BiDaML: A Suite of Visual Languages for Supporting End-user Data Analytics

Hourieh Khalajzadeh[1], Mohamed Abdelrazek[2], John Grundy[1], John Hosking[3], Qiang He[4]
[1]Faculty of Information Technology, Monash University, Australia
[2]School of Information Technology, Deakin University, Australia
[3]Faculty of Science, University of Auckland, New Zealand
[4]School of Software and Electrical Engineering, Swinburne University of Technology, Australia
{hourieh.khalajzadeh, john.grundy}@monash.edu, mohamed.abdelrazek@deakin.edu.au,
j.hosking@auckland.ac.nz, qhe@swin.edu.au

***Abstract*—We introduce Big Data Analytics Modeling Languages (BiDaML), a novel integrated suite of visual languages aimed at supporting end users during the process of designing big data analytics solutions. BiDaML comprises five diagrammatic notations: a data analytics brainstorming diagram; a process diagram; technique diagrams; data diagrams; and output diagrams. Tool support in the form of an integrated design environment for creating BiDaML diagrams has also been developed. To demonstrate the utility of BiDaML, we illustrate our approach with a real-world example of property price prediction and evaluate BiDaML using the physics of notations and cognitive walkthroughs with target end users - data scientists and software engineers.**

Keywords-big data analytics; big data modeling; big data toolkits; domain specific visual languages; end user tools

I. INTRODUCTION

Using big data to improve decision-making through data analytics is an active area [1, 2]. Traditionally, advanced ML knowledge and experience of complex data science toolsets were required to use data analytics. Emerging analytics approaches seek to automate many of these steps, making ML technology more accessible to those who lack deep quantitative analysis and tool building skills [3].

Several data analytics and ML tools including Azure ML Studio, Amazon AWS ML, Google Cloud ML, and BigML have become popular. Many do not require programming language knowledge and are based on using simple drag and drop interfaces. However, they mostly focus on ML algorithms and simple deployment, but lack domain knowledge and business problem capture, modeling, solution traceability and solution validation. They also lack explanation of the model from an end-user perspective.

To unleash value, data analytics and ML steps need to be tightly connected to business and requirements engineering processes. Current data analytics tools rarely focus on capturing and modeling the end-to-end data analytics development lifecycle and the improvement of end-to-end processes [4]. This causes a lack of traceability and poor communication between stakeholders involved in the solution development. To address this, better integration of data science, data technology and process science has been advocated [4]. Such tools would be useful for end-users and data scientists to model the problem, extract insights/patterns, and develop predictive and clustering models before involving software engineers.

We present BiDaML, a set of domain specific visual models at different levels of abstraction, to capture and refine requirements and specify different parts of the data analytics process. Through these domain specific visual languages (DSVLs), we aim to make data analytics design more accessible to end users and facilitate dialogue with expert data scientists and software engineers. BiDaML and its integrated design environment, provide better tool support and collaboration between different users while improving the speed of implementing data analytics solutions. We do not aim to replace common big data analysis tools, such as Azure ML Studio, Amazon AWS ML, Google Cloud ML, etc. Our goal is to augment these platforms to provide an end-to-end specification of what needs to be developed, the tasks to be completed, and techniques used.

II. MOTIVATING EXAMPLE

A. Data Analytics Process Steps

Business owners, business analysts, data analysts and data scientists have regular meetings and interviews with domain experts and users, acquire datasets from different resources, get access to government information, integrate data items with different formats, analyze and visualize them, communicate with each other to discuss the analyses and finally use different tools to design their approaches and develop their models. However, there is no unified language that facilitates communication between them, and they need to wait until their models are usable and deployable. We use a property price prediction example to discuss some of the problems they face during the solution design process.

B. Example: Property Price Prediction

Consider a property price prediction problem, where, a real estate agency wants to improve agents' focus and outcomes by looking for patterns in a large amount of real estate, government and financial data. To solve this problem using conventional approaches, end users need to have a basic knowledge of a data analytics programming languages such as Python and R and basic knowledge of data science. The company employs a team of software engineers and data scientists to work on the problem. The end user chooses features based on their quality, applies data type casting and data cleansing, and filters the features to build the dataset as an input for the prediction model. Then the user chooses the best model, algorithm, and validation method and adapts the characteristics based on problem requirements. Users

978-1-7281-2772-9/19/$31.00 ©2019 IEEE
DOI 10.1109/BigDataCongress.2019.00025

initially apply low-level ML operations on widely-used datasets such as the Ames Housing dataset [5] using ML tools such as Azure ML Studio. Users need data science knowledge to choose modules and change properties. If a user wants to use a preprocessing method or apply a model which is not in the list of modules, knowledge of Python or R is required to embed custom code.

However, the company realizes the team lacks sufficient understanding of domain knowledge. The company appoints a senior real estate agent, highly experienced with domain nuances, to help the team build the analytics solution. The team struggles with a lack of a common dialect between engineers, scientists and domain experts. Eventually, the team realizes that the solution is not ready due to issues in the available dataset that need to be rectified. It takes a long time for the team to design and build a working solution. A few months later, after disbanding the technical team, the company wished to add new features e.g. to build a new model to predict who would be willing to sell their property in a given suburb. The company also noticed a degradation in the performance of the property price prediction model, which needed updating. The company recruits a new team which struggles to develop models for the new capability, due to the lack of business knowledge, and to update the existing model and requirements to integrate both.

C. Key Challenges

A common challenge in using current ML tools is that there is no trace back to the business needs/requirements that triggered the project. Also, communicating and reusing existing big data analytics information and models is a challenge for many companies new to data analytics. Users need to be able to collaborate with each other through different views and aspects of the problem and possible solutions. Current practices and tools do not cover most activities of data analytics analysis and design, especially critical business requirements. Current tools focus on low-level data analytics process design, coding and basic results visualization. They mostly assume data is in a form amendable to processing. In reality, data items are in different formats and are not clean or integrated. Great effort is needed to source data, integrate, harmonize, pre-process and cleanse it. Moreover, few ML tools offer the ability for the data science expert to embed new code and expand algorithms and provide visualizations for their needs.

Data processing and ML tasks are only a small component in the building blocks necessary to build real-world deployable data analytics systems [6], as shown in Figure 1. Business/management modeling tools usually do not support many key data analytics steps including data pre-processing and ML steps. There is a need to capture high-level goals and requirements for different users such as domain expert, business analyst, data analyst, data scientist, software engineer, and customers and relate them to low-level diagrams, and capture details such as different tasks for different users, requirements, objectives, etc. Finally, most of the ML tools require data science and programming knowledge to embed code and change features based on user requirements.

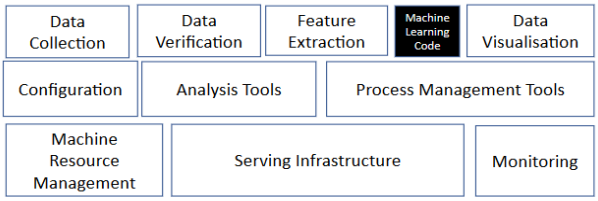

Figure 1. Data Analytics Steps (adapted from [6])

III. Our Approach

We present BiDaML, a set of domain-specific visual languages using five diagram types at different levels of abstraction to support key aspects of big data analytics:

- ***Brainstorming diagram*** provides an overview of a data analytics project and all the tasks and sub-tasks involved in designing the solution at a very high level;
- ***Process diagram*** specifies the analytics processes/steps including key details related to the participants (individuals and organizations), operations, and data items in a data analytics project capturing details from a high-level to a lower-level;
- ***Technique diagrams*** show the step by step procedures and processes for each sub-task in the brainstorming at a low level of abstraction;
- ***Data diagrams*** document the data and artifacts that are produced in each of the above diagrams in a low level, i.e. technical AI based layer;
- ***Output diagrams*** define in detail the outputs associated with different tasks e.g. output information, reports, results, visualizations, outcomes, etc.

Figure 2 shows how our diagrams are connected to each other from a high level to a low level. A brainstorming diagram is defined for every data analytics project. Then, at a lower level to include more details and involve the participants, we use a process diagram. Every operation in a brainstorming diagram can be further extended by technique and data diagrams, and finally, data and technique diagrams are connected to a result output diagram.

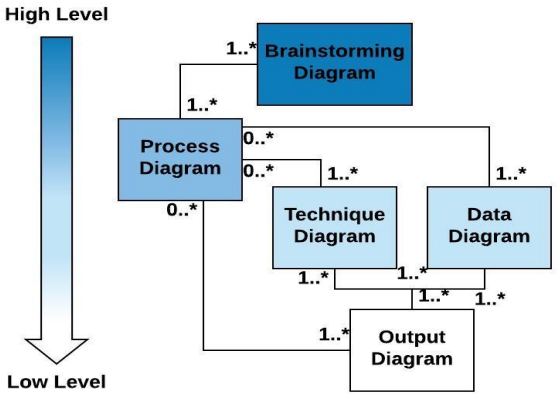

Figure 2. BiDaML Overview.

1) Brainstorming Diagram

A brainstorming diagram covers the entirety of a data analytics project expressed at a high-level. It supports interactive brainstorming to identify key aspects of a data analytics project such as its requirements implications, analytical methodologies and specific tasks. Figure 3(a) shows the visual notation used. It comprises an icon representing the data analytics problem, tasks which the problem is associated with, a hierarchy of sub-tasks for each task, and finally the specific information used or produced.

We group the building blocks of an AI-powered system into four groups: Domain and business-related activities (BusinessOps); data-related activities (DataOps); AI and ML-related activities (AIOps); and development and deployment activities (DevOps). BusinessOps covers domain and business knowledge, requirement gathering, modeling and analysis. DataOps includes data collection/ingestion, validation, cleansing, wrangling, filtering, union, merge, etc. AIOps covers feature and model engineering, model training and tuning. Finally, DevOps covers model integration and deployment, monitoring and serving infrastructure. Figure 3(b) depicts a high-level brainstorming diagram for our property price prediction example in Section 2.

2) Process Diagram

The key business processes in a data analytics application are shown in a process diagram, whose basic notation is shown in Figure 4(a). We adapt the Business Process Modeling Notation (BPMN) [7] to specify big data analytics processes at several levels of abstraction. Process diagrams support business process management, for both technical users such as data analysts, data scientists, and software engineers as well as not-technical users such as domain experts, business and users, by providing a notation intuitive to business users, yet able to represent complex process semantics.

In this diagram type, we use different "pools" for different organizations and different "swim lanes" for the people involved in the process within the same organization. Different layers are also defined based on different tasks such as business-related tasks (BusinessOps), technical (DataOps and AIOps), and operational tasks (DevOps and application-based tasks). Data and artifacts produced and used in each step can be shown as icons specific to the source and type of data. Preparation of data items or different events trigger other events and redirect the process to the other users in the same or different pool. Particular detailed activities or tasks performed by different users and the order of them are represented using rectangles and arrows. Diamonds show different decision points that can adjust the path based on conditions and double circles show unexpected events that can change the process at any step. A high-level process diagram for our property price prediction example is shown in Figure 4(b).

3) Technique Diagram

Technique diagrams extend the brainstorming diagram to low-level detail specific to different big data analytics tasks and sub-tasks. For every sub-task, the process is broken down to the specific stages and the technique used to solve a specific sub-task specified.

4) Data Diagram

In data diagram, data and artifacts related to all tasks and sub-tasks in brainstorming and process diagrams are connected to different data entities. Different data items, features, outliers, the algorithms used, parameters related to these algorithms, model created based on different algorithms, and the evaluation metrics used for the model are captured for the AIOps entity. To document the data and artifacts consumed and produced in different phases, a low-level data diagram is presented. Data diagrams support the design of data and artifacts collection processes. They represent the structured and semi-structured data items involved in the data analytics project in different steps.

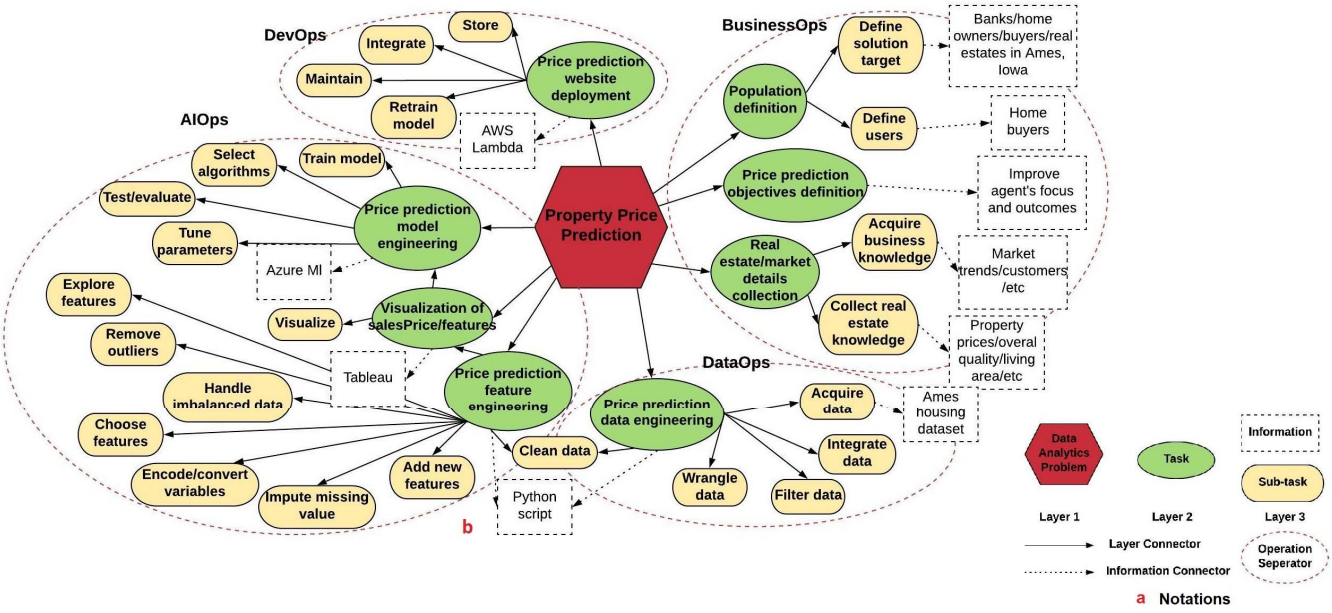

Figure 3. A brainstorming Diagram Example for the Property Price Problem

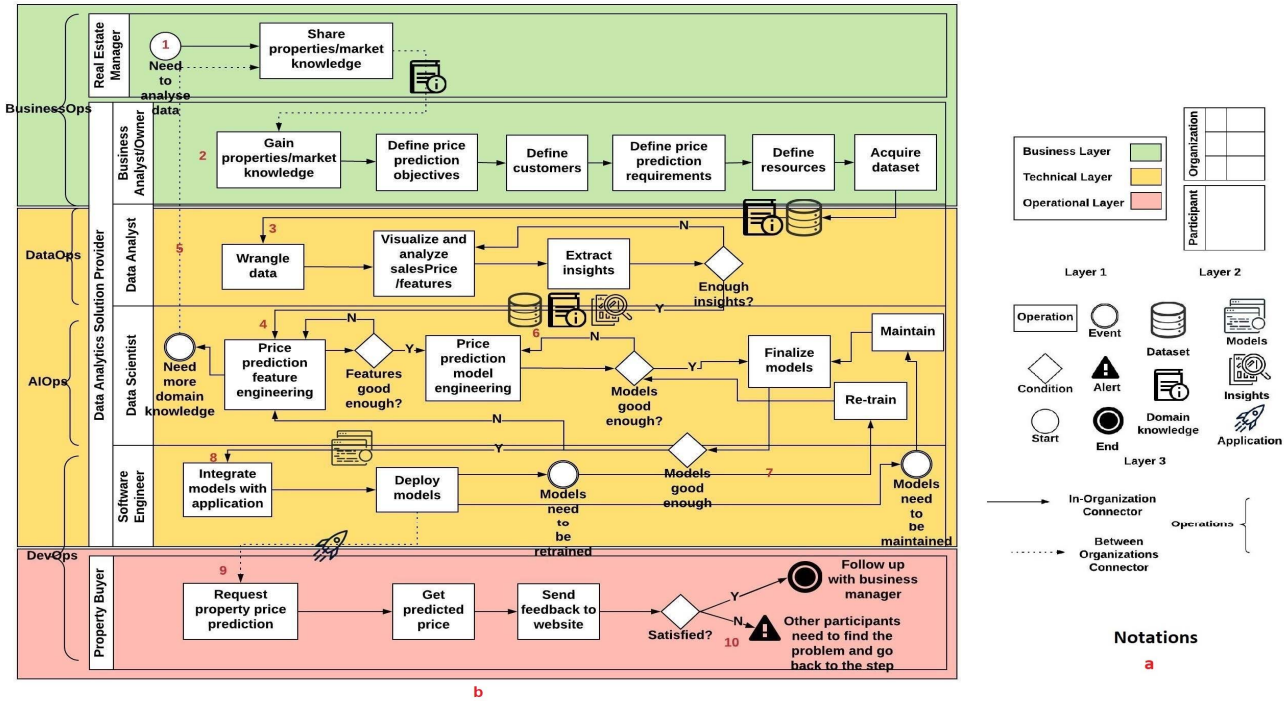

Figure 4. A Process Diagram Example for the Property Price Problem

5) Output Diagram

Finally, output diagrams specify an individual technique in more detail. This diagram type reuses and merges technique and data diagrams and adds information on the technique (logic entity) and the data produced by it (as output ports). This diagram type shows the outputs and reports that can be extracted using current techniques and data items, such as predicted price versus sale price report, and accuracy as well as analysis of property prices in different areas based on nearby schools, public transport, and shopping centers.

We have developed an integrated design environment for creating BiDaML diagrams. BiDaML tool support aims to provide a platform for efficiently producing BiDaML visual models and to facilitate their creation, display, editing, storage, code generation and integration with other tools.

IV. EVALUATION

We have evaluated BiDaML in terms of its usability and suitability using two evaluations. The first was an extensive physics of notations evaluation [8]. This was a useful end user perspective evaluation without having to involve a large-scale usability trial. To understand how easy BiDaML diagrams are to learn and use, we also conducted a cognitive walkthrough using several target end-users.

A. Physics of notations evaluation

Semiotic clarity: all visual symbols in BiDaML have 1:1 correspondence to their referred concepts; ***Perceptual discriminability***: all symbols in BiDaML use different shapes as their main visual variable, plus redundant coding such as color and/or textual annotation; ***Semantic transparency***: icons are used to represent visual symbols and minimize the use of abstract geometrical shapes; ***Complexity management***: we used hierarchical views for representation in BiDaML and as our future work, we will hide visual construct details for complex diagrams; ***Cognitive integration***: all the diagrams in BiDaML have a hierarchical tree-based structure relationship; ***Visual expressiveness***: various visual variables, such as shape, color, orientation, texture, etc are used in designing BiDaML visual symbols; ***Dual coding***: all visual symbols have a textual annotation; ***Graphic economy***: as few visual symbols as possible are used in BiDaML; ***Cognitive fit***: all the symbols in BiDaML are usable for different users and tasks.

B. Cognitive walkthrough

We asked three data scientists and two software engineers (all experienced in big data analytics tasks) to carry out a task-based end-user evaluation of BiDaML. The objective was to assess how easy it is to learn to use BiDaML and how efficiently it can solve the diagram complexity problem. BiDaML diagrams were briefly introduced to the participants who were then asked to perform three predefined modeling tasks. The first was to design one part of a brainstorming diagram for a data analytics problem of their choice. In the second, they were given a process diagram and asked to explain, comment and provide suggestions to improve it. The third involved subjects designing a technique diagram related to a specific task of the data analytics problem they chose for the first part of the evaluation.

Overall, user feedback indicated that BiDaML is straightforward to use and understand. Users felt they could easily communicate with other team members and managers

and present their ideas, techniques, expected outcomes and progress in a common language before the final solution is ready. They liked how different layers and operations are differentiated. They could capture and understand business requirements and expectations and make agreements on requirements, outcomes and results through the project. Using this feedback we have made some minor changes to our diagrams such as the shape and order of some notations, and the relationships between different objects.

However, several limitations and potential improvements have also been identified in our evaluations. Some users prefer to see technique and data diagrams components together in a single diagram, while some others prefer to have these separate. Moreover, in the process diagram, some users prefer to only see the operations related to their tasks and directly related tasks. Finally, one of the users wanted to differentiate between tasks/operations that are done by human versus by a tool. In future tool updates we will provide different views for different users and will allow users to hide/unhide different components based on their preference.

V. Related Work

There are many data analytics tools, such as Azure ML Studio, Amazon AWS ML, and Google Cloud ML, as reviewed in [9]. However, these tools only cover a few phases of DataOps, AIOps, and DevOps and none cover business problem description, requirements analysis and design. Some DSVLs have been developed for supporting enterprise service modeling and generation using end user-friendly metaphors. EML [10], MaramaAIC [11], and SDLTool [12] are some examples. These tools provide environments supporting end-users in different domains. However, they do not support data analytics processes, techniques, data and requirements specifications, and do not target end users for such applications. There are some tools for designing, reusing, and sharing scientific workflows such as Kepler [13],Taverna [14], VisTrails [15], and Workspace [16]. Different projects can be built on top of these drag and drop based graphical tools and these tools are used in a variety of applications and domains. However, they only offer a limited number of data analysis steps and do not offer data analytics and ML capabilities and libraries. Finally, some software tools called model-based approaches to ML [17] implement algorithms specific to a given graphical model such as Infer.NET [18] and its extension in [19]. However, they are in very early stages and do not cover many of the data analytics steps in real-world problems.

VI. Conclusions

We have described a set of visual notations for specifying data analytics projects. Our DSVL, BiDaML, aims to provide a similar modeling framework for data analytics solution design as UML does for software design. It is comprised of five high and low level diagrammatic types representing both data and technique oriented components of a data analytics solution design. A physics of notations analysis and a cognitive walkthrough with several end-users were undertaken to evaluate the usability of our data analytics visual language. We have used our diagrams to model several complex big data analytics problems. We will improve our tool in future by including multiple view/elision support, integration with other tools, code generation, etc.

Acknowledgements

Support for this research from ARC Discovery grant DP170101932 is gratefully acknowledged.

References

[1] I. Portugal, P. Alencar, and D. Cowan, "A Preliminary Survey on Domain-Specific Languages for Machine Learning in Big Data," presented at the IEEE International Conference on Software Science, Technology and Engineering (SWSTE), Beer-Sheva, Israel, 2016.

[2] S. Landset, T. M. Khoshgoftaar, A. N. Richter, and T. Hasanin, "A Survey of Open Source Tools for Machine Learning with Big Data in the Hadoop Ecosystem," *Journal of Big Data,* vol. 2, no. 24, 2015.

[3] J. B. Rollins, "Foundational Methodology for Data Science," IBM Analytics2015.

[4] W. v. d. Aalst and E. Damiani, "Processes Meet Big Data: Connecting Data Science with Process Science," *IEEE Transactions on Services Computing,* vol. 8, no. 6, pp. 810-819, 2015.

[5] D. D. Cock, "Ames, Lowa: Alternative to the Boston Housing Data as an End of Semester Regression Project," *Journal of Statistics Education,* vol. 19, no. 3, 2011.

[6] D. Sculley *et al.*, "Hidden Technical Debt in Machine Learning Systems," presented at the 28th International Conference on Neural Information Processing Systems (NIPS), Montreal, Canada, 2015.

[7] *Business Process Model And Notation (BPMN)*. Available: https://www.omg.org/spec/BPMN/2.0/

[8] D. Moody, "Theory Development in Visual Language Research: Beyond the Cognitive Dimensions of Notations," in *IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*, Corvallis, OR, USA, 2009, pp. 151-154.

[9] H. Khalajzadeh, M. Abdelrazek, J. Grundy, J. Hosking, and Q. He, "A Survey of Current End-user Data Analytics Tool Support," in *IEEE International Congress on Big Data 2018*, San Francisco, USA, 2018, pp. 41-48.

[10] L. Li, J. Grundy, and J. Hosking, "A Visual Language and Environment for Enterprise System Modelling and Automation," *Journal of Visual Languages & Computing,* vol. 25, no. 4, pp. 253-277, 2014.

[11] M. Kamalrudin, J. Hosking, and J. Grundy, "MaramaAIC: Tool Support for Consistency Management and Validation of Requirements," *Automated Software Engineering,* vol. 24, no. 1, pp. 1-45, 2017.

[12] C. H. Kim, J. Grundy, and J. Hosking, "A Suite of Visual Languages for Model-Driven Development of Statistical Surveys and Services," *Journal of Visual Languages and Computing,* vol. 26, no. C, pp. 99-125, 2015.

[13] *Kepler*. Available: https://kepler-project.org/

[14] *Apache Taverna*. Available: https://taverna.incubator.apache.org/

[15] *VisTrails*. Available: https://www.vistrails.org/index.php/Main_Page

[16] *Workspace*. Available: https://research.csiro.au/workspace/

[17] C. M. Bishop, "Model-based Machine Learning," *Philosophical Transactions of the Royal Society A, Mathematical, Physical and Engineering Sciences,* vol. 371, no. 1984, 2012.

[18] T. Minka, J. Winn, J. Guiver, and D. Knowles, "Infer .NET 2.4, 2010. Microsoft Research Cambridge," 2010.

[19] D. Breuker, "Towards Model-Driven Engineering for Big Data Analytics – An Exploratory Analysis of Domain-Specific Languages for Machine Learning," presented at the 47th Hawaii International Conference on System Science, 2014.