Exploring the integration of teaching capabilities into a CSCP framework through help agents

Jean-Paul Sansonnet[1], Evandro Milletto[2], François Bouchet[1], Marcelo Pimenta[2]

[1]LIMSI-CNRS, BP 133, 91403 Orsay cedex, France

[2]Instituto de Informática – Universidade Federal do Rio Grande do Sul (UFRGS)
Caixa Postal 15.064 – 91.501-970 – Porto Alegre, RS – Brazil

{miletto,mpimenta}@inf.ufrgs.br, {jps,bouchet}@limsi.fr

Abstract*. In this paper we describe and we evaluate the first stages of a strategy for attaching basic teaching capabilities to Computer Supported Cooperative Prototyping (CSCP) environments. The strategy is based on the integration of learning functions in the reasoning module of an Embodied Conversational Agent in charge of the Help Function of the application. This paper presents a CSCP framework dedicated to music prototyping on the Web which enabled us to conduct a first experiment to evaluate the grounds of our strategy.*

1. Introduction: adding *acceptable* pedagogical viewpoint to a CSCP

Computer Supported Learning Environments (CSLE) have long been a top-down approach to propose learning functions for study and training [Dillenbourg 1995, Selwyn 2007]: given an educational strategy, CSLE are intended to mediate and facilitate the task of a tutor who manages students in their individual learning process. More recently, the development of the Internet made it possible to propose Computer Supported Collaborative/Cooperative Learning Environments (CSCL), where students can interact through Web pages [Stahl 2006]. In this new context, the roles of tutors and students are unchanged but the *cooperation* between the students (a traditional concept in the learning community [Slavin 1989, Totten 1991]) is strongly encouraged [Nett 2008, Nussbaum 2007] to enhance *critical thinking* [Amobi 2007], thus inducing new learning strategies based on Communities of Practice (CoP) [Brown 1991, Wenger 1998]. Indeed, this pedagogical potential of cooperative and creative systems may be explained by Vygotsky's theories [Vygostsky 1980], which suggest that social interactions play a fundamental role in shaping internal cognitive structures. According to Vygotsky, cognitive development derives from the person's engagement in cooperative problem solving. In these situations, the learner is forced to examine his thinking when challenged by others, and in turn to keep an eye out for possible mistakes made by his collaborators.

On the other hand, web-based collaborative environments are being developed for many purposes, not necessarily related to learning: first for cooperative prototyping [Bodker 1991] and more recently for ludo-creative social communities sharing images (e.g. Flickr, Picasa…), video (e.g. YouTube) or music making [Iazzetta 1998, Weinberg 2002]. Computer Supported Cooperative Prototyping (CSCP) is an activity based on Internet frameworks where users can: a) build artifacts and share their artifacts with other users; b) learn by doing/modifying and by sharing artifacts and knowledge (mainly help information from peers) with other users. Here users build *a priori* and they learn from experience and help-seeking among peers; moreover the static and hierarchical roles of tutor/students are replaced by the more dynamic roles of experts/novices regarding different topics (the same person can be both expert about topic 1 and novice about topic 2). Along with the fast spreading of the notion of Serious Games [Susi 2007, Luckin 2007], there is an enticement to take advantage of the unquestionable appeal of CSCP to a large population plagued by the 'motivation paradox' [Carroll 1987] and who feels uncomfortable with the top-down approach provided by CSLE/CSCL environments.

However, as stated by [Nitzke 2002], CSCP cannot be viewed as CSCL and even less CSLE, for two reasons: a) typically there is no human or artificial tutor; b) more crucial, there is no explicit educational strategy implemented. Nonetheless, one can see the benefit to build an educational strategy over CSCP frameworks. Note that an alternative would be to equip traditional CSLE (or more appropriately CSCL) environments with collaborative prototyping (CP) functionalities but we believe that approach would be less appropriated since: a) existing CSCL are complex and technically not easy to transform into CSCP; b) the lack of motivation would remain unchanged.

Attaching educational functions to (existing or envisaged) CSCP environments leads to two main problems: a) the educational objectives must be limited and the educational functions adapted. Finding how much users can learn by building/experimenting/playing is a major problem which is not discussed in this paper; b) the tutoring system must be acceptable in the sense of [Davis 1989], otherwise it will be discarded, or the complete CSCP framework will be abandoned. Resolving this second problem is not a theoretical prerequisite to the first one, but in practice it will strongly influence/restrain the tractable domain of educational objectives. This is the reason why we will focus on this key issue. Our approach to the issue of acceptability is based on three main propositions:

1) To provide educational functions at the initiative of the user exclusively, not to be considered intrusive and/or patronizing.
2) To take advantage of the opportunity of interaction provided by the user when he/she is in need for help. Consequently, the educational functions will be provided together with the Assistance Function of the help system. Thus, novice users who often request help are more likely to interact with the educational system while expert users will be less affected. We are fully aware of idiosyncrasies related to *help functions* (e.g. see [Godbole-Chaudhuri, 2006]) which makes it more difficult to attach learning functions to a help agent.
3) Assistance and Educational functions will be implemented through an Embodied Conversational Agent (ECA) [Cassell 2000], attached to the CSCP framework. ECAs are virtual characters displayed in the application window that can interact with the users in a multimodal way and play various roles (presenters, helpers, companions, tutors…). In our context, they have the three following advantages: a) they are considered friendly; b) despite the 'Clippy effect', they have been proven to be efficient tutors [Lester 1997]; c) they are more enticing to develop a transfer of the dialogue topic to educational concerns related to the focus of the help request at hand.

The outline of the paper is as follow: Section 2 presents the supporting CSCP software architecture (dedicated to music prototyping) and describes the three stages of our strategy. Section 3 presents the experiment carried out in order to collect a corpus of user's natural language requests and the results of the data analysis. The paper is concluded by a discussion about the feasibility of an automatic integration of learning functions into the help function of CSCP environments.

2. Architecture and strategy

Architecture. In the context of the new web-based frameworks for shared prototyping, we have been developing the Web 2.0 platform CODES (**CO**operative **M**usic Prototype **DES**ign) dedicated to cooperative music prototyping, and freely available at [CODES url]. It aims at allowing novice users (i.e. users with no particular music knowledge) to experiment with music samples and to interact with each other on the Web, in order to create simple music pieces; herein called *music prototypes*. CODES is a large-size Web 2.0 application and hence a good candidate for an experiment involving a large-scale problem. Moreover, music is a handy topic to study relationships between prototyping and learning, as it involves both ludo-creative and formal-knowledge skills.

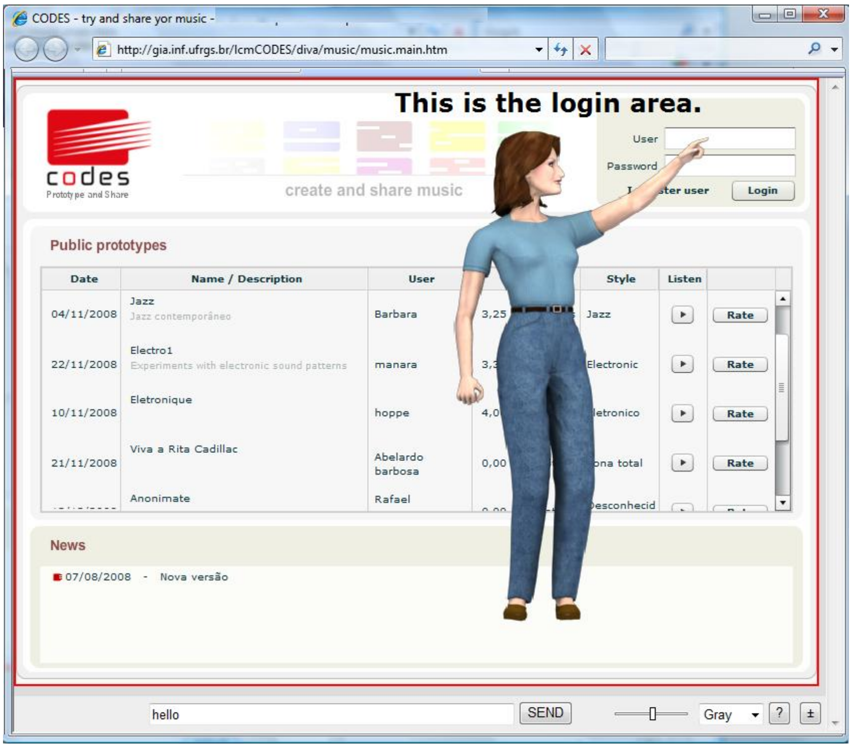

Figure 1. The CODES main page displays a DIVA agent providing information about an area of the screen, in reaction to a user's click. Natural Language requests can be typed in the chat box of the agent's control bar (at the bottom).

In order to study the Function of Assistance in the context of Rich Internet Applications (RIA), we have developed an experimental toolkit called DIVA, freely available for research and education purposes at [DIVA url]. DIVA stands for "**D**OM-**I**ntegrated **V**irtual **A**gents", which emphasizes its full Web 2.0 approach to assisting tools: the toolkit is written in JavaScript with additional symbolic computing available through a server technology. DIVA supports a) virtual graphic characters that personify the assisting agent, b) a NLP-chain that analyzes and resolves the users' questions and c) an asynchronous link to the server for access to resources and client information storage.

The CODES and DIVA are two Web 2.0 applications which have been developed separately and independently. However, being based on the same technology, DIVA has been viewed as a good candidate for the integration of a Virtual Assisting Agent into the CODES framework in order to support the studies about the integration of learning functions in the CSCP context.

Consequently, the DIVA toolkit has been integrated into the CODES framework, producing the CODIVA application [CODIVA url], shown in Figure 1.

Strategy. The rapid and successful development of the CODIVA architecture has proven that the technical part of our approach is quite feasible. Now, the main issue remains to be investigated: is it possible and useful to integrate learning functions in CSCP applications with embedded Assisting Conversational Agents? In order to answer that question, at least three successive stages of increasing complexity must be carried out:
1) Measuring the degree of interaction between the users and the agent. Do they actually need help? If so, do they call the agent? Note that in the particular context of CODIVA, we are privileging Natural Language modality (requests typed in the chat box), putting aside for the moment the multimodal dimension of the interactions. To achieve this task, we have to conduct some experiments, involving human subjects using CODIVA, in order to collect a controlled corpus of Natural Language requests.
2) Analyzing the user's natural language requests. This stage has three complementary purposes: a) categorizing the elements of the requests of the collected corpus; b) categorizing the subjects' behaviors; c) exhibiting elements of the requests that can be linked to further learning functions.
3) Integrating a learning strategy into the assisting agent. This strategy can be integrated to the rational part of the agent, i.e. the symbolic reasoning process in charge of resolving the users' requests. Hence, the agent will not just provide direct answers, but could add extra information about the focus of the request; moreover it can also enter into a dialogical session to check the user's knowledge and so on.

Because this work is still in progress, in this paper we restrict ourselves to the first and second stages; the third shall only be part of the discussion at the end of the paper. In the next section we describe the principle of the experiment carried out to collect and analyze a first corpus of users' requests.

2.1. Three Stages of Experiment procedure

Stage 1 *Corpus collecting conditions*: This experiment aims at collecting a small corpus of Natural Language Requests of Assistance uttered by novice users while interacting with the CODES framework to perform simple tasks about music prototyping. Subjects, both male and female, of age 20 to 30, are college students with no expert skills in computer science or music. Subjects are told to 'think aloud' [Ummelen 2000], that is to express orally the requests they could have while performing the proposed tasks as if there was an expert 'friend behind their shoulder' [Capobianco 2001] played by a Wizard of Oz (WoZ) operator. This choice was made to let them be as free as possible in the utterance of their requests.

Stage 2 *Requests analysis*: In order to answer the question of the degree of interaction, it is possible to make a quantitative analysis of the subjects' requests by measuring their loquacity (Section 3.3). Beyond that, a qualitative analysis can be performed using Natural Language Processing (NLP) techniques. Because the collected corpus is small (about 150 requests), it is possible to perform a manual transcription of the subject's natural language requests into a formal syntax of the general structure (see actual examples in table 3):

Speech Act(Predicate(Reference-1, Reference-2, ..))

From the experience of the DIVA agents, we know that it is possible to exhibit three main types of entities in the subjects' requests, as described in the following list:

Type	Description	Number
Speech Acts	They express the mental position taken by the speaker about the propositional content of the utterance [Searle 1969, Harnish 1994].	6 classes
Predicates	They are mainly linked to action verbs identifying the operations in the CODES graphical user interface but also to mental verbs about the status of the elements.	35 classes
References	They are the elements categorized by the users and targeted through the referential expressions.	14 classes

Note: The numbers of classes shown in the right column are taken from the collected corpus (see 3.3).

These elements can be divided into two main categories:

- Domain independent (or generic) elements: mainly speech acts and mental predicates. They could be significant when we will assess the generic help-profile of the subjects, i.e. their behavior in front of **any** application they are not familiar with.
- Domain dependant (or specific) elements: concern mainly the predicates of operations and most of the references. They could be significant when we will find a way to integrate learning information within the agent's answers.

3. Experiment

Description of the tasks. A typical CODES scenario requires from the users that they edit a music prototype, which means choosing a music style together with their related sound patterns, then create and play a sequence of patterns, and finally edit the sequence they have created. These three stages have been decomposed into five basic tasks (described in Table 1 and illustrated in Figure 2), proposed tasks in sequential order (one needs to finish T_i before T_{i+1}) and of increasing difficulty (T_i requires more CODES skills than T_{i+1}).

Table 1. Description of the five basic tasks users have to accomplish in CODES.

ID	Task *summary* and objective
T1	*"Choose a) a preferred musical style and b) 3 preferred sound patterns in that style."* Identifying the sound library area (cf. (1) in figure 2b), the musical styles available and their sound patterns, selecting three of them. The idea is to check if the users could identify and navigate through the different musical styles described by the tabs (Rock, Funk, Jazz, etc.) and listen to different sound patterns of each style.
T2	*"Put the sound patterns in the editing area to compose a musical sequence."* Adding sound patterns in the editing area to create musical sequences, to check if the users are able to drag icons from the sound library and to drop them onto the editing area (cf. (3) in figure 2b). Red arrow in figure 2a shows the action to achieve.
T3	*"Listen to the sequence you have created."* Checking if the users identify and associate the execution control buttons with the editing area. When clicked on, the "Play button" (cf. (2) in figure 2b) changes itself into a "Pause button" and is filled with a gray color in order to give a feedback to the user about how to stop or pause the listening (figure 2a).
T4	*"Delete one sound pattern you didn't like, changing for another one."* Checking if the user understands how to exclude a sound pattern and to replace it by another one. The metaphor adopted here is the *non-intuitive* usage of a 'broom' button (see figure 2b) to enable the erase function.
T5	*"Create a music prototype where some sound patterns (three at least) must be played at the same time; use for that at least five sound patterns."* Checking the understanding of the notion of musical track, allowing users to test the combination of samples by playing them at the same time. Users should be able to put sound patterns under or above each other and to listen to that combination.

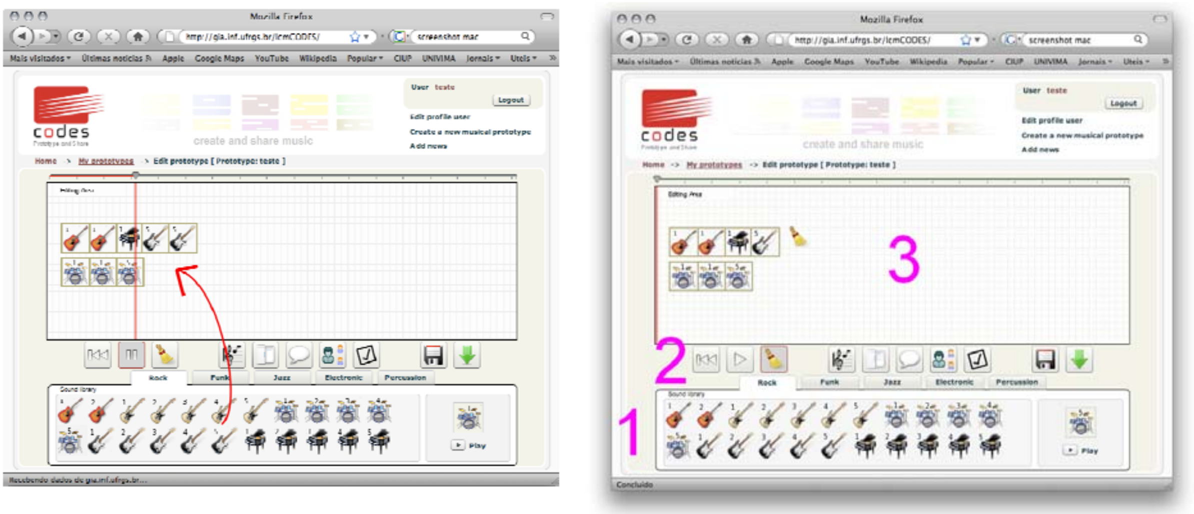

Figure 2. a) on left, Task T3: creating and then listening to a musical sequence; b) on right, Task T4: Deleting a sound from the editing area with the broom tool.

Quantitative analysis of the loquacity. The collected corpus made it possible to produce a first quantitative analysis of the requests. A full statistical analysis is described in [Manara 2009] from which is extracted the table 2 that displays the number of produced requests, by subject and by task.

Table 2. Number of produced requests, by subject and by task.

Subject	Task 1	Task 2	Task 3	Task 4	Task 5	Subject total
Aleyda	3	0	7	5	0	15
Francisco	5	2	2	0	2	11
Hector	1	2	0	0	2	5
Mariano	4	3	0	1	1	9
Priscilla	5	3	1	1	2	12
Judith	1	2	1	3	2	9
Taysa	4	3	1	2	2	12
Rodrigo	3	1	2	2	0	8
Fabiola	4	1	1	3	0	9
Fernanda	3	1	2	1	1	8
Julieta	3	3	4	6	1	17
Julia	2	1	1	0	0	4
Task total	38	22	22	24	13	119

The last line of table 2 gives the count of requests for each task over all the subjects. Whereas a lot of questions are put at task 1, the number of questions tend to decrease during the session. Indeed we have observed that the subjects rapidly become familiar with the application and need less assistance although the tasks are proposed with an increasing difficulty. We have also noted that tasks T4 and T5 require a new way to use the GUI: this leads to a momentary surge of questions at T4 before they drop again. The last column of table 2 gives the count of requests for each user over all the tasks. It shows a large spectrum of interpersonal differences (ratio of ~4) between the subjects. This allows a classification of the subjects into three main behavioral classes: mute{Julia, Hector}, normal{others}, loquacious{Julieta, Aleyda}.

Natural language analysis of the requests. A manual transcription of the collected corpus into the formal syntax described in section 2.3 has been carried out. An excerpt is given in Table 3.

Table 3. Examples of transcriptions from natural language into formal requests.

Can I put it back to the library? [it = SP]	`CANI(Move(SP,SL))`
Can I try all the styles?	`CANI(Try(MUSIC)`
How can I stop it? [it = SP]	`HOWTO(Stop(SP))`
How one can do to put the sound pattern there [there = EA]	`HOWTO(Move(SP,EA))`
How can I take off the broom?	`HOWTO(Quit(BROOM))`
How can I play the whole sequence?	`HOWTO(Play(MP))`
How can I come back? [previous = EA]	`HOWTO(Go(EA))`
What do you mean with 3 sound patterns?	`ASK(Meaning(SP))`
Should I listen to all of them? [all of them = SL]	`SHOULDI(Listen(SL))`
Should I drag and drop? [to put them=SP in the EA]	`SHOULDI(Move(SP,EA))`
Where is the editing area?	`ASK(Location(EA))`
Etc.	`Etc.`

Note: Between [] is the contextual information collected during the session which enables various kinds of deictics to be resolved.

The transcription of the collected corpus exhibited three categories of formal entities:

Speech acts: HOWTO (27%), CHECK (26%), SHOULDI (16%), ASK (15%), CANI (10%), WHY (3%), misc. (1%). The distribution is in a close range, at least for the first classes.

Predicates: Meaning, Listen, Choose, Click, Record, Find, Not, Change, Go, Stop, Delete, Quit, Move, Location, Available, Possibility, Put, Play, Significance, Obligation, Effect, Undo, Listof, Equals, Difference, Do, Try, Angry, Repeat, Use, Like, Correct, Found, Say, Necessity. They are listed from most frequent to least frequent; unlike the speech acts, their distribution follows a Zipf curve. The gloss of the predicates is detailed in [Manara 2009].

References: SP (47), EA (18), SL (14), MUSIC (11), GUI (10), BROOM (9), NUM (6), SPL (4), MP (3), MANUAL (1), MUSICSTYLE (1), ALIGNEMENT (1), SPEA (1), Misc (). Their frequency is given within (). References denote entities of the CODES application as they are actually categorized by the subjects. Often they have a graphical representation on screen, e.g. SP = Sound pattern; EA = Editing area; SL = Sound library; MUSIC = a musical instrument; GUI = any generic GUI item (like a button); NUM = a number; etc.

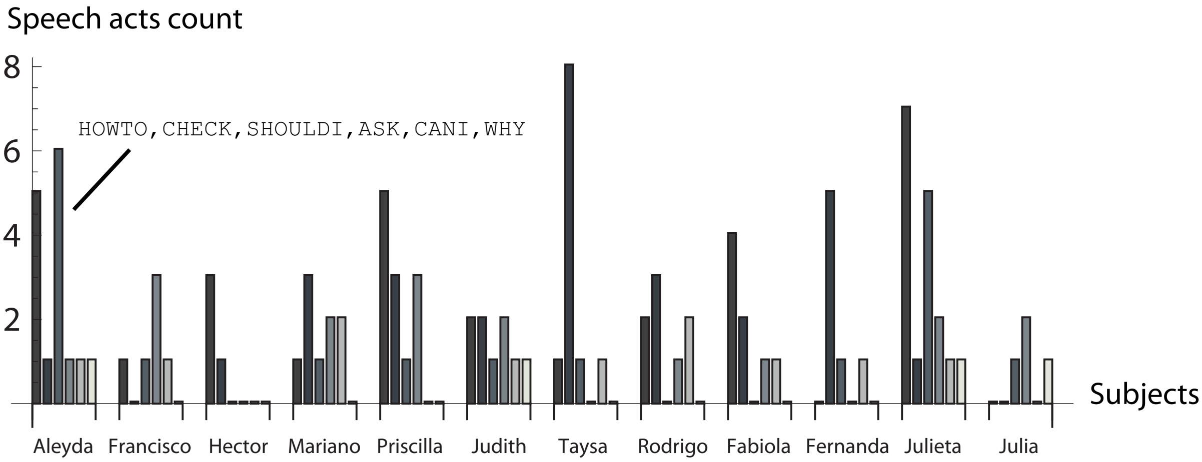

Figure 3. Distribution of the speech acts. For each subject, the occurrences of the 6 speech acts classes are given, from left to right in order listed above.

The distribution of the speech acts displayed in Figure 3 shows a large interpersonal difference, especially with the three most used speech acts (HOWTO, CHECK, SHOULDI). Therefore it would be difficult to exhibit a direct categorization of the subjects on this unique basis. However, the speech acts can be associated with behavioral categories. For example, CHECK, SHOULDI, CANI are typical of a person that constantly asks for reassurance to the agent before doing an action, whereas HOWTO, ASK and WHY are typical of a person that consider the agent as a mere service. To evaluate this hypothesis, the profile of the subjects has been plotted on a 2D plan where X = count(CHECK, SHOULDI, CANI) and Y = count(HOWTO, ASK, WHY) as shown in Figure 4.

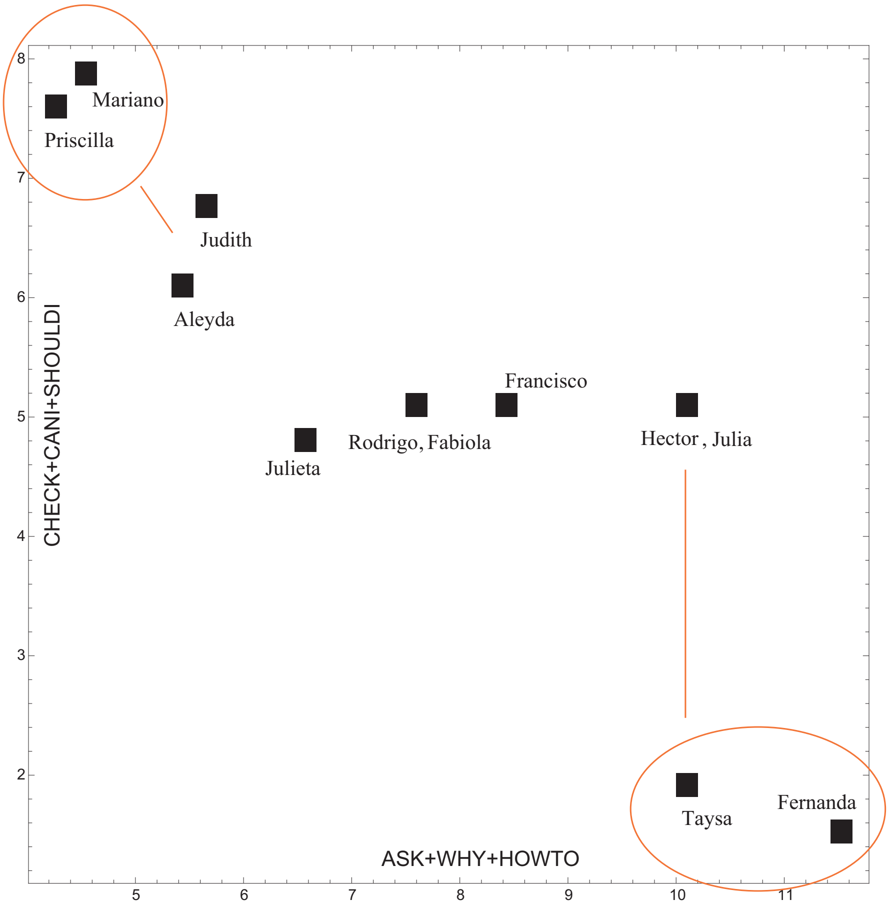

Figure 4. Distribution of the subjects on a Dominant × Hesitant plan.

The subjects are distributed along the diagonal which entails that extremes can be categorized as two main opposite behaviors on a Dominant ↔ Hesitant axis: Dominant = {1, 0} = {**Taysa**, **Fernanda**, Judith, Aleyda} while Hesitant = {0, 1} = {**Mariano**, **Priscilla**, Hector, Julia}. The third group is Balanced = {0.5, 0.5} = {**Julieta**, **Rodrigo**, **Fabiola**, **Francisco**}.

The predicates that occur in the corpus can be divided into two subclasses:
1) *Operations*: are generic GUI actions {Click, Delete, Move, Undo …} but also domain dependant actions {Listen, Record, Play …}.
2) *Mental verbs*: are always generic {Meaning, Available, Possibility, Significance, Obligation, Effect …}.
All mental verbs are generic and most operations are also domain independent, amounting to a total of 90% while domain dependant predicates amount only to 10%.

Concerning the references, except for the references grouped into the class GUI and some operators (12%), all the references are domain dependant (88%).

4. Discussion and further work

In stage 1 we had to deal with the two questions: Do users actually need help? If so, do they actually call the agent? The experiment showed that all users needed help even for very basic tasks: this is encouraging if we want to integrate learning functions in a non-intrusive assisting agent. However there is a large interpersonal difference between the subjects in the degree of interaction (min 4 – max 17): this means that not all the users will have the same access to the learning functions; hopefully, because we can easily detect non-loquacious people, we can provide them with a specific learning policy.

In stage 2, since the experiment is simple, the collected corpus made easy the transcription and the categorization of requests' significant elements. The limited data does not enable a full statistical approach of the categorization of the subject's behaviors in help-seeking conditions, but if the distribution of the speech acts can't be used directly, it is possible to exhibit some behaviors like Dominant/Hesitant that are related to the usage of speech acts subclasses. This is a first step towards the characterization of the users with the aim of integrating adapted learning policies in the agent's rational module. Finally, the requests reveal few domain dependant predicates (10%) and a large amount of domain-dependant references (90%): this suggests that we will have to mainly rely on nominal groups to link the request with the learning content. Nonetheless the generic predicates can be useful to control the learning content in reaction to Meaning, Effect, Significance, and Available Etc.

The inherent limitation of this first experiment on the CODIVA CSCP environment entails that further work has to proceed with the integration of basic learning functions into the assisting agent in order to evaluate their feasibility and the obtained efficiency, as stated in stage 3. We are planning to report this in future papers.

5. References

Amobi F., (2007) Making It Real: Promoting Novice Teachers' Critical Perspectives on Educational Issues through Structured Debates, Journal of Teaching and learning, Vol. (4)2

Bodker S., Gronbek K. (1991) Cooperative prototyping: users and designers in mutual activity, International Journal of Man-Machine Studies, Vol. 34(3), pp 453-478

Brown J. S., Duguid P. (1991) Organizational Learning and Communities-of-Practice: Toward a Unified View of Working, Learning, and Innovation, Organization Science, Vol. 2(1), pp. 40-57

Capobianco A., Carbonell N. (2001) Contextual online help: elicitation of human experts' strategies, Proceedings of HCI'01, New Orleans, pp. 824-828

Carroll J.M., Rosson M.B. (1987) Paradox of the active user, Interfacing thought: cognitive aspects of human-computer interaction, MIT Press, pp. 80-111

Cassell J., Sullivan J., Prevost S., Churchill E. (Eds.) (2000) Embodied Conversational Agents, MIT Press

Davis F., (1989) Perceived Usefulness, Perceived Ease of Use, and User Acceptance of Information Technology, MIS Quarterly, Management Information Systems Research Center, University of Minnesota, Vol. 13(13), pp 319-340

Dillenbourg P. et al. (1995) The evolution of research on collaborative learning, in Learning in humans and machines: towards an interdisciplinary learning science, Ed. S. P. Reiman, Elsevier, p. 189-211, Oxford, UK

Godbole-Chaudhuri P., Winters F. I., Azevedo R., Hofman N., (2006) Help-seeking behavior and learning with hypermedia, Proceedings of the 7th International Conference on Learning Sciences, Bloomington, Indiana, pp 928-929

Harnish R. M. (1994) Mood, meaning and speech acts. In Foundations of Speech Act Theory: Philosophical and Linguistic Perspectives, ed. Savas L. Tsohatzidis, Routledge, London, pp 407–459

Iazzetta F., Kon K. (1998) Internet music: Dream or (virtual) reality. In: Proc. of Brazilian Symposium on Computer Music, pp 69–81, Belo Horizonte, Brasil.

Lester J.C., Converse S.A., Kahler S.E., Barlow S.T., Stone B.A., Bhogal R.S. (1997) The persona effect: affective impact of animated pedagogical agents, Proceedings of the SIGCHI conference on Human factors in computing systems, ACM, pp 359-366

Luckin R., Koedinger K. R., Greer J. E. (2007) Artificial intelligence in education: building technology rich learning contexts that work, Frontiers in Artificial Intelligence and Appl., Vol. 158, IOS Press

Manara Miletto E., Sansonnet J. P., Soares Pimenta M., Bouchet F. (2009) Corpus-based design of a Web 2.0 Assisting Agent, 8th International Workshop on Web-Oriented Software Technologies, IWWOST 2009, San Sebastian, Spain

Nett B. (2008) A Community of Practice among tutors enabling student participation in a seminar preparation, International Journal of Computer-Supported Collaborative Learning, Springer, Vol. (3)1, pp 53-66

Nitzke J. A., Carneiro M. L. F., Franco S. R. K. (2002) Are computer supported learning environments really learning environments?, The 11th WWW conference, Honolulu

Nussbaum E. M., Winsor D. L., Aqui Y. M., Poliquin A. M. (2007) Putting the pieces together: Online argumentation vee diagrams enhance thinking during discussions, International Journal of Computer-Supported Collaborative Learning, Springer, Vol. (2) 4, pp 479-500

Searle J. R. (1969) Speech Acts: An Essay in the Philosophy of Language, Cambridge University Press

Selwyn N. (2007) The use of computer technology in university teaching and learning: a critical perspective, Journal of Computer Assisted Learning, Vol. 23 (2), pp 83-94, Blackwell Publ. Ltd.

Slavin R. E. (1989) Research on cooperative learning: An international perspective, Scandinavian Journal of Educational Research, Vol. 33(4), pp 231-243

Stahl G., Koschmann T., Suthers D. (2006) CSCL: An Historical Perspective, in Cambridge Handbook of the Learning Sciences, R. K. Sawyer (Ed.), Cambridge, UK

Susi T., Johannesson M., Backlund P. (2007) Serious Games - An Overview, Technical Report HS- IKI -TR-07-001, School of Humanities and Informatics

Totten S., Sills T., Digby A., Russ P. (1991) Cooperative learning: A guide to research, Garland, N.Y

Ummelen N., Neutelings R. (2000) Measuring reading behavior in policy documents: a comparison of two instruments, IEEE Trans. on Prof. Comm., vol. 43, pp. 292-301

Weinberg G. (2002) The aesthetics, history, and future challenges of interconnected music networks. In Int. Computer Music Conference, Goteborg, Sweden

Wenger E. (1998) Communities of practice Learning, meaning, and identity, Cambridge Un. Press.

CODES url: http://gia.inf.ufrgs.br/lcmCODES

DIVA url: http://www.limsi.fr/~jps/online/diva/divahome

CODIVA url: http://gia.inf.ufrgs.br/lcmCODES/diva/music/music.main.htm