Corpus-based design of a Web 2.0 Assisting Agent

Evandro Manara Miletto[1], Jean-Paul Sansonnet[2],
Marcelo Soares Pimenta[1], François Bouchet[2]

[1]Instituto de Informatica – UFRGS, Porto Alegre, RS, Brasil,
[2]LIMSI-CNRS, BP 133, 91403 Orsay cedex, France
{miletto@inf.ufrgs.br, jps@limsi.fr, mpimenta@inf.ufrgs.br, bouchet@limsi.fr}

Abstract. We present an approach to facilitate the design of Assisting Conversational Agents dedicated to the Function of Assistance to ordinary people interacting in Natural Language with assisting agents on Web 2.0 pages. For each new assisted RIA, an experiment is carried out to collect a specific corpus of textual questions. It is then analyzed to exhibit the specific linguistic entities of the application required by the generic skeleton of the agent. Using a real large-scale example of a cooperative music prototyping application, we study the feasibility and evaluate the cost-effectiveness of this approach.

Keywords: RIA Assisting Agents, Corpus-based design, Music prototyping.

1 Introduction

1.1 Context: the Need for Assistance in New Rich Internet Applications

Day by day, the Web 2.0 based on Rich Internet Applications (RIA), is growing as an environment for communication, providing interaction facilities that support the growth of virtual communities. Thus, the Web has become a rich and ideal way to study the new so-called Distributed Collective Practices (DCP) [1]. In this context, we have recently been involved in the development of the Web 2.0 platform **CODES** (**CO**operative Music Prototype **DES**ign) dedicated to cooperative music prototyping, and freely available at [2]. It aims at allowing *novice* users (i.e. with no particular music knowledge) to experiment with music samples and interact with each other on the Web, in order to create simple music pieces; herein called *music prototypes.*

Along with the rapid evolution of the Internet, the population of computer applications has evolved from the relatively small number of expert or corporate users to the huge amount of general public users who are mainly novices. This phenomenon is accentuated in the Distributed Collective Practices and it has been observed in the CODES environment. This is the reason why, we think the Function of Assistance, dedicated to novice users, is a key issue of the future Web 2.0.

Assisting users of computer applications has been a challenging task since personal computers have started to be available to a broader audience of non-expert users at the beginning of the 1980s. As a mere transposition of a paper-based documentation into an electronic version has been shown to be not very efficient [3], research has mainly been focusing on the notion of adaptation: a) to bridge the semantic gap between novice users and expert developers – this has been undertaken Contextual Help

Systems (CHS) [4] ; and b) depending on the users' profiles: either through the use of static classes of users [5], or with a regularly updated dynamic model of the user [6].

From a strictly computational point of view, the main qualities of a help system are the *precision* and the *completeness* of its informational content. But as far as general public users are concerned, it has been shown that *ease of use* is by far the primary factor (otherwise the help system is merely left aside) since according to the 'motivational paradox' [7], users prefer to ask from expert friends 'behind their shoulder' [8]. In relation to this statement, recent studies have shown the positive impact of multimodality for help systems, and particularly the linguistic modality [9] (*i.e.* Natural Language interaction). One of the consequences has been the development of Assisting Conversational Agents (ACA), software tools typically taking as input textual questions from the users and capable of reasoning over the dynamic model of the application to provide pertinent answers.

1.2 Key Issue: a Corpus-Based Approach to the Function of Assistance

A typical ACA is built upon a Natural Language processing chain (NLP-chain). There are two main approaches to the design of dialogue-oriented NLP-chains:

a) Human/machine dialogue stand-alone systems, like Allen's TRAINS [10], are very complex to build but work well, especially with trained people in corporate environments. Their main drawback, stated by Allen as the "genericity problem" [11], is that they are difficult to reuse or adapt cost-effectively to new applications;

b) Web Chatbots are long time successors of ELIZA [12], like today the web-based chatbots Hal [13], Jabberwacky [14] or ALICE [15], to consider only the latest winner of the Loebner Prize given each year to the most convincing ones. Chatbots rely on trivial NLP-chains based on simple word-matching techniques. A comparative study [16] has shown they are more adapted to social/game chatting than to task-oriented dialogue. However, they are very easy to develop, customizable for new applications and quite light to deploy.

In the context of Web 2.0 applications and services, where the cost of the assisting agent has to be small compared to the cost of the application itself, the three advantages mentioned above appear crucial. This is the reason why we have favored this approach to develop a generic assisting agent *i.e.* an agent that is supposed: a) to be easily 'pluggable' into new Web 2.0 applications; and b) to provide a NLP-chain *skeleton* quick and easy to adapt to the new application.

Therefore, the key issue to the filling of the NLP-chain skeleton is the elicitation of the specific linguistic entities that appear to be actually occurring in the users' textual requests. For this purpose, we have been relying on a corpus collected on a group of novice users carrying out CODES basic tasks during an experimental session.

In Section 2.1, we present the architecture of our ACA and its NLP-chain skeleton. The rest of the paper is then dedicated to the experiment carried out with a large-scale Web 2.0 application (the CODES framework) in order to assess a) the integration of the agent into CODES; b) the feasibility of the corpus-based phase of adaptation; and c) the global cost-effectiveness.

2 Methodology

2.1 Architecture of the DIVA NLP-chain

In order to study the Function of Assistance in the context of RIA, we have developed an experimental toolkit called DIVA, freely available for research and education purposes at [17]. 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 completely written in JavaScript for the support of a) the virtual characters that personify the assisting agent, b) the NLP-chain that analyzes and resolves the users' questions and c) the AJAX link to the server for access to resources and client information storage. The general architecture of a DIVA assisting agent is given in figure 1.

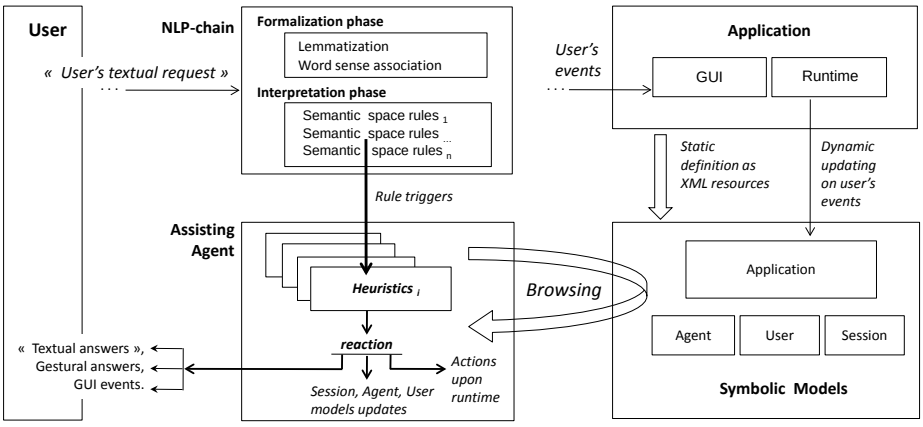

Fig. 1. General architecture of a DIVA Assisting Agent.

Following the guidelines stated in section 1.2, the NLP-chain of the DIVA toolkit is based on a typical chatbot approach but with a much more sophisticated structure as shown in figure 1 (middle–top):

1) The formalization phase: based on two sets of filtering rules applied sequentially:

- *Syntactical level:* first a typical chunking phase is applied, then words' flexions are transformed into their corresponding lemmas (root words);
- *Word-sense association level:* lemmas are then transformed into semantic so-called synsets, as done in WordNet [18]

At the end of the formalization phase, the request is transformed into an intermediate formal form, called the Formal Request Form (FRF). In the FRF language, a request is expressed by a sequence of abstract keywords, each of them being associated to a semantic concept defined by a textual gloss.

2) The interpretation phase: based on a set of rules of the form *pattern* → *reaction*, where the *pattern* is expressed in FRF and the *reaction* is a procedural heuristic defining the behavior of the agent in response to the user's request. To build a reaction, the triggered heuristic uses two kinds of information: a) a representation of the current dialogical session, and b) a symbolic model of the application describing its specific features. The set of interpreting rules is organized into so-called *semantic spaces* dedicated to specific domains.

2.2 Application to the CODES Framework

Given the proposed structure of the DIVA NLP-chain, the key point is now to evaluate its quick and easy adaptation to existing Web applications and services. This task is carried out in two steps: a) exhibiting the linguistic entities expressed and/or referred to by the users; and b) writing the formalization and interpretation rules according to these entities. In a first stage, several small Web pages (also available on the DIVA homepage) were developed to test this approach, with success in terms of cost-effectiveness. However, this work had to be improved on two main points:
1) By evaluating the procedure to a real large-size application (large-scale problem);
2) By defining a more systematic procedure to exhibit the application's entities.

The CODES framework is a large-size Web 2.0 application dedicated to collaborative music prototyping on the Internet involving general public users, and hence, a good candidate for an experiment involving a large-scale problem. This experiment was carried out in two main steps:
- The first step was to integrate the DIVA toolkit into the CODES framework (this was done quite easily, in less than a week);
- The second step was to define an experimental protocol: a) to collect a corpus of assistance in the CODES environment, and b) to exhibit the linguistic entities occurring in the corpus (which was more time-consuming – see table 4).

The experimental protocol is described in the next section and the systematic procedure to exhibit the application's entities is detailed with its results in section 3.

2.3 Experimental Protocol

Objective. 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. This corpus is meant to be manually analyzed and annotated to exhibit the entities required to build rules of the ACA (cf. section 3).

Subjects. Experiment sessions involved 12 subjects (4 male / 8 female) between 23 and 35 years old, most of them being MSc and PhD students (from very diverse domains: architecture, law, cooking, acrobatic dance, urbanism, history, psychology and physics). To keep the homogeneity of the sample group and hence of the collected corpus, no subject had any background knowledge neither in computer science (beyond surfing the web) nor in music (theory or practice) and can thus be considered as novices in both domains.

Conditions. In this experiment, all subjects were told to 'think aloud' [19], that is to express orally the questions they could have while performing the tasks as if there was an expert 'friend behind their shoulder' [8] 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 questions. Interactions were recorded since even when being briefed about the think aloud protocol, subjects often uttered generic or vague questions (using anaphora and coverbal deictic gestures) such as: "why does **it** not work?", "should I put **it** there?" (cf. figure 2a) – which could be a problem during the subsequent analysis phase. So watching back and analyzing the recorded videos has allowed us to resolve many issues and to identify the objects referred to in the requests. After the experiment, the subjects had to fill a questionnaire.

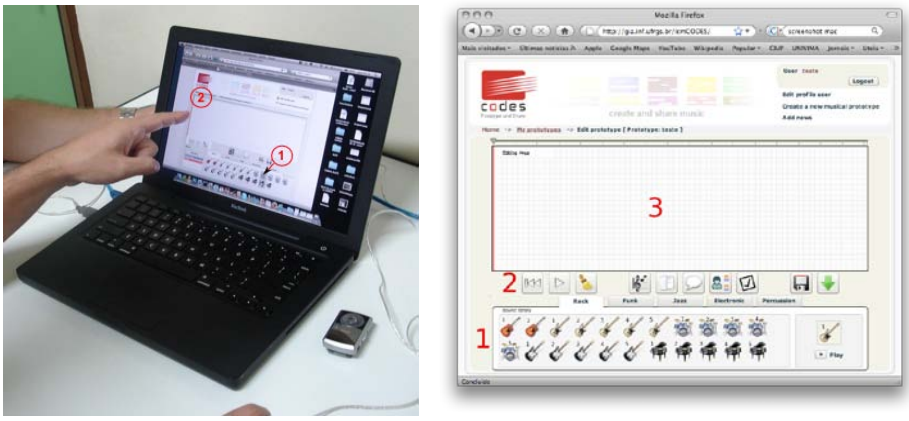

Fig. 2. A photo taken during a recorded session. a) on left, a subject is asking: « Should I put it there? » while dragging a selected pattern icon (1) = 'it' and pointing his finger on the editing area (2) = 'there' ; b) on right, the window of the CODES Musical Prototype Editing Level.

Tasks Definition. 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) which are:

a) in sequential order: one need to finish T_i in order to perform T_{i+1};
b) with 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.

Task	Task *summary* and objective
T1	*"Choose a) a preferred musical style and b) three preferred sound patterns in that musical style."* The goal of this task is merely to identify the sound library area (situated at (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."* The goal of this task is to add the sound patterns in the editing area to create musical sequences, in order to check if the users are able to drag icons from the sound library and to drop them onto the editing area (see (3) in figure 2b). Figure 3a illustrates with a red arrow the action to be achieved.
T3	*"Listen to the sequence you have created."* The goal of this task is to check if the users identify and associate the execution control buttons with the editing area. When clicked on, the "Play button" (see (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 3a).
T4	*"Delete one sound pattern you didn't like, changing for another one."* The goal of this task is to check 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 3b) 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." The goal of this task is to check 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 the sound patterns under or above each other and listen to the result of that combination.

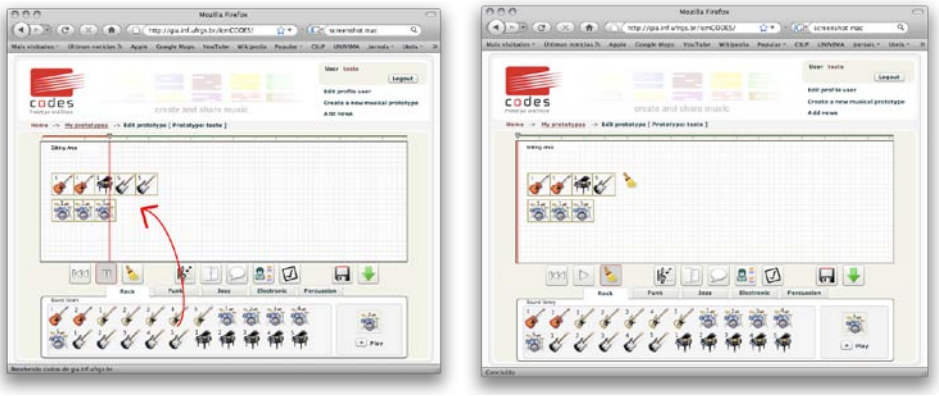

Fig. 3. a) on left, Task T3: creating and then listening to a musical sequence; b) on right, Task T4: excluding a sound from the editing area with the broom tool.

3. Results

3.1 Corpus Data

During this experiment, all subjects asked for assistance from the CODES expert. Hence, they were given the best possible help information and consequently all subjects achieved successfully all the tasks. It resulted in the constitution of a corpus of 115 Natural Language requests, acquired over a month. They have been collected in Portuguese, Spanish and English and all translated into English for further processing. Table 2 displays an excerpt of 30 oral utterances, transcribed off-session from the audio data. Using the recorded sessions, utterances have been associated with additional contextual information, thus enabling the analyst to resolve anaphora (e.g. 'it' pronoun — lines 1, 5, 6, 9…) and indexical items (e.g. 'this'— line 15, 18). Even if few utterances aren't assistance requests about the structure or the functioning of the application (*e.g.* line 20 expresses a subject preference), most actually are.

Table 2. An excerpt of 30 questions extracted from the CODES corpus.

1	Why doesn't it sound?
2	If they are not well aligned the sound will play?
3	Can I use the same sound pattern?
4	Can I repeat?
5	How can I stop it?
6	How should I listen to it?
7	How can I take off the broom?
8	How can I come back?

9	Doesn't it function?
10	Are the sound patterns the instruments?
11	What do you mean with 3 sound patterns?
12	Should I choose only 3?
13	To play together should I put them behind one another?
14	Where is the login area?
15	Is the editing are this blank area?
16	I don't know where is the editing area?
17	Are the musical styles: rock, funk, jazz..?
18	Is it this way?
19	3 instruments?
20	I like pianos!
21	How can I undo the broom?
22	Why it changes?
23	Are they the sound or the instruments?
24	Is it necessary to click on play to listen?
25	Should I record before?
26	Where can I play?
27	How can I play the whole sequence?
28	Can I put them back?
29	How do I know the sound patterns?
30	I don’t understand why when I click in the #1 and the #5 appears?

3.2 Quantitative Analysis of the Assistance Turns

We have observed that assistance was often provided by answering a single question, the CODES expert’s answer being enough for the subject to be able to continue the task at hand. So each assistance turn corresponds to an independent pair question/answer and not to a real dialog (in the worst case, there sometimes was a ‘re-phrasing dialog’ where some questions were asked twice or three times with some linguistic variations).

Subject Loquacity. The general data from the assistance turns related to the five tasks is represented in figure 4, with subjects sorted from the most loquacious (i.e. help-seeking) to the least loquacious. We can easily note that for each task, in the group of subjects, there is a large inter-subjects variation. It is also shown in figure 5a, with a range factor of 4 between Julia and Julieta. This may be proportional to the familiarity with others Web-based applications, not necessarily related to music.

The Novice Effect. In figure 5b, we can see that whereas the five tasks were defined with *increasing* complexity, the total amount of turns indicates that while the subjects progress from task 1 to task 5 they tend to require *less* assistance. This is emphasized by the turn count of task 1 exhibiting a peak of assistance turns. If we analyze the interrelationships among the tasks, we can suppose that some questions related to one action performed, in relation to a given task, were not repeated when a similar action was performed in another task execution. This clearly shows a so-called ‘novice effect’ where assistance is mostly needed when the users enter the application, and tend to decrease even when the users are involved in more complex tasks.

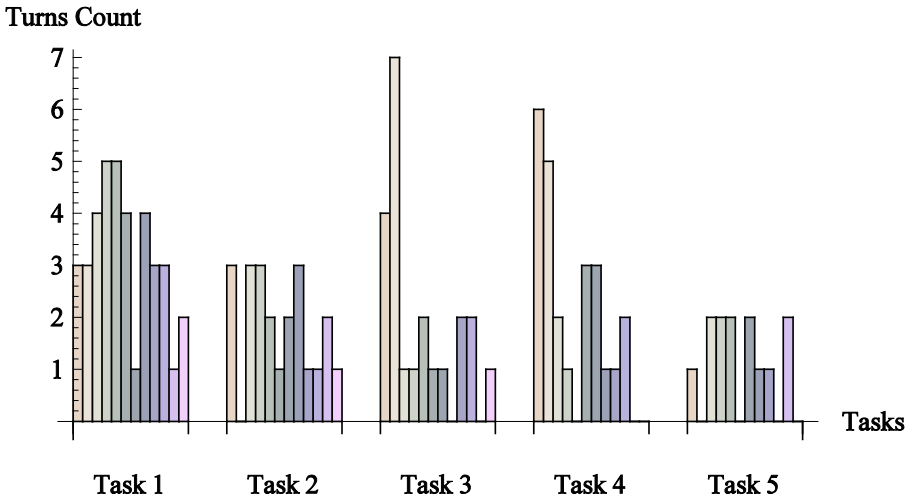

Fig. 4. Overview of turn counts per task and subject (a bar = a subject). Subjects are sorted by their global loquacity (from maximum to minimum).

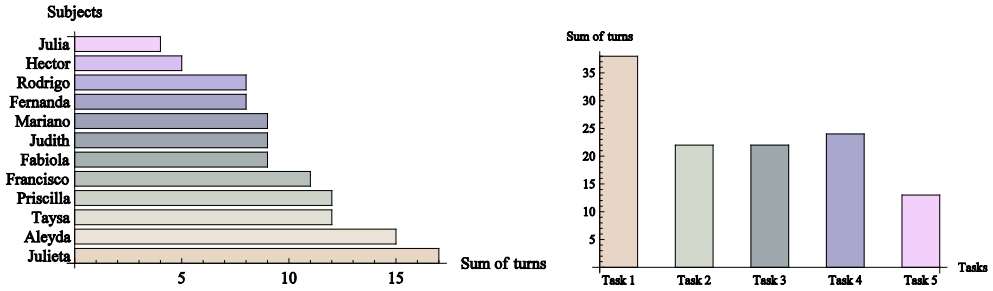

Fig. 5. a) on left, loquacity of the subjects: sum of turn counts on all tasks for each subject; b) on right, need for assistance of the tasks: sum of turn counts over all subjects for each task.

3.3 Exhibiting the Entities in the Utterances

Once collected, the corpus was transcribed manually into a formal form. Table 3 presents some examples of transcriptions of users' questions into formal requests. Note that deictic items (e.g. 'it', 'there' while pointing with the finger on screen) or session anaphoric items (e.g. 'come back' which requires registering the previous states) were registered by the WoZ operator so that they could be filled here within brackets (left column in table 3), making it possible for the transcription to be accurately completed.

Table 3. Examples of manual transcriptions, from natural to 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(Put(SP, EA))
How can I take off the broom?	HOWTO(Quit(Broom))
How can I deactivate the broom?	HOWTO(Quit(Broom))
How can I play the whole sequence?	HOWTO(Play(MP))
How can I come back? [previous = EA]	HOWTO(Comeback(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))
Are the musical styles: rock, funk, jazz..?	CHECK(Listof(MUSIC))
Etc.	…

We were able to exhibit three main types of entities in the subjects' utterances, 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.	5 classes
Predicates	They are mainly linked to action verbs identifying the operations in the CODES GUI.	28 classes
References	They are the elements categorized by the users and targeted through the referential expressions.	10 classes

Speech Acts. For simplicity, we limited the speech acts to five subclasses[1]. Their definition, limited here to a shortened *gloss* and one example, is as follows:

Speech Act	**Definition and example**	**% corpus**
HOWTO	The user asks how to achieve some function e.g. *"How can I listen to the sound pattern?"*	21 %
ASK	The user asks the value of an attribute e.g. *"what do you mean with 3 sound patterns?"*	20 %
CHECK	The user checks for the confirmation of a proposition e.g. *"This does not correspond to this?"*	18 %
SHOULDI	The user asks for a suggestion so as to proceed e.g. *"Should I record before?"*	14 %
CANI	The user ask for the possibility to perform an action e.g. *"Can I put them in any level I want?"*	11 %

Predicates. They were manually exhibited and classed into semantic classes, like in Wordnet synsets [18]. Each of the 28 semantic classes is associated with a symbol and a gloss, shortened in the right part of the following excerpt:

[1] Note that the Austin-Searle notion [20] is used here in a metaphorical way: this is because the linguistic domain is drastically restricted to the Function of Assistance, where not all speech acts are occurring; on the other hand, we were able to make a fine grained distinction between the subclasses of help questions issued by the subjects.

Predicate	Gloss
Play	Action of playing a sample of music.
Listen	Action of listening to a sample of music.
Put	Action of moving an item into a place (e.g. by drag & drop actions).
Delete	Action of deleting an item (often represented by an icon).
Meaning	The sense/purpose attached to a GUI object, a function name etc.
Possibility	Logical predicate indicating that its argument can be activated.
Etc.	…

The distribution of the most occurring predicates is given in the figure 6.

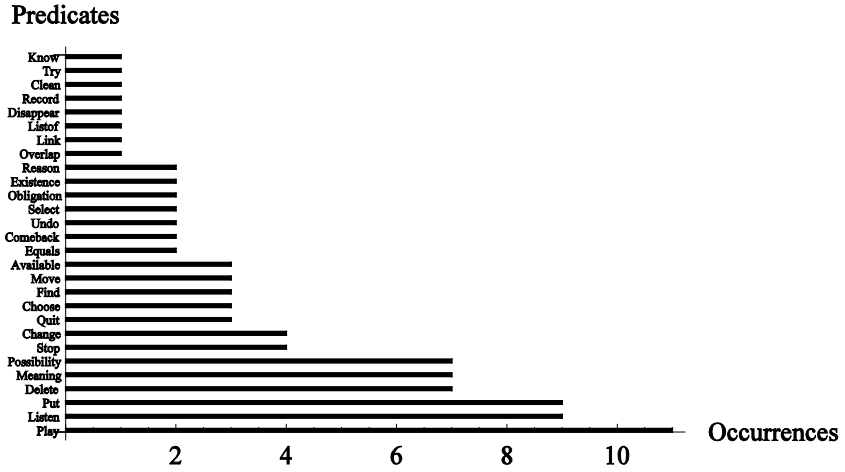

Fig. 6. Distribution of the most occurring predicates over the corpus.

References. Referential objects are 'things' that the users categorize 'in their mind' and refer to in their linguistic expressions. They can be actual GUI objects (buttons, browser, window…), screen areas and panels, icons representing application items like the musical prototypes (MP) or the sound patterns (SP); other significant elements referenced in the corpus are the "broom" button, numbers, and application-specific concepts like musical-styles or musical-instruments (for simplicity, they are all collected in a single class called MUSIC). Their notation, definition and corpus distribution is given in the following list:

Reference	Description	% corpus
SP	sound pattern	59 %
EA	editing area	33 %
SL	sound library	16 %
GUI	GUI objects = keyboard, button …	10 %
MUSIC	music-related objects = styles, instruments …	9 %
Broom	button used to delete de sound patterns	8 %
SPL	sound pattern library	5 %
NUM	any integer number	4 %
MP	musical prototype	4 %
SPEA	sound pattern editing area	2 %

3.4 Discussion about Cost-Effectiveness

As stated before, this experiment was done with two objectives in mind: a) to demonstrate that the structure of the resource definition files, associated with a particular application, could be revealed from the analysis of the data; b) that the task of analyzing and programming the resources files would be cost-effective. Section 3.3 has attempted to bring an answer to the first point so we can now add some qualitative remarks about the cost-effectiveness.

Table 4. Time taken by the steps of the experimentation.

Step	Task description	Time (in days)
1	Software integration of DIVA into CODES	5
2	Definition of the tasks (discussions, pre-experiment and description)	6
3	Selection of the subjects and experimental sessions	**11**
4	Manual transcriptions of experimental sessions into the corpus	2
5	Manual annotation of the corpus and exhibition of the entities (stats)	3
6[2]	Programming of the NLP-chain rules for the spaces	5 (approx.)

As shown in table 4, the total experiment was achieved in about a month time. Globally, even if not completely cost-effective, this can be considered encouraging compared to the development duration of classical dialogue systems which often amounts in months, even in years. One can see that the part dedicated to the experiment with the subjects took more than half the time, even with a small number of subjects. This is certainly the weak point of this approach which relies on ordinary people availability and WoZ techniques. In the future, an effort should be made to adapt this phase with automated Web-based procedures, using virtual characters instead of WoZ operators. This will allow to reach a large population of potential users and to automate the transcription phase. However, it will then be difficult to keep performing manually the annotation and exhibition steps, thus leading to a tradeoff between the loss of accuracy and the larger coverage of the linguistic phenomena.

4 Conclusion

We have presented an approach to facilitate the design of Assisting Conversational Agents dedicated to the Function of Assistance to ordinary people interacting in Natural Language with assisting agents on Web 2.0 pages. In this particular context, the key issue is the feasibility and the cost effectiveness of the adaptation of the assisting agent to new assisted Rich Internet Applications. Our approach is based on a) the collection of a specific corpus of textual requests with novice subjects asking help while performing predefined tasks; and b) the elicitation of the specific linguistic entities through an analysis of the collected data.

[2] Actually, step 6 which is related to the integration of the exhibited entities within the agent rules is not detailed in this paper which focuses on the linguistic part.

In this paper, we have described an example of such an experiment with a real, large-scale, typical Web 2.0 application which shows that it is feasible and relatively cost effective to carry out the proposed procedure. The concluding discussion points some weak points among them the extent of the WoZ experiments and of the manual analysis: this will prompt future effort on the automation of this part of the procedure.

References

1 Turner W. et al., 2006, Information Infrastructures for Distributed Collective Practices, Special Issue of Computer Supported Cooperative Work: the Journal of Collaborative Computing, Vol. 15, Nos. 2-3.
2 CODES, http://gia.inf.ufrgs.br/CODES2/
3 Capobianco A., 2003, questioning the effectiveness of contextual online help: some alternative propositions, Human-Computer Interaction INTERACT'03, M. Rauterberg, M. Menozzi and J. Wesson, eds., IOS Press, 2003, pp. 65-72.
4 Carenini G., and Moore J.D., 1993, Generating explanations in context. Proceedings of the ACM Conf.on Intelligent User Interfaces, Orlando (FL), ACM Press, pp. 175-182.
5 Shneiderman B., 1992, Designing the user interface (2nd ed.): strategies for effective human-computer interaction, Addison-Wesley Longman Publishing Co., Inc.
6 Jameson A., 2003, Adaptive interfaces and agents, The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications, Lawrence Erlbaum Associates, Inc. pp. 305-330
7 Carroll J. M., Rosson M. B., 1987, The paradox of the active user. In J.M. Carroll (Ed.), Interfacing Thought: Cognitive Aspects of HCI. Cambridge, Mass: MIT Press (pp. 80-111).
8 Capobianco A., Carbonell N., 2001, Contextual online help: elicitation of human experts' strategies, Proceedings of HCI'01, New Orleans, pp. 824-828
9 Carbonell N., 2003, Towards the design of usable multimodal interaction languages," Universal Access in the Information Society, vol. 2, Jun. 2003, pp. 143-159.
10 Allen J.F. et al., 1995, The TRAINS project: a case study in building a conversational planning agent. Journal of Experimental & Theoretical Artificial Intelligence, 7(1), 7-48.
11 Allen J.F. et al., 2001, Towards Conversational Human-Computer Interaction. AI Magazine, 22(4), 27-37.
12 Weizenbaum J., 1966, ELIZA, a computer program for the study of natural language communication between man and machine. Communications of the ACM, Volume 9 # 1.
13 Hal, http://zabaware.com/
14 Jabberwacky, http://www.jabberwacky.com/
15 ALICEBOT, by R. Wallace, http://alicebot.blogspot.com/
16 Wollermann C., 2006, Chatbots a comparative study, Proc.of the Young Researchers' Roundtable on Spoken Dialogue Systems, 75-76. Pittsburgh, PA.
17 DIVA toolkit home page, http://www.limsi.fr/Individu/jps/online/diva/divahome/index.html
18 Fellbaum C. Ed., 1998, WordNet: An Electronic Lexical Database, The MIT Press, Cambridge MA.
19 Ummelen N., Neutelings R., 2000, Measuring reading behavior in policy documents: a comparison of two instruments, IEEE Transactions on Professional Communication, vol. 43, pp. 292-301.
20 Searle J. R., 1969, Speech Acts: An Essay in the Philosophy of Language, Cambridge University Press