In: Ali R., Kaindl H., Maciaszek L.A. (eds) Evaluation of Novel Approaches to Software Engineering. ENASE 2020. Communications in Computer and Information Science, vol 1375. Springer

BiDaML in Practice: Collaborative Modeling of Big Data Analytics Application Requirements

Hourieh Khalajzadeh[1], Andrew J. Simmons[2], Tarun Verma[1], Mohamed Abdelrazek[2], John Grundy[1], John Hosking[3], Qiang He[4], Prasanna Ratnakanthan[5], Adil Zia[5], and Meng Law[5]

[1] Monash University, Clayton VIC 3800, Australia {hourieh.khalajzadeh, john.grundy}@monash.edu and tver0005@student.monash.edu
https://www.monash.edu/it/humanise-lab
[2] Deakin University, Burwood VIC 3125, Australia
{a.simmons,mohamed.abdelrazek}@deakin.edu.au
[3] University of Auckland, Auckland 1010, New Zealand
j.hosking@auckland.ac.nz
[4] Swinburne University, Hawthorn VIC 3122, Australia
qhe@swin.edu.au
[5] Alfred Health, Melbourne VIC 3000, Australia
{P.Ratnakanthan,A.Zia,Meng.Law}@alfred.org.au

Abstract. Using data analytics to improve industrial planning and operations has become increasingly popular and data scientists are more and more in demand. However, complex data analytics-based software development is challenging. It involves many new roles lacking in traditional software engineering teams – e.g. data scientists and data engineers; use of sophisticated machine learning (ML) approaches replacing many programming tasks; uncertainty inherent in the models; as well as interfacing with models to fulfill software functionalities. These challenges make communication and collaboration within the team and with external stakeholders challenging. In this paper, we describe our experiences in applying our BiDaML (Big Data Analytics Modeling Languages) approach to several large-scale industrial projects. We used our BiDaML modeling toolset that brings all stakeholders around one tool to specify, model and document their big data applications. We report our experience in using and evaluating this tool on three real-world, large-scale applications with teams from: realas.com — a property price prediction website for home buyers; VicRoads — a project seeking to build a digital twin (simulated model) of Victoria's transport network updated in real-time by a stream of sensor data from inductive loop detectors at traffic intersections; and the Alfred Hospital — Intracranial hemorrhage (ICH) prediction through Computed Tomography (CT) Scans. These show that our approach successfully supports complex data analytics software development in industrial settings.

Keywords: Big data analytics · Big data modeling · Big data toolkits · BiDaML · Domain specific visual languages · End-user tools.

1 Introduction

Big data analytics applications have become increasingly widespread in business [24, 18]. However, building such software systems requires considering roles from many different skill backgrounds compared to traditional software development teams. Therefore, it is not straightforward to manage collaborations, teamwork and task specification. Nor is it easy to choose a language that is communicable for the diverse range of users from programmers and analysts to business managers. Such systems require complex, ML-based approaches, deployed at scale and that undergo rapid evolution, as business goals change and new data sources become available. A challenge reported by data scientists in [17] is that it is hard to convey the resulting insights to leaders and stakeholders in an effective manner and to convince teams that data science approaches are in fact helpful. Moreover, results of a large-scale survey [28] of data science workers show that even though they engage in extensive collaboration across all stages of data science work, there are gaps in the usage of collaborative tools. In order to successfully develop such big data analytics systems, a range of perspectives, tasks and interactions need to be taken into consideration [11]:

- Business perspective, including management need for the solution;
- Domain experts, who understand the various datasets available and how analysis of these can lead to usable value;
- Target end-users of the data analytics solution, i.e. the data visualizations produced - sometimes this is business management and/or domain experts, and sometimes other end users e.g. business staff, planners, customers and/or suppliers;
- Data analysts who have deep knowledge of available analytics toolsets to integrate, harmonize, analyze and visualize complex data;
- Data scientists or ML experts who have the expertise to deploy sophisticated ML software solutions;
- Software engineers with expertise to deploy solutions on large scale hardware for data management and computation, and end-user devices for data presentation;
- and Cloud computing architects who deploy and maintain large-scale solutions and datasets.

Existing ML-oriented tools only cover the technical ML and data science part of such problems, i.e. a very small part of the data analytics software engineering life cycle [25]. Current frameworks do not adequately capture multiple stakeholder perspectives and business requirements and link these to support the development of domain models. In this paper, we discuss the challenges in multidisciplinary data analytics teams. We then report on our experiences using our BiDaML approach [10, 14, 12], to help stakeholders to collaborate (using visual diagrams) in specifying, modeling and documenting what and how the software should perform. BiDaML is a suite of domain-specific visual languages (DSVL) that we created to support the teams through the development of data

analytics systems. Different visual languages support modeling of complex, big data software at differing levels of abstraction, using big data analytics domain constructs, and can be translated into big data solutions using Model-Driven Engineering (MDE)-based partial code generation. We also describe our experience working on three different industry use-cases to model and capture the requirements of their big data analytics applications. This paper is an extended version of an earlier one that appeared at ENASE 2020 [14]. The key contributions of this paper include:

- Important new insights into the key challenges in developing big data software solutions;
- Validating these challenges though three large, real-world data intensive industry projects and reporting on the experiences of using our approach for these usecases, and
- Identifying key future directions for researchers in the field of data analytics software development.

The rest of this paper is organized as follows. Section 2 presents our three large scale real world data intensive system examples. Section 3 provides key background and related work analysis. Section 4 outlines our approach to tackling such challenging big data system development and Section 5 presents the results of our industry case studies. Section 6 presents the results of research studies conducted to evaluate the usability of the BiDaML notations and tools. Section 7 discusses key findings and key future work directions, while Section 8 summarizes conclusions from this work.

2 Our Motivating Industrial Case Studies

In this section, we will show examples of real world data analytics projects to discuss some of the problems data scientists and software engineers face during the solution design process.

2.1 ANZ REALas

REALas[6] is a property price prediction website owned by the Australia and New Zealand Banking Group (ANZ). Launched in 2011, REALas claims to provide Australia's most accurate price predictions on properties listed for sale. ANZ had acquired Australian property start-up REALas to help home buyers access better information about the Australian property market in 2017. Being acquired by ANZ means more users and customers, and consequently, more data, leading to the need for an updated algorithm and retrained models, and therefore a need for data scientists. In this use-case, a complex new model needed to be developed to improve the accuracy and coverage of the property price prediction model. The project team originally comprised a project leader, a business manager, a product

[6] https://realas.com/

owner, three software engineers, and one data scientist. There were an existing working website and an ML model, as well as a dataset purchased from a third party. Two new data analysts/scientists were appointed to this project in order to create new models and integrate them with the existing website. The solution had initially been developed without the use of our tool, and the challenges the team faced to communicate and collaborate through the process was a key motivation for our research. Data scientists initially lacked an understanding of the existing dataset and solution as well as domain knowledge. Therefore, it took them some time to be able to start the project. Communicating progress to the business manager and other members of the team was another challenge. With the REALas team, we used our tool to document the process from business analysis and domain knowledge collection to deployment of final models.

2.2 VicRoads

VicRoads[7], the Victorian road traffic authority, utilizes the Sydney Coordinated Adaptive Traffic System (SCATS) to monitor, control and optimize traffic intersections. Transport researchers within the Monash University Civil Engineering department sought to build a traffic data platform that would ingest a real-time feed of SCATS data from VicRoads and integrate it with other transport datasets such as public transport travel history and traffic incidents reported through social media. Initially, the Civil Engineering department consulted with a software outsourcing company, who proposed a platform composed of industry standard big data tools. However, the software outsourcing company lacked understanding of the datasets and intended use of the platform, thus were unable to begin work on the project. Furthermore, it was unclear who would maintain the computing infrastructure, monitor data quality, and integrate new data sources after the initial phase of the project. We worked with transport researchers and used our tool to document the intended software solution workflow from data ingestion to traffic simulation and visualization. This allowed us to assist in the formation of an alternative software solution making better use of systems and services already available.

2.3 Alfred Hospital & Monash Clinical Data Science

In this project, a group of radiologists, researchers and executives from Alfred hospital have used AI for predicting Intracranial hemorrhage (ICH), pulmonary embolism, spine/rib fractures, lung nodules/cancer through CT Scans, work traditionally done by radiologists. These AI platforms would enable them to prioritize the CT Scans based on the results and forward them to the radiologist for an urgent double check and follow up. Hence, a CT Scan with positive outcome could be reported in a few minutes instead of a few days. The team wanted to analyze the data before and after using the AI platform and based on the

[7] https://www.vicroads.vic.gov.au/traffic-and-road-use/traffic-management/traffic-signals/scats

turnaround time (TAT) and cost analysis decide whether to continue using the AI platform or not. Human radiologists would then also spend more time interrogating scans which have been flagged to be abnormal by AI, and perhaps less time on scans analysed as normal. These clinical AI algorithms have already been found to detect abnormalities that have been missed by human radiologists, even though as that time, they did have some false positives. Currently with another AI clinical product used to detect lung nodules, the AI has in a few months detected 4-5 nodules which radiologists have missed. They were looking towards improving these models to provide near human or supra-human accuracy. However, due to the diversity of the team, it was difficult to communicate the medical terms to the data analysts and software engineers, and the analysis methods and software requirements and solution choices to the radiologists and the executive team. We used our approach with clinical, data science and software team members to model and document steps and plan further key project stages.

3 Data Analytics Software Development Challenges and Related Works

As illustrated using our motivating examples, there is no trace back to the business needs/requirements that triggered the project. Furthermore, communicating and reusing existing big data analytics information and models is shown to be 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 design, especially the critical business requirements. Most current tools focus on low-level data analytics process design, coding and basic visualization of results and they mostly assume data is in a form amenable to processing. In reality, most data items are in different formats and not clean or integrated, and great effort is needed to source the data, integrate, harmonize, pre-process and cleanse it. Only a few off-the-shelf 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 [25]. These tasks only cover a small part of data and ML operations and deployment of models. Business and management modeling tools usually do not support many key data analytics steps including data preprocessing and ML steps. There is a need to capture the high-level goals and requirements for different users such as domain expert, business analyst, data analyst, data scientist, software engineer, and end-users and relate them to low level diagrams and capture details such as different tasks for different users, requirements, objectives, etc. Finally, most of the tools covering ML steps require data science and programming knowledge to embed code and change features based on the user requirements.

3.1 Key Challenges

Data analytics are widely used in different organizations to improve decision making. Developing big data software solutions to support an organization's data analytics needs requires a multidisciplinary team of data analysts, data scientists, domain experts, business managers, software engineers, etc. Domain experts, business analysts and business managers do not necessarily have a background in data science and programming, and therefore, they do not know how to convert their problem to a data analytics problem, where to start the project from and how to use the myriad of existing data science tools. Similarly, data scientists may be able to create small, bespoke solutions but lack software engineering skills to scale solutions. Software engineers generally lack detailed data science and domain expertise. As identified in [9, 11], while many techniques and tools exist to support the development of such solutions, they have many limitations. In general, developing big data software solutions suffers from several key challenges.

Challenge 1 (C#1) : ***Domain experts, business analysts and business managers do not have a background in data science and programming.*** Domain experts and business users of big data analytics solutions know the target domain, the data in the domain and the intended benefits from the solution. However, they lack the expertise to design and develop, and sometimes to adequately understand such solutions.

Challenge 2 (C#2) : ***Data analysts, data scientists and software engineers do not have domain knowledge.*** Data analytics is applied to a variety of different applications from health and education to finance and banking, and data scientists with technical data science and programming background do not necessarily have a background in any of the applications they work on. Therefore, it takes some time for them to collect background knowledge, get familiar with the domain, what has been done so far, the existing solutions, etc.

Challenge 3 (C#3) : ***Data scientists lack software engineering expertise.*** Data analysts and data scientists are an emerging IT workforce and are to describe a problem domain, analyze domain data, extract insights, apply ML models, do evaluations and deploy models. However, most do not have many of the skills of software engineers, including solution architecture, coding and large-scale data analytics software deployment. Often data science models lack ways to describe these aspects of the solution requirements and do not scale.

Challenge 4 (C#4) : ***Lack of a common language between team members.*** Domain experts, business analysts and business managers have a high-level knowledge of the problem, objectives, requirements, users, etc while data analysts and data scientists are more from a technical background with expertise

in data science and programming. Communicating and collaborating between users from different backgrounds is a bottleneck in data analytics application development. Data scientists spend most of their time preparing data to make it usable by downstream modeling algorithms. There are no communicable models or outputs for different stakeholders to enable mutual understanding and agreement.

Challenge 5 (C#5) : ***Evolution of the solution is poorly supported.*** After the solution is developed and deployed, emerging new data, changing business needs and usage of the application all typically result in a need to update and re-train the models to improve performance. However, the data science group originally employed to develop the initial solution is often disbanded upon completion, leaving others to attempt to maintain their models. In contrast to software artifacts, the processes and decisions involved in gathering, cleansing and analyzing data are rarely fully documented even in scientific research [2], let alone industry projects with tight deadlines and limited resources for producing documentation.

Challenge 6 (C#6) : ***Re-using of the existing solutions is not feasible.*** Whether a new group of data scientists is appointed to update and improve out-of-date models and software or the team is left struggling with it, traditional documentation approaches mean it will take a long time for the new team to understand the existing model, as different data scientists have their own method of modeling and programming. In addition, as they often use specific tools, it is not normally the case for them to spend time understanding and modifying the existing solutions. Often, the new team ends up creating new models and therefore repeating all these steps and facing all the same problems again. This is not only a problem within a single group but in different parts of organizations, where data analytics solutions are created without common models being shared and reused.

3.2 Related Work

There are many data analytics tools available, such as Azure ML Studio[8], Amazon AWS ML[9], Google Cloud ML[10], and BigML[11] as reviewed in [11]. However, these tools only cover a few phases of DataOps, AIOps, and DevOps and none cover business problem description, requirements analysis and design. Moreover, since most end-users have limited technical data science and programming knowledge, they usually struggle using these tools. Some DSVLs have been developed for supporting enterprise service modeling and generation using end-user friendly

[8] https://studio.azureml.net/
[9] https://aws.amazon.com/machine-learning/
[10] https://cloud.google.com/ai-platform
[11] https://bigml.com/

metaphors. An integrated visual notation for business process modeling is presented and developed in [19] using a novel tree-based overlay structure that effectively mitigates complexity problems. MaramaAIC [8] provides end-to-end support between requirements engineers and their clients for the validation and improvement of the requirements inconsistencies. SDLTool [16] provides statistician end-users with a visual language environment for complex statistical survey design/implementation. These tools provide environments supporting end-users in different domains. However, they do not support data analytics processes, techniques, data and requirements, and do not target end-users for such applications. Scientific workflows are widely recognized as useful models to describe, manage, and share complex scientific analyses and tools have been designed and developed for designing, reusing, and sharing such workflows. Kepler [20] and Taverna [27] are Java-based open source software systems for designing, executing, reusing, evolving, archiving, and sharing scientific workflows to help scientists, analysts, and computer programmers. VisTrails [5] is a Python/Qt-based open-source scientific workflow and provenance management system supporting simulation, data exploration and visualization. It can be combined with existing systems and libraries as well as your own packages/modules. Finally, Workspace [6], built on the Qt toolkit, is a powerful, cross-platform scientific workflow framework enabling collaboration and software reuse and streamlining delivery of software for commercial and research purposes. Users can easily create, collaborate and reproduce scientific workflows, develop custom user interfaces for different customers, write their own specialized plug-ins, and scale their computation using Workspace's remote/parallel task scheduling engine. 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 no data analytics and ML capabilities and libraries. Finally, some software tools implement algorithms specific to a given graphical model such as Infer.NET [21]. This approach for implementing data analytics techniques is called a model-based approach to ML [3]. An initial conceptualization of a domain specific modeling language supporting code generation from visual representations of probabilistic models for big data analytics is presented in [4] by extending the analysis of the Infer.NET. However, it is in very early stages and does not cover many of the data analytics steps in real-world problems.

4 Our Approach

Since the exisitng big data analytics tools provide only low-level data science solution design, despite many other steps being involved in solution development, a high-level presentation of the steps to capture, represent, and communicate the business requirements analysis and design, data pre-processing, high-level data analysis process, solution deployment and data visualization is presented in [10, 14].

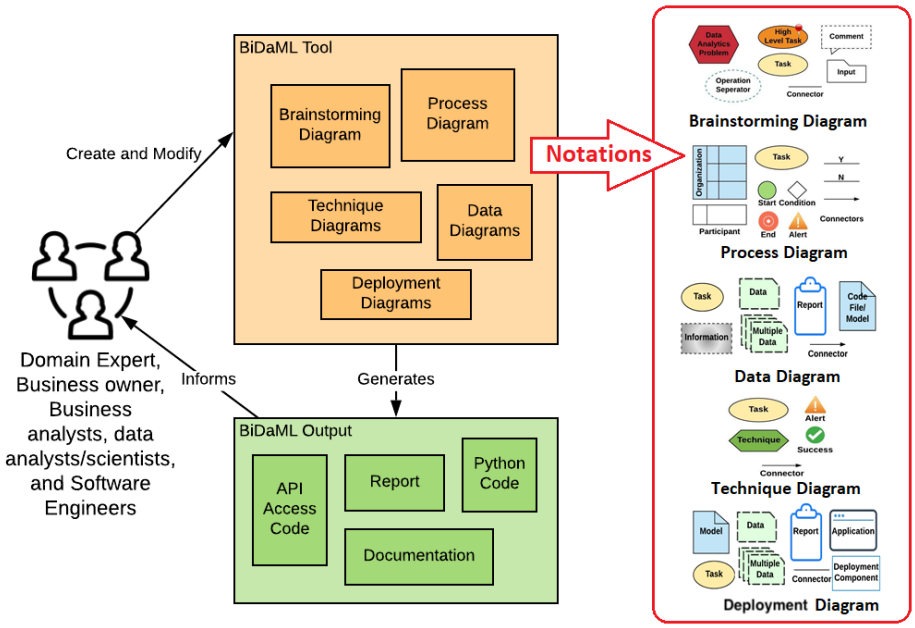

Fig. 1. BiDaML Notations (from [14])

4.1 BiDaML Visual Language

BiDaML, presented in [10] and extended in [14, 12], is a set of domain-specific visual languages using different diagram types at different levels of abstraction to support key aspects of big data analytics. Overview of the BiDaML approach, high level to low level diagrams, and their notations are shown in Fig. 1. 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 process diagram can be further extended by technique and data diagrams, and then, the technique and data diagrams are connected to a result output diagram. Finally, the deployment diagram, defined for every data analytics problem, models deployment related details at a low level. The updated diagrams presented in [12] include:

Brainstorming Diagram A data analytics brainstorming diagram's scope covers the entirety of a data analytics project expressed at a high-level. There are no rules as to how abstractly or explicitly a context is expanded. The diagram overviews a data analytics project in terms of the specific problem it is associated with, and the task and subtasks to solve the specific problem. It supports interactive brainstorming to identify key aspects of a data analytics project such as its requirements implications, analytical methodologies and specific tasks. Brainstorming diagram comprises an icon representing the data analytics prob-

lem, tasks which the problem is associated with, a hierarchy of sub-tasks for each task, and finally the specific information about sub-systems 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); artificial intelligence and ML-related activities (AIOps); and development and deployment activities (DevOps).

Process Diagram The key business processes in a data analytics application are shown in a process diagram. We adapt the Business Process Modeling Notation (BPMN) [23] 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 non-technical users such as domain experts, business users and customers, by providing a notation that is 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). Preparation of data items or different events trigger other events and redirect the process to the other users in the same or different pool.

Technique Diagram Data analytics 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 into the specific stages and the technique used to solve a specific sub-task specified.

Data Diagram To document the data and artifacts consumed and produced in different phases described by each of the above diagrams, one or more low-level data diagrams are created. 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. A high-level data diagram can be represented by connecting the low-level diagrams for different BusinessOps, DataOps, AIOps, and DevOps. We initially had an output diagram to represent the reports and outputs, that has eventually been merged with data diagram.

Deployment Diagram Deployment diagrams represent the software artifacts and the deployment components and specify the deployment related details. In the deployment diagram, we focus on distributed cloud platforms, services, and frameworks rather than individual nodes/devices. We had initially adopted the deployment diagram concepts from the context of Unified Modeling Language (UML) that has eventually changed to our new deployment representation. Details can be found in [12].

4.2 BiDaML Support Tool

We have developed an integrated design environment for creating BiDaML diagrams. The 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. We have used MetaEdit+ Workbench [26] to implement our tool. Using MetaEdit+, we have created the objects and relationships defined as the notational elements for all the diagrams, different rules on how to connect the objects using the relationships, and how to define low level sub-graphs for the high level diagrams. Figs. 2 and 3 show examples of BiDaML tool in use for creating brainstorming diagram and an overview of all the diagrams. They show the outputs generated from the tool including Python code and word document.

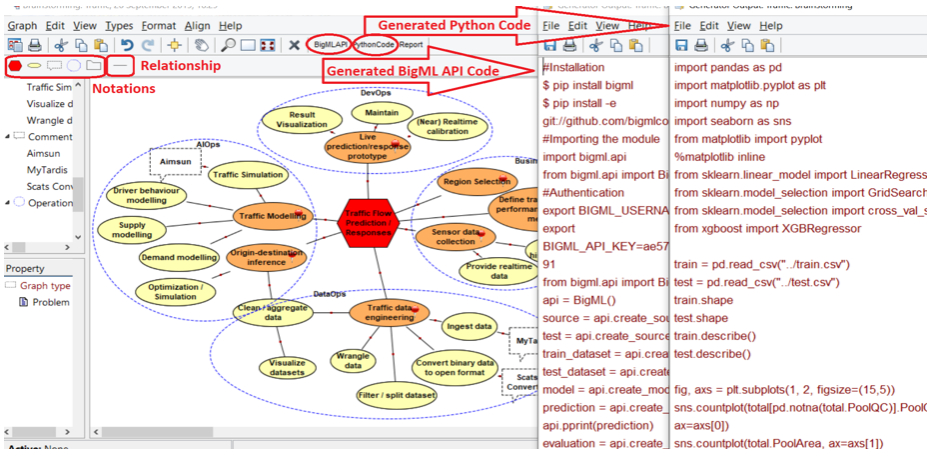

Fig. 2. Brainstorming Diagram Created in BiDaML Tool for the Traffic Analysis Example and Snippets of the Generated Python Code (from [13])

4.3 BiDaML-web

We have implemented a web based auto layout user interface for BiDaML. BiDaML-web[12] [15] uses Node.js as a runtime environment that executes JavaScript and for server-side scripting. The application uses Vue.js which is a JavaScript framework for building user interfaces. Our web-based implementation of BiDaML[13] is based on the auto-layout web-based tool vue-graphViz[14][7] which includes features to adjust placement of items to avoid clutter (such as lines crossing symbols

[12] https://bidaml.web.app/
[13] https://github.com/tarunverma23/bidaml
[14] https://github.com/yusufades/vue-graphViz

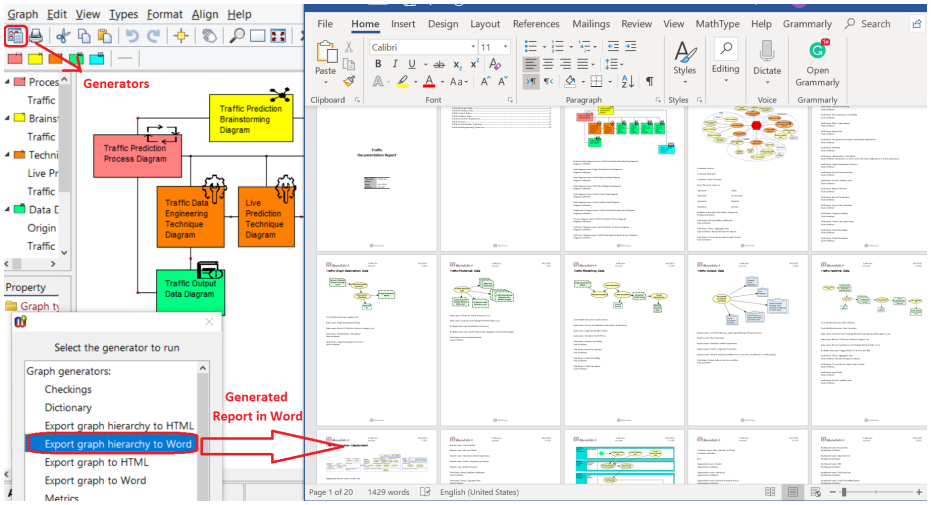

Fig. 3. Overview of all the Diagrams Created in BiDaML Tool for the Traffic Analysis Example and the Final Report in Word Generated from the Overview Diagram (from [13])

on the graph) in order to improve readability. Hosting of the application, Real-time database, Google Analytics, and authentication services have been also been implemented using Firebase's APIs. We include a set of quick-start questions to help the user rapidly generate the initial diagram with minimal clicks, then provide a minimal interface through which the user can modify the diagram as needed. To increase the user's awareness of relevant algorithms and datasets specific to their problem, we utilise Papers with Code and Google/Kaggle Datasets Search for recommending algorithms and datasets. Finally, our tool includes a technique recommender in order to help end users decide which techniques are appropriate given their dataset, and to consider questions such as the type of prediction or classification task, and whether they have access to sufficient labelled data. An example of using BiDaML-web for creating a brainstorming diagram for the property price prediction is shown in Fig. 4.

5 BiDaML in Industry Practice

In this section, as the main contribution of this paper, we report on the experiences using our tool in three industry projects to validate the challenges we identified earlier as well as evaluating the usability and suitability of our approach in a real setting. We used our tool in 3 different real-world industry problems. In each case, we create a high-level view of an existing problem and use it to uncover assumptions and allow communications and collaborations to successfully evolve in a measured manner.

Fig. 4. BiDaML-web used for Creating a Brainstorming Diagram for the Property Price Prediction Example

ANZ REALas Our tool was used to model REALas development from initial requirement analysis and data collection through the entire life cycle of its deployment. The first issue *(C#1)* appeared when the team was unable to develop data analytics solutions or update the existing models due to the lack of necessary knowledge and therefore appointed new data scientists to work on it. There was no documentation available from the existing models, and the data scientist who had developed the existing model was not available to work on the project, therefore data scientists needed to spend hours talking to the existing members, and going through the existing source code, where available, to understand the existing solution. Moreover, new data scientists were not from finance and banking backgrounds and needed to go through documents in different formats, and dataset dictionaries to understand the concepts *(C#2)*.

Since there was no common language *(C#4)*, it took a long time for them to transfer information and convert them from domain knowledge to data science knowledge. Moreover, new data scientists needed to spend weeks to even months to analyze the existing dataset, clean and wrangle datasets, acquire more datasets, integrate all these models and try many new features to be able to improve the existing model since data scientists spend most of their time preparing data. However, due to the lack of a platform or common language, they could not share and communicate their progress with the business manager/leader/owner. Data scientists finally recreated new models instead of reusing and modifying the existing models due to the lack of documentation and also since there was no common framework and they had to use a different platform than used for the existing models *(C#6)*. They eventually created a new model to replace the existing model and finally had communicable results. They still needed to integrate it with the website, requiring software engineers in the final step *(C#3)*. How-

ever, this was not the end of the story for the team, and the models needed to be later updated and retrained after a while, and therefore, the need for updated models, new data scientists, and another round of all these challenges *(C#5)*.

We used our tool to redesign the whole project, in order to communicate and collaborate through the project, as well as automatically document all the above development steps. A brainstorming diagram was designed in the first place to help all the stakeholders fully understand and communicate the steps and existing solutions through a visualized drag-and-drop based platform. Once agreements were made on all the steps, a process diagram was created to assign the tasks to the existing stakeholders. These two steps only took a few hours from the whole team, and they could also incrementally modify these two during the process. Data scientists were now fully aware of the domain knowledge, background of the project and existing models and could further leave comments and ask questions if further information was required.

In the next step, data scientists worked on the data and ML parts, however this time, needed to visually record and keep track of the data items, artefacts, models, reports, etc that they tried, whether they were successful, failed, or were just being planned. This made it easy for them to communicate their progress and also reach agreement on the results. Our tool has a code generation feature that could help data scientists start from a template instead of starting from scratch. Our tool provides a way to automatically document and embed their code templates for future usage. They gradually developed new models, and finally, worked with software engineers on a deployment diagram to define where and how to deploy the items generated in different steps. The new method was efficient in the way it took less time for the stakeholders to communicate and collaborate, and the step-by-step automatic documentation made the solution reusable for future reference. Based on the product owner's feedback *"this tool would have been helpful to understand and communicate the complexity of a new ML project within an organisation. It would assist the wider team to collaborate with data scientists and improve the outputs of the process"*.

Examples of some of the diagrams generated throughout the process are shown in Fig. 5. A full list of the diagrams and the generated report is available in [1]. In Fig. 5, a brainstorming diagram shows high-level tasks designed for the problem and how they are divided based on the nature of the operations (BusinessOps, DataOps, AIOps and DevOps). For instance, "price prediction website deployment" is a high-level DataOps task consisting of lower-level tasks such as "store", that is later supported by the "Real Estate MySQL DB" component on top of an AWS Cloud DB Server shown in the deployment diagram. Report data diagram enables agreement on the reports expected to be generated or shown on the website.

VicRoads In this use-case, there was a need to formally capture detailed requirements for a traffic data platform that would ingest a real-time stream of traffic data received from VicRoads (the Victorian road transport authority), integrate this with other transport data sources, and support modeling and vi-

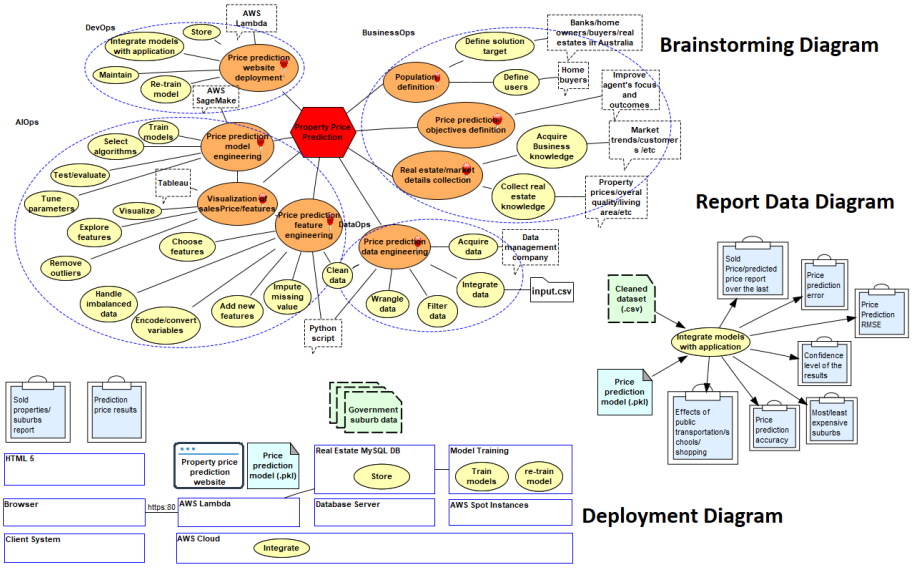

Fig. 5. Examples of the Diagrams Created for the Property Price Prediction Use-Case

sualization of the transport network at a state-wide level. The first issues (*C#1* and *C#3*) arose in the initiation of the project. The project leader and traffic modeling experts identified the need for a big data platform. However, without a background in software engineering or familiarity with modern data science tools, they were unable to determine whether the technology stack offered by the software outsourcing company would meet their needs.

The second issue *(C#2)* arose in requirements elicitation; the software outsourcing company lacked understanding of the domain and thus did not understand what tasks were required of them. To overcome the communication difficulties, a meeting was arranged between the project leader, the traffic modeling expert, a data engineer/visualization designer, the project team from the software outsourcing company, and the eResearch High Performance Computing services team. However, the lack of a common language *(C#4)* meant that communication could only take place at a high-level rather than at the level of detail necessary to initiate direct technical action. The software outsourcing company produced a plan for the software they intended to deploy; however, no plan existed for who would monitor and maintain the software and systems after deployment *(C#5)*, such as responding to faults in real-time data ingestion or adding support for new types of data. To justify the cost and time investment into the project, the project leader wanted to be able to reuse *(C#6)* the platform for related projects, such as a smart city. However, it was unclear whether the work invested in the design of the transport data platform could be reused in other projects.

We performed in-depth interviews with the project leader and traffic modeling expert, then used our tool to document the entire data analytics workflow including data ingestion, transport modeling, and result visualization. The traffic prediction brainstorming diagram was initially created as a handwritten sketch on paper, then later recreated using the tool. The process diagram, technique diagram, data diagram, and deployment diagram were created directly using the tool. While most diagram types took only 15-30 minutes to create, the process diagram proved the most time consuming, taking almost 3 hours due to the need to detail tasks to integrate each system and determine roles of individuals (we have since simplified the process diagram notation in order to streamline the process). As our tool forces the user to consider all phases of the project, the modeling process helped reveal gaps in planning that required attention. Notably, no budget or personnel had been assigned to maintain the system after initial deployment, integrate new data sources, and monitor data quality/security. Indeed, in the process diagram, we were forced to label both the organization and participant for these tasks as To Be Determined (TBD).

We presented the diagrams to the traffic modeling expert for feedback. This took place over a course of an hour session, in which we presented each diagram in the tool. The tool supported live corrections to the diagrams such as creation, modification or reassignment of tasks as we discussed the diagrams with the traffic modeling expert. Feedback from the expert was positive: *"I think you have a good understanding of the business... how do you know about all of this? I think this is very interesting, very impressive what you are proposing. It covers a lot of work that needs to be done."* While the expert stated that the diagrams were helpful to *"figure out all the processes and what tasks need to be done"* they were reluctant to use our tool to communicate with external stakeholders in other organizations: *"to use this tool, it will be likely not possible, because they [the other organizations involved] have their own process, they don't want to follow a new one."* We subsequently presented printouts of the diagrams to the project leader who expressed some uncertainty about the purpose of the notation; however, noted that an adaptation of the data diagrams as a means to document data provenance (i.e. the ability to trace the origins of data analysis results back to raw data used) would be *"very useful"*.

Examples of the diagrams generated throughout the process are shown in Fig. 6. A full list of the diagrams and the generated report is available in [1]. For instance, the "Simulation Data Diagram" shows that the Origin-Destination Matrix (where vehicles enter the traffic network and where they travel to) used for traffic modeling is partially derived from "15 Minute Volume Average" sensor data fed into an optimization/simulation process to find the most likely Origin-Destination matrix given the reported sensor readings. Documenting this using our approach can assist future users of the resultant Origin-Destination Matrix to better understand the original source of their data and to recompute the result. For the purpose of live traffic prediction, it was desired to automatically recompute the Origin-Destination Matrix from recent data. The "Live Prediction Technique Diagram" shows the consideration of different techniques to achieve

this. Periodic re-execution of the workflow every 5 to 15 minutes emerged as an option to facilitate live predictions without requiring traffic modeling experts to have a software engineering background.

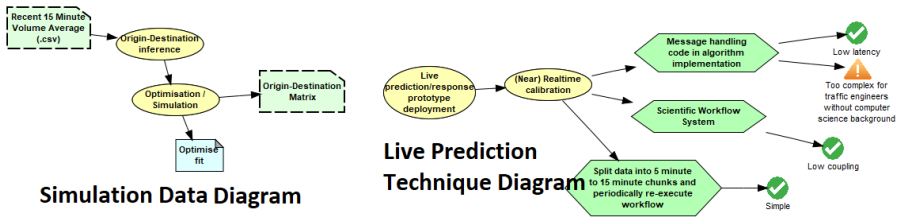

Fig. 6. Examples of the Diagrams Created for the VicRoads Use-Case

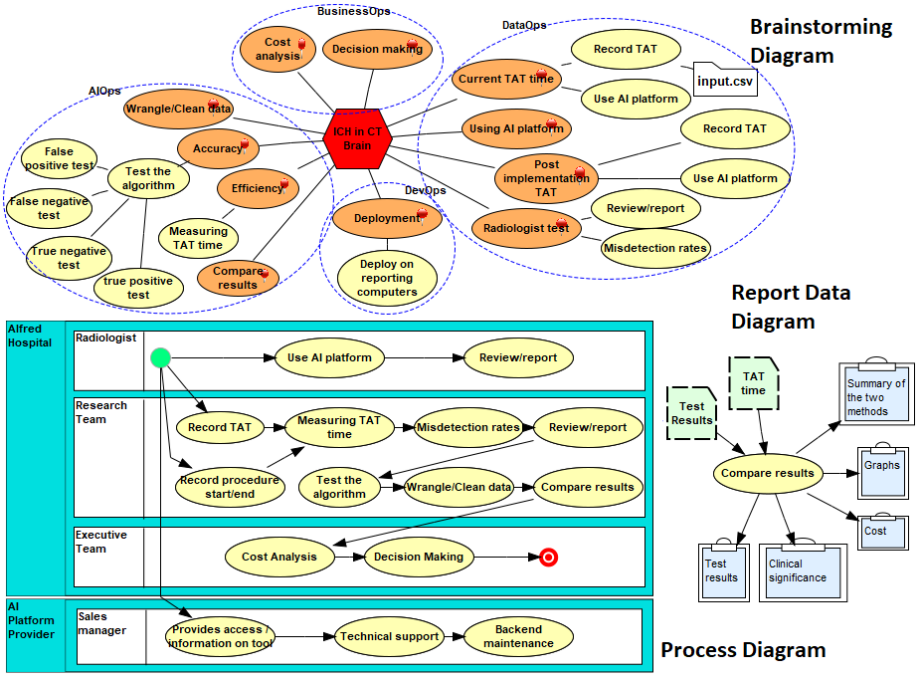

Fig. 7. Examples of the Diagrams Created for the Alfred Hospital Use-Case

Alfred Hospital & Monash Clinical Data Science The team at the Alfred hospital needed to analyze data before and after using an AI platform to decide

whether to continue using the tool or not. The team consisted of radiologists and medical researchers without a background in data analytics. The AI platform provider claimed that by using the AI platform they would be able to reduce the TAT time from days to minutes. Without a background in data analytics and ML they would not be able to collect, integrate, cleanse, analyze and compare the results after using the AI platform and ensure the AI platform was able to meet their requirements *(C#1)*. They decided to talk to a data analysis researcher to discuss the possible solutions, however, the data analysis researcher had no background in medical and radiology terms and concepts and communicating the requirements was a challenge *(C#2)* as there initially was a lack of a common language between these people *(C#4)*. Deploying the final software *(C#3)*, who would be responsible for different tasks *(C#5)* and whether they could reuse the analysis for the future projects *(C#6)* were other challenges they would face if they wanted to continue the project with no clear definition and documentation of the detailed tasks.

We briefly introduced our tool to the team and since it seemed to be a well-designed fit for the project due to the diverse nature of the stakeholders, we decided to use the tool to model and analyze the requirements and capture the details. We had an initial one-hour meeting with one of the radiologists and started developing the models and collecting a deep understanding of the project, requirements, concepts, and objectives through the brainstorming diagram. Then we spent almost 30 minutes to document the entire data analytics workflow including data collection and wrangling, comparing the methods, making the final decision and deploying the final product through process, technique, data and deployment diagrams. Since we needed to deeply think about all the details and plans, the tool forced us to consider all phases and details of the project. We then organized a follow-up meeting with the team from the hospital, including two radiologists and the team leader and presented the diagrams for their feedback. The meeting took 30 minutes. During the meeting, we modified the organizations and users involved as well as the expected reports and outcomes and the infrastructure in the deployment diagram. Going through the diagrams made us think about these and plan for them. We then shared the report generated from the tool with the team for their feedback.

Feedback from the team was that *"BiDaML offered a simplified visual on different components of the project. These diagrams could be circulated to the project team and would clarify the workflow, requirements, aims and endpoints of each role and the entire project. In large-scale projects, BiDaML would be of even greater benefit, with involvement of multiple teams all working towards a common goal."* However, *"The user interface seemed quite challenging to navigate. However, this could be easily negated with appropriate training and instructional material."* Examples of the diagrams generated throughout the process are shown in Fig. 7. A full list of the diagrams and the generated report is available in [1]. For instance, the process diagram, as one of the high-level diagrams generated for this example, clarified that there are three groups from the hospital involved in this project, and there are currently no data analysts involved in the

project and the research team are recording TAT, testing the algorithm, planning to compare the results, etc before the executive team decides on purchasing the AI platform or not.

6 Evaluation

In addition to our experiences of using BiDaML in practice within Section 5, we have evaluated the usability and suitability of our visual languages and tool suite in two ways (preliminary results originally reported in [10, 14] and the comprehensive extended experiments originally presented in [12, 15]). The first was an extensive physics of notations evaluation [22]. This was a useful end-user perspective evaluation without having to involve a large-scale usability trial. The second was a series of user studies to understand how easy BiDaML is to learn and use. The user studies performed were: a cognitive walkthrough of the original BiDaML support tool with several target domain expert end-users, including data scientists and software engineers, as test participants; a group user study to compare handwritten BiDaML diagrams to other notations; and finally a user study of BiDaML-web.

6.1 Physics of Notations Evaluation

Semiotic clarity specifies that a diagram should not have symbol redundancy, overload, excess and deficit. All our visual symbols in BiDaML have 1:1 correspondence to their referred concepts. Perceptual discriminability is primarily determined by the visual distance between symbols. All our symbols in BiDaML use different shapes as their main visual variable, plus redundant coding such as color and/or textual annotation. Semantic transparency identifies the extent to which the meaning of a symbol should be inferred from its appearance. In BiDaML, icons are used to represent visual symbols and minimize the use of abstract geometrical shapes. Complexity management restricts a diagram to have as few visual elements as possible to reduce its diagrammatic complexity. We used hierarchical views in BiDaML for representation and as our future work, we will add the feature for users to hide visual construct details for complex diagrams. Cognitive integration identifies that the information from separate diagrams should be assembled into a coherent mental representation of a system; and it should be as simple as possible to navigate between diagrams. All the diagrams in BiDaML have a hierarchical tree-based structure relationship.

Visual expressiveness defines a range of visual variables to be used, resulting in a perceptually enriched representation that exploits multiple visual communication channels and maximizes computational offloading. Various visual variables, such as shape, color, orientation, texture, etc are used in designing BiDaML visual symbols. Dual coding means that textual encoding should also be used, as it is most effective when used in a supporting role. In BiDaML, all visual symbols have a textual annotation. Graphic economy discusses that the number of different visual symbols should be cognitively manageable. As few

visual symbols as possible are used in BiDaML. Cognitive fit means that the diagram needs to have different visual dialects for different tasks or users. All the symbols in BiDaML are usable for different users and tasks. However, in the future, we will provide different views for different users in our BiDaML support tool, and users will be able to navigate between views based on their requirements.

6.2 Cognitive Walk-through of BiDaML Support Tool

We asked 3 data scientists and 2 software engineers (all experienced in big data analytics) to carry out a task-based end-user evaluation of BiDaML. The objective was to assess how easy it is to learn to use the visual models 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 BusinessOps, DataOps, AIOps, or DevOps part of a brainstorming diagram for a data analytics problem of their choice from scratch. In the second, each participant was given a process diagram and asked to explain it, comment on the information represented and provide suggestions to improve it. The third involved participants 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 from the participants indicated that BiDaML is very 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 during the project before the final solution. They liked how different layers and operations are differentiated. Moreover, they could capture and understand business requirements and expectations and make agreements on requirements, outcomes, and results through the project. These could then be linked clearly to lower-level data, technique and output diagrams. 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 altogether 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 humans versus a tool. In future tool updates, we will provide different views for different users and will allow users to hide/unhide different components of the diagrams based on their preference. Moreover, in our future code generation plan, we will separate different tasks based on whether they are conducted by humans or tools.

6.3 Group User Study of Handwritten BiDaML Diagrams

To address limitations of the first user study, we performed a second user evaluation in a more structured manner, with feedback collected anonymously. In order to evaluate the suitability of the BiDaML notation for new users in a diversity of scenarios, the participants were asked to create BiDaML diagrams to model the project of their choice. Moreover, to see how BiDaML compared to other notations, participants were asked to create both a BiDaML diagram as well as diagram using another notation, then share their diagrams with other participants. The results of our study, reported in [12], showed that users prefer BiDaML for supporting complex data analytics solution modeling more than other modeling languages.

6.4 BiDaML-web User Study

To evaluate BiDaML-web, we performed a user study with a group of 16 end-users. Given we had conducted comprehensive evaluations of BiDaML notations and its comparison in our previous works [10, 12], we only evaluated the auto-layout web based user interface and the recommender tools in this study. Our aim was to evaluate the usability of BiDaML-web and whether users paid attention to recommendations and found the code, paper, project and dataset recommendations, helpful. In this study (originally reported in [15]), we first introduced the BiDaML concept, notations, and diagrams and then asked a group of 16 data analysts, data scientists, domain experts and software engineers to use BiDaMl-web tool to model and describe a project of their choice. We finally asked participants to fill in a questionnaire and asked to rate whether BiDaML-web is easy to understand/learn/use and how they found the recommender tool. The group study consisted of 9 PhD students, and 7 academic staff. 8 participants categorised themselves as software engineers, 4 as data analysts/scientists, 5 as Domain expert/business analyst/business manager, and 2 as "other" (some participants identified as multiple categories). The distribution of data analytics/data science experience was: 7 participants with less than 1 year; 2 participants with 2 years; 2 participant with 3 years; and 5 participants with 5 to 9 years. The distribution of programming experience was: 2 participants with 0-1 year, 1 participant with 1 year, 3 participants with 2 years, 2 participants with 3 years, 2 participants with 4 years, 2 participants with 5 to 9 years, and 4 participants with 10 or more years. Study participants found the integrated recommender tools helpful,and also responded positively to the tool overall, as detailed in [15]. The primary reasons selected were "it made me think of details that I never noticed" (9 of 16) and "introduced resources I wasn't aware of" (9 of 16); it was possible for a participant to select multiple reasons or provide a custom response to this question.

7 Discussions

We applied our approach on three different real world usecases to validate these challenges as well as our approach in a real setting. Our aim was to evaluate

and gain experience with applying our method to conduct requirements analysis and modeling part of complex data analytics applications. We have found that our method: has been practical to a variety of real-world large-scale applications. It helped communication and collaboration between team members from different backgrounds by providing a common platform with mutual language (*C#1-C#4*). It also helped identify and agree on details in the early stages *(C#5)*. Thus our tool can reduce costs and improve the speed of business understanding by addressing these details during the requirement analysis stage. It also provided automatic documentation that can be re-used for retraining and updating of the models *(C#6)*. Based on our radiologist users' experience: *"As the frequency of multidisciplinary, collaborative projects is increasing, there is a clear benefit with the use of BiDaML as a tool for designing data analytics processes. Furthermore, the automatic code generation capabilities of BiDaML would greatly aid those who do not have experience in large-scale data analysis. We do see use of BiDaML in this specific project and would be interested in seeing its results."* There are some notable issues we faced while working with industrial partners on these data analytics requirement engineering problems. Our tool can be accessed by all the stakeholders in different geographical locations. However, our intervention has been required so far, as our original BiDaML tool depends on MetaEdit+ modeling development tool [26] and a license required to be purchased by users. Users make benefits of the early requirement engineering part. However, they continue using existing tools and programming languages to develop the ML and application development parts once they have completed the requirement analysis, modeling and planning part of the project. To overcome the first issue, we re-implemented the tool as a stand-alone web-based tool that users can work on individually without us being required to manage the modeling part. To overcome the second, we aim to develop recommendations and integrations for popular existing tools to encourage users to continue using our approach through the entire development of the final product. We see considerable scope for providing back end integration with data analytics tools such as Azure ML Studio, RapidMiner, KNIME, etc. Our tool can be used at an abstract level during requirements analysis and design, and then connect to different tools at a low-level. Therefore, our DSVLs can be used to design, implement and control a data analytics solution.

8 Conclusions

We have identified several key challenges in data analytics software engineering, compared to traditional software teams and processes. We described our BiDaML domain-specific visual-language based technique for requirements modeling and documentation of big data analytics systems. Our experience in three different real-world case studies in finance, transportation and health has been that our method is easy to apply to diverse real-world large-scale applications and greatly assisted us in identifying the requirements as well as domain and business knowledge that will potentially lead to improvements in planning and

developing the software. Additionally, an initial web-based interface for BiDaML is presented and evaluated in this paper. Being able to successfully apply our practical method in designing and analyzing different big data analytics applications encouraged us to further extend BiDaML as a stand-alone web-based tool and connect it to the existing ML recommendation toolboxes as a step toward realising our vision of an integrated end-to-end modelling platform for big data solutions.

Acknowledgements

Support for this work from ARC Discovery Projects DP170101932 and from ARC Laureate Program FL190100035 is gratefully acknowledged. We would also like to acknowledge Prof. Hai Vu and Dr. Nam Hoang from the Monash Institute of Transport Studies for their collaboration, and thank the Department of Transport (VicRoads) for sharing the transport data.

References

1. Bidaml big data analytics modeling languages, http://bidaml.visualmodel.org/
2. Baker, M.: 1,500 scientists lift the lid on reproducibility (2016)
3. Bishop, C.M.: Model-based machine learning. Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences **371**(1984), 20120222 (2013)
4. Breuker, D.: Towards model-driven engineering for big data analytics–an exploratory analysis of domain-specific languages for machine learning. In: 2014 47th Hawaii International Conference on System Sciences. pp. 758–767. IEEE (2014)
5. Callahan, S.P., Freire, J., Santos, E., Scheidegger, C.E., Silva, C.T., Vo, H.T.: Vistrails: visualization meets data management. In: Proceedings of the 2006 ACM SIGMOD international conference on Management of data. pp. 745–747 (2006)
6. Cleary, P., Thomas, D., Bolger, M., Hetherton, L., Rucinski, C., Watkins, D.: Using workspace to automate workflow processes for modelling and simulation in engineering. in 'modsim2015. In: 21st International Congress on Modelling and Simulation', Broadbeach, Queensland, Australia. pp. 669–675 (2015)
7. Dwyer, T., Marriott, K., Wybrow, M.: Dunnart: A constraint-based network diagram authoring tool. In: International Symposium on Graph Drawing. pp. 420–431. Springer (2008)
8. Kamalrudin, M., Hosking, J., Grundy, J.: Maramaaic: tool support for consistency management and validation of requirements. Automated software engineering **24**(1), 1–45 (2017)
9. Khalajzadeh, H., Abdelrazek, M., Grundy, J., Hosking, J., He, Q.: A survey of current end-user data analytics tool support. In: 2018 IEEE International Congress on Big Data (BigData Congress). pp. 41–48. IEEE (2018)
10. Khalajzadeh, H., Abdelrazek, M., Grundy, J., Hosking, J., He, Q.: BiDaML: A Suite of Visual Languages for Supporting End-User Data Analytics. In: 2019 IEEE International Congress on Big Data (BigDataCongress). pp. 93–97. IEEE (2019)
11. Khalajzadeh, H., Abdelrazek, M., Grundy, J., Hosking, J.G., He, Q.: Survey and analysis of current end-user data analytics tool support. IEEE Transactions on Big Data (2019)

12. Khalajzadeh, H., Simmons, A., Abdelrazek, M., Grundy, J., Hosking, J., He, Q.: An end-to-end model-based approach to support big data analytics development. Journal of Computer Languages p. 100964 (2020)
13. Khalajzadeh, H., Simmons, A.J., Abdelrazek, M., Grundy, J., Hosking, J., He, Q.: End-user-oriented tool support for modeling data analytics requirements. In: 2020 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC). pp. 1–4. IEEE (2020)
14. Khalajzadeh, H., Simmons, A.J., Abdelrazek, M., Grundy, J., Hosking, J.G., He, Q.: Visual languages for supporting big data analytics development. In: ENASE. pp. 15–26 (2020)
15. Khalajzadeh, H., Verma, T., Simmons, A.J., Grundy, J., Abdelrazek, M., Hosking, J.: User-centred tooling for the modelling of big data applications. In: MODELS '20: ACM/IEEE International Conference on Model Driven Engineering Languages and Systems. ACM (2020)
16. Kim, C.H., Grundy, J., Hosking, J.: A suite of visual languages for model-driven development of statistical surveys and services. Journal of Visual Languages & Computing **26**, 99–125 (2015)
17. Kim, M., Zimmermann, T., DeLine, R., Begel, A.: Data scientists in software teams: State of the art and challenges. IEEE Transactions on Software Engineering **44**(11), 1024–1038 (2017)
18. Landset, S., Khoshgoftaar, T.M., Richter, A.N., Hasanin, T.: A survey of open source tools for machine learning with big data in the hadoop ecosystem. Journal of Big Data **2**(1), 24 (2015)
19. Li, L., Grundy, J., Hosking, J.: A visual language and environment for enterprise system modelling and automation. Journal of Visual Languages & Computing **25**(4), 253–277 (2014)
20. Ludäscher, B., Altintas, I., Berkley, C., Higgins, D., Jaeger, E., Jones, M., Lee, E.A., Tao, J., Zhao, Y.: Scientific workflow management and the kepler system. Concurrency and computation: Practice and experience **18**(10), 1039–1065 (2006)
21. Minka, T., Winn, J., Guiver, J., Knowles, D.: Infer .net 2.4, 2010. microsoft research cambridge (2010)
22. Moody, D.: The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Transactions on software engineering **35**(6), 756–779 (2009)
23. OMG: Business process model and notation (bpmn) (2011), Retrieved from https://www.omg.org/spec/BPMN/2.0/
24. Portugal, I., Alencar, P., Cowan, D.: A preliminary survey on domain-specific languages for machine learning in big data. In: 2016 IEEE International Conference on Software Science, Technology and Engineering (SWSTE). pp. 108–110. IEEE (2016)
25. Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.F., Dennison, D.: Hidden technical debt in machine learning systems. In: Advances in neural information processing systems. pp. 2503–2511 (2015)
26. Tolvanen, J.P., Rossi, M.: Metaedit+ defining and using domain-specific modeling languages and code generators. In: Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications. pp. 92–93 (2003)
27. Wolstencroft, K., Haines, R., Fellows, D., Williams, A., Withers, D., Owen, S., Soiland-Reyes, S., Dunlop, I., Nenadic, A., Fisher, P., et al.: The taverna workflow

suite: designing and executing workflows of web services on the desktop, web or in the cloud. Nucleic acids research **41**(W1), W557–W561 (2013)
28. Zhang, A.X., Muller, M., Wang, D.: How do data science workers collaborate? roles, workflows, and tools. arXiv preprint arXiv:2001.06684 (2020)