EDITOR INFO. Name here. Affiliation here.

SECTION TITLE. Section Title

Collaborative modelling and group decision-making using chatbots within social networks

Sara Pérez-Soler, Esther Guerra, Juan de Lara
Modelling and Software Engineering Research Group
http://miso.es
Computer Science Department
Universidad Autónoma de Madrid (Spain)

Modelling is used in early phases of software and system development to discuss and explore problems, understand domains, evaluate alternatives and comprehend their implications. In this setting, modelling is inherently collaborative as it involves stakeholders with different backgrounds and expertise, who cooperate to build solutions based on consensus. However, modelling tools typically provide unwieldy diagrammatic editors that may hamper the active involvement of domain experts and lack mechanisms to ease decision-making.

To tackle these issues, we embed modelling within social networks, so that the interface for modelling is natural language which a chatbot interprets to derive an appropriate domain model. Social networks have intuitive built-in discussion mechanisms, while the use of natural language lowers the entry barrier to modelling for domain experts. Moreover, we facilitate the choice among modelling alternatives using* soft consensus *decision-making. This approach is supported by our tool* Socio*, which works on social networks like Telegram.

EDITOR INTRO HEAD. For example, "From the Editor"

EDITOR INTRO PARAGRAPH. This is the main text for the editor intro.

Keywords: Collaborative modelling, social networks, chatbots, decision-making

1. Introduction

Many software development activities are not individual but require collaboration among teams of stakeholders. Modelling is no exception to this rule since the initial stages of development generally involve heterogeneous partners with diverse background and likely distributed. Among them, the participation of domain experts is essential for building successful domain models. However, the use of highly technical and unwieldy modelling tools may hinder their engagement.

As any group activity, collaborative modelling needs mechanisms for decision-making and consensus building. This includes support for the proposal, discussion and selection of modelling alternatives. Given that large-scale collaboration is often the norm in software projects, modelling tools should provide handy and scalable means for discussion and collaboration[3].

Nowadays, social networks and messaging systems are pervasive to discuss among peers, keep contact with friends and organize all sorts of activities. Not only general-

purpose networks like Twitter[1], Facebook[2], Whatsapp[3] or Telegram[4] have boosted, but specialized work-team nets like Slack[5], Workplace[6] or Yammer[7] have spread in enterprises. The reason of this success is their agility, simplicity of use, and the possibility to use them everywhere and in mobility, covering the need to stay connected while being familiar to people.

The use of social media has also disrupted how software engineers work[9], changing the way developers communicate with colleagues and participate in open communities. Advances in natural language (NL) processing have enabled the proliferation of *chatbots* which run on social networks and offer services to users upon NL requests, thereby mimicking human responses. Developers use bots, e.g., to automate deployment tasks, schedule tasks like sending reminders, integrate communication channels, or customer support[6]. They have also been proposed to access API documentation[11] and analyse software projects[1].

Previously[8], we have used social networks for collaborative modelling to leverage on their ubiquity, extended use, scalability and discussion support. Since interaction within social networks is performed through NL, we presented a chatbot called SOCIO that creates domain models out of requirements expressed in NL. This promotes the participation of non-modelling experts in the modelling task. Here, we extend SOCIO to ease decision-making by combining facilities for creating model branches, with soft consensus mechanisms[5] that assist the team in selecting among alternatives.

2. Collaborative modelling in social networks using chatbots

Collaborative modelling can occur offline or online[2]. The former yields asynchronous interactions where users check out models from a version control system, perform local changes, and commit them back to the server. Online collaboration is synchronous, with users meeting in collaborative sessions likely from remote places. This collaboration mode is more appropriate in our context as it supports early discussions and knowledge building, but it demands advanced tooling in terms of context awareness (i.e., knowing who is doing what), discussion (e.g., chats) and coordination (e.g., collaboration protocols).

Existing tools for online collaborative modelling are either diagramming web applications (e.g., GenMyModel[8], Lucidchart[9], AToMPM[10], the online MONDO collaboration framework[2]) or dedicated modelling tools (e.g., MetaEdit+[10], SPACE-DESIGN[4]). In all cases, building models requires the direct manipulation of diagrams, which is laborious and may hinder the active involvement of domain experts with low technical profile. Although some of them provide discussion channels, they are ad-hoc and must be learned.

Our work radically departs from these approaches and looks at modelling as an add-on to a truly discussion environment. In particular, we rely on the discussion and interaction mechanisms offered by social networks based on micro-blogging – like

[1] https://twitter.com/
[2] https://facebook.com/
[3] https://www.whatsapp.com/
[4] https://telegram.org/
[5] https://slack.com
[6] https://www.facebook.com/workplace/
[7] https://www.yammer.com/
[8] https://www.genmymodel.com/
[9] https://www.lucidchart.com/
[10] http://www.metacase.com/

Twitter or Telegram – as they are familiar to many people and do not require installing or learning new applications. Thereby, a collaborative session occurs within a social network, and may involve any number of stakeholders with both technical and non-technical expertise, who can discuss and coordinate via regular short messages.

To assist in the modelling task, a chatbot called SOCIO can be added as a participant to the session. The bot interprets domain requirements in NL and creates a domain model out of them. NL lowers the entry barrier of modelling to domain experts, who may not necessarily know about modelling and may find modelling tools intimidating. Moreover, text interfaces are lightweight and simple to use compared to diagrammatic editors, where one may need to take care of the model layout.

Interacting with SOCIO depends on the social network; in Telegram, the bot is addressed through commands starting by `/`. For example, `/talk` permits describing a domain requirement to the bot. Upon receiving this command, the bot parses the requirement using the Stanford NL parser[7], and applies a set of extensible NL processing rules that trigger update actions on the model. Then, it sends a picture of the resulting model back to the users, highlighting the modified elements. SOCIO also provides other modelling facilities, like direct model manipulation using NL, model validation, exporters to Ecore/EMF (the backend technology used for diagrams), and statistics of user contributions. More information about SOCIO[8] is available at https://saraperezsoler.github.io/ModellingBot/.

Each domain model is developed within a modelling project. The project creator is its owner and specifies its access policy, which can be *public* (anyone can read and modify), *protected* (anyone can read, but only the owner can modify), or *private* (only the specified users can read and modify). After configuring the project, the modelling session can start.

Figure 1 illustrates a collaborative modelling session in Telegram, where SOCIO assists a modelling expert (ME) and a domain expert (DE) to build a model for marketing campaigns. The DE is the leader of a marketing department and wants an application to manage campaigns and their resources. The session occurs within a Telegram group to which all belong. The figure has two columns to be read top-down, left-to-right. First, the ME sends a message to initiate the discussion, and the DE describes a domain requirement to the bot using the `/talk` command. These messages are received by all group members. As a response to the `/talk` command, the bot first echoes the command, and then adds three classes in the domain model that represent the subject and direct objects of the sentence, and two relationships for the verb. The relationships are compositions because the verb is *to contain* (or a synonym), and are unbounded because the direct objects (e.g., *employees*) are in plural. The figure shows further interactions, which seamlessly mix discussions among human participants and commands addressing the bot. The last interactions show how to validate and download the model.

3. Soft consensus for group decision-making

The participants in a collaborative session may require exploring several solutions to a modelling problem, and eventually, they will have to opt for one of them. If collaboration is distributed or involves many participants, tool assistance to facilitate consensus is essential for agile coordination.

The DEs in our example have expressed the need for a communication channel between the team members of a marketing campaign. The following options are being considered:

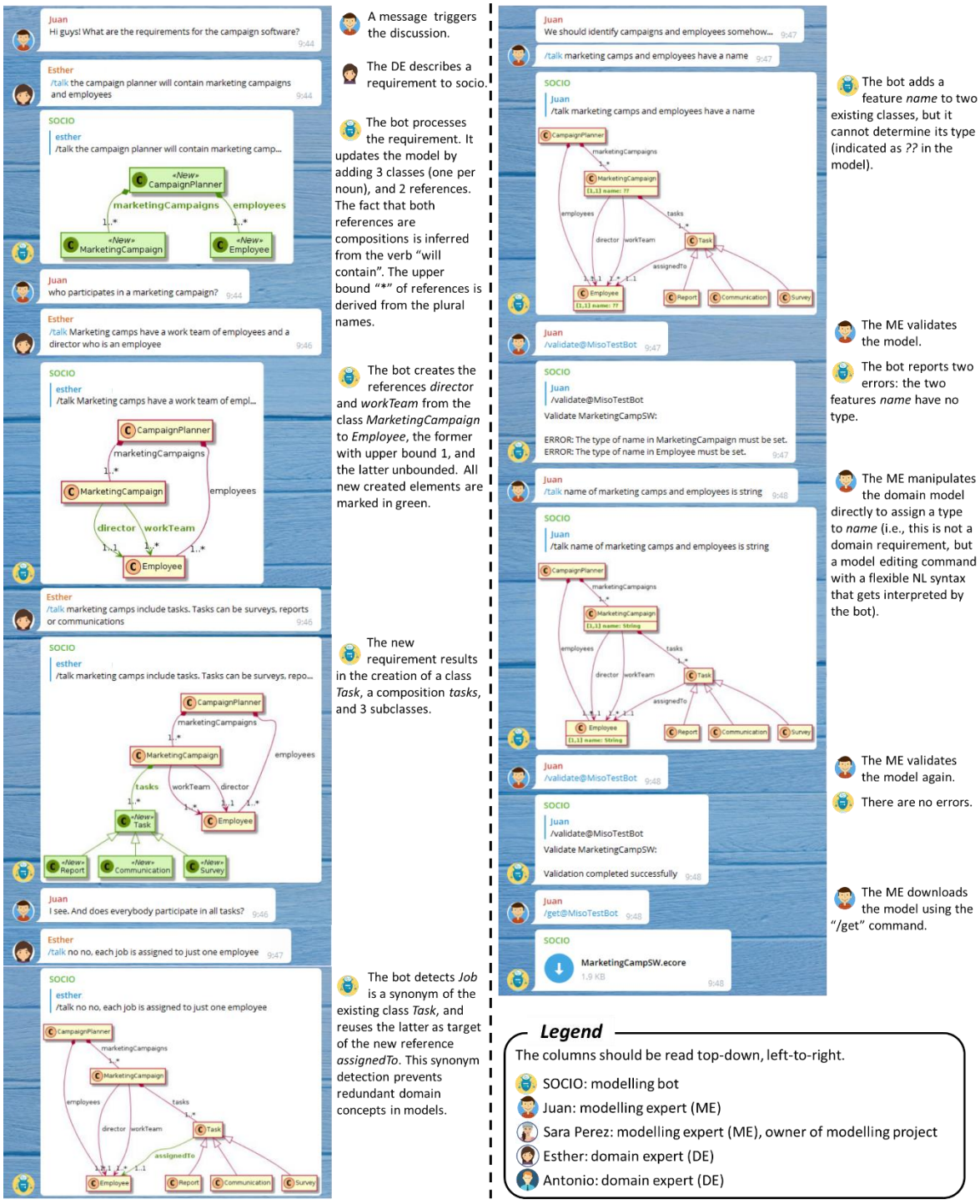

Figure 1. Collaborative modelling session in Telegram with SOCIO

- A message box per employee, not necessarily working in the same campaign. This would be like an e-mail or peer-to-peer messaging system.
- A special type of work task for discussion, where any employee can post comments and reply to other comments.
- A forum associated to each marketing campaign, where work-team members can contribute news organized into threads, like in bulletin boards.

To study different possibilities, SOCIO supports *branch groups*. These collect the alternatives and discussions to model an aspect of a system as different branches of the

current model. In our example, we use this facility to create a group called *communication* with three branches, one for each considered solution: `p2pMessages`, `DiscussionTasks`, and `BulletinBoard`. Anyone with editing access can create a branch group, and the branch creator configures its participants. In each branch, the domain model evolves separately from the other branches, according to the bot-directed messages sent within the branch. SOCIO distinguishes the elements created in a branch from the trunk elements using different colours and the `<Old>` stereotype (see models in Figure 2(a)).

After outlining the alternative solutions in branches, the participants need to agree on the most suitable one. For this, SOCIO incorporates a soft consensus mechanism for multi-person decision-making[5] that assists in choosing the option that is acceptable to all group members. Figure 2(c) shows its working scheme. Participants can express their favourite solution in several ways, like ordering the alternatives from better to worse, or giving a score to each option (e.g., from 1 to 10). Then, an iterative consensus process identifies the preferred alternative based on the expressed preferences. We use "soft" consensus because unanimous agreement can be difficult to achieve, especially in numerous groups or with experts with dissimilar backgrounds. Soft consensus models[5] permit measuring the degree of consensus in a group, provide feedback to each participant on the current consensus, and iterate to improve the consensus and converge towards a shared consensus threshold.

Figure 2(a, b) illustrates the use of SOCIO to select by consensus one solution in the *communication* branch group. Part (a) displays the project owner starting the polling and picking the voters, as well as one voter (Antonio) expressing his preferences through private vote. During the polling, SOCIO will only show the new elements added in each solution and their context, to highlight the variations.

Figure 2(b) shows how consensus is measured. When all voters have indicated their preferences or when a predefined deadline is reached, the system aggregates all answers into a collective preference vector, and computes a global ranking for the alternatives, and a consensus measure ranging from 0 to 1. If the consensus is below a threshold (0.75 by default), another iteration is performed. In the figure, the consensus is 0.57 after the first iteration, which is below the threshold. Hence, voters are ranked according to how far their preference is to the collective preference, and the farthest ones (according to a threshold) are invited to change their choice, while the rest remain unchanged. This process promotes the convergence to an acceptable group consensus[5]. When the consensus reaches the threshold or when the project owner decides so, the most preferred branch is integrated with the main model trunk, and the other branches closed. The branches and voting results can be consulted in the project history.

4. Discussion and outlook

In this paper, we have stressed the collaborative nature of modelling and have argued that this collaboration can take place within social networks and mediated by chatbots which are interfaced by NL and, under the hood, perform modelling tasks. Using NL as modelling interface has the advantage of lowering the entry barrier to modelling, and does not interrupt the group discussion flow because messages for discussion and modelling are intertwined. Moreover, any element included in the model is immediately justified by the NL message used for its creation, hence documenting its provenance. To enrich traceability and rationale of modelling decisions, we also produce a *history* model tracking user contributions to model elements[8].

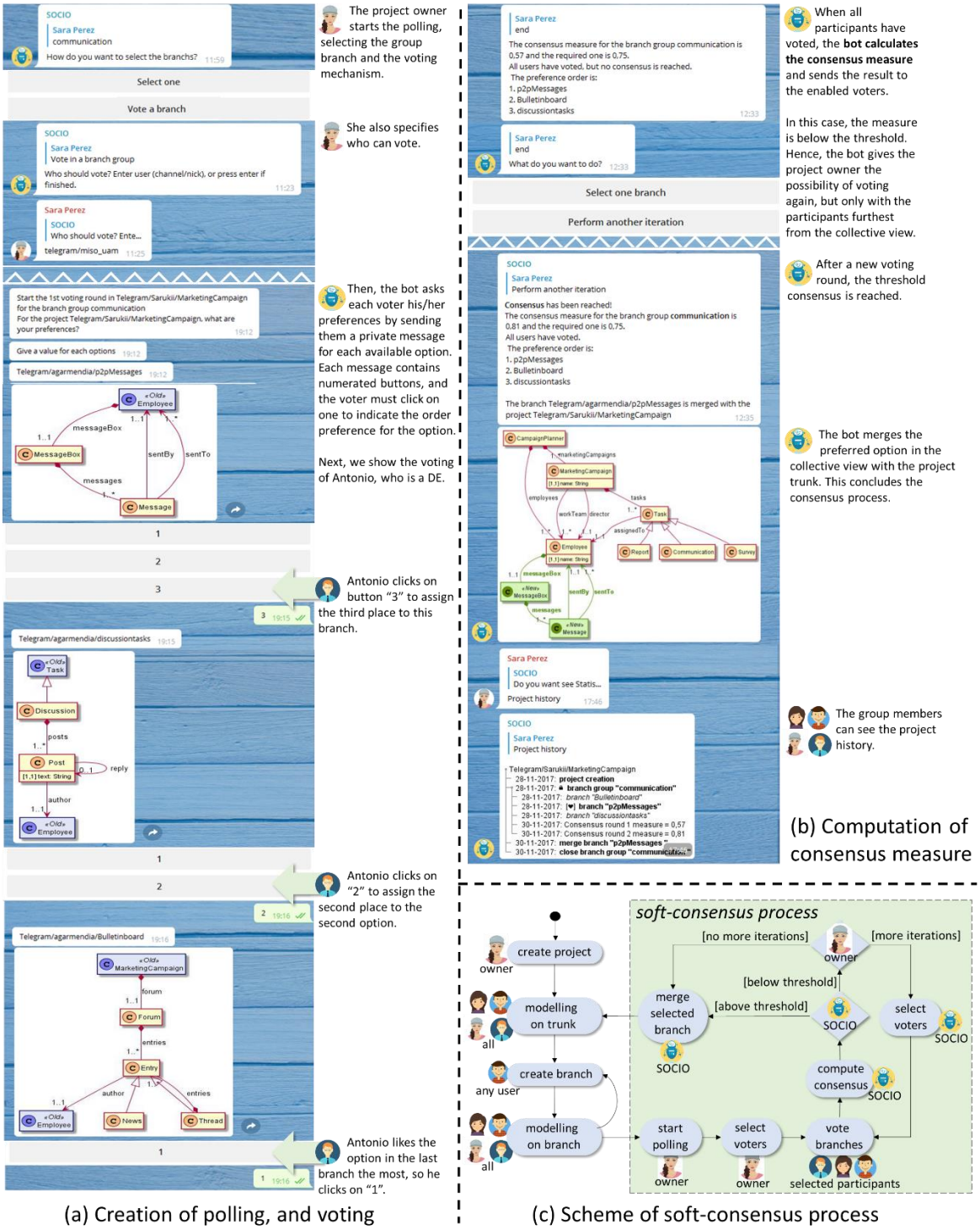

Figure 2. Consensus decision-making with SOCIO (a, b). Consensus process scheme (c)

To ease decision-making by a potentially large heterogeneous group, we incorporate a soft-consensus mechanism to measure the degree of agreement based on the group preferences, and avoid the bias that a human moderator may introduce[5]. To assess this hypothesis, we performed a small-scale evaluation with 8 participants recruited from the Master and Doctoral programs of the Department of Computer Science of our university. 6 participants were computer scientists, 1 engineer in telecommunication, and 1 physicist. After attending a 10-minutes tutorial about SOCIO, they used it to select the best solution

among three possibilities for two different projects, first without consensus mechanism, and then using it. Interestingly, without the consensus mechanism, they ended up organizing a public polling within Telegram, but discrepancies among participants remained until the end of the experiment. They also answered a five-point Likert scale survey on the consensus mechanism, which was considered especially useful for large groups (average 4.7/5), and with an outcome that reflected the opinion of the majority (4.8/5) and was deemed objective because of the private voting (4.3/5).

The practical usability of modelling chatbots depends on their ability to interpret NL. A preliminary assessment of a previous version of SOCIO revealed that modelling using NL was liked more than using graphical editors (75%), while its NL processing was reasonable but improvable (participants gave an average of 62.5% for accuracy)[8]. To mitigate possible mistakes of the NL processor, SOCIO permits manipulating models directly through NL (i.e., using sentences like "remove employee" which are parsed and interpreted) and undoing actions. While messages in micro-blogging systems are short, which facilitates NL processing, they foster the use of slang and abbreviations, which we will consider in future work. We also plan to extend SOCIO with the ability to answer NL questions about a model, to assist domain experts in subsequent decisions.

Enabling social networks for collaborative modelling brings exciting possibilities, like involving large groups of people (i.e., crowdsourced modelling), or its use in Software Engineering education. We also foresee chatbots for other modelling tasks, like model quality monitoring and refactoring suggestions, and for other diagram types.

Acknowledgements

Work funded by the Spanish MINECO (TIN2014-52129-R) and the R&D programme of Madrid (S2013/ICE-3006).

References

1. Beschastnikh, Lungu, Zhuang. Accelerating software engineering research adoption with analysis bots. *ICSE-NIER*, pages 35-38. IEEE, 2017.
2. Debreceni, Bergmann, Búr, Ráth, Varró. The MONDO collaboration framework: secure collaborative modeling over existing version control systems. *ESEC/FSE*, pages 984–988. ACM, 2017.
3. Franzago, Di Ruscio, Malavolta, Muccini. Collaborative model-driven software engineering: A classification framework and a research map. *IEEE Transactions on Software Engineering*, in press, 2017. DOI 10.1109/TSE.2017.2755039.
4. Gallardo, Bravo, Redondo. A model-driven development method for collaborative modeling tools. *J. Network and Computer Applications*, 35(3):1086–1105, 2012.
5. Herrera-Viedma, Herrera, Chiclana. A consensus model for multiperson decision making with different preference structures. *Trans. Sys. Man Cyber*. Part A, 32(3):394–402, 2002.
6. Lin, Zagalsky, Storey, Serebrenik. Why developers are slacking off: Understanding how software teams use slack. *CSCW*, pages 333–336. ACM, 2016.
7. Marneffe, Maccartney, Manning. Generating typed dependency parses from phrase structure parses. *LREC*, pages 449–454. ELRA, 2006.
8. Pérez-Soler, Guerra, de Lara, Jurado. The rise of the (modelling) bots: towards assisted modelling via social networks. *ASE*, pages 723–728. ACM, 2017.
9. Storey, Zagalsky, Figueira Filho, Singer, German. How social and communication channels shape and challenge a participatory culture in software development. *IEEE Transactions on Software Engineering*, 43(2):185–204, 2017.
10. Syriani, Vangheluwe, Mannadiar, Hansen, Mierlo, Ergin. AToMPM: A web-based modeling environment. *MODELS Satellite Events*, volume 1115 of CEUR, pages 21–25, 2013.
11. Tian, Thung, Sharma, Lo. APIbot: question answering bot for API documentation. *ASE*, pages 153–158. ACM, 2017.

Sara Pérez-Soler is a Master's student in Computer Science at the Universidad Autónoma de Madrid, and the main developer of SOCIO. Contact her at sara.perezs@estudiante.uam.es.

Dr. Esther Guerra is a professor at the Universidad Autónoma de Madrid. Her research interests include model-driven engineering, automated software development and domain-specific languages. Contact her at esther.guerra@uam.es.

Dr. Juan de Lara is a professor at the Universidad Autónoma de Madrid. His research interests are in model-driven engineering, including meta-modelling, model transformations and domain-specific languages. Contact him at juan.delara@uam.es.