Comparing the Usability of two Multi-Agents Systems DSLs: SEA_ML++ and DSML4MAS *Study Design*

João Silva*, Ankica Barišić*, Vasco Amaral*, Miguel Goulão*,
Baris Tekin Tezel†, Omer Faruk Alaca‡, Moharram Challenger‡, and Geylani Kardas‡
*Universidade NOVA de Lisboa, NOVA LINCS, DI, FCT, Lisboa, Portugal
†Dokuz Eylul University, Izmir, Turkey
‡Ege University, International Computer Institute, Izmir, Turkey
Email: (ji.silva | a.barisic)@campus.fct.unl.pt, (vma | mgoul)@fct.unl.pt
baris.tezel@deu.edu.tr, omerfarukalaca@gmail.com, (moharram.challenger | geylani.kardas)@ege.edu.tr

***Abstract*—Context: The "Physics of Notations" (PoN) supports a systematic improvement of the cognitive effectiveness of visual modelling languages. Problem: PoN focuses on the concrete syntax of a language, building on a predefined abstract syntax. We should also consider the abstract syntax of a language when developing efforts to improve it by choosing the most adequate language constructs (concepts and their relationships). We instantiate this challenge by comparing two Multi-Agent Systems Domain Specific Languages: *SEA_ML++* and *DSML4MAS*, and assessing the extent to which their respective constructs affect the developer experience. Method: We will perform a quasi-experiment for comparing how practitioners use both languages to solve similar modelling challenges. The experiment will have a cross-over within-subjects design and will focus on the extent to which the different language constructs impact on developer experience. These tasks will be monitored, so that we can assess their success and effort involved, including eye-tracking information. Results: This paper reports on the planned study design for this empirical comparison of two DSLs for MAS.**

I. Introduction

In the last two decades, technologies like modelling workbenches made it easier to design, prototype and deploy diagrammatic languages used more often for capturing abstractions in modelling. Extensive experience with the development of domain-specific languages (DSLs) lead to a new discipline, Software Languages Engineering (SLE), with the goal of making systematic the process of developing a software language.

SLE follows an iterative life-cycle [1], [2] that starts with domain analysis, followed by language design, implementation and evaluation. Unfortunately, the first and the last steps are still not at a mature phase. Besides taking into account the evaluation of expressiveness of a given language, the language design (coverage of the language goals) needs to make use of empirical studies to assess the language usability.

The "Physics of Notations" (PoN) [3] created a valuable framework to evaluate the language's concrete syntax, and is extensively used to support a systematic improvement of the cognitive effectiveness of visual modelling languages with a fixed abstract syntax in some language metamodel or grammar.

Improving the concrete syntax is very important, but we should also consider the abstract syntax of a language. We should be able to choose and validate the adequate language constructs (concepts and their relationships) and the models (or language sentences) we can express with those. However, there is a lack of guidelines reported in the literature for this new level of assessment that could give a languages engineer a "recipe" for doing this sort of evaluations. These would be valuable when developing and improving a given language.

We are currently improving a Multi-Agent Systems (MAS) DSL: SEA_ML++ [4], [5]. In this context, we are planning an empirical comparison with another MAS DSL: DSML4MAS [6], to assess the extent to which these DSLs respective constructs, and combinations, affect language usability.

We will perform a quasi-experiment for comparing how practitioners use both languages to solve similar modelling challenges. We will have a cross-over within-subjects design with a focus on how the different language constructs impact on their usability by modellers. We will monitor these tasks to assess their success and effort involved, including eye-tracking information and a usability questionnaire (SUS [7]).

This paper is organised as follows: Section II describes DSMLs for MAS languages as the object of our case study. Section III presents the planned quasi-experiment, followed by a discussion in Section IV. Section V summarises this paper.

II. Background

A. Multi-Agent Systems DSMLs

Software agents are autonomous entities which contain intelligence that serves for solving their selfish or common problems and to achieve certain goals. The study of Multi-Agent Systems (MASs) focuses on those systems in which many intelligent agents interact with each other. In agent-oriented software engineering (AOSE), the application of model-driven development (MDD) and the use of domain-specific modelling languages (DSMLs) for MAS development are quite popular since the implementation of MAS is naturally

complex, error-prone and costly due to the autonomous and proactive properties of the agents [8].

In the last decade, several MAS modelling languages and DSMLs (e.g. [4], [6], [9], [10], [11], [12]) were proposed to support development of MASs. For example, DSML4MAS [6] introduces a general MAS metamodel with various viewpoints that enable the development of MAS for many application domains. A DSL is introduced in [10] to provide a language for the development of mobile agents. In addition, [11] introduces a modelling language enabling the model-driven development within the scope of Prometheus methodology for agent development. In [4] and [13] a graphical DSML (called SEA_ML) and textual DSL (called SEA_L) are proposed for MAS working in semantic web environments including 8 viewpoints. MAS-ML 2.0 [12] is a modelling language which supports the MAS modelling with different agent architectures such as: Simple Reflex Agents, Model-Based Reflex Agents, Goal-Based Agents and Utility-Based Agents. DSML4BDI [14] is another modelling language specific for Jason agent programming language. In [15], the authors propose a tool-supported development method that applies MDD techniques to design and implement agents based on the belief-desire-intention architecture with a sophisticated plan selection process.

B. The "Physics of Notations" (PoN)

Moody proposed the "Physics of Notations" [3] to support the construction of more effective software languages. A major concern is on how to evaluate the cognitive effectiveness of visual languages (see, for example [16]). The framework concentrates on the physical properties (concrete syntax) of the symbols and not on their structure (abstract syntax) or semantics (ignoring semantics of both the ontological and language target semantic Domain). In figure 1, we present the dimensions at the instance level (in grey) that are explored by the current work. Here we study the composition of visual elements and its structure to form sentence instances. This figure is adapted from [17], where the authors refer to their focus on the top left corner (Visual Notations).

C. Related studies

Most of the available DS(M)Ls proposed for MASs have been evaluated by just providing a case study demonstrating how the related language can be used for design and implementation of MAS. A quantitative analysis and/or qualitative evaluation considering e.g. the development time performance, generation performance, and/or the usability of the language are not considered in these studies.

In [18], we proposed an evaluation framework which provides the systematic assessment of both the language constructs and the use of agent DSMLs according to various dimensions and criteria. The study also provides an assessment of SEA_ML [4]. However, it does not take into account the effect of language constructs in the developer's modelling process while using the languages. This evaluation framework is adopted in [14], [19] and [15] for the assessment of the proposed MAS DSMLs. Another MAS DSML evaluation feature exists in [20] for a textual DSL, called JADEL, providing four abstractions, namely agents, behaviours, communication ontologies, and interaction protocols to the well-known JADE agent development framework. However, the study evaluates solely JADEL's code generation performance.

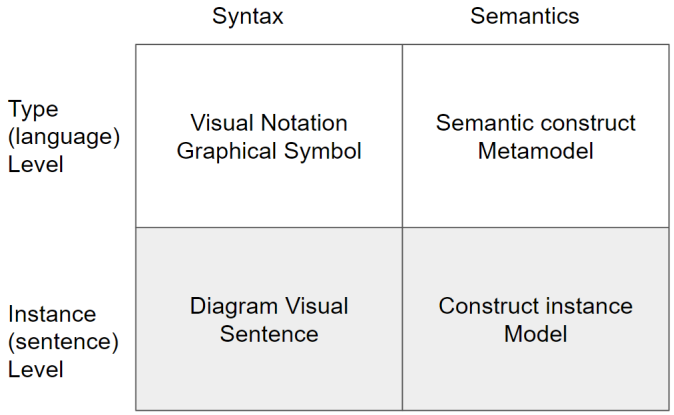

Fig. 1. Dimensions in grey explored by the current work. Adapted from [17].

In recent years, we have seen several studies to identify language improvement opportunities, identifying problems with their concrete syntax and how they impact developer experience. These studies have covered a diversity of languages, including UML [16], [21], BPMN [22], [23], KAOS [24], [25], [26], *i** [17], [27], [28], [29], OutSystems BPT [30], and SEA_ML++ [5]. Some of these languages were also analysed from the perspective of the impact of diagram layout in the understandability of models, namely UML [31], [32], [33] and *i** [34]. Other studies have compared alternative DSLs for a similar domain (e.g. Lego Mindstorms *vs.* Gyro [35]).

III. Experiment planning

This section describes the experimental planning for this evaluation. Further details, including documentation and evaluation materials, can be found in this paper's companion site[1].

A. Goals

Broadly, we are interested in assessing the usability of two MAS DSMLs, SEA_ML++ [4], [5] and DSML4MAS [6] in the context of solving modelling challenges. We use the Goal-Question-Metric [36] template to describe our research goals:

Our first goal (**G1**) is to *analyse* the effect of using SEA_ML++ or DSML4MAS, *for the purpose of* evaluation, *with respect to* the **correctness** with which a developer models a MAS system, *from the viewpoint of* researchers, *in the context of* an experiment conducted with graduate students from Universidade Nova de Lisboa and Ege University. Our second goal (**G2**) is to *analyse* the effect of using SEA_ML++ or DSML4MAS, *for the purpose of* evaluation, *with respect to* the **speed** with which a developer models a MAS system, *from*

[1]https://sites.google.com/fct.unl.pt/hufamo2018masstudydesign/home

the viewpoint of researchers, *in the context of* an experiment conducted with graduate students from Universidade Nova de Lisboa and Ege University. Our third goal is to *analyse* the effect of using SEA_ML++ or DSML4MAS, *for the purpose of* evaluation, *with respect to* the **rework** involved in modelling a MAS system, *from the viewpoint of* researchers, *in the context of* an experiment conducted with graduate students from Universidade Nova de Lisboa and Ege University.

B. Experimental units

The participants in this evaluation will be professional software developers from Lisbon, and graduate students trained in several universities, namely Universidade Nova de Lisboa, Instituto Superior Técnico and Instituto Universitário de Lisboa. We will have a close replica of these evaluations with subjects from the Ege University, in Turkey. We will use convenience sampling to recruit participants, in all these sites. Each participant will be randomly assigned to one of four groups, keeping a balanced sample on each of the four groups.

C. Tasks

Each subject will be asked to perform two modelling tasks: one using SEA_ML++, the other DSML4MAS. The two tasks will have similar complexity and will consist in modelling a MAS system from a natural language description of that system. They will use an Eclipse-based editor, which is essentially similar. The editor only varies in the language constructs and composition rules offered to participants, depending on which language is being used. The participant will make his best to correctly model a system with each of these languages. Regardless of the particular development task, the user will see a split screen, with the majority of it being occupied by the editor, on the left side, and a smaller portion with the case study the user is to model, on the right side. Figure 2 presents the starting point for performing the task with SEA_ML++. Both the textual description of the model to build, on the right side, and the editor, on the left, are sized so that the whole exercise can be performed without the need to resizing or scrolling any window. Figure 3 presents the starting point for performing the task with DSML4MAS. Again, window sizes will be similar, and no need to resize or scroll is expected. Indeed, participants will be instructed not to change windows sizes, to increase comparability among sessions. After performing both modelling tasks, participants are asked to answer a System Usability Scale (SUS) test on SEA_ML++ and DSML4MAS.

The tasks involve three different viewpoints: the agent viewpoint, the MasAndOrg viewpoint and the Interaction viewpoint. For the sake of illustration, we provide here the agent viewpoint, in both languages, Figure 4 (SEA_ML++) and Figure 5 (DSML4MAS). Further materials, including large-sized versions of these diagrams can be found in our companion site.

D. Hypotheses, parameters and variables

For each of our high-level goals, we define the null (H_0) and alternative (H_1) hypotheses. Similar hypotheses can be written for contrasting *SEA_ML++* with *DSML4MAS* in terms of their effect on **correctness**, **speed**, amount of **rework**, visual **effort**, and perceived **usability** of the languages.

$H_{0Correctness}$: Using *SEA_ML++* rather than *DSML4MAS* does not influence the produced models **correctness**.
$H_{1Correctness}$: Using *SEA_ML++* rather than *DSML4MAS* influences the produced models **correctness**.
H_{0Speed}: Using *SEA_ML++* rather than *DSML4MAS* does not influence the **speed** of model production.
H_{1Speed}: Using *SEA_ML++* rather than *DSML4MAS* influences the **speed** of model production.
$H_{0Rework}$: Using *SEA_ML++* rather than *DSML4MAS* does not influence the amount of **rework** during model production.
$H_{1Rework}$: Using *SEA_ML++* rather than *DSML4MAS* influences the amount of **rework** during model production.
$H_{0Effort}$: Using *SEA_ML++* rather than *DSML4MAS* does not influence the **visual effort** involved during model production.
$H_{1Effort}$: Using *SEA_ML++* rather than *DSML4MAS* influences the **visual effort** involved during model production.
$H_{0Usability}$: Using *SEA_ML++* rather than *DSML4MAS* does not influence the perceived **effort** involved during model production.
$H_{1Usability}$: Using *SEA_ML++* rather than *DSML4MAS* influences the perceived **usability** of model production.

1) Assessing correctness: For each of the proposed challenges, we have a "gold standard" model defined in both languages, with which we can compare the models built by our participants. The correctness of the proposed models is measured in terms of their *precision*, *recall*, and *F-measure*, defined here as follows:

- *precision* – the percentage of model elements and relationships in the model built by the participant that correctly address the challenge (even if the participant chose alternative ways of modelling the MAS when compared to the *"gold standard"*, as long as they are considered correct.
- *recall* – the percentage of model elements and relationships in the *"gold standard"* model that are correctly addressed by the participant's model.
- *F-measure* – a measure that combines precision and recall, computed as $\frac{2*(Precision*Recall)}{(Precision+Recall)}$; this measure provides an harmonic mean of precision and recall.

Higher values of *precision*, *recall* and the *F-measure* support the claim for higher correctness, with 0 representing totally incorrect and 1 totally correct models.

2) Assessing speed: We assess *speed* by measuring the amount of time (measured in seconds) taken by our participant to build a MAS model. Lower values of this metric support the claim of better language usage efficiency.

3) Assessing rework: We assess *rework* by identifying, through the analysis of the model building screencast, the moments where the participant discarded parts of the solution he was building (*e.g.* by removing a previously added element, or relationship).

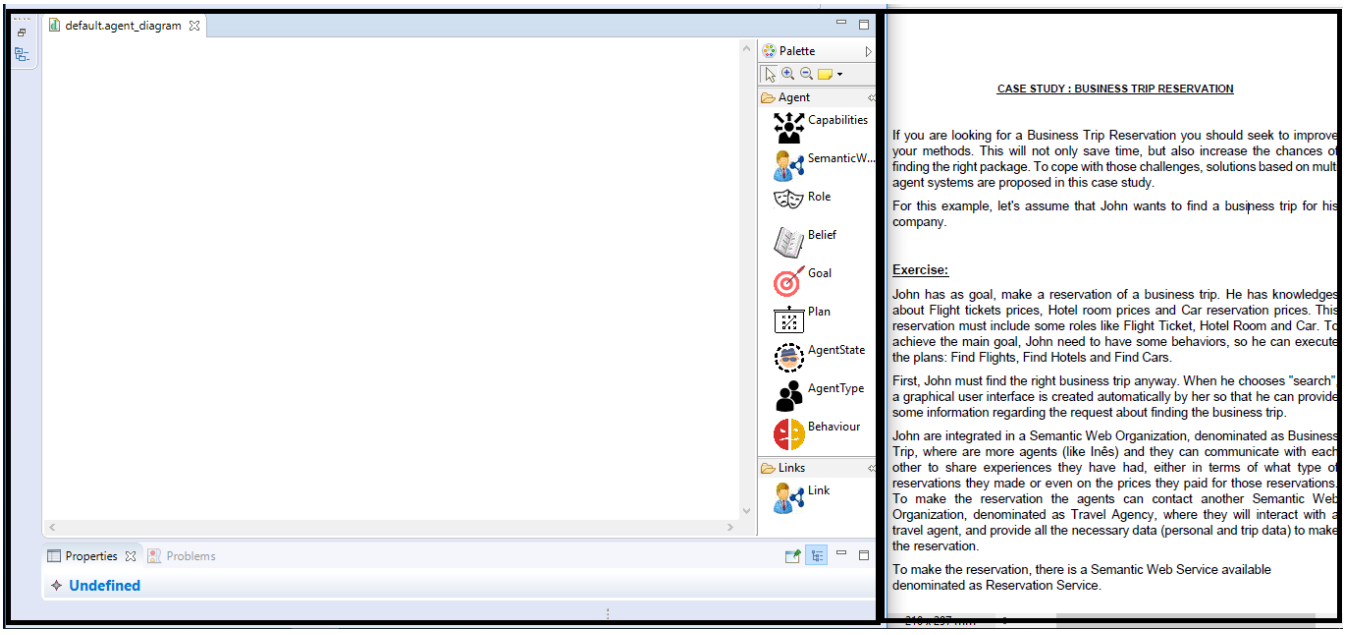

Fig. 2. Environment for performing the SEA_ML++ modelling task.

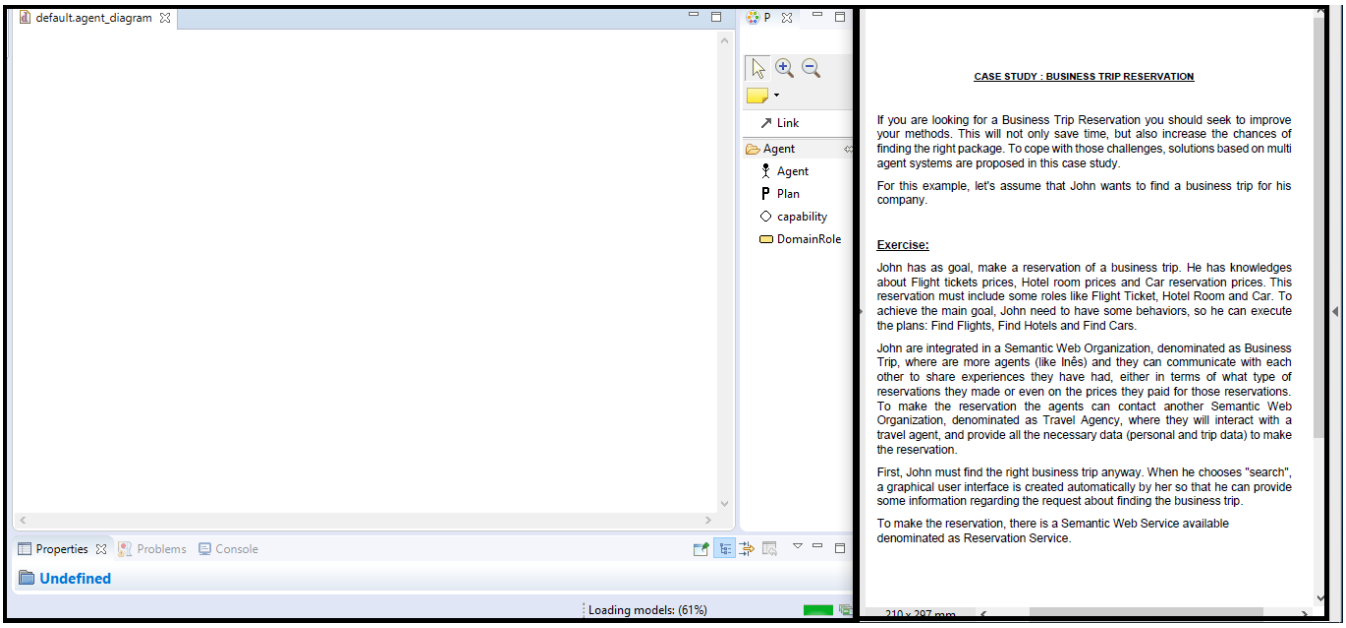

Fig. 3. Environment for performing the DSML4MAS modelling task.

4) Assessing visual effort: We assess *visual effort* using eye tracking data collected through the screencast. In particular, we will analyse heat-maps of the screencasts to compare, for example, whether there are significant differences in the amount of time spent exploring modelling options available in the language toolbar and whether there is some relationship between these exploring moments and patterns of rework.

5) Assessing the perceived usability: We assess the *perceived usability* through an SUS questionnaire which provides a SUS score from 0 to 100, with an average value of 68 [7]. Higher values support the claim for a better usability.

E. Design

Table I outlines our cross-over within subjects design, with two challenges from different domains (D1 and D2), but with a similar complexity. Each participant will solve those two challenges using a different language in each of them. To cancel learning effects, we will balance the number of times the participants start with each of the languages and each of the problems. In other words, we will balance the participants in groups A, B, C and D.

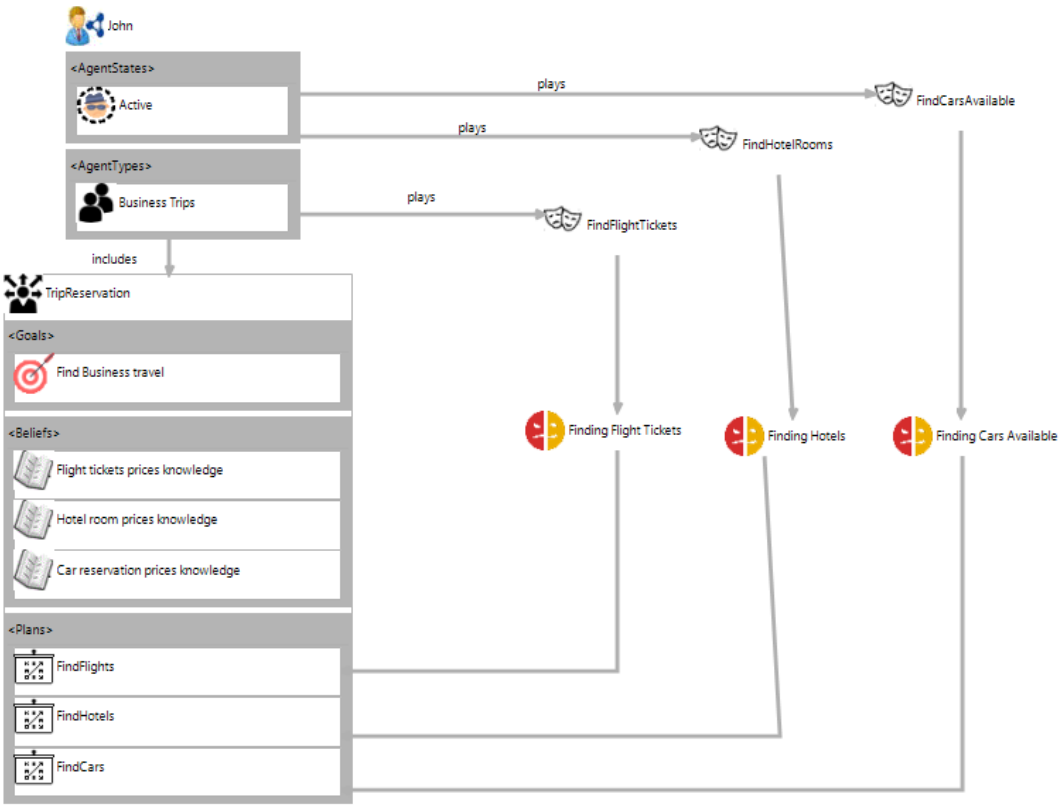

Fig. 4. SEA_ML++ agent viewpoint possible solution.

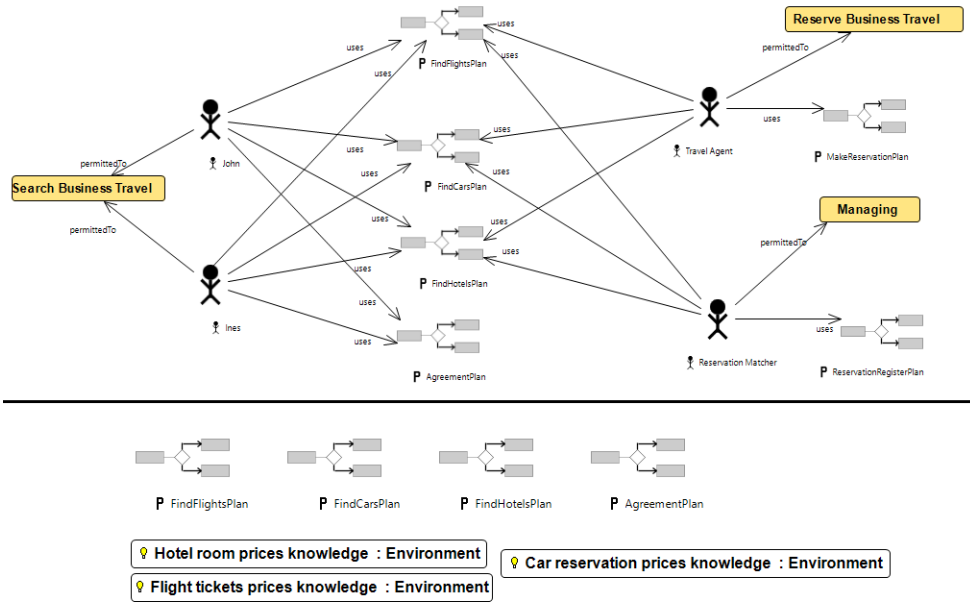

Fig. 5. DSML4MAS agent viewpoint possible solution

TABLE I
EXPERIMENTAL DESIGN AND TASKS SEQUENCE

Gr	Ltr	Dem	Tut	Cal	Challenge 1	Challenge 2	SUS
A	✓	✓	✓	✓	D1 / SEA_ML++	D2 / DSML4MAS	✓
B	✓	✓	✓	✓	D1 / DSML4MAS	D2 / SEA_ML++	✓
C	✓	✓	✓	✓	D2 / SEA_ML++	D1 / DSML4MAS	✓
D	✓	✓	✓	✓	D2 / DSML4MAS	D1 / SEA_ML++	✓

F. Procedure

As depicted in Table I, before starting, each participant will sign a letter of consent, adapted from [37] and fill in a demographic questionnaire, so that we record information about our participants, including country, age, genre, academic level, previous experience with MAS and, in particular, with each of the two analysed languages. This is followed by viewing a short tutorial on both languages. Then, the subject will perform an eye tracking device calibration, so that the eye tracking data of the session can be recorded with precision. To maximise eye tracking recording precision, participants will be comfortably seated at a distance of about 60cm from a full HD 22 inch monitor and instructed not to move much during the whole session. An EyeTribe eye tracker [2] will be placed below the monitor. The participant will also have a keyboard and a mouse, to be able to build a MAS model. After these preparatory tasks, the experiment itself can start. During the whole session, a screencast of the contents of the screen will be recorded. Furthermore, eye tracking data will also be collected, in sync with the screencast of the session. The participant will have no time limit to finish his task, but our pilot sessions point to a duration of about 20 minutes to perform the given tasks. Finally, the subject answers a SUS test [7], so that we may contrast his opinions on the usability of SEA_ML++ and of DSML4MAS.

G. Analysis procedure

The data collected during the experiment sessions will be analysed using a combination of automated data collection for the questionnaires and eye tracking data, with manual data collection, combining the visual inspection of the screencast with the synchronised recorded audio of the think aloud protocol. Concerning descriptive statistics, we will normally collect the following ones, adjusting the actual set of descriptive statistics to the scale type (nominal, ordinal, interval or ratio) of each variable: number of cases, mean, median, mode, standard deviation, skewness, kurtosis, the *p-value* of the Shapiro-Wilk normality test. We will then use appropriate statistics tests. For example, we plan to use the Welch t test, which is a more robust alternative to the t-test [38]) to compare the distributions of correctness obtained with SEA_ML++ *vs.* DSML4MAS. The statistics analysis will be run using SPSS [3].

1) Correctness: The data concerning *correctness* will be collected through visual inspection of the solutions created by the participants in our study. This implies a qualitative assessment of those solutions in a process which is somewhat similar to grading the result of a modelling exercise, in an academic context, following the criteria detailed in section III-D1. We will then compute descriptive statistics for the collected metrics and test for significant differences between the level of correctness achieved with each language.

2) Speed: The data concerning *speed* will be collected during the visual inspection of the screencast of the sessions, by annotating the timestamps marking the begin and the end of each task. We will then compute descriptive statistics for the collected metrics and test for significant differences between the duration of the tasks using each language.

3) Rework: The data concerning *rework* will be collected through visual inspection of the screencast. In particular, we will collect and annotate with timestamps events of creation, deletion, or update of model elements and associations among those elements. This will provide us with a timeline of the model construction process for further analysis. Concerning rework, we will analyse activities that undo previous work (e.g. a model element that was previously added to the model and now is deleted). This will allow identifying when the participant is convinced he made a mistake and decides to backtrack. Ultimately, we will explore whether the different languages lead to different levels of rework, both in general, and with particular sub-groups of participants, divided according to their background (e.g. by level of expertise with MAS).

4) Visual effort: The eye tracking data is collected automatically during the execution of the experiment. This produces a time series of eye tracking events, namely fixations and saccades, with their duration, location, etc. The screen area will be annotated with relevant areas of interest, so that we can use the eye tracking data to monitor how each participant navigated through those areas, during the process. We will use custom-made tools from the NOVA LINCS team to support this analysis. In the end, we expect to use heat maps to analyse where the most important focuses of visual attention were, and scanpath analysis to better understand the model navigation strategies of our participants.

5) Perceived usability: We will assess usability through a SUS test. The SUS instrument is available in the testing environment as a web form. The collected data will be directly fed into SPSS so that we may proceed with the comparative analysis of the distributions of the usability scores.

IV. DISCUSSION

A. Expected results and implications

We are interested in assessing how the usability is influenced by the selection of one of these languages over the other. Rather than using these results as a way of promoting the usage of one of the languages, our goal is to identify language improvement opportunities, on the one hand, and learning from the "competition", on the other. This process is, in that sense, similar to the one the NOVA LINCS team has followed for supporting the Gyro language evolution [35] through a series of developer experience evaluations. We have advocated elsewhere [1] that software language development should

[2]http://www.theeyetribe.com/

[3]https://www.ibm.com/analytics/spss-statistics-software

be iterative and incremental, including (possibly lightweight) evaluations after each iteration, so that improvement opportunities are identified as soon as possible, and, when feasible and adequate, followed on in the next version of the language.

Apart from the more "traditional" analysis of *effectiveness*, here regarded from the perspective of *correctness*, and *efficiency*, viewed considering the *speed*, we expect our exploratory study on the process of building the models, with an analysis of the time annotated sequences of insertions, deletions and changes while constructing models to provide us insights on the main bottlenecks language users experience during the model building process and, conversely, where they seem to experience less difficulties. The eye tracking data is expected to provide further context for better identifying language improvement opportunities. In a longer run, the lessons learned in this and similar studies have the potential for helping us designing more usable software modelling languages. This will also help us better understanding how people from different backgrounds interact with each modelling language, building on earlier works that explored how different personal characteristics (e.g. gender) impacted on the learning, problem solving and information processing style [39]. Finally, the SUS usability questionnaire will help us better understanding how the differences between both languages impact usability.

B. Threats to validity

1) Conclusion validity: Although we plan to have a reasonable amount of participants (over 30), considering the nature of this study, sample size is a likely threat, due to the difficulty in recruiting participants. Our mitigation strategy is to have two teams performing the study in two different countries. The exercise of preparing the experimental replication package so that it can be run both in Portugal and Turkey will help us fine tune it making the package more reusable to third-party replications. This will directly mitigate the sample size risk, as we will have participants in both countries, and indirectly, by facilitating potential third-party replications.

2) Internal validity: There is a potential learning effect from solving one challenge to the next. We mitigate this risk by having the crossover design so that half of the participants start with a SEA_ML++ model while the other starts with DSML4MAS. Another threat could be that a particular problem would by accident favour one of the languages. To mitigate it, both problems will be modelled in both languages, by different participants. We chose two languages for which the tool support is at a similar level, and with a close look and feel, so that tooling does not play a role in differentiating among the two DSLs. We also made efforts so that all materials were easily readable in a 22 inch monitor and that the models to be developed would fit nicely in a canvas on this kind of monitor, without requiring the user to scroll or zoom the image. Monitor size and the general layout for the experiment, including the distance of the participant to the monitor were constrained by the technical specifications of our eye tracking device. In spite of these constraints, the tasks are already challenging to our participants.

3) External validity: Our participants will not have, in general, much experience with MAS and with the two languages. As such, our participants are better representatives of developers who are learning these languages. Further research is necessary to assess how these languages compare, when used by modellers who are experienced with the two languages. The conclusions of this study will be applicable to these two MAS DSMLs. Replications with other languages, not necessarily for MAS, are required before we can generalise this study's conclusions to other contexts.

4) Construct validity: After watching a short tutorial about both languages, participants will solve a couple of challenges, one with each language. This may cause an evaluation apprehension threat. We mitigate this by informing participants that the languages are being evaluated, not the participants. The experimental process is built so that we express no bias toward any of the languages, to mitigate the risk of accidentally favouring SEA_ML++. Our goal is to identify opportunities to improve SEA_ML++ rather than the comparison with DSML4MAS itself. Our measures to mitigate this risk include choosing for the author of the recorded tutorials someone with no vested interest in any of the languages and doing the same for the researchers performing the data analysis. Further, in the interest of transparency and replicability, the data used in these evaluations and data analysis scripts for SPSS will be made publicly available. Last, but not the least, this paper discussing the experimental design to be used in this evaluation serves as a manifest of interest in performing this particular experiment. This creates an opportunity for a sanity check, where the initial goals of this study will be directly comparable with what is actually tested in the experiment, and reported later, mitigating the potential for selective publishing, where only favourable results would be published.

V. Summary

We presented the experimental planning for the evaluation of the case study of DSLs for Multi-agents Systems. Our goal is to go beyond the evaluation of the language's notations (concrete syntax) and evaluate the constructs composition at the level of the instance sentence level (abstract syntax).

It is expected that the results of the evaluation planned in this paper will help in identifying effective improvement opportunities for the developer experience with SEA_ML++.

The work triggers future research in that it departs from the more commonly explored part of visual modelling languages (their visual notation) to other relevant perspectives, namely at the instance (sentence) level.

Acknowledgment

The authors would like to thank the following: i) the Scientific and Technological Research Council of Turkey (TUBITAK) under grant 115E591, and ii) Portuguese grants NOVA LINCS Research Laboratory (Grant: FCT/MCTES PEst UID/ CEC/04516/2013) and DSML4MAS Project (Grant: FCT/MCTES TUBITAK/0008/2014).

The authors would also like to thank the COST Action networking mechanisms and support of IC1404 Multi-Paradigm Modeling for Cyber-Physical Systems (MPM4CPS). COST is supported by the EU Framework Programme Horizon 2020.

REFERENCES

[1] A. Barisic, V. Amaral, and M. Goulão, "Usability driven DSL development with USE-ME," *Computer Languages, Systems & Structures*, vol. 51, pp. 118–157, 2018.

[2] M. Mernik, J. Heering, and A. M. Sloane, "When and how to develop domain-specific languages," *ACM Comput. Surv.*, vol. 37, no. 4, pp. 316–344, 2005.

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

[4] M. Challenger, S. Demirkol, S. Getir, M. Mernik, G. Kardas, and T. Kosar, "On the use of a domain-specific modeling language in the development of multiagent systems," *Eng Appl Artif Intel*, vol. 28, pp. 111–141, 2014.

[5] T. Miranda, M. Challenger, B. T. Tezel, O. F. Alaca, V. Amaral, M. Goulão, and G. Kardas, "Improving the usability of a mas dsml," in *6th International Workshop on Engineering Multi-Agent Systems (EMAS 2018)*. Stockholm, Sweden: Springer, July, 14 2018.

[6] C. Hahn, "A domain specific modeling language for multiagent systems," in *Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1*, 2008, pp. 233–240.

[7] J. Brooke *et al.*, "Sus-a quick and dirty usability scale," *Usability evaluation in industry*, vol. 189, no. 194, pp. 4–7, 1996.

[8] G. Kardas and J. J. Gomez-Sanz, "Special issue on model-driven engineering of multi-agent systems in theory and practice," *Comput Lang Syst Str*, vol. 50, pp. 140–141, 2017.

[9] G. Beydoun, G. Low, B. Henderson-Sellers, H. Mouratidis, J. J. Gomez-Sanz, J. Pavon, and C. Gonzalez-Perez, "Faml: a generic metamodel for mas development," *IEEE T Software Eng*, vol. 35, no. 6, pp. 841–863, 2009.

[10] G. Ciobanu and C. Juravle, "Flexible software architecture and language for mobile agents," *Concurr Comp-Pract E*, vol. 24, no. 6, pp. 559–571, 2012.

[11] J. M. Gascueña, E. Navarro, and A. Fernández-Caballero, "Model-driven engineering techniques for the development of multi-agent systems," *Eng Appl Artif Intel*, vol. 25, no. 1, pp. 159–173, 2012.

[12] E. J. T. Gonçalves, M. I. Cortés, G. A. L. Campos, Y. S. Lopes, E. S. Freire, V. T. da Silva, K. S. F. de Oliveira, and M. A. de Oliveira, "Masml 2.0: Supporting the modelling of multi-agent systems with different agent architectures," *J Syst Software*, vol. 108, pp. 77–109, 2015.

[13] S. Demirkol, M. Challenger, S. Getir, T. Kosar, G. Kardas, and M. Mernik, "A dsl for the development of software agents working within a semantic web environment," *Computer Science and Information Systems*, vol. 10, no. 4, pp. 1525–1556, 2013.

[14] G. Kardas, B. T. Tezel, and M. Challenger, "Domain-specific modelling language for belief-desire-intention software agents," *IET Softw*, vol. 12, no. 4, pp. 356–364, 2018.

[15] J. Faccin and I. Nunes, "A tool-supported development method for improved bdi plan selection," *Engineering Applications of Artificial Intelligence*, vol. 62, pp. 195–213, 2017.

[16] D. Moody and J. van Hillegersberg, "Evaluating the visual syntax of uml: An analysis of the cognitive effectiveness of the uml family of diagrams," in *International Conference on Software Language Engineering*. Springer, 2008, pp. 16–34.

[17] D. L. Moody, P. Heymans, and R. Matulevičius, "Visual syntax does matter: improving the cognitive effectiveness of the i* visual notation," *Requir Eng*, vol. 15, no. 2, pp. 141–175, 2010.

[18] M. Challenger, G. Kardas, and B. Tekinerdogan, "A systematic approach to evaluating domain-specific modeling language environments for multi-agent systems," *Software Qual J*, vol. 24, no. 3, pp. 755–795, Sep. 2016.

[19] G. Kardas, E. Bircan, and M. Challenger, "Supporting the platform extensibility for the model-driven development of agent systems by the interoperability between domain-specific modeling languages of multi-agent systems," *Comput Sci Inf Syst*, vol. 14, no. 3, pp. 875–912, 2017.

[20] F. Bergenti, E. Iotti, S. Monica, and A. Poggi, "Agent-oriented model-driven development for jade with the jadel programming language," *Comput Lang Syst Str*, vol. 50, pp. 142–158, 2017.

[21] A. El Kouhen, A. Gherbi, C. Dumoulin, and F. Khendek, "On the semantic transparency of visual notations: Experiments with uml," in *International SDL Forum*. Springer, 2015, pp. 122–137.

[22] N. Genon, P. Heymans, and D. Amyot, "Analysing the cognitive effectiveness of the bpmn 2.0 visual notation," in *Proceedings of the Third International Conference on Software Language Engineering*, 2010, pp. 377–396.

[23] D. L. Moody, "Why a diagram is only sometimes worth a thousand words: An analysis of the bpmn 2.0 visual notation," Hämtat 2012-06-19 från http://www. business. uq. edu. au/sites/default/files/event/supportingD ocs/Analysis% 20of% 20BPMN% 202.0% 20Visual% 20Syntax. pdf, Tech. Rep., 2011.

[24] R. Matulevičius and P. Heymans, "Visually effective goal models using kaos," in *International Conference on Conceptual Modeling*. Springer, 2007, pp. 265–275.

[25] R. Matulevicius and P. Heymans, "Comparing goal modelling languages: An experiment," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*, 2007, pp. 18–32.

[26] M. Santos, C. Gralha, M. Goulão, and J. a. Araujo, "Increasing the semantic transparency of the kaos goal model concrete syntax," in *37th International Conference on Conceptual Modeling (ER 2018)*. Xi'an, China: Springer, October, 22–25 2018.

[27] P. Caire, N. Genon, P. Heymans, and D. L. Moody, "Visual notation design 2.0: Towards user comprehensible requirements engineering notations," in *RE'13*. IEEE, 2013, pp. 115–124.

[28] N. Genon, P. Caire, H. Toussaint, P. Heymans, and D. Moody, "Towards a more semantically transparent i* visual syntax," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*, 2012, pp. 140–146.

[29] M. Santos, C. Gralha, M. Goulão, J. a. Araujo, and A. Moreira, "On the impact of semantic transparency on understanding and reviewing social goal models," in *26th IEEE International Conference on Requirements Engineering (RE 2018)*. Banff, Canada: IEEE, August, 20–24 2018.

[30] H. Henriques, H. Lourenço, V. Amaral, and M. Goulão, "Improving the developer experience with a low-code process modelling language," in *ACM/IEEE 21st International Conference on Model Driven Engineering Languages and Systems (MODELS)*. Copenhagen, Denmark: ACM, October 2018.

[31] H. Störrle, "On the impact of layout quality to understanding uml diagrams," in *Visual Languages and Human-Centric Computing (VL/HCC), 2011 IEEE Symposium on*. IEEE, 2011, pp. 135–142.

[32] H. Storrle, "On the impact of layout quality to understanding uml diagrams: Diagram type and expertise," in *Visual Languages and Human-Centric Computing (VL/HCC), 2012 IEEE Symposium on*. IEEE, 2012, pp. 49–56.

[33] H. Störrle, "On the impact of layout quality to understanding uml diagrams: size matters," in *International Conference on Model Driven Engineering Languages and Systems*. Springer, 2014, pp. 518–534.

[34] M. Santos, C. Gralha, M. Goulão, J. Araújo, A. Moreira, and J. Cambeiro, "What is the impact of bad layout in the understandability of social goal models?" in *24th IEEE International Requirements Engineering Conference (RE'16)*. Beijing, China: IEEE, September, 12–16 2016.

[35] A. Barišić, J. Cambeiro, V. Amaral, M. Goulão, and T. Mota, "Leveraging teenagers feedback in the development of a domain-specific language: the case of programming low-cost robots," in *Proceedings of the 33rd Annual ACM Symposium on Applied Computing*. ACM, 2018, pp. 1221–1229.

[36] V. Basili, G. Caldiera, and H. Rombach, "Goal Question Metric Paradigm," *Encyclopedia of Software Eng.*, vol. 1, pp. 528–532, 2001.

[37] P. Runeson, M. Host, A. Rainer, and B. Regnell, *Case study research in software engineering: Guidelines and examples*. Wiley, 2012.

[38] B. L. Welch, "The generalization of 'student's' problem when several different population variances are involved," *Biometrika*, vol. 34, no. 1-2, pp. 28–35, 1947. [Online]. Available: http://dx.doi.org/10.1093/biomet/34.1-2.28

[39] L. Beckwith and M. Burnett, "Gender: An important factor in end-user programming environments?" in *Visual Languages and Human Centric Computing, 2004 IEEE Symposium on*. IEEE, 2004, pp. 107–114.