A Tool for Introducing Computer Science with Automatic Formative Assessment

Luciana Benotti, María Cecilia Martínez, and Fernando Schapachnik

Abstract—In this paper we present a software platform called Chatbot designed to introduce high school students to Computer Science (CS) concepts in an innovative way: by programming chatbots. A chatbot is a bot that can be programmed to have a conversation with a human or robotic partner in some natural language such as English or Spanish. While programming their chatbots, students use fundamental CS constructs such as variables, conditionals, and finite state automata, among others. Chatbot uses pattern matching, state of the art lemmatization techniques, and finite state automata in order to provide automatic formative assessment to the students. When an error is found, the formative feedback generated is immediate and task-level. We evaluated Chatbot in two observational studies. An online nation-wide competition where more than 10,000 students participated. And, a mandatory in-class 15-lesson pilot course in three high schools. We measured indicators of student engagement (task completion, participation, self reported interest, etc.) and found that girls' engagement with Chatbot was higher than boys' for most indicators. Also, in the online competition, the task completion rate for the students that decided to use Chatbot was five times higher than for the students that chose to use the renown animation and game programming tool Alice. Our results suggest that the availability of automatic formative assessment may have an impact on task completion and other engagement indicators among high school students.

Index Terms—Interactive learning environments, K-12 education, computer science education, automatic formative assessment

1 Introduction

There is a worldwide need to promote youth engagement in Computer Science (CS). Taking our country as an example, we know that Argentinean universities graduate approximately 4,000 CS students a year (compared to 10,000 in Law and 15,000 in Economics) while the national industry needs to hire twice that amount [1], [2].

Previous studies suggest that the lack of early CS education can influence career choices: students may not be selecting CS simply because they do not know what CS is [3], [4]. The typical K-12 student in Argentina never encounters CS topics during his/her school years. The curriculum focuses on user training rather than on CS content: students learn how to use a word processor, a spreadsheet or how to create an online blog. This context is not unique to Argentina; many developed countries share the same problem [5], [6], [7]. There are some exceptions such as Israel, where CS has been taught at (some) high schools for many years now [8], and other countries are starting to follow. This is the case, for example, in the US [9], New Zealand [10], and the UK [6].

- *L. Benotti is with the Department of Computer Science, Universidad Nacional de Córdoba, Córdoba X5016 GCA, Argentina. E-mail: benotti@famaf.unc.edu.ar.*
- *M.C. Martínez is with the Department of Education, Universidad Nacional de Córdoba, Córdoba X5016 GCA, Argentina. E-mail: cecimart@gmail.com.*
- *F. Schapachnik is with the Fundación Sadosky, Caba C1054AAT, Argentina, and the Deparmento de Computación, Facultad de Ciencias Exactas y Naturales, Universidad de Buenos Aires, Buenos Aires 1053, Argentina. E-mail: fschapachnik@fundacionsadosky.org.ar.*

Manuscript received 17 May 2016; revised 30 Jan. 2017; accepted 4 Mar. 2017. Date of publication 0 . 0000; date of current version 0 . 0000.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TLT.2017.2682084

Given the current situation, there is increasing consensus that introducing students to CS in high school (and even primary school) is necessary to help them make educated choices about their professional future but also to include them in the technological world as active and creative citizens. Institutions, companies, universities and teachers around the world are working towards this goal with several initiatives. In Argentina, one of them is an online programming contest based on the renown animation and game programming tool Alice [11], [12], that despite having attracted tens of thousands of students, faced the issues of low female participation and low task completion rates.

With the goal of addressing these issues, we developed Chatbot. Chatbot is an educational software tool designed to introduce high school students to CS concepts in an innovative way. Chatbot is innovative in at least three aspects.

First, students program chatbots rather than animations, games or physical robots as most initiatives around the world do. A chatbot is a bot that can be programmed to have a conversation with a human or robotic partner in some natural language such as English or Spanish. Inspired by previous research on gender and technology [13], [14], we intended to address low female participation in the online programming contest through a chat related activity. The contest participants could decide to participate with Chatbot, Alice or both.

Second, Chatbot was designed to serve as an educational software tool suitable for massive online open courses or competitions that include low teacher support. Hence, the tool includes automatic formative assessment capabilities with immediate, task level feedback. Chatbot uses pattern matching, state of the art lemmatization techniques, and finite state automata in order to generate the feedback.

Third, Chatbot is innovative because, differently from most available software tools for CS education, it can be used for teaching not only basic programming but also high level CS concepts (such as finite state automata). For example, finite state automata are necessary to model topic shifts during a conversation.

Besides describing Chatbot and how it generates formative feedback, this article presents two observational studies that analyze student engagement while programming chatbots in a classroom environment and in the online contest, making the following contributions:

- We present Chatbot, an educational tool for introducing high school students to CS concepts in an innovative way. Chatbot is innovative because it generates formative assessment automatically while the students program bots that can chat, using basic programming constructs (e.g., variables) and high level CS concepts (e.g., finite state automata).
- We measured indicators of student engagement when using Chatbot. Following [15], we define engagement as cognitive investment in learning and completing the task, and we measure it through several indicators: task completion, enthusiasm, participation, self reported interest and easiness, willingness to learn more, among others.
- We compared Chatbot effects on student engagement in two observational studies: an online competition that also uses the well known platform Alice where more than 10 thousand students participated, and an in-class 15-lesson course in three high schools.
- We found that girls' engagement with Chatbot was significantly higher than boys' for most indicators. We found evidence suggesting that Chatbot was easier to use with formative feedback.
- In the online competition we found that, although Alice was initially more appealing to the contestants, the task completion rate for the students using Chatbot with formative feedback was five times greater than for the students that chose to produce a game or animation using Alice.

The rest of the article is organized as follows. The next section positions our work with respect to related work. Section 3 describes the Chatbot tool and the technologies implementing its automatic formative assessment capabilities. Section 4 reports our results of the evaluation of Chatbot in the online student contest and in the classroom. Section 5 discusses the results of the studies. Final remarks and our future research agenda conclude the article in Section 6.

2 Related Work and Formative Assessment

This section presents an analytical review of the related work and highlights the role and place of the research reported in this paper with respect to the existing work.

2.1 Initiatives for Computer Science Promotion

Chatbot development was motivated by an online contest called Dale Aceptar (DA) (Spanish for "just hit OK"). Dale Aceptar is a free online competition organized by the Sadosky Foundation at http://www.daleaceptar.gob.ar. It is performed annually with the aim of interesting students into pursuing CS related careers. The competition targets high school students, with no prior background in CS, who sign in on their own because they see an advertisement.

DA is not the only initiative in the world to promote K-12 student engagement in CS. Code.org [16] in the US organizes the "One hour of code" campaign, a massive online challenge for CS promotion. In this challenge students solve fixed puzzles by programming. Since programming assignments in Code.org are fixed, it provides rich automatic formative feedback. It implements verification feedback as well as elaborated feedback with links to videos that explain the core programming concepts it teaches. It has reached millions of K-12 students and teachers all over the world. It only focuses on teaching programming and does not target other high level CS concepts. It targets elementary school students (6+ years old). In contrast, Dale Aceptar targets a high school audience (13+ years old).

Another initiative to promote K-12 student engagement in CS is the New Zealand based CS Unplugged [17]. This program also proposes to teach high level CS concepts other than programming (such as finite state automata, criptography and protocols) but without using a computer and through simple but effective games that involve physical movements and manipulation of objects. This approach is quite widespread but, to the best of our knowledge, its impact has not been measured. Like our Chatbot project, CS Unplugged is based on the assumption that learning high level CS concepts can contribute to developing higher order thinking skills. Some studies, such as the one conducted by Doran et al. [18] support this assumption. Doran and her colleagues evaluated the impact of teaching abstract CS concepts other than programming on high school student performance and engagement in other subjects. They found that performance and engagement indicators improved in subjects as diverse as Math and English.

Code.org and CS Unplugged are the most widespread initiatives for CS promotion at K-12 level but there are others (e.g., [6], [10]) that use different tools for their purposes. The first edition of Dale Aceptar in 2012 was solely based on Alice [11]. Alice was the tool of choice because, at the time the competition was planned, it complied with the following features that were *sine qua non* conditions for the online student competition. 1) User interface appealing for teenagers between 13 and 20 years old. Some tools with a more childish look and feel (e.g., Scratch [19]) could not be used. 2) Provided for free with an Open Source license and it had both Windows and Linux versions. Available in Spanish (unlike most tools from other countries). Most Argentinean students are not fluent in English. 3) Usable with low teacher support (unlike CS Unplugged activities). A website made available 23 short-video lessons, going from basic concepts up to building a turn-taking, timer-based game. A fora provided support for Q&A.

In 2012, the contest attracted more than 10,000 students by various promotional methods that focused on having fun while attempting to win prizes. In this first edition, the task completion rate was only 1 percent and almost 90 percent of the students were male. Thus, attracting more female students and improving the task completion rate were reasons to develop Chatbot.

2.2 Chatbots and Tutoring Systems

Chatbots have been used in different contexts as *tutoring systems* not only for teaching CS but also other subjects. These tutoring systems teach different topics by chatting with the students. Differently from our work, when using such tutoring systems, students do not *program* chatbots but they *chat* with them. Kerly et al. [20] describe a chatbot that chats with students in order to influence their opinions about programming in C. Oscar [21] is another intelligent tutoring system which leads a tutoring conversation and dynamically predicts and adapts to students' learning style. Oscar can discuss about programming errors and code style with the students. There is a vast amount of work on intelligent tutoring systems that teach topics other than CS. For example, AutoTutor [22] is an intelligent tutoring system that helps students learn Newtonian physics, computer literacy, and critical thinking topics through tutorial conversations in natural language. AutoTutor uses computational linguistics techniques such as latent semantic analysis, regular expression matching, and speech act classifiers in order to understand natural language.

Tutoring systems are different from Chatbot, where students have to program their own chatbots and learn basic CS concepts by programming, not by chatting. As most intelligent tutoring systems do, Chatbot provides formative feedback and assessment. However, we do not consider Chatbot to be a full-fledged intelligent tutoring system since it tries neither to model nor to adapt to students' learning style. As well as AutoTutor, Chatbot uses some computational linguistics techniques. In particular, Chatbot uses lemmatization (to parse word declinations) and regular expression matching. For predictability, since the students are doing the chatbot programming, we decided not to include non-determinism. Hence, no statistically based technique (such as latent semantic analysis or speech act classifiers) is used. Semantics (such as topic shifts and dialogue referents) are modeled through the finite state automata that the students design. We describe the technologies used to generate formative feedback automatically in Section 3.

A few researchers have used *chatbot programming*, as we do, as a method for interesting students in CS. For example, Shaw [23] taught some basic artificial intelligence concepts in an introductory CS course at his university by making his students program a chatbot. Keegan et al. [24] presented Turi, a chatbot programming tool that was used in a workshop to explain the Turing Test. Bigham et al. [25] used chatbot programming to inspire a group of blind high school students to pursue a career in Computer Science. To the best of our knowledge, these experiences have not been described in detail and their engaging effect has not been reported. Moreover, differently from Chatbot, none of them provides automatic formative assessment.

2.3 Gender and Tools for Computer Science Education

One of the reasons to develop Chatbot was to have a tool that can promote girls engagement in CS.

Based on teenager focus groups and self reported questionnaires that gather information about what they use computers for and for how long, research showed that most Latin American girls prefer to use computers in order to foster interpersonal and social relationships and, in contrast, boys are more likely to use computers to play games [13], [14], [26], [27]. Recent studies at university level concentrated on analyzing teachers' opinions about student engagement [28], [29], [30], [31]. These studies found that instructors' responded differently to females and males in CS learning environments, and that these differences may affect achievement and interest in CS. Other studies [32], [33] described how to engage students through specific techniques such as game design and they reported positive impact on engagement indicators. However these studies did not control for gender differences.

Research also shows that boys are more likely to know more programming than girls before entering CS majors. They usually have previous experiences learning programming and are more likely to chose programming courses in high school [26], [34], [35]. Margolis [26] documents that girls are more attracted to more abstract and high-level concepts of CS rather than to programming. Unlike Chatbot, most of the tools currently available for CS education focus on teaching programming rather than on other CS topics.

Reviewing the most reknown tools for teaching programming at a high school level we found that the first ones were Karel [36] and Logo [37]. While Karel design focused more on how to properly structure basic programs, Logo also included notions of spatial reasoning. Recently, several educational programming environments for high school were developed. Utting et al. [38] compared three of the most well known environments with respect to age, gender, and logical thinking. We briefly summarize their findings. To start with, Greenfoot [39], that teaches object orientation with Java, emphasizes logical thinking more than the other tools. Moreover, Alice [11], [40], which can be used to program in a three dimensional environment, is particularly appealing for girls (however, see our own findings related to gender in Section 4). Finally, Scratch [19] is similar to Alice, but has a user interface that targets younger children. All three of them allow for easy development of animations and interactive games while teaching programming with block based programming languages that prevent parsing problems. Since all these tools allow for open ended programming, they do not offer automatic formative feedback or assessment. More tools exist for teaching programming at university level than at high school level. They have been surveyed in [41], [42] and more recently in [43]. In [43], an online architecture for classifying and increasing the access to such tools is proposed. Some of these tools provide different forms of automatic formative feedback but they are probably too complex for the background and age of the students that we target.

There are a few tools that teach high level CS concepts. For example, Kara [44] is a declarative programming environment where programs are graphically created as finite state machines. Stagecast Creator [45] is based on the programming by demonstration paradigm, where rules are created by giving examples of what actions should take place in a given situation. Neither of these tools provide automatic formative feedback. To the best of our knowledge, there are not many *tools to teach CS concepts other than programming* at high school level. We believe that the availability of tools with different characteristics and design goals is good because the range of students, teachers and CS concepts that they serve are also diverse.

2.4 Desiderata for a New Tool: Formative Assessment

Taking all this research into account, Dale Aceptar organizers added a new category based on Chatbot's ability to connect to social networks (e.g., Facebook) for Dale Aceptar second edition (ran during the second semester of 2012). The idea was simple: build a chatbot that impersonates yourself. Participants got points for long conversations that their chatbots had with their friends; the one with the highest score won.

Online videos taught students how to build chatbots that favored engagement of the other party through a series of strategies: answer a question with another question, make your chatbot expert in one topic (e.g., movies) and always tilt the conversation towards that topic, etc. All videos for both Chatbot and Alice lessons had the same characteristics: discovery based teaching as the predominant teaching strategy, scripted by the same group of educators; and taught by the same teacher. All of the videos—in Spanish—are available from the competition's site http://www.daleaceptar.gob.ar.

However, the Chatbot online contest in 2012 was not successful. Although 489 students signed in to participate with Chatbot, only a few uploaded complete bots. The task completion rate was still 1 percent, as with Alice. The female participation did increase however, with 27 percent of females signing up in the Chatbot contest (versus 10 percent in Alice). While interviewing 2012 Dale Aceptar participants, we found the following potential explanations for Chatbot low task completion rate in the online contest.

- Impersonation: teenagers' online profiles are not just "accounts", they are their daily socialization channel, their own personal exhibit window. The possibility of having a bot interacting with others making them look bad in public was a show stopper for turning Chatbot on or having it chat with many people.
- Unbounded score: although their own score was shown all the time at the top of Chatbot's screen, they had no idea what others' score was. As a result, there was no way of inferring if they were making adequate progress in the competition context.
- Open ended task: students had no way of knowing when they had finished programming a good chatbot. In order to program a good chatbot, one needs to predict what others' might ask from it. The variability of such questions may well be infinite.
- Too difficult: predicting what others' might say is a difficult task, more related to linguistics than to CS. Chatbot had a log of questions it did not have an answer for, and some features to turn that question into new code. However, leaving the chatbot on for a few hours and then finding a great number of misunderstood questions built an idea of never ending task.

It is well-known that retention rates are very low in online courses [46]. As argued by Newmann [15], engagement in the classroom can be seen as the product of three main factors, 1) the need for personal competence (which varies with socio-economic status), 2) the types of tasks students are required to do (mechanical, fun, authentic), and 3) the school environment (support, care, fairness, academic status). Positive school and classroom environment includes teachers providing personal support to avoid frustration when difficulties arise, and caring about students as individuals in a context where academic expectations are clear and school success is promoted for all. How does that work in the online world? What is necessary to do to retain students who have not logged in to a site for a while, so that they can come back? How is it possible to offer help to thousands of students widely spread geographically when there are only a few teachers? These are hard questions related to the open problem of low completion rates in all online courses.

One potential solution is to provide automatic formative feedback and assessment to students about their work. Previous work on formative feedback and its effects, can be classified according to whether they focus on its effects on learning [47], [48], [49], [50] or on its effect on motivation and engagement [51] (most previous work fall in the first category). Wiliam [47] found that formative feedback is most effective for *learning* when the feedback tells participants not just what to improve but also how to go about it. Butler [51] found that *motivation* was higher after receiving verification feedback (i.e., grades) than after receiving feedback that explains how to improve. Wiliam [50] argues that there remains much more work to be done to integrate research on formative assessment with more fundamental research on instructional design, feedback, motivation and engagement.

Considering this body of research, we modified Chatbot in order to include four different kinds of feedback as classified by Shute [48]: 1) a percentage of correctness of the chatbot with respect to predefined questionnaires, 2) a classification of the errors into "Not addressing the question" or "Addressing the question wrongly", 3) a direct link to the part of the program that needs revision, and 4) hints about the concepts needed to solve the task (for example, by pointing the student to a video or reading material). We describe the Chatbot tool and the technologies implementing its automatic formative assessment capabilities in the next section.

3 Chatbot Design and Implementation

We begin this section by presenting Chatbot and explaining how we use it to introduce fundamental CS concepts (Section 3.1). We then explain how Chatbot generates formative assessment automatically in Section 3.2.

The examples used in this section are based on a murder story that can be played with Chatbot where five suspects, a murder victim and a detective are left alone in a mountain (see Fig. 1). Students have to choose one suspect to defend and program their chatbot so that it answers the detective questions properly. The detective's questionnaire files are programmed by the teachers using regular expressions for the expected answers.[1]

3.1 Chatbot Basics

In Chatbot students program chatbots to answer in different ways depending on who they are talking to, what the person is saying, which topic they talked about before, etc.

While designing Chatbot, we emphasized on *transparency*. Unlike commercial chatbots, our tool does not use any "black box" approach to "magically" fabricate replies (e.g., using

1. The game story and the detective questionnaires can be downloaded from www.daleaceptar.gob.ar (in Spanish).

Fig. 1. The suspects and the victim before the murder in the murder story game that can be played with Chatbot.

machine translation techniques). However, Chatbot can also be used to explore specialized concepts such as the Turing Test and Natural Language Processing topics. It includes state-of-the-art natural language processing techniques. We use FreeLing [52] as a C++ library providing natural language analysis functionalities (morphological analysis, named entity detection, PoS-tagging, parsing, word sense disambiguation, semantic role labeling, etc.). Freeling works for a variety of languages: English, Spanish, Portuguese, Italian, French, German, Russian, Catalan, Galician, Croatian, Slovene, among others. Freeling's licence is GPL Affero. Chatbot is open source, distributed under the GNU General Public License GPLv3. It is available in Spanish and English (both its interface and the natural language processing). It can be downloaded from http://bit.ly/1iglAf6.

In this section we illustrate how Chatbot can be used to teach some fundamental CS concepts such as *variables* and *conditionals*, as well as the more advanced concept of *finite state automata*.

Chatbots are programmed in Chatbot by writing sets of *(pattern, effect)* pairs. These pairs are called *rules* in Chatbot. The chatbot responds with the *effect* when the *pattern* matches the stimulus received by the chatbot. Patterns are regular expressions that may include wildcards and variables, and effects may include variables and conditionals (among more advanced structures).

The example shown in Fig. 2 is a dialogue between the suspect of a murder and the detective in charge of the investigation. The teacher can give this dialogue to the students and challenge them to write a single (pattern, effect) pair to program a bot that can answer like this suspect. In order to solve this exercise with a single pair, a *conditional* and a *variable* need to be used. The detective chatbot is programmed by the teacher. A student can monitor whether her chatbot is fulfilling its goal by observing both bots chatting on Facebook as shown in the figure.[2]

A correct answer to the exercise is shown in a screenshot of the Chatbot platform in Fig. 3. In the screenshot, the upper part contains a basic menu, the left hand panel contains a hierarchy of all the pair of rules that have been programmed, and the right hand panel shows the rule

Fig. 2. Sample dialogue provided to students between a murder suspect bot and a detective bot that motivates the use of a conditional if-then-else as the one shown in Fig. 3.

highlighted in the left hand panel. This rule is a solution to the exercise: the pattern (under the textbox *Writes*) includes the wildcard $*$ that can match any number of words (e.g., "*Do you think that*") and the variable [person] that stores the value of the word (e.g., "*cook*") that comes right before the phrase "*is the murderer?*". The effect (under the textbox *Chatbot replies*) is a conditional expression that, depending on the value of the variable [person], may give different answers. This single rule is enough for the suspect bot to answer as required by the dialogue in Fig. 2. This suspect bot will say that the millionaire is not the murderer.

The rest of the screenshot shown in Fig. 3 contains the following elements. The textbox *If* indicates that the suspect chatbot will only use this rule when talking to the detective. This textbox can be filled by selecting any of the contact names obtained from the contact list of the social network to which it is connected. The textbox *Or any of these variants* includes a list of alternative patterns that can also trigger the effect of this rule. These variants are not required for the challenge posed by Fig. 2 but allow the bot to answer consistently even if the question is asked in a different way.

The teacher can give the students the dialogue shown in Fig. 4 in order to motivate the need for *finite state automata*. This dialogue contains two utterances that are exactly the same—"*when did you meet him?*"—but that occur at different points or states of the dialogue and, as a result, they need to receive different answers—"*last year at the dinner*" and "*few years ago*". Under the same input pattern, the effect has to be different. In this way, the teacher can show that not only the input pattern but also the current state of the dialogue, can alter the output that needs to be generated.

While solving this exercise the students have to design a finite state automaton that contains, at least, two *states* which model the two referents of this dialogue fragment (one for the millionaire and one for the cook). They also have to

2. The Facebook messenger here plays the role of an interface where a student can see how the dialogue evolves between the suspect and the detective in order to debug her bot. A similar visualization can be done in the Chatbot interface without Internet connection.

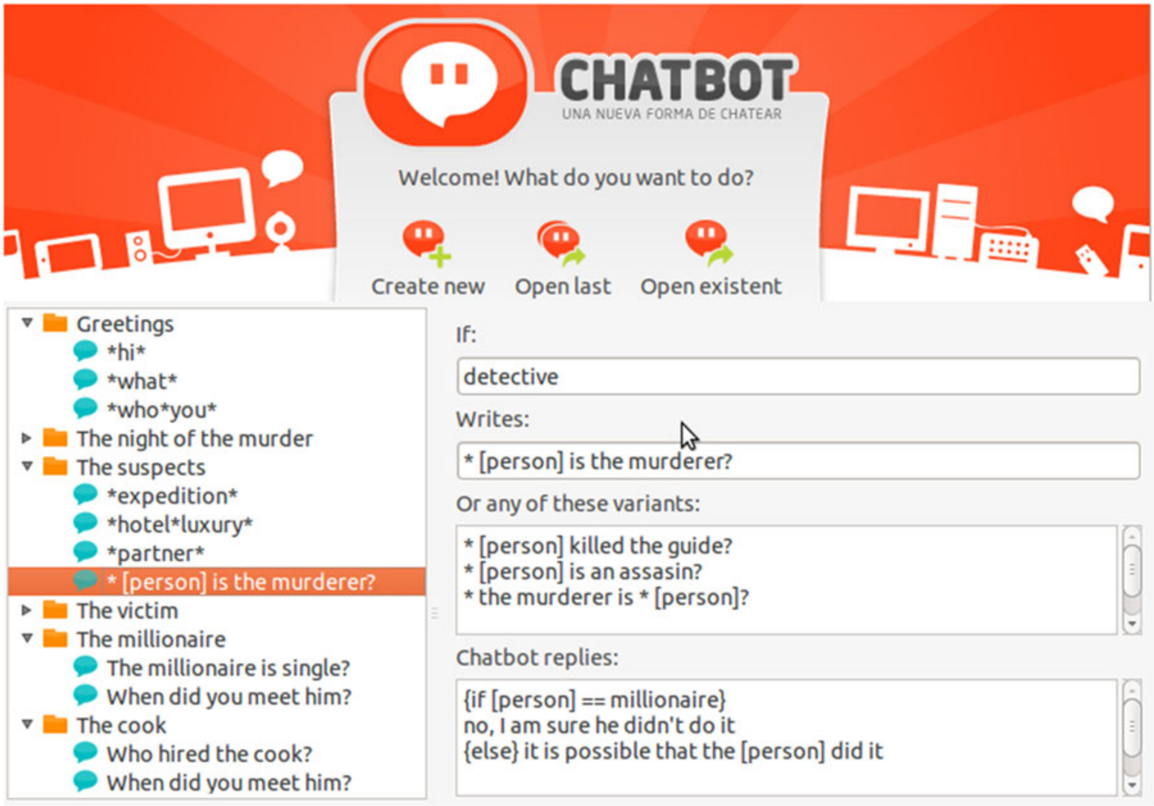

Fig. 3. Screenshot of Chatbot. The selected (pattern, effect) pair uses variables, wildcards, and conditionals in order to produce the behavior of the suspect in Fig. 2. The folders in the left hand panel represent the states that the chatbot can be in. The states The millonaire and The cook are necessary to produce the behavior shown in Fig. 4 and represented as a finite state automata in Fig. 5.

model how the dialogue utterances make the automaton enter the appropriate states. The students need to realize that the utterance "*is the millionaire single?*" takes the chatbot into

Fig. 4. Sample dialogue between a suspect and the detective that motivates the understanding of finite state automata. The same utterance "*when did you meet him?*" occurs twice in different states of the dialogue.

the state that we call The millionaire. When the question "*when did you meet him?*" is asked for the first time in the dialogue, the chatbot responds with the rule triggered in The millionaire state producing the answer "*last year at the dinner*". Later on, the utterance "*You hired the cook?*" takes the chatbot into the state The cook. When the second occurrence of the utterance "*when did you meet him?*" appears in the dialogue the chatbot answers "*Few years ago*". Fig. 5 shows a finite state automaton that models this dialogue.

After discovering the finite state automaton shown in Fig. 5, the students have to codify it in the Chatbot tool as shown in Fig. 3. The folders The millionaire and The cook represent the two states shown in Fig. 5. Hence, both of them have to contain the rule "*when did you meet him?*". States are represented as folders in the tool; as shown in the left hand panel of the screenshot in Fig. 3. Each folder contains all the rules that are incoming arrows of the state in the finite state automata. When a rule is used during a dialogue—"*The millionaire is single?*"—the chatbot enters the state represented as its containing folder—The millionaire. Using finite state automata terminology, the rules define the *transitions* between states, and the *topics* that the chatbot can talk about are represented as *states*. The different topics that the dialogue can go through are different states that the chatbot can be in. In this way, the chatbot topics and topic transitions can be formalized as a finite state automaton.

To the best of our knowledge, although chatbot programming has been used before to engage students in CS (see Section 2), no educational tool similar to Chatbot has been

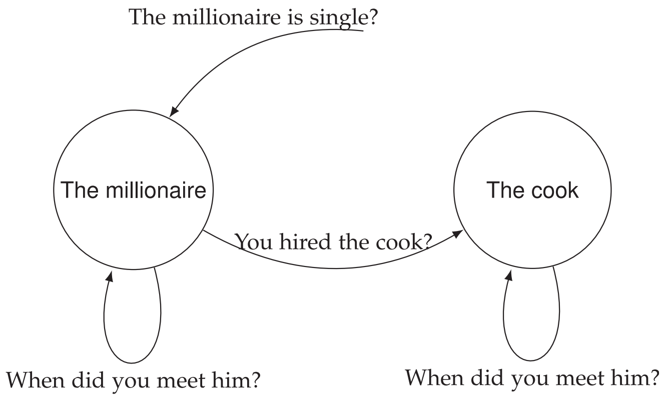

Fig. 5. Finite state automata representing the states and transitions used by the dialogue in Fig. 4.

developed before where fundamental concepts of CS such as variables, conditionals and finite state automata, can be introduced.

3.2 Implementing Formative Assessment in Chatbot

Alibi is the name of the version of Chatbot that generates automatic formative assessment. As we already mentioned, students can play a murder story using Chatbot by choosing one suspect to defend, and by programming their chatbots so that they answer the detective questions properly. In Dale Aceptar 2013 the detective's interrogation questionnaire for each of the suspects along with a log book with his findings and speculations, was made available weekly to the participants by the contest teachers. The formative assessment implementation is based on the detective questionnaires.

In order to win the contest, students need to have a good score in all of the 10 published questionnaires and also in a final one, which is not disclosed before the deadline to upload the produced chatbots. A jury of experts then picks the winners among the top ranked bots, which must be programmed using concepts such as variables and finite state automata in order to handle properly the questionnaires. As in the case of Alice, students learn by watching online Chatbot tutorial videos and using the support fora.

Once the detective bot questionnaires are loaded into Chatbot, the tool simulates a conversation between the detective bot and the suspect bot programmed by the student as shown in Fig. 6. During the simulation Chatbot unifies different declinations of words and uses the most probable parse tree of the questions by including natural language lemmatization and parsing techniques as described in [52]. The answers to the questions are programmed by the student, as explained in Section 3.1, and are matched against the rules that are programmed by the teacher in the detective bot. Fig. 6 shows that the student did not program the finite automata behind the bot properly because it is answering the question about the cook with the same answer as the question about the millionaire. Such answer does not match the patterns $[\text{Some}|\text{Few}|\text{Four}|4] * \text{years}*$ programmed in the detective bot. A correct answer is shown in Fig. 4.

Fig. 6 illustrates the different types of formative feedback implemented in the Alibi version of Chatbot. Questions are flagged as ok (in black) if the answer is correct. Otherwise, Chatbot confesses guilt if it cannot find a matching rule (red flag), or tags an answer as incorrect if there is a rule but the

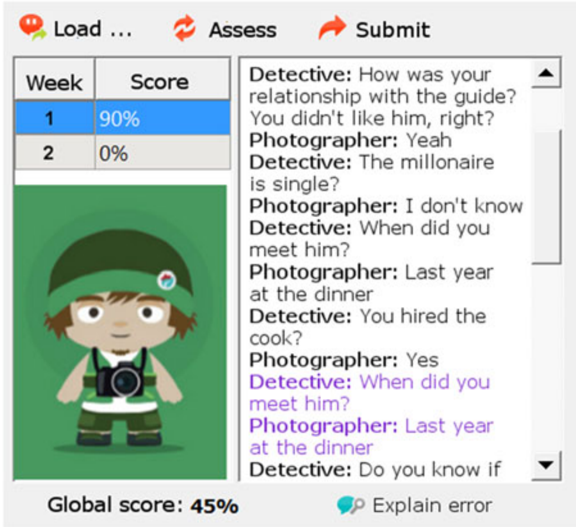

Fig. 6. Chatbot's interface showing formative feedback. The selected questionnaire has been 90 percent correctly solved. The question shown in violet indicates that the answer does not comply with the expected regular expression. The student can get further explanations by pressing the button "Explain error".

output does not match the (encrypted) regular expression that the questionnaire file has for identifying correct answers. In this last case, the incorrect answer is marked in violet as in Fig. 6. Students must keep their bot from confessing but also from flagging answers as incorrect. Based on how well the bot answers, a global score is calculated. In Fig. 6 the global score is 45 percent (the first questionnaire is 90 percent correct and the second is 0 percent). It reaches 100 percent if all questions of the loaded questionnaires are answered properly.

The student can use the button "Explain error" in order to inspect any question that is red or violet. When programming the detective bot, the teacher can associate hints to different questions in the interrogatory such as a link to the part of the detective log that registers how long the suspects have known each other, or a video that explains how finite state automata are used in Chatbot.

The different kinds of formative feedback that Chatbot uses can be classified following Shute [48]. First, the percentage of correctness of the programmed chatbot with respect to predefined questionnaires can be considered as a *verification* type of feedback that provides an assessment for the outcome. Second, Chatbot generates a classification of the errors into "Not addressing the question" and "Addressing the question wrongly". This feedback cannot be directly mapped to a feedback type in [48]. It is quite common for teachers to classify the kind of errors the students make while programming (syntax error versus semantic error for example), but this may not be so common in other disciplines. Third, Chatbot provides the student with a direct link to the part of the program that needs revision. This type of feedback corresponds to *error flagging* according to Shute. Finally, Chatbot can give a hint about the concepts needed to solve the task (for example, by pointing the student to a video or reading material). This last type of feedback can help the student figure out how to fix the error. It can be classified as *hint* according to Shute. Summing up, Chatbot is able to provide verification, error

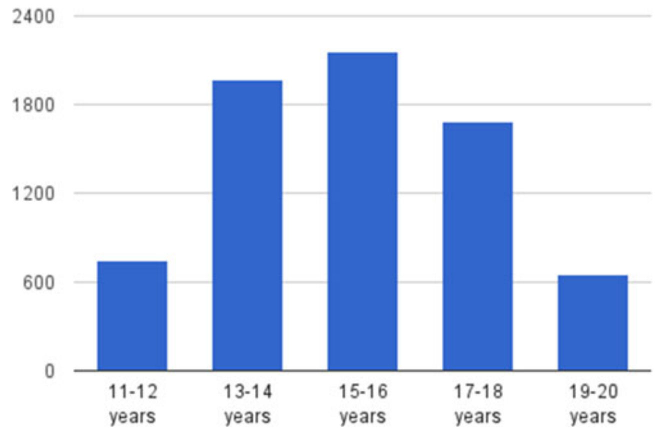

Fig. 7. Age distribution of the Dale Aceptar 2013 participants.

flagging and hint feedback that is immediate and related to the task, in the context of introductory CS.

Alibi addresses the problems found in the first version of Chatbot as follows:

- Impersonation: the "social" component is not present in Alibi, chatbots do not need to be tied to a personal account anymore.
- Unbounded score: there is a maximum achievable score of 100 percent, students always know how they are doing.
- Open ended task: the previous open ended task was turned into a structured one: program an answer for each of the detective's questions.[3] Chatbot has a color code that provides students automatic formative feedback about their progress.
- Too difficult: the questionnaire format helped overcome the difficulties coming from not knowing what others may answer or ask. If an error is found, the student can ask for feedback about what rules need revision.

Altogether, the different elements of Alibi provide a structured task with automatic formative feedback decreasing uncertainty. They also allow questionnaire designers to decide which CS concepts are required to be mastered in order to solve a questionnaire making students focus more on the fundamental CS concepts taught rather than on linguistic issues.

An important advantage of the formative feedback provided by Chatbot is that it is automatic. This feature makes feedback available all the time at no cost, letting teenagers advance at their own pace in their learning process.

4 Chatbot Evaluation

In this section we present the result of two observational studies that analyze the effects of Chatbot on student engagement. The first study compares the use of Chatbot with and without the formative assessment tool in the online contest. The second study compares the use of Chatbot with and without formative assessment in the classroom. We decided to use these studies to analyze the effect of formative assessment in student engagement in Chatbot because although the contexts of the studies were different, the findings were similar.

4.1 Chatbot Evaluation in the Online Contest

As we already mentioned, besides the issue of lack of female participation, with Alibi (which included formative assessment) we intended to improve the completion rate which was only 1 percent in Dale Aceptar 2012 using Chatbot without formative assessment. Although many students signed in 2012, only a very small percentage were self-motivated enough to complete the task and participated until the end of the competition. In this section we report our results using Chatbot with Alibi in the 2013 edition of Dale Aceptar.

During 2013, 9,612 students signed in to participate in Dale Aceptar. They were required to indicate whether they had previous background in programming or CS. If they did, they registered in an advanced track and did not compete with the students with no background. We include in this study only those students with no background in CS or programming: 9,371 teenagers, 8,137 (86.83 percent) being male. Of those, 8,502 participants decided to participate with Alice, and 1,454 decided to go with Chatbot. 585 participants signed in to participate with both (and hence they are included in the number of participants for individual systems). Birth date was self reported and optional, and its verification was relaxed because at the time it was a popular belief that children should not report their true birth date in online forms to protect their personal information from grooming and other forms of abuse. From the 9,371 that registered, 7,216 participants reported their age. Age distribution is shown in Fig. 7. Inside each age group, gender distribution follows the overall proportion (around 85 percent being male).

Argentina has 24 provinces with most of the population living in Buenos Aires Province (38 percent, also taking into account Buenos Aires City, which is politically independent but geographically included), Córdoba (8 percent), and Santa Fe (8 percent). The rest of the population is spread in lower percentages over the remaining 21 provinces. The geographical distribution of the participants closely resembles the population distribution (Buenos Aires City plus Buenos Aires Province having 37 percent, Córdoba having 7.5 percent, Santa Fe having 7.8 percent, etc.).

Table 1 reports the participation and completion rate by gender in Dale Aceptar 2013. More people decided to participate with Alice than with Chatbot (8,502 versus 1,454). We attribute the difference to the fact that most teenagers do not know what a chatbot is while Alice was presented as a tool to program video games and animations, two concepts very familiar to them. Also, the prize of the competition was a gaming console,[4] which attracts gamers.

3. It should be noted that the existence of a final undisclosed questionnaire leads students into not using rules that textually match each given question because some variation of it could reappear under a slightly different form. Thus, they should use the learned concepts to accept more general inputs.

4. Choosing a prize what would be appealing was not easy. Before setting up the competition, a focus group study was conducted to find out how to attract teenagers into the competition. When asked about a prize they would desire, no clear consensual option was self suggested by the female participants, while males immediately and affirmatively suggested a gaming console. When such a prize was presented as an option to females, it was accepted although not as effusively as their male counterparts.

TABLE 1
Comparison of 2013 Dale Aceptar Participants That Registered (Start) versus Those That Uploaded Their Work to the Competition Web Page (End) by Tool

	Alice			Chatbot		
	Start	End	*%End*	Start	End	*%End*
Female	1,022	16	*1.57%*	337	27	*8.01%*
Male	7,480	93	*1.24%*	1,117	75	*6.71%*
%Female	*12%*	*15%*		*23%*	*26%*	
Total	8,502	109	*1.28%*	1,454	102	*7.01%*

The percentage of female participation is also reported.

In order to gain further insights on the differences (if any) in the chatbots produced by male and female participants we asked two Chatbot experts to mark the submitted chatbots based on three criteria: 1) Character modeling: how well the chatbot responses fitted the Alibi character personality and story, and how interesting the responses were, 2) Programming concepts mastery: whether conditionals, variables and pattern matching were used when appropriate, and 3) High level concept mastery: How well states, topics and topic transitions of the Alibi questionnaires were modeled. The marks ranged between zero (failed) and ten (excellent). No significant differences were observed in criteria 1: participants of both genders got both high and low marks in character modeling. But there were considerable differences between criteria 2 and 3 for male and female students. On the one hand, 80 percent of the female chatbots had clearly identified, meaningful, disjoint and exhaustive topics covering all turns in the Alibi conversations and properly modeled topic shifts, while only 33 percent of the male chatbots did. On the other hand, females had more trouble coming up with generic rules (pattern matching and variables) to match several utterances that correspond to a pattern. For example, a female participant programmed 30 rules to match 30 different utterances that could be matched with a single rule that used a variable. That is, most female chatbots rated higher in criteria 3 (high-level CS concept mastery) than in criteria 2 (low-level CS concept mastery). Most initiatives that seek to engage students into CS propose programming challenges (see related work in Section 2). According to these results such initiatives could be more successful attracting females if they proposed CS challenges more related to design and modeling.

Two observations can be made from Dale Aceptar 2013 results from Table 1. First, the percentage of female participation with Chatbot (23 percent) is almost twice the one with Alice (12 percent). This proportion is maintained for those students that completed the competition and handed in a product (Chatbot 25 percent versus Alice 15 percent). Second, completion rate, as measured by number of students who completed their work in the competition, reaches 7.01 percent in Chatbot while it is only 1.28 percent in Alice. Alice completion rate was similar to that of the Dale Aceptar 2012 editions. We also repeat here the results of using Chatbot in Dale Aceptar 2012 (already introduced in Section 2.4). In Dale Aceptar 2012 although 489 students signed in to participate with Chatbot, only a few uploaded complete bots. The task completion rate was 1.2 percent, similarly to with Alice. The female participation with Chatbot in 2012 was 27 percent of females signing up in the Chatbot contest (versus 10 percent in Alice).

We observe that the task completion rate for the students that decided to use Chatbot with formative feedback (in 2013) was almost six times greater (7 percent) than for the students that used Chatbot without formative feedback in 2012. Also, in 2013 the task completion rate with Chatbot was five times greater than the completion rate with Alice (1.28 percent). The female participation rate with Chatbot with and without formative assessment was similar (23 percent versus 27 percent).

Based on these results we pose the following two directional hypotheses for them to be tested in a more observable classroom environment as explained below. Ha is related to gender. *Ha0 (null): There is no significant difference in interest in Chatbot between girls and boys. Ha1 (alt): Girls are significantly more interested in Chatbot than boys.* Hb is related to formative feedback. *Hb0 (null): There is no significant difference in ease of use in Chatbot with formative feedback and without. Hb1 (alt): Chatbot is significantly easier to use with formative feedback.*

4.2 Chatbot Evaluation in the Classroom

At the same time Dale Aceptar 2013 was launched, we conducted a pilot study using both Chatbot with Alibi in three public high schools in the city of Córdoba, Argentina, through a 15-lesson course. Our goal was to evaluate Chatbot in a classical classroom context and not in the self-learning context that Dale Aceptar provides. We also wanted to know how students from poor context, and specially girls, with no previous interest in CS, engaged in programming using Chatbot with Alibi. In these two high-schools the 15-lesson course (which lasted 4 months) was mandatory for students.

Introducing Chatbot in the context of public schools also lets us understand how students use the platform. We agree with Pears et al. [53] that researchers often spend a great amount of time developing a teaching tool, but very little effort disseminating it. Tools need customization and pedagogical work before educational institutions can adopt them.

The Chatbot course was designed to teach students how to program chatbots that play the role of a suspect in Alibi. Tutors visited the schools once a week to teach Chatbot. The lesson design for teaching Chatbot followed a discovery based approach [54]. All lessons had four different segments:

1) Motivation. It aimed to challenge students to use some CS concept. In this segment, the tutor presented students with a goal, such as programming a bot that could reproduce a given dialogue in Chatbot.
2) Tutorial. In this segment the tutor gave a short lecture consisting of an introduction to a CS concept that can be used to solve the problem, e.g., showing how variables are used in Chatbot. Intentionally, the tutor did not solve the problem, leaving room for student discovery.
3) Exploration and production. In this segment students explore the platform, combining the concepts necessary to solve the challenge. The purpose of the

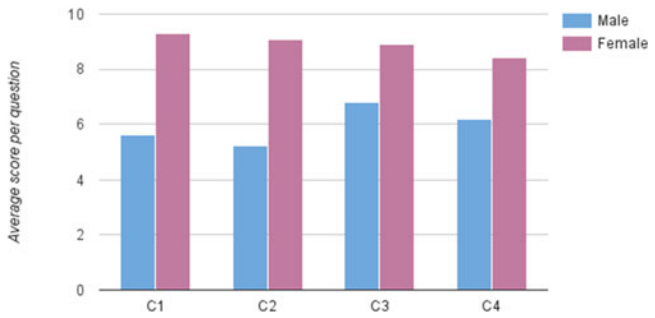

Fig. 8. Average of questions C1, C2, C3, and C4 obtained in the high schools pilot study, discriminated by gender. Data obtained from the student's post-test, N = 46, 25 female, 21 male. Key: 1 = interesting, 2 = learn more, 3 = easy, 4 = successfully.

segment is exposing students to experimentation as a way to gain understanding [55]. This is a central activity of the software development process when there is a general objective but the details of how to accomplish it are unclear. Students work in groups.

4) Show and assess. In the last part of each lesson students share their progress on their chatbots with other students. Student construction, presentation and evaluation of their products has been shown to improve the learning process [56]. Chatbot offers the obtained score and incorrect question flags as a self assessment mode that provides students with feedback on the quality of their rules. Other students could inspect these automatic assessment results and help improve it, or chat with the chatbot in a tab provided for this purpose.

The pilot course was held in two public schools and was attended by 46 students. The average age was 15.4. 55 percent of the students were female. The average age in School 1 was 14.7 and the students were attending their third high school year, while the average age in School 2 was 16.5 and the students were attending their fifth high school year. Students in Córdoba typically attend primary school from age 6 to 11 and then high school from age 12 to 17.

Students from both schools had similar socio-economical situations: all students came from impoverished families. The course was mandatory and taught during school hours, thus there was no student self selection based on their previous interest in CS. The course did not include exams or provide extra credits.

Two tutors with a Masters degree in Computer Science and 3 years of teaching experience taught the course, one in each school. Both tutors were male and 25 years old. A female assistant (majored in Education Sciences) made classroom observations and both the assistant and the tutors filled in post observations notes after each lesson. They were acquainted with the teaching materials having previously participated in the design of the lesson plans. The lesson plans for Chatbot lessons are available at http://masmas.unc.edu.ar/(in Spanish).

For the evaluation, we asked the students to complete a post-test at the end of the course. The post-test included multiple choice questions about their experience with Chatbot as well as open ended questions. At the end of the experience we had multiple sources of data. Quantitative data came from students' answers to the post-test questions, and qualitative data from the assistant lesson observations, tutors post lesson reflections and the students' open questions. We present the quantitative results about Chatbot here and we compare them with the qualitative results in the following sections.

All the following quantitative questions in our questionnaires are based on a scale ranging from 0 (meaning "not at all") to 10 (meaning "very much"). We list here the questions asked to the students.

C1) How *interesting* was learning Chatbot for you?
C2) Do you want to *learn more* using Chatbot?
C3) How *easy* was learning Chatbot for you?
C4) Could you *successfully* do the tasks in Chatbot?

Questions C1 and C2 can be seen as indicators of perceived engagement or usefulness while C3 and C4 are intended to measure perceived ease of use. The standard Technology Acceptance Model (TAM) [57] suggests that when users are presented with a new technology, perceived usefulness and perceived ease of use are the most important factors that influence their decision about how and when they will use it.

In Fig. 8 we compare the average response for the four questions in the post-test discriminated by gender (N = 46, 25 female, 21 male). Girls' self-reported interest is higher than boys' for Chatbot. Girls' interest with Chatbot had an average of 9.6 over 10. Boys' interest with Chatbot had an average value of 5.6 over 10. Based on the data histograms we assume a normal distribution for the answers to the four questions. Hence, we performed independent t-tests on the data and found that the difference is statistically significant with $p = 0.01$. After finishing the course, girls want to learn more using Chatbot (average 9.1) while half of the boys do not (average 5.2), the difference between genders is statistically significant (independent t-tests, $p = 0.01$). These results give evidence for rejecting the null hypothesis Ha0 (null) in favor of the alternative one: girls are significantly more interested in Chatbot than boys. These results are in line with the results reported in Section 4.1 where the female participation percentage was higher with Chatbot than with Alice.

In terms of easiness, again girls found Chatbot easier (average 8.9) than boys (average 6.8). Finally, also girls (8.4) agreed more than boys (6.2) with the question that addressed whether they could successfully complete the tasks posed during Chatbot lessons. These differences were not statistically significant (independent t-texts, $p = 0.09$ and $p = 0.10$).

It is clear that automatic formative assessment is a useful feature for an online massive course, but we wanted to see whether in the classroom it also made a difference as posed by our hypothesis Hb. In order to evaluate this hypothesis we reproduced our 15-lesson course in a third high-school using Chatbot without its automatic formative assessment feature. We also asked the 34 participating students to complete the same post-questionnaire. In this third school 60 percent of the students were female and the average age was 15.7. The course was taught by the teacher that taught at School 1. The results of the post-questionnaire are shown in Fig. 10 and are compared to the average results of schools 1 and 2 reported in Fig. 9.

The figure shows that the self-reported interest and the willingness to learn more is similar for the students no matter

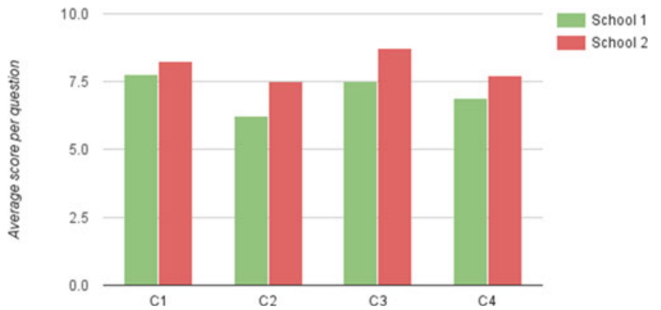

Fig. 9. Average of questions C1, C2, C3, and C4 obtained in the high schools pilot study, discriminated by school. Data obtained from the student's post-test, N = 46, 22 school 1, 24 school 2. Key: 1 = interesting, 2 = learn more, 3 = easy, 4 = successfully.

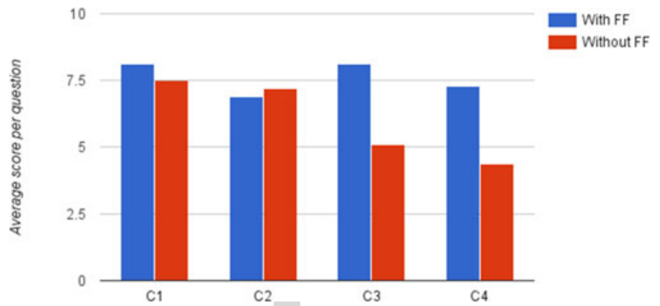

Fig. 10. Average of questions C1, C2, C3, and C4 that compares the two highschools of the pilot study that used Chatbot with automatic formative feedback with a third school that used Chatbot without the automatic formative feedback feature. Data obtained from the student's post-test, N = 78, 46 with formative feedback, 34 without formative feedback. Key: 1 = interesting, 2 = learn more, 3 = easy, 4 = successfully.

whether they had the automatic formative feature, however, there is an statistically significant difference (p=0.01, independent t-tests) in easiness and self-reported task completion. These results provide evidence for rejecting the null hypothesis Hb0. The course was considered significantly easier by the students that had access to the automatic formative feature. In an online massive course, where a teacher is not available, the availability of automatic formative feedback might be even more relevant as suggested by the increase in task completion reported in Section 4.1.

5 Discussion and Qualitative Evidence

We discuss here the reliability and validity issues on our post-test instrument. With respect to *reliability*, we tested the stability of our measurements by discriminating by school the results obtained in the post-test. That is, we tried the reliability of our post-test by a split-half technique. In Fig. 9 we show the results obtained in the post-test discriminated by school (N = 46, 22 school 1, 24 school 2). We performed independent t-tests on the data. The differences found are not statistically significant. Therefore, the results are stable on two different student samples. The most important differences on the two samples are student age and tutor. The average age in School 1 was 14.7, while the average age in School 2 was 16.5. Also, the tutors that taught at each school were different as explained above.

Another aspect of reliability is internal consistency among the questions. In order to verify whether similar questions in our post-test gave rise to similar answers we did a correlation analysis between them. As expected we found that C1 and C2 (which are intended to measure engagement) are strongly correlated with Pearson correlation coefficient r equal to 0.8. That is, the desire of learning more is strongly correlated to whether they found the tool interesting. We also did a correlation analysis between C3 and C4 which are intended to measure ease of use. We found that C3 and C4 are strongly correlated with Pearson correlation coefficient r equal to 0.82.

A potential threat to our *validity* is that, in the classroom, we only asked two questions related to engagement and two questions related to ease of use. However, the results found in the classroom are consistent with the results found in the online competition Dale Aceptar. In Dale Aceptar, Chatbot also increased the participation of girls, which is an indicator of girls' engagement. The completion rate with Chatbot (with formative feedback) was 5 times greater than with Alice, which can be considered an indicator of ease of use. In order to increase the validity of these observational empirical studies, we analyzed the evidence about ease of use and engagement found in the qualitative data collected in the classroom from tutors, students and observers. We describe this analysis in the next two sections.

In Section 5.1 we present the qualitative study on engagement with Chatbot. The Section 5.2 presents qualitative results about ease of use.

5.1 A Deeper Analysis of *Engagement*

Using the quantitative data, we found students average responses on the questionnaires and related them to school and gender. For the analysis of qualitative data, following grounded theory analysis [58], we tagged observation passages, tutors' excerpts and student discourses. In this inductive reasoning process, we sought evidence to generate a category. Then, we compared and contrasted qualitative incidents within the same tag, identifying emerging themes and analytic categories. Following standard qualitative reporting [59], the description of our findings includes an explanation of the emerging theme as well as discourse transcriptions to illustrate and clarify our results.

Tutors reported that introducing Chatbot was fun for students. In the first lecture, we used Chatbot's ability to connect to social networks by telling the students to connect to Facebook and chat with the tutor, all the students at the same time. The tutor connected a chatbot to his account so that he was able to answer automatically to all students. The tutor reported: "*The activity was a success. All of the students were engaged and I saw that they were having fun while chatting with my chatbot. At first they were surprised, after a while they realized they were chatting with a chatbot and found this interesting.*"

In general, almost all lesson observations contained incidents tagged in the engagement category. When describing the attitude of students working on their lesson tutors and assistants used the words "engaged", "interested", "fun" and other synonyms very frequently, all indicators of engagement according to [15]. This data was obtained from 20 post lesson observations written by tutors and observers. One possible explanation is that programming with Chatbot was part of playing the game Alibi.

As with the online experience, we believe that Alibi provided a source of fun. For example, based on post lesson reflections, one lesson included collective testing of some interesting pre-made chatbots (the psychologist and a chatbot that chats about his birthday). As those worked well, students showed interest in seeing how they were programmed. After that segment of the lesson, Alibi was presented. Students got hooked into the characters, and all of them preferred to start creating their own Alibi chatbot instead of trying to build one of the topic of their choice.

A piece of evidence that seemed to suggest that the game setup could have provided a source of engagement comes from the students "exit tickets" where students mentioned what they liked about each day's class. 22 percent of them mentioned they liked the game Alibi ("*I like the questionnaires*", "*I like being the character*", "*I like Alibi*"), while 36 percent of them mentioned they liked the platform Chatbot or chatting with Chatbot. 11 percent used the word "play" or "game" to describe what they liked the most, and 9 percent reported enjoying learning specific CS concepts ("*I liked it when we learned variables/conditionals/states*").

5.2 A Deeper Analysis of *Easiness*

The second emerging theme in the qualitative analysis was that most CS concepts tackled with Alibi were "easy" for the majority of the students. Tutors and assistants reported in their observations that students learned "easily", solved most of the challenges and discovered new rules or instructions to develop their programs. For example one reflection mentioned: "*I asked them to write something that required a conditional and gave the class time to find the right tool to solve the problem. In particular, one of the students found the option 'create a conditional rule' and solved the challenge.*" Other classroom observations considered for this theme reported that students could solve challenges "rapidly". Tutors observed that students easily understood and applied conditionals.

However, some concepts were harder for our students. As an example, in one school, the tutor reported students had difficulties understanding finite state automata, despite eventually being able to apply it into their chatbots. In the other school the opposite happened: when the teacher presented a problem requiring the use of states, two groups of students discovered how to make the Chatbot change from one state to the other on their own.

Based on the analysis of classroom observations we found that students can discover and learn some concepts by themselves, following their own intuitions. For example, variables and conditionals were concepts that students discovered when exploring the platform. Some other concepts, such as finite state automata in Chatbot seemed to require much more thought, practice and analysis for some students. In spite of this, students reported finding Chatbot easy mainly because they could immediately evaluate the results of their programming through the automatic formative feedback feature.

6 Conclusions and Future Work

In this article we documented the creation of Chatbot, a chatbot programming platform whose intent is increasing student task completion and engagement, specially in girls, while teaching basic CS concepts, as a way of promoting interest towards CS-related careers and as a way of contributing to the increasingly important discussion of how to introduce high-school students to CS concepts in an engaging way.

We evaluated Chatbot in two observational studies: an online competition that included the use of the well known educational tool Alice and in an in-class 15-lesson pilot course in two high schools. Combining the results of these studies allowed us to identified general findings with largely quantitative data, and confirm, construct explanations and understandings of these findings, from classroom field work conducted in real school settings. In both studies, most indicators of engagement (participation, task completion, interest, willingness to learn more and self reported interest) were higher for girls than for boys when using Chatbot. In the online study, task completion for the students that decided to use Chatbot was five times higher than for those that used Alice. In the in-classroom pilot course, girls' self-reported interest was considerably higher than boys' as was their willingness to learn more using Chatbot. Moreover, in the online study, girls' participation rates doubled with Chatbot (23 percent) compared to Alice (12 percent).

The interest and willingness to learn more observed in the classroom could be due to good lesson design and highly motivated tutors being in charge of lessons. The same tutors taught boys and girls in the classroom and we found significant differences that cannot be explained by good teaching. Qualitative data showed also that students had "fun".

The differences observed in engagement for boys and girls may be due to the dissimilar concepts that each tool covers. For instance, Alice includes a complete programming language, which could make it harder to use, and although most of the concepts we taught during our courses appear in both tools, not all of them do. If the difference in engagement could be attributed to Chatbot being somehow "incomplete", a teaching strategy could be depicted for girls: start with more structured albeit "incomplete" tools, get them to the "want to learn more" state (Fig. 8) and then move to more powerful platforms.

Learning from our previous experiences, Chatbot was designed so that it easily lends itself to the use of structured tasks and to provide automatic formative assessment. As shown with the qualitative data, at least partially, the results of our observational studies with Chatbot may be explained by these two aspects of Alibi. Structured tasks and formative assessment guide students during the task resolution and give a clear sense of progress. These two factors were absent in the experience implemented with Alice. It is yet to be determined by future research if some type of scaffolding could improve the retention rate of those students that decide to participate with Alice. This is a plausible hypothesis but has its own set of challenges. For instance, work with Alice could also be made more structured, focusing on building a particular type of game instead of each student choosing their favorite, and requiring students to follow some sort of schedule where each week a particular aspect of the game is tackled. However, nothing prevents students from e.g., adding more characters or music (i.e., diverge from the structure). Even in the case of a structured assignment being given, providing support and feedback on their programming in Alice can be done in class, while an online contest would need an immense amount of resources to provide the same level of individual assistance. This feedback has no cost with Chatbot because the tool provides it automatically.

ACKNOWLEDGMENTS

This work was partially supported by grants ANPCyT-PICT-2014-1833, UBACyT 20020130200032BA, CIN-CONICET-PDTS-2015, PID-MINCYT-CBA-2014 and grants by the Argentinean Ministry of Science, Technology and Productive Innovation's Manuel Sadosky Foundation. It was also partially supported by scholarships and grants from the Universidad Nacional de Córdoba and by two Google RISE awards. The authors wish to thank the Google in Education team that organized the Google RISE Summit where we presented the tool and got interesting feedback. They also would like to thank the people from the Manuel Sadosky Foundation that took part in the development of Alibi, Santiago Ceria, Gabriela Di Piazza, Vanina Klinkovich, Mariana Rodrigo, Aaron Wang, and people from the Universidad Nacional de Córdoba that took part in the development and field study: María Emilia Echeveste, Candela Casas, Emmanuel Gunther, Marcos Gómez, Joshep Cortez Sanchez, Eduardo Rodríguez Pesce, Andrés Pagliano, and Pablo Pérez De Angelis.

REFERENCES

[1] M. L. Alonso and G. Molino, "Síntesis de información estadística universitaria argentina 2014–2015," Secretaría de Políticas Universitarias, Ministerio de Educación de la Nación, Argentina, pp. 1–27, 2016.
[2] Observatorio Permanente de la Industria del Software y Servicios Informáticos, "Estado y Perspectivas de la Transformación Digital en las Empresas Argentinas," Cámara de la Industria Argentina del Software, 2016.
[3] L. Carter, "Why students with an apparent aptitude for computer science don't choose to major in computer science," *ACM SIGCSE Bulletin*, vol. 38, no. 1, pp. 27–31, 2006.
[4] S. Grover, R. Pea, and S. Cooper, "Remedying misperceptions of computer science among middle school students," in *Proc. 45th ACM Tech. Symp. Comput. Sci. Educ.*, 2014, pp. 343–348.
[5] C. Wilson, L. A. Sudol, C. Stephenson, and M. Stehlik, "Running on empty: The failure to teach K-12 computer science in the digital age. association for computing machinery," Comput. Sci. Teachers Assoc., New York, NY, 2010.
[6] S. Furber, "Shut down or restart? The way forward for computing in UK schools," Roy. Soc., London, U.K., Tech. Rep. DES2448, 2012.
[7] T. Bell, P. Andreae, and L. Lambert, "Computer science in New Zealand high schools," in *Proc. 12th Australasian Conf. Comput. Educ.*, 2010, pp. 15–22.
[8] I. Zur Bargury, "A new curriculum for junior-high in computer science," in *Proc. 17th ACM Annu. Conf. Innovation Technol. Comput. Sci. Educ.*, 2012, pp. 204–208.
[9] Computer science for all iniative. [Online]. Available: https://www.whitehouse.gov/blog/2016/01/30/computer-science-all, Accessed on: Sep. 24, 2016.
[10] T. Bell, "Establishing a nationwide CS curriculum in New Zealand high schools," *Commun. Assoc. Comput. Machinery*, vol. 57, no. 2, pp. 28–30, 2014.
[11] S. Cooper, W. Dann, and R. Pausch, "Teaching objects-first in introductory computer science," in *Proc. 34th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2003, pp. 191–195.
[12] W. Dann, S. Cooper, and D. Slater, "Alice 3.1 (abstract only)," in *Proc. 44th ACM Tech. Symp. Comput. Sci. Educ.*, 2013, pp. 757–757.
[13] G. Bonder, "Youth, gender & ICT: Imaginaries in the construction of information society in latin america," *Arbor*, vol. 184, no. 733, pp. 917–934, 2008.
[14] R. S. C. Espinosa, I. G. Medina, and Z. González, "Consumption of digital media by children and pre-teenagers in Catalonia, Spain," *Zer: Revista de Estudios de Comunicacion*, vol. 20, no. 39, pp. 145–162, 2015.
[15] F. Newmann, *Student Engagement and Achievement in American Secondary Schools*. New York, NY, USA: Teachers College Press, 1992.
[16] C. Wilson, "Hour of code: We can solve the diversity problem in computer science," *ACM Inroads*, vol. 5, no. 4, pp. 22–22, Dec. 2014.
[17] T. Bell, P. Curzon, Q. Cutts, V. Dagiene, and B. Haberman, "Introducing students to computer science with programmes that don't emphasise programming," in *Proc. 16th ACM Annu. Joint Conf. Innovation Technol. Comput. Sci. Edu.*, 2011, pp. 391–391.
[18] K. Doran, A. Boyce, S. Finkelstein, and T. Barnes, "Outreach for improved student performance: A game design and development curriculum," in *Proc. 17th ACM Annu. Conf. Innovation Technol. Comput. Sci. Educ.*, 2012, pp. 209–214.
[19] M. Resnick, et al., "Scratch: Programming for all," *Commun. ACM*, vol. 52, no. 11, pp. 60–67, Nov. 2009.
[20] A. Kerly, P. Hall, and S. Bull, "Bringing chatbots into education: Towards natural language negotiation of open learner models," *Knowl. Based Syst.*, vol. 20, no. 2, pp. 177–185, Mar. 2007.
[21] A. Latham, K. Crockett, D. McLean, and B. Edmonds, *Oscar: An Intelligent Adaptive Conversational Agent Tutoring System*. Berlin, Germany: Springer, 2011, pp. 563–572.
[22] A. C. Graesser, P. Chipman, B. C. Haynes, and A. Olney, "AutoTutor: An intelligent tutoring system with mixed-initiative dialogue," *IEEE Trans. Educ.*, vol. 48, no. 4, pp. 612–618, Nov. 2005.
[23] A. Shaw, "Using chatbots to teach socially intelligent computing principles in introductory computer science courses," in *Proc. IEEE 9th Int. Conf. Inf. Technol.*, 2012, pp. 850–851.
[24] M. Keegan, R. D. Boyle, and H. M. Dee, "Turi: Chatbot software for schools in the turing centenary," in *Proc. 7th Workshop Primary Secondary Comput. Educ.*, 2012, pp. 153–154.
[25] J. P. Bigham, M. B. Aller, J. T. Brudvik, J. O. Leung, L. A. Yazzolino, and R. E. Ladner, "Inspiring blind high school students to pursue computer science with instant messaging chatbots," in *Proc. 39th SIGCSE Tech. Symp. Comput. Sci. Educ.*, 2008, pp. 449–453.
[26] J. Margolis and A. Fisher, *Unlocking the Clubhouse: Women in Computing*. Cambridge, MA, USA: MIT Press, 2003.
[27] E. F. Gross, "Adolescent internet use: What we expect, what teens report," *J. Appl. Develop. Psychology*, vol. 25, no. 6, pp. 633–649, 2004.
[28] M. M. Voyles, S. M. Haller, and T. V. Fossum, "Teacher responses to student gender differences," in *Proc. 12th Annu. SIGCSE Conf. Innovation Technol. Comput. Sci. Educ.*, 2007, pp. 226–230.
[29] J. Black, J. Brodie, P. Curzon, C. Myketiak, P. W. McOwan, and L. R. Meagher, "Making computing interesting to school students: Teachers' perspectives," in *Proc. 18th ACM Conf. Innovation Technol. Comput. Sci. Educ.*, 2013, pp. 255–260.
[30] M. Lemaire, "Incorporating computer science into an elementary school curriculum," Student Projects, Computer Science Undergraduate Society, McGill University, 2014.
[31] S. J. Wille and D. Kim, "Factors affecting high school student engagement in introductory computer science classes (abstract only)," in *Proc. 46th ACM Tech. Symp. Comput. Sci. Educ.*, 2015, pp. 675–675.
[32] M. Papastergiou, "Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation," *Comput. Educ.*, vol. 52, no. 1, pp. 1–12, Jan. 2009.
[33] Y.-T. Carolyn Yang and C.-H. Chang, "Empowering students through digital game authorship: Enhancing concentration, critical thinking, and academic achievement," *Comput. Edu.*, vol. 68, pp. 334–344, Oct. 2013.
[34] L. Murphy, B. Richards, R. McCauley, B. B. Morrison, S. Westbrook, and T. Fossum, "Women catch up: Gender differences in learning programming concepts," *ACM SIGCSE Bulletin*, vol. 38, no. 1, pp. 17–21, 2006.
[35] B. Barron, "Learning ecologies for technological fluency: Gender and experience differences," *J. Educational Comput. Res.*, vol. 31, no. 1, pp. 1–36, 2004.
[36] R. E. Pattis, *Karel the Robot: A Gentle Introduction to the Art of Programming*, 1st ed. New York, NY, USA: Wiley, 1981.
[37] B. Harvey, *Computer Science LOGO Style. Vol. I: Intermediate Programming*. Cambridge, MA, USA: Massachusetts Inst. Technol., 1985.
[38] I. Utting, S. Cooper, M. Kölling, J. Maloney, and M. Resnick, "Alice, greenfoot, and scratch–a discussion," *Trans. Comput. Educ.*, vol. 10, no. 4, pp. 17:1–17:11, Nov. 2010.

[39] M. Kölling, "The greenfoot programming environment," *ACM Trans. Comput. Educ.*, vol. 10, no. 4, pp. 14:1–14:21, Nov. 2010.

[40] W. Dann, D. Cosgrove, D. Slater, D. Culyba, and S. Cooper, "Mediated transfer: Alice 3 to Java," in *Proc. 43rd ACM Tech. Symp. Comput. Sci. Educ.*, 2012, pp. 141–146.

[41] P. Brusilovsky, E. Calabrese, J. Hvorecky, A. Kouchnirenko, and P. Miller, "Mini-languages: A way to learn programming principles," *Educ. Inf. Technol.*, vol. 2, no. 1, pp. 65–83, Jan. 1998.

[42] C. Kelleher and R. Pausch, "Lowering the barriers to programming: A taxonomy of programming environments and languages for novice programmers," *ACM Comput. Surveys*, vol. 37, no. 2, pp. 83–137, Jun. 2005.

[43] P. Brusilovsky, et al., "Increasing adoption of smart learning content for computer science education," in *Proc. Working Group Rep. Innovation Technol. Comput. Sci. Educ. Conf.*, 2014, pp. 31–57.

[44] W. Hartmann, J. Nievergelt, and R. Reichert, "Kara, finite state machines, and the case for programming as part of general education," in *Proc. IEEE Symposia Human Centric Comput. Languages Environ.*, 2001, pp. 135–141.

[45] C. Seals, M. B. Rosson, J. M. Carroll, T. Lewis, and L. Colson, "Fun learning Stagecast Creator: An exercise in minimalism and collaboration," in *Proc. IEEE Symposia Human Centric Comput. Languages Environ.*, 2002, pp. 177–191.

[46] D. Koller, A. Ng, C. Do, and Z. Chen, "Retention and intention in massive open online courses: In depth," *Educause Rev.*, Louisville, CO, USA, Tech. Rep. 3613, 2013.

[47] D. Wiliam, "Five "key strategies" for effective formative assessment," presented at the Nat. Council Teachers Math., Reston, VA, USA, 2007.

[48] V. J. Shute, "Focus on formative feedback," *Rev. Educational Res.*, vol. 78, pp. 153–189, 2008.

[49] P. Black and D. Wiliam, "Developing the theory of formative assessment," *Educational Assessment Eval. Accountability*, vol. 21, no. 1, pp. 5–31, 2009.

[50] D. Wiliam, "What is assessment for learning?" *Studies Educational Eval.*, vol. 37, no. 1, pp. 3–14, 2011.

[51] R. Butler, "Task-involving and ego-involving properties of evaluation: Effects of different feedback conditions on motivational perceptions, interest, and performance," *J. Educational Psychology*, vol. 79, no. 4, pp. 474–482, 1987.

[52] L. Padr and E. Stanilovsky, "Freeling 3.0: Towards wider multilinguality," presented at the Language Resources Eval. Conf., Istanbul, Turkey, May 2012.

[53] A. Pears et al., "A survey of literature on the teaching of introductory programming," *ACM SIGCSE Bulletin*, vol. 39, no. 4, pp. 204–223, 2007.

[54] M. J. Prince and R. M. Felder, "Inductive teaching and learning methods: Definitions, comparisons, and research bases," *J. Eng. Educ.*, vol. 95, no. 2, pp. 123–138, 2006.

[55] J. O'Kelly and J. P. Gibson, "RoboCode & problem-based learning: A non-prescriptive approach to teaching programming," *ACM SIGCSE Bulletin*, vol. 38, no. 3, pp. 217–221, 2006.

[56] T. L. Naps, et al., "Exploring the role of visualization and engagement in computer science education," *ACM SIGCSE Bulletin*, vol. 35, no. 2, pp. 131–152, 2002.

[57] N. Marangunić and A. Granić, "Technology acceptance model: A literature review from 1986 to 2013," *Universal Access Inf. Soc.*, vol. 14, no. 1, pp. 81–95, 2015.

[58] J. Creswell, *Research Design: Qualitative, Quantitative, and Mixed Methods Approaches*, 4th ed. Newbury Park, CA, USA: Sage, 2013.

[59] A. Berglund, M. Daniels, and A. Pears, "Qualitative research projects in computing education research: An overview," in *Proc. 8th Australasian Conf. Comput. Educ.*, 2006, pp. 25–33.

Luciana Benotti received the Erasmus Mundus MS degree in computational logic from the Free University of Bolzano, Italy, and the PhD degree in computer science from the Universitè de Lorraine, France at INRIA. She is currently a professor with the Universidad Nacional de Córdoba and a researcher in the Argentinean National Council of Technical and Scientific Research (CONICET). Her research interests include natural language processing, automated inference, and computer science education. She has authored 47 papers in these areas. She is a member of the Association for Computing Machinery (ACM) and the Association for Computational Linguistics (ACL).

María Cecilia Martínez received the PhD degree in educational policy from Rutgers University. She is a professor with the Universidad Nacional de Córdoba, and a researcher in the Argentinean National Council of Technical and Scientific Research (CONICET). Her research interests include teacher professional development, educational innovation, and computer science teaching and learning. She has authored 28 papers in these areas. She is a member of the Association for Computing Machinery (ACM).

Fernando Schapachnik received the PhD degree in computer science from the Universidad de Buenos Aires, where he is an assistant professor and researcher. He also serves as director of the Program.AR Iniciative of the Argentinean Ministry of Sciences. His research interests include automatic legal reasoning, deontic logic, and introducing computer science in schools. He is a member of the Association for Computing Machinery (ACM).