, budget, quality, resources, limited software licenses, etc.) [24].
Customer Needs	A customer can be internal or external to the organization [34]. Customers are stakeholders of the project and as such their ideas, needs and whishes are central to the project [35]. Costumer's needs describe their expectations regarding the product to be developed.
Data Capability Requirements	Deal with infrastructure issues such as the need to support legacy and advanced software packages, legacy and advanced computing platforms, data storage and elastic data transmission, hardware, for example [17].
Data Consumer Requirements	Refers to the set of requirements related to the presentation of the processed results of Big Data to the users (e.g., processed results in text, table, visual, and other formats) [19].
Data Source Requirements	Refer to the set of requirements the system should address to support or deal with the different characteristics of the data sources (e.g., data size, file formats, rate of growth, at rest or in motion, etc.) [19].
Data Transformation Requirements	Refer to the type of requirement that relate to data analytics, data fusion and data processing requirements.
Domain Knowledge	Valid knowledge used to refer to an area of human endeavour, an autonomous computer activity, or other specialized discipline [23].
Functional Requirements	Specifies a function that a system or system component must be able to perform [20]. Describes what the system should do, how the system should react to particular inputs, and how the system should behave in particular situations [18].

Non-Software Requirements,	In the context of this paper, represent all requirements that are not related to the software itself (e.g., infrastructure requirements, project requirements, etc.).
Quality Attributes Scenarios	Scenarios that describe the usage of the software with respect to the quality attributes it might address (e.g., performance, privacy, etc.). A quality attribute scenario helps to derive quality-attribute-specific requirements applicable to the system [27].
Quality Requirements	State additional quality-related properties (e.g., performance, security, etc.) that the functional effects of the software should have [15, p.24].
Software Requirements	Prescriptive statement to be enforced by the software to be developed and formulated in terms of phenomena shared between the software and the environment [15, p. 19].
Systems Requirements	Include all the requirements necessary for build the whole system and that includes hardware requirements, infrastructure requirements, project requirements, software requirements, for example.
Test Case	Consists of a set of conditions or actions which are performed on the software application in order to verify the expected functionality of the feature [28]. Test cases are used to validate the implemented requirements.

TABLE II. ARTEFACTS AND THEIR RELATIONSHIPS

Artefact/Cardinality	**Relationship**	**Artefact/Cardinality**
Project Constraints	*Is Identified from*	Organisational Goals and needs
Project Constraints	*Is Identified from*	Application Domain Knowledge
Big Data Scenarios	*Is Identified from*	Application Domain Knowledge
Big Data Scenarios	*Is Identified from*	Organisational Goals and Needs
Quality Attribute Scenarios	*Is Identified from*	Organisational Goals and Needs
Quality Attribute Scenarios	*Is Identified from*	Application Domain Knowledge
Quality Requirements	*Is Derived from*	Application Domain Knowledge
Quality Requirements	*Is Derived from*	Quality Attribute Scenarios
Quality Requirements	*Is Derived from*	Big Data Scenarios
Quality Requirements	*Is Derived from*	Organisational Goals and Needs
Functional Requirements	*Is Derived from*	Application Domain Knowledge
Functional Requirements	*Is Derived from*	Use Cases
Data Transformation Requirements	*Is Derived from*	Application Domain Knowledge
Data Transformation Requirements	*Is Derived from*	Big Data Scenarios
Data Source Requirements	*Is Derived from*	Application Domain Knowledge
Data Source Requirements	*Is Derived from*	Big Data Scenarios
Data Capability Requirements	*Is Derived from*	Application Domain Knowledge
Data Capability Requirements	*Is Derived from*	Big Data Scenarios
Data Consumer Requirements	*Is Derived from*	Big Data Scenarios
Data Consumer Requirements	*Is Derived from*	Application Domain Knowledge
Functional Requirements	*Is part of*	Software Requirements
Quality Requirements	*Is part of*	Software Requirements
Data Transformation Requirements	*Is part of*	Big Data Requirements
Data Source Requirements	*Is part of*	Big Data Requirements
Data Consumer Requirements	*Is part of*	Big Data Requirements
Data Capability Requirements	*Is part of*	Non-Software Requirements
Big Data Requirements	*Is part of*	Software Requirements
Software Requirements	*Is part of*	Systems Requirements
Non-Software Requirements	*Is part of*	Systems Requirements
Software Requirements	*Contains*	Analysed Requirements
Software Requirements	*Contains*	Specified Requirements
Software Requirements	*Contains*	Prioritized Requirements
Software Requirements	*Contains*	Negotiated Requirements
Project Constraints	*Used in*	All artefacts in the RE process
Big Data Scenarios	*Used in*	Test Cases
Quality Attribute Scenarios	*Used in*	Test Cases
Systems Requirements	*Used in*	Test Cases



Legend: ADK - Application Domain-knowledge | Cons - Constraints | ⟶ Association line | ⟷ Aggregation Line | Artefact

Figure 1. Big Data Requirements Engineering Artefact Model (BD-REAM) depicting Elements, Relationships and Cardinalities.

IV. ILLUSTRATIVE EXAMPLE

In this section, we present an illustrative scenario aligned with the proposed BD-REAM. In this scenario, we also provide some examples of the main elements discussed in this paper (see Table I).

A financial services company has a critical data processing task that requires joining several large data sets of approx. 6TB (Volume) each with different formats of data (Variety) in a daily basis (Velocity). However, the company's current legacy proprietary integration platform does not support the data integration for heterogeneous datasets. Thus, there is a need to build a Big Data solution to support the task to capture, store, integrate and process the collected data. The solution would enable the data integration task of from disparate sources to be completed within one day, speeding up each iteration and providing the data to the consumer within time constraints.

The development of the envisaged application starts with the identification and definition of the system's requirements (like in any software engineering project). However, in the context of Big Data, some Big Data specific requirements must be taken into consideration while eliciting, specifying and documenting these requirements. The requirements engineer and the business analyst then start to gather the necessary information (through interviews and other means from the stakeholders of the project) which is then formulated as requirements. While interviewing the stakeholders, the organisational goals, objective and needs are identified.

Example of an Organisational Goal [21]

"beat competitor's product that has a transaction time of 0.25 seconds"

The needs of the users of the system as well as the projects constraints are also identified. Based on the organisational goals, objectives, users' needs and project constraints – System quality attribute scenarios are identified. In order to make the content of this model easier to read and understand, we created a conceptual separation of the "content" of these system quality attribute scenarios into: (i) Big Data and (ii) Quality Attribute Scenarios. The Big Data scenarios focus on the usage of the system taking into consideration the data characteristics (such as volume, velocity, variety, etc.). The quality attribute scenarios describe system usage scenarios with focus on the traditional quality attributes (such as reliability, privacy, security, etc.). Examples of these two are presented below:

Example of a Quality Attribute Scenario [30]

"Users initiate 1,000 transactions per minute stochastically under normal operations, and these transactions are processed with an average latency of two seconds (performance attribute)"

Example of a Big Data Scenario [31]

"Automate the process of measuring the value of potential acquisitions. This comprehensive analysis needed to include public and private information, including patents and trademarks, company press releases, annual and quarterly reports, corporate genealogies, and IP ownership and patents ranked by citation (variety)"

Then, the identified scenarios can be used as input for the formulation of the systems requirements, both functional and non-functional, as well as the Big Data specific requirements such as data capability, data source, data consumer and data transformation type of requirements. As shown in our model some of these requirements are not considered software requirements (e.g., infrastructure). Examples of data specific requirements are presented below:

Example of a Data Capability Requirement [19]

"System needs to support legacy and advanced distributed computing clusters, coprocessors, input output (I/O) processing (infrastructure)"

Example of a Data Source Requirement [19]

"System needs to support reliable real-time, asynchronous, streaming, and batch processing to collect data from centralized, distributed, and cloud data sources, sensors, or instruments"

Example of a Data Transformation Requirement [19]

"System needs to support batch and real-time processing and needs to support processing large diversified data content and modelling"

Example of a Data Consumer Requirement [19]

"System needs to support diversified output file formats for visualization, rendering, and reporting"

For simplicity - as this is an artefact oriented model – not a process model - we present the elicited, analysed, negotiated and prioritized requirements under the software requirements artefact class. The application domain knowledge (ADK) and the identified project constraints are used to define the scope of the scenarios, and therefore the requirements.

V. LIMITATIONS

The construction of a RE artefact model itself is not difficult [12]. However, one must be wary about the elements and inter-relationships of interest in the target domain (e.g., artefacts and relationships involved in Big Data centric requirements for end-user applications) – in the context of development, maintenance, evolution and use of the system of interest. Since Big Data Software Engineering is a relatively new topic, and much of the focus in scientific literature on Big Data analytics, it is possible that some important elements and relationships may have been missed while creating this model. Also, not all elements and inter-relationships identified in this study had clear-cut information. Thus, we interpreted some inter-relationships based on our previous experience in software development projects. We anticipate that with empirical validation of the proposed model, its quality should improve over time.

VI. SUMMARY AND FUTURE WORK

Software engineering of Big Data end-user applications is relatively new. Currently, to our knowledge, there is no domain or artefact model for Requirements Engineering (RE). This paper thus aims to fill that gap though the artefact model depicted in Figure 1 is still preliminary and pre-empirical-validation. The model has been created through systematic literature review (SLR).

One observation from the SLR results is that there isn't much research addressing the RE aspects of Big Data. While this necessarily limits the richness of the artefact model we have created, the resultant model does provide a preliminary baseline to the research community for further research in this field and to the practicing community an understanding of the key artefacts and relationships in the model which can be used in the design of RE projects. The current version of the Big Data RE Artefact Model depicted in Figure 1 is composed of 22 artefacts and numerous inter-relationships (see Tables I and II).

For further work we highlight the following: (i) validation of the model through empirical studies and evaluation; (ii) definition of RE life-cycle processes in the context of Big Data projects.

ACKNOWLEDGMENT

This research is in part supported by grants from CNPq, The National Council of Technological and Scientific Development – Brazil and NSERC, Natural Science and Engineering Research Council of Canada. We thank the reviewers for providing helpful feedback that has improved this paper.

REFERENCES

[1] B. Penzenstadler, D. M. Fernandez, and J. Eckhardt, "Understanding the impact of artefact-based RE - Design of a replication study," *Int. Symp. Empir. Softw. Eng. Meas.*, no. October, pp. 267–270, 2013.

[2] D. Méndez Fernández and B. Penzenstadler, "Artefact-based requirements engineering: the AMDiRE approach," Requir. Eng., vol. 20, no. 4, pp. 405–434, 2015.

[3] Capgemini, "The deciding factor: Big Data & decision making," Capgemini Ecomomist Web, 2012.

[4] Economist Intelligence Unit, "The evolving role of data in decision making," 2013.

[5] H.-M. Chen, R. Kazman, S. Haziyev, and O. Hrytsay, "Big Data System Development: An Embedded Case Study with a Global Outsourcing Firm," 2015 IEEE/ACM 1st Int. Work. Big Data Softw. Eng., pp. 44–50, 2015.

[6] N. H. Madhavji, A. Miranskyy, and K. Kontogiannis, "Big Picture of Big Data Software Engineering: With Example Research Challenges," Proc. - 1st Int. Work. Big Data Softw. Eng. BIGDSE 2015, pp. 11–14, 2015.

[7] I. Gorton, "Software Architecture for Big Data Systems," 2016.

[8] M. Chen, S. Mao, and Y. Liu, "Big Data: A survey," Mob. Networks Appl., vol. 19, no. 2, pp. 171–209, 2014.

[9] Nadkarni, ashish; Vesset, Dan. Worldwide Big Data Technology and Services Forecast, 2016–2020. Available at https://www.idc.com/getdoc.jsp?containerId=US40803116. Last Access: Nov 20th, 2017.

[10] K. M. Anderson, "Embrace the Challenges: Software Engineering in a Big Data World," Proc. - 1st Int. Work. Big Data Softw. Eng. BIGDSE 2015, pp. 19–25, 2015.

[11] Pressman, Roger S. Software engineering: a practitioner's approach. 5th edition. McGraw-Hill series in computer science, 2001.

[12] Berenbach,B.D. J. Paulish, J. Kazmeier, and A. Rudorfer. 2009. Software & Systems Requirements Engineering: In Practice. McGraw-Hill, New York.

[13] D. Mendez Fernandez, B. Penzenstadler, M. Kuhrmann, and M. Broy. A Meta Model for Artefact-Orientation: Fundamentals and Lessons Learned in Requirements Engineering. In D. Petriu, N. Rouquette, and O. Haugen, editors, *MoDELS'10*, volume 6395, pp. 183–197. Springer, 2010.

[14] G. booch, J. Rumbaugh, I. Jacobson. The Unified modelling language User guide. 2nd edition. Addison-Wesley Professional, 2005.

[15] A.V. Lamsweerde. Requirements Engineering: from system goals to UML models to software specifications. Wiley. 2009.

[16] A. Mockus, "Engineering Big Data solutions," Proc. Futur. Softw. Eng. - FOSE 2014, pp. 85–99, 2014.

[17] G. Fox and W. Chang, "Big Data Use Cases and Requirements," 1st Big Data Interoperability Framew. Work. Build. Robust Big Data Ecosyst. ISO/IEC JTC 1 Study Gr. Big Data , 2014.

[18] Sommerville, Ian. Software Engineering. 9th edition. Pearson, Boston, Massachusetts, 2009.

[19] NIST Big Data Public Working Group: Use Cases and Requirements Subgroup, "NIST Big Data Interoperability Framework: Volume 3, Use Cases and General Requirements," vol. 3, p. 260, 2015.

[20] A. Geraci, F. Katki, L. McMonegal, B. Meyer, and H. Porteous, "IEEE Standard Computer Dictionary. A Compilation of IEEE Standard Computer Glossaries," *IEEE Std 610*. p. 1, 1991.

[21] P. Clements and L. Bass, "Relating Business Goals to Architecturally Significant Requirements for Software Systems," *Tech. Note - C.*, no. May, p. 72, 2010.

[22] Dictionary.com "business plan," in Collins English Dictionary - Complete & Unabridged 10th Edition. Source location: HarperCollins Publishers. http://www.dictionary.com/browse/business-plan. Available: http://www.dictionary.com/. Last Accessed Oct 9th, 2017.

[23] Fujita, H., A. Selamat, and S. Omatu, eds. New Trends in Intelligent Software Methodologies, Tools and Techniques: Proceedings of the 16th International Conference SoMeT_17. Vol. 297. IOS Press, 2017.

[24] Siegelaub, J. M. (2007). Six (yes six!) constraints: an enhanced model for project control. Paper presented at PMI® Global Congress 2007—North America, Atlanta, GA. Newtown Square, PA: Project Management Institute.

[25] Project Management tips: Guidance for real life situations. Available at: <http://pmtips.net/blog-new/defining-project-constraints>. Last access: Oct 9th, 2017.

[26] Arms, William. Scenarios and Use Cases. Lecture Notes. Cornell University. Available at: https://www.cs.cornell.edu/courses/cs5150/2014fa/slides/D2-use-cases.pdf. Last Access: Oct 9th, 2017.

[27] L. Bass, P. Clements, and R. Kazman. *Software Architecture in Practice* (3rd ed.). Addison-Wesley Professional, 2013.

[28] International Software Testing Qualifications Board. What is a test case?. Available at http://istqbexamcertification.com/test-case/ Last access: Oct 9th, 2017.

[29] V. Dipti Kumar and P. Alencar, "Software engineering for big data projects: Domains, methodologies and gaps," *Proc. - 2016 IEEE Int. Conf. Big Data, Big Data 2016*, pp. 2886–2895, 2016.

[30] Quality attribute scenario. Available at http://etutorials.org/Programming/Software+architecture+in+practice,+second+edition/Part+Two+Creating+an+Architecture/Chapter+4.+Understanding+Quality+Attributes/4.4+Quality+Attribute+Scenarios+in+Practice/. Last Access Nov 20th, 2017.

[31] IBM. Research and business development. Available at https://www.ibm.com/support/knowledgecenter/SSPT3X_4.1.0/com.ibm.swg.im.infosphere.biginsights.product.doc/doc/bi_research_develop.html. Last Access Nov 20th, 2017.

[32] Humphrey, W.S. and M. Kellner, Software Process Modeling: Principles of Entity Process Models, Proc. 11th. Intern. Conf. Software Engineering, IEEE Computer Society, Pittsburgh, PA, 331-342, 1989.

[33] B. Selic, "The pragmatics of model-driven development," *IEEE Softw.*, vol. 20, no. 5, pp. 19–25, 2003.

[34] IEEE. IEEE Standard Glossary of Software Engineering Terminology, vol. 121990, no. 1. 1990.

[35] S. M. Wundenberg and M. Bächle, Requirement engineering for knowledge-intensive processes: Reference architecture for the selection of a learning management system. 2015.

[36] [C. E. Otero and A. Peter, "Research directions for engineering big data analytics software," *IEEE Intell. Syst.*, vol. 30, no. 1, pp. 13–19, 2015.