2020 IEEE Symposium on Visual Languages and Human-centric Computing, Dunedin, New Zealand, 10-14 August 2020.

End-User-Oriented Tool Support for Modeling Data Analytics Requirements

Hourieh Khalajzadeh
Faculty of Information Technology
Monash University
Melbourne, Australia
hourieh.khalajzadeh@monash.edu

Andrew J. Simmons
Applied Artificial Intelligence Institute (A^2I^2)
Deakin University
Melbourne, Australia
a.simmons@deakin.edu.au

Mohamed Abdelrazek
School of Information Technology
Deakin University
Melbourne, Australia
mohamed.abdelrazek@deakin.edu.au

John Grundy
Faculty of Information Technology
Monash University
Melbourne, Australia
john.grundy@monash.edu

John Hosking
Faculty of Science
University of Auckland
Auckland, New Zealand
j.hosking@auckland.ac.nz

Qiang He
School of Software and Electrical Engineering
Swinburne University
Melbourne, Australia
qhe@swin.edu.au

***Abstract*—Big data and analytics are increasingly used in different domains to gain insights and to improve decision-making. Developing big data analytics solutions is a complex task involving multidisciplinary teams and users - with no data science and programming background - to professional data scientists and software engineers. Different stakeholders work with a variety of data types, tasks and concepts in different languages from high-level domain concepts to low level programming languages and technical concepts. In order to advance the level of abstraction beyond low-level data analysis technical details, we demonstrate our BiDaML tool. BiDaML brings all stakeholders around one tool to specify, model and document their big data applications using a novel set of domain-specific visual languages (DSVLs).**

***Index Terms*—big data analytics, domain-specific visual languages, BiDaML**

I. Introduction

Big data analytics, as a very active area [1], [2] in research and industry, brings together stakeholders with a variety of skills, knowledge, and technical knowledge and languages to communicate and collaborate in order to improve decision-making. Based on Gartner's technical professional advice [3], a data analytics project involves many steps e.g. classifying the problem, acquiring data, processing data, modeling the problem, validation and execution, and finally deploying. In fact, data processing and Machine Learning (ML) tasks are only a small component in the building blocks necessary to build real-world deployable data analytics systems [4].

Currently, data analytics tools, such as Azure ML Studio, Amazon AWS ML, and Google Cloud ML, as reviewed in [5], [6] are being used to automate the machine learning part of the project. However, these tools rarely focus on capturing and modeling the end-to-end data analytics development lifecycle, requirements and the improvement of end-to-end analytics processes [7]. This causes a lack of traceability, poor communication between stakeholders, and much manual effort to implement solutions. Better integration of data science, data technology and process science will enable end-users and data scientists to model the problem, extract insights/patterns, and develop predictive models.

We have identified and specified six main challenges in multidisciplinary data analytics software development teams: (1) Domain experts, business analysts and business managers do not have a background in data science and programming. (2) Data analysts, data scientists and software engineers do not have domain knowledge. (3) Data scientists lack software engineering expertise. (4) Team members lack a common language to describe solutions. (5) Evolution of the solution is poorly supported. (6) Re-using existing solutions is difficult. A challenge reported by data scientists in [8] is that it is hard to convey the resulting insights to leaders and stakeholders in an effective manner and convincing teams that data science approaches are in fact helpful. Moreover, results of a large-scale survey [9] 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.

To make data analytics solution design more accessible to end users and facilitate high-level dialogue with expert data scientists and software engineers, we present BiDaML [10], [11], a tool consisting of a set of domain specific visual models at different levels of abstraction. BiDaML provides modeling and collaboration tool support between different users and can be translated into big data solutions using Model-Driven Engineering (MDE)-based partial code generation. All users and stakeholders involved in the project, including domain experts, business managers, data scientists, software engineers, etc can work together to define, confirm and make agreements on different tasks assigned to them through a collaborative drag and drop interface.

We demonstrate our approach using a real-world traffic data analysis example detailed in [10]. This research has published in [10] and [11]. Showcasing this work as a novel way of using domain specific visual languages for data analytics applications would demonstrate its applicability to industry sponsors, help in building community interest and will lead to new collaborations. We will present the tool as a demo in the conference.

II. Our Approach

In order to model such data analytics applications, from business needs, requirements analysis and design, to development of the final solution, we present BiDaML (Big Data Analytics Modeling Languages). Our BiDaML tool consists of five different diagrams: two high level diagrams, brainstorming and process diagrams, and three low level diagrams, data, technique, and deployment diagrams. Each of the diagrams includes a collection of visual notations for modeling different objects, their relationships and rules. Users, based on their responsibilities, work with the related diagrams through dragging and dropping notations and connecting them through relationships. They can then collaborate and communicate using diagrams and visual notations as a common language. Once the modeling part is completed, users can decide to create different reports and generate a range of boilerplate Python code from the models. They can also add their own code or modify generated code for reusability purposes. 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). BiDaML supports all of these groups in designing and developing a data analytics related project. We used MetaEdit+ 5.5 [12] to develop our domain specific visual languages and tool. Here, we briefly describe different BiDaML diagrams as well as more details of BiDaML tool and its visual notations [13]. Notations and relationships can be dragged and dropped to the designing environment and used to generate Python code structures for the data analytics problem to assist in implementing the solution. Details of the diagrams and their notations can be found in [11]

III. Example Usage

We worked with transport researchers and used our method to specify the intended VicRoads software solution workflow. We performed in-depth interviews with the project leader and traffic modeling expert, then used our tool to document the entire data analytics workflow. This allowed us to assist in the formation of an alternative software solution that made better use of the systems and services already available. As BiDaML forces the user to consider all phases of the project, the modeling process helped in revealing the gaps. A brainstorming diagram listing all the tasks, and high-level information is shown in Appendix Fig. 2. A process diagram generated for the example and a sample of the generated report for diagram explanation and clarification is shown in Appendix Fig. 3. Process diagram lists all the organizations and users involved, e.g., VicRoads, Monash Transport Group, eResearch, etc, with a detailed list of tasks and conditions. For any of the tasks, the responsible user can create a sub-graph to add more detailed information, such as techniques used, data items generated, and expected outputs. Samples of technique and data diagrams created in BiDaML tool are shown in Fig. 1. Users can choose to generate Python code, reports, or BigML API code from the brainstorming diagram that includes details of brainstorming, data and technique diagrams. Fig. 2 in Appendix also shows snippets of the Python code generated from the brainstorming diagram and Fig. 3 in Appendix shows a sample of the report generated from the process diagram. Finally, Fig. 4 shows an overview of all the diagrams and a report exported to Word from a hierarchy of all the diagrams and their explanations.

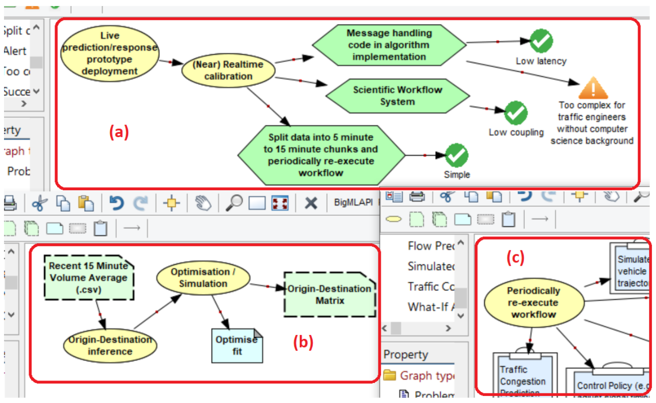

(a)
Live prediction/response prototype deployment
(Near) Realtime calibration
Message handling code in algorithm implementation
Scientific Workflow System
Split data into 5 minute to 15 minute chunks and periodically re-execute workflow
Low latency
Too complex for traffic engineers without computer science background
Low coupling
Simple
BigMLAPI
(b)
Recent 15 Minute Volume Average (.csv)
Optimisation / Simulation
Origin-Destination Matrix
Origin-Destination inference
Optimise fit
(c)
Periodically re-execute workflow
Flow Pred
Simulated
Traffic Co
What-If
Property
Graph type
Traffic Congestion
Control Policy (e.g

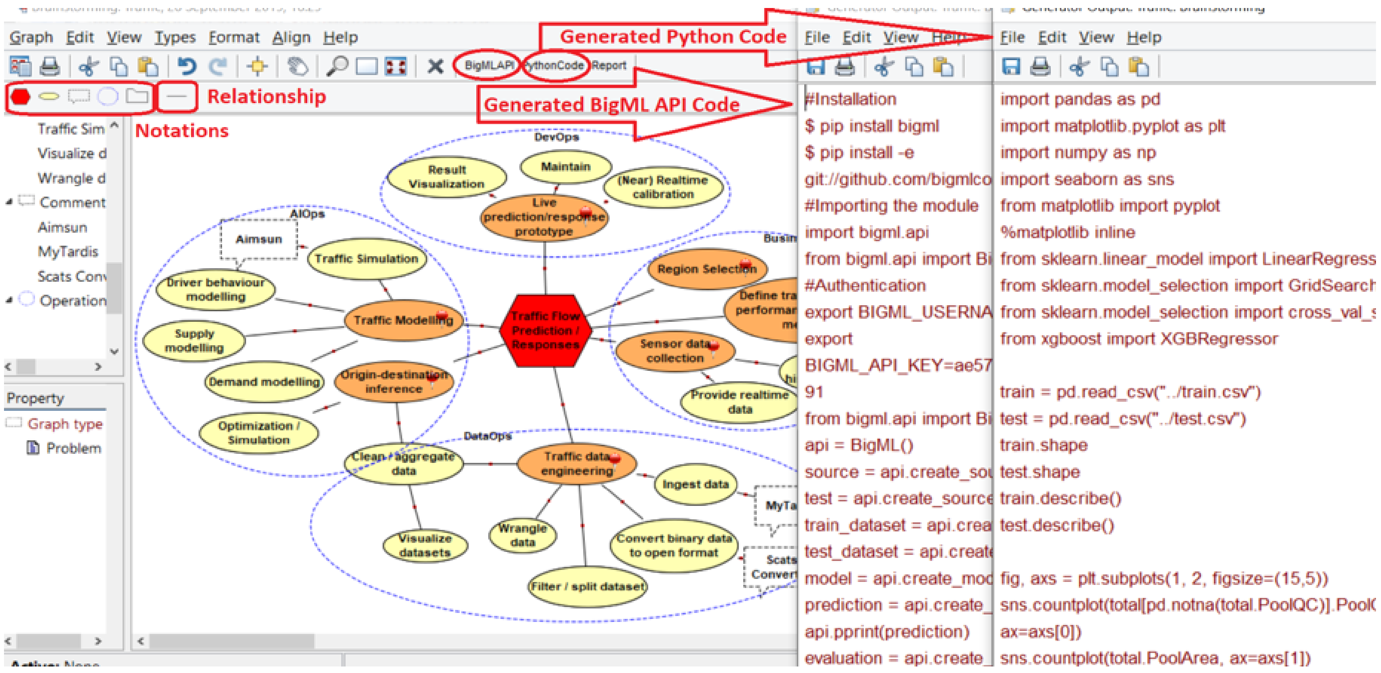
Fig. 1. (a) Technique and (b, c) Data Diagrams Created in BiDaML Tool for the Traffic Analysis Example

IV. Evaluations

We have evaluated the usability and suitability of BiDaML using an extensive Physics of Notations evaluation [14], an empirical evaluation with a group of 12 data analysts, data scientists and software engineers, and a detailed cognitive walkthrough of the tool with 3 data scientists and 2 software engineers. Details of the evaluations can be found in [11], [13].

V. Conclusion and Future Work

We designed our BiDaML tool to support big data analytics requirements specification and modeling. BiDaML uses five high and low level diagrammatic domain specific visual languages to ensure diverse stakeholders can effectively communicate and capture information and requirements during the process of designing data analytics solutions. Furthermore, our testing environment supports code and report generation to help users getting descriptive reports and Python code templates based on the created diagrams. In the future, we will provide a web-based version of BiDaML to enable easier integration with other tools and better access for different stakeholders.



```
#Installation
$ pip install bigml
$ pip install -e
git://github.com/bigmlco
#Importing the module
import bigml.api
from bigml.api import Bi
#Authentication
export BIGML_USERNA
export
BIGML_API_KEY=ae57
91
from bigml.api import Bi
api = BigML()
source = api.create_so
test = api.create_source
train_dataset = api.crea
test_dataset = api.create
model = api.create_moc
prediction = api.create_
api.pprint(prediction)
evaluation = api.create_
```

```
import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
import seaborn as sns
from matplotlib import pyplot
%matplotlib inline
from sklearn.linear_model import LinearRegress
from sklearn.model_selection import GridSearch
from sklearn.model_selection import cross_val_s
from xgboost import XGBRegressor

train = pd.read_csv("../train.csv")
test = pd.read_csv("../test.csv")
train.shape
test.shape
train.describe()
test.describe()

fig, axs = plt.subplots(1, 2, figsize=(15,5))
sns.countplot(total[pd.notna(total.PoolQC)].Pool(
ax=axs[0])
sns.countplot(total.PoolArea, ax=axs[1])
```

Fig. 2. Brainstorming Diagram Created in BiDaML Tool for the Traffic Analysis Example and Snippets of the Generated Python Code.

ACKNOWLEDGMENT

Support for this research from ARC Discovery grant DP170101932 and ARC Laureate Program FL190100035 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," in *2016 IEEE International Conference on Software Science, Technology and Engineering (SWSTE)*, pp. 108–110, IEEE, 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. 1, p. 24, 2015.

[3] C. E. Sapp, "Preparing and architecting for machine learning," *Gartner Technical Professional Advice*, pp. 1–37, 2017.

[4] D. Sculley, G. Holt, D. Golovin, E. Davydov, T. Phillips, D. Ebner, V. Chaudhary, M. Young, J.-F. Crespo, and D. Dennison, "Hidden technical debt in machine learning systems," in *Advances in neural information processing systems*, pp. 2503–2511, 2015.

[5] H. Khalajzadeh, M. Abdelrazek, J. Grundy, J. Hosking, and Q. He, "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.

[6] H. Khalajzadeh, M. Abdelrazek, J. Grundy, J. G. Hosking, and Q. He, "Survey and analysis of current end-user data analytics tool support," *IEEE Transactions on Big Data*, 2019.

[7] W. Van der 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.

[8] M. Kim, T. Zimmermann, R. DeLine, and A. Begel, "Data scientists in software teams: State of the art and challenges," *IEEE Transactions on Software Engineering*, vol. 44, no. 11, pp. 1024–1038, 2017.

[9] A. X. Zhang, M. Muller, and D. Wang, "How do data science workers collaborate? roles, workflows, and tools," *arXiv preprint arXiv:2001.06684*, 2020.

[10] H. Khalajzadeh, A. Simmons, M. Abdelrazek, J. Grundy, J. Hosking, and Q. He, "Visual Languages for Supporting Big Data Analytics Development," in *15th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE)*, 2020.

[11] H. Khalajzadeh, A. Simmons, M. Abdelrazek, J. Grundy, J. Hosking, and Q. He, "An end-to-end model-based approach to support big data analytics development," *Journal of Computer Languages*, p. 100964, 2020.

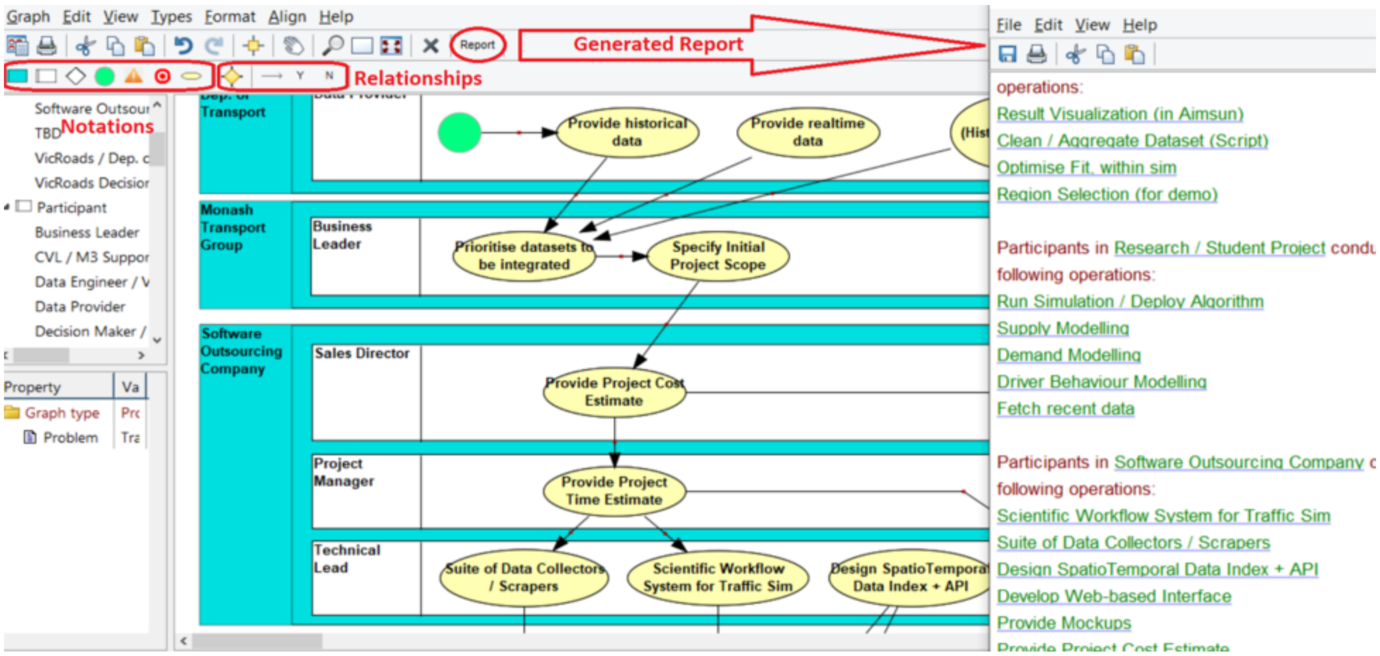
[12] J.-P. Tolvanen and M. Rossi, "Metaedit+ defining and using domain-specific modeling languages and code generators," in *Companion of the 18th annual ACM SIGPLAN conference on Object-oriented programming, systems, languages, and applications*, pp. 92–93, 2003.

[13] H. Khalajzadeh, M. Abdelrazek, J. Grundy, J. Hosking, and Q. He, "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.

[14] D. Moody, "The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering," *IEEE Transactions on software engineering*, vol. 35, no. 6, pp. 756–779, 2009.

APPENDIX

Brainstorming, process and overview diagrams created for the traffic analysis project as well as the code snippet and reports generated from the diagrams. The generated traffic documentation report is available at http://bidaml.visualmodel.org/cases/traffic.pdf



operations:
Result Visualization (in Aimsun)
Clean / Aggregate Dataset (Script)
Optimise Fit, within sim
Region Selection (for demo)

Participants in Research / Student Project condu
following operations:
Run Simulation / Deploy Algorithm
Supply Modelling
Demand Modelling
Driver Behaviour Modelling
Fetch recent data

Participants in Software Outsourcing Company c
following operations:
Scientific Workflow System for Traffic Sim
Suite of Data Collectors / Scrapers
Design SpatioTemporal Data Index + API
Develop Web-based Interface
Provide Mockups
Provide Project Cost Estimate

Fig. 3. Process Diagram Created in BiDaML Tool for the Traffic Analysis Example and a Report Generated from the Process Diagram.

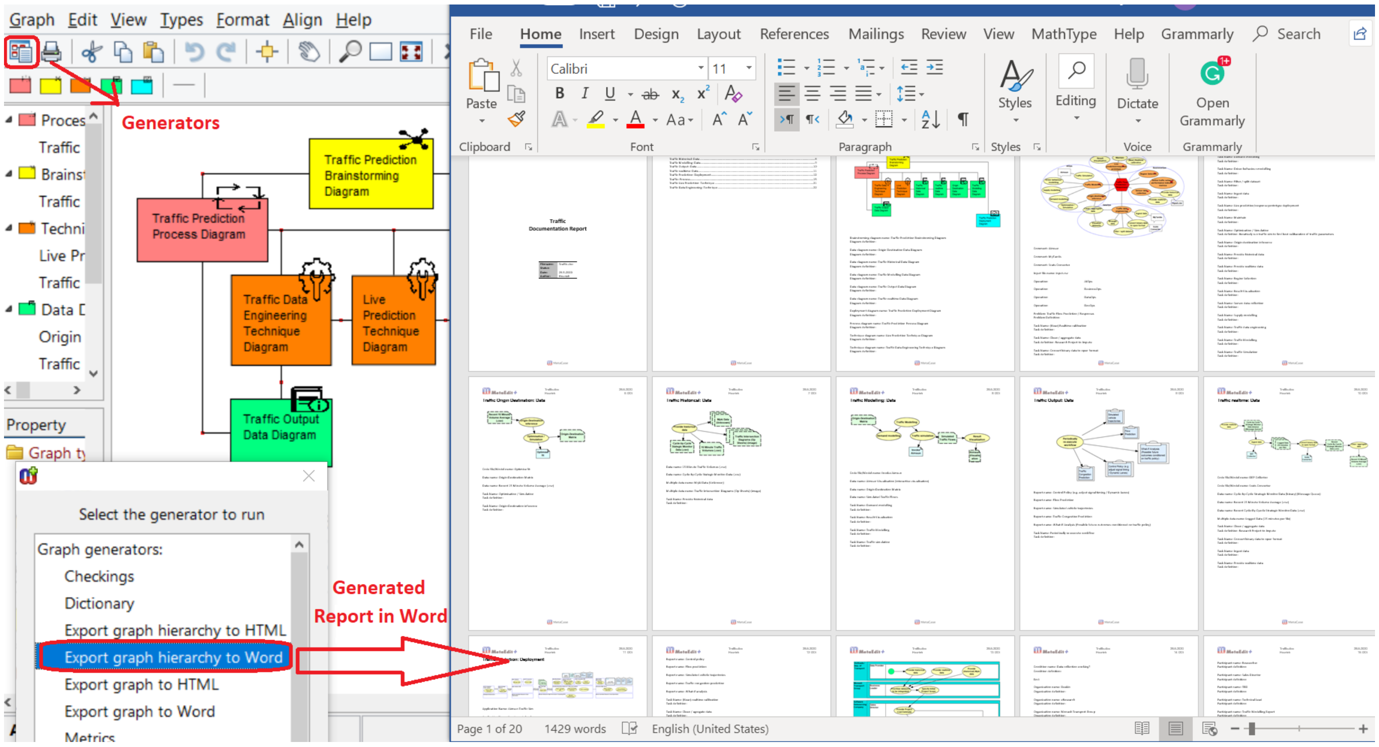

Graph Edit View Types Format Align Help
Generators
Proces
Traffic
Brains
Traffic
Techni
Live Pr
Traffic
Data D
Origin
Traffic
Property
Graph ty
Traffic Prediction Brainstorming Diagram
Traffic Prediction Process Diagram
Traffic Data Engineering Technique Diagram
Live Prediction Technique Diagram
Traffic Output Data Diagram
Select the generator to run
Graph generators:
Checkings
Dictionary
Export graph hierarchy to HTML
Export graph hierarchy to Word
Export graph to HTML
Export graph to Word
Metrics
Generated Report in Word
File Home Insert Design Layout References Mailings Review View MathType Help Grammarly Search
Calibri 11
Paste
Clipboard
Font
Paragraph
Styles
Editing
Dictate
Voice
Open Grammarly
Grammarly
Traffic Documentation Report
Page 1 of 20 1429 words English (United States)

Fig. 4. 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.