Requirements Elicitation Driven by Interviews: The Use of Viewpoints

Julio Cesar Sampaio do Prado Leite*
Departamento de Informática PUC- Rio
R. Marquês de São Vincente 225
22453-900 - Rio de Janeiro, Brasil
e-mail: julio@inf.puc-rio.br

Ana Paula Pinho Gilvaz
Johnson Wax
Av. Comandante Guaranys 599
22275-610 - Rio de Janeiro, Brasil

Abstract

Requirements elicitation in the context of organizational information systems is well know to be a very hard task, much dependent on the experience and cleverness of the team performing the elicitation. In such a context the use of interviews is frequent and pointed out as the major technique for getting the requirements from the actors in the organization. We have been working with the idea of a general interview assistant and our first results are promising. In this article we elaborate on our original proposal in order to augment its assistant capability, without loosing its simplicity. We show how the use of viewpoint analysis improves the inference capability of our assistant.
Key-words: Interview, requirements elicitation, conceptual model, intelligent assistance.

1. Introduction

Our work is aimed at supporting the software engineer (systems analyst) in eliciting information for corporate information systems. We used well established IS techniques to build a prototype CASE tool called FAES [4]. FAES was designed to support an interview process based on a general framework of questions. Using a conceptual model and some analysis heuristics we managed to provide to the software engineer an automated support for his work of finding out important information in a given information system. As such, the work we will describe here is focused on a particular instance of computer automated support, namely the elicitation of information by interviews. FAES was built with the purpose of supporting and evaluating our strategy. Preliminary data supports our hypothesis that in using FAES there is an increase in productivity during interviews [4].

During FAES's presentation at Case'95 [4], several questions from the audience encouraged us to rework some of its original architecture, making it more flexible and more powerful, but still maintaining its simplicity. FAES uses a simple conceptual model. Its shallowness is a positive factor in the tool performance, not only because it requires less computing power, but also because its structure is well understand by its users.

The literature [10] [8] has been pointing out the need of intelligent assistance to support upstream activities. The gap from informal to formal is not well addressed by the existing CASE technology. Although some, like [10] [9], believe that it is necessary to use deep representation strategies to bridge this gap, thus relying heavily on previous encoded domain knowledge, we firmly believe that it is possible to provide assistance, and thus decreasing the gap, by using simpler models at least at the stage of "reconnoitering the requirements" [2]. A previous work on viewpoints [7] does also use this approach, that is exploring the very first step in understanding a future software system.

In this article we will address two "problems" observed in FAES. The first one is related to the lack of flexibility in questioning. The second one is related to the lack of more powerful heuristics. Regarding the

* This work was supported by CNPq

flexibility aspect we re-designed the flow of questions, making it possible to interleave the fixed questions of the original FAES with questions on demand. With respect to more powerful heuristics we decided to fully explore the FAES capability of looking into previous interviews, by reusing our experience and heuristics described in Leite and Freeman's viewpoint work [7].

Section 2 provides a general description of the interview process and of the original FAES. Section 3 describes the new questioning schema. Section 4 details the viewpoint approach proposed for the new FAES. We conclude by pointing out how the improvements will impact on the automation support for interviews and how our work relates to other work in the area.

2. FAES

This section summarizes the original FAES work [4] and uses parts of that paper to explain the general context. FAES is the central part of an interview process that covers three basic interview questions: What to ask? How to ask? Whom to ask? The process has automation support for the first two questions and relies on general guidelines to the third question. Following the process, we build a knowledge base organized according to a conceptual model and analyzed according to special heuristics. The conceptual model was built upon three well-know information system techniques: BSP (Business System Planning) [6], CSF (Critical Success Factors) [11] and E/M (End Means Analysis) [12] and follows the integration model proposed by [13]. FAES knowledge base is an important factor for providing an organized model of corporate information, and its automation strategy supports the boring clerical tasks associated with interviews.

2. 1 The Process

The interviews are conducted individually with each person found to be important to interview [4]. The software engineer asks the questions suggested by FAES and annotates the answers trying to be as factual as possible given the respondents answer and trying to be as clear as possible. The software engineer can also comment on the answers he is annotating. It is important to note that the tool offers two feedback mechanisms: one at the time the question is being annotated and the other as the interview ends. At the end of the interview a report is generated which mirrors the knowledge base and provides some diagnoses of the captured information.

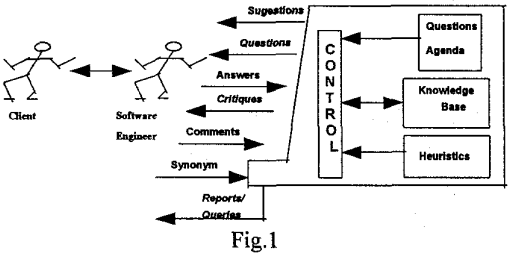


Fig.1

The interview assistant (Figure 1) applies a basic set of questions that would fill in the conceptual model. The interview assistant has a set of heuristics based on the conceptual model and on general common sense. These heuristics have been written to validate the answers, verify the existence of relationships between the answers and discover the need for more answers. FAES has four basic components: control, questions, knowledge base and heuristics. Control deals with the interface, the order of questions and heuristic's application. Heuristics are activated by a particular question or by the end of the interview. The questions are based on the conceptual model and contain the information necessary to instantiate the model. The knowledge base stores the answers, the diagnoses and the entries made by the software engineer (observations and synonyms).

2. 2 The Conceptual Model

The conceptual model is based on Wetherbe's work on executive information requirements [13]. According to Wetherbe, a common mistake made in determining information requirements is to ask the wrong question: "What information do you need from the new system ? " . Although this is the obvious question, it is not all helpful to clients attempting to determine what information they need. In order to minimize this problem, Wetherbe proposes an approach to interviewing that uses indirect questions. The interview scheme is composed of types of questions from three methods/techniques defined mentioned before: BSP, CSF and E/M analysis. Figure 2 shows the conceptual model we developed based on Wetherbe's approach. The conceptual model links the different types of questions and includes a lot of new information found necessary to support the interview. The model nodes represent the information to be

defined and the arcs the relationships among the information. The nodes serve as basis for the questions posed by the assistant. The arcs provide relationships that will help the analysis of the answers. The model has passed through several versions, as we tested its instantiation with a couple
of case studies.

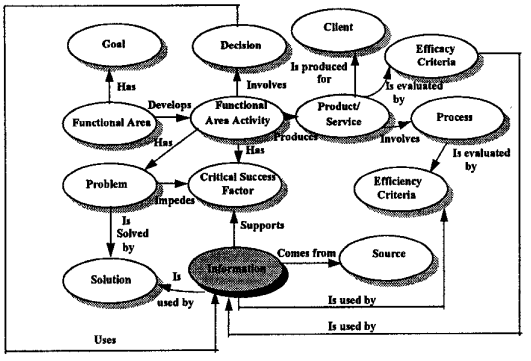


Fig. 2

2.3 Automation Strategy

In order to fill in the model with information, FAES uses 22 instantiation's questions. These questions have a fixed and a variable part. The fixed part is determined by the model and the variable part is used to establish the chain of questions. Below we list some of the instantiations questions.

4. What are the best solutions for < problem > ?
5. What information does support < solution > ?
6. Who does provide < information > ?
7. What are the decisions related to < activity > ?
8. What information does support < decision > ?

The questions that are used to fill in the model use the concept of information chaining, that is each question is composed of a fixed part and a variable part (< >). The variable part is an answer given to another question already answered, thus making a chaining process, since each of the answers of a given question will produce a different question for one of the fixed patterns. Besides the basic 22 questions, there are some heuristics that trigger other kind of questions in order to elaborate or to criticize the answers provided by users. Below we exemplify how the heuristics may trigger questions.

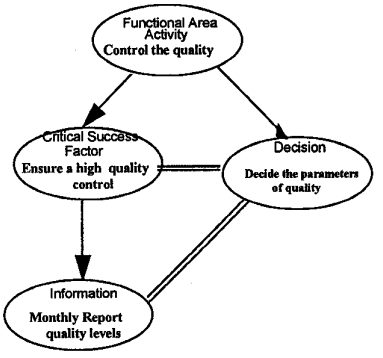
Completeness Heuristics - happen whenever a relation heuristic is confirmed and has an objective of relating model entities with information already available for other entity.

Example:

Given a critical factor and a decision and also considering that a previous heuristic found a relation between them, then the information that supports the critical factor may also be relevant to the decision.

Question:

Does < Information > support < Decision > ?



Does <*Monthly report of quality levels* > support <*decide quality parameters*>?

In this case the critical factor <ensure a high quality control> and the decision <Decide the quality parameters> were related by the confirmation of a relation heuristic that happened during the interview. As a consequence, the completeness heuristic was activated, thus creating a link between information and decision, which does not exist in the original model (Fig. 2).

FAES uses a standard production system scheme for dealing with the heuristics. Once a given node in the conceptual model is filled in by one of the questions from the automation strategy, the control mechanism activates the production memory to check if a rule will fire given the state of the knowledge base. These types of rules fire during the interview process. Other types will only be fired once the interview has ended.

2. 4 The Assistant

FAES was developed using an object oriented language, ENFIN, and a database tool, SQLBase. ENFIN is a Windows compatible software and as a

result has the advantage of easily interfacing with other software. It implements the functionality described above, but has problems regarding performance, mainly due to the use of SQLBase.

Figure 4 shows the main window. On its top it poses a question to be asked to the client. The software engineer will use the <Answer> frame to type the answer. A frame labeled <Preview Answers> shows answers previously given by other respondents for this same question and this same functional area. The <Questions triggered by heuristics> frame shows all the questions generated by the heuristics. In order to answer a question posed by a heuristic, a special window is activate. The OBS bottom makes it possible to add comments to the answer. The Synonym bottom makes it possible to associate chosen terms in the <Answer> with other terms.

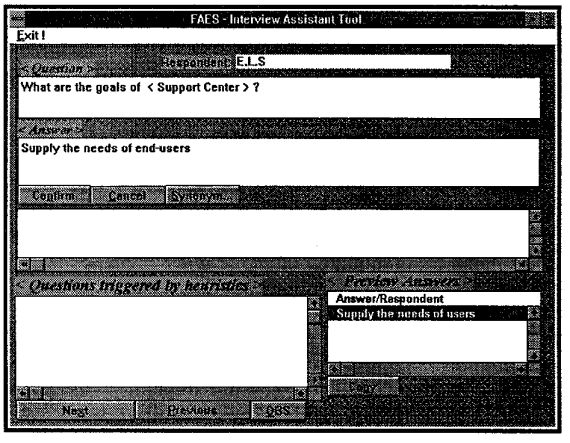


Fig. 4

3. The New Questioning Schema

One of the aspects in the original proposal was it's fixed set of questions. It was a feature in the sense that by following the script the elicitor would have filled the necessary information according to the meta-model, on the other hand it was also a barrier to the elicitor in terms of adding new information or following a different pattern of questioning. Considering that, we have analyzed our original approach and modified it in order to create an alternated form of questioning. The basic idea was to create an escape mechanism to allow the software engineer to follow, or investigate a different pattern of questioning, without loosing the original fixed set of questions.

As we can observe from Figure 5, there are four main paths we can follow in questioning. Each path of these paths instantiates their questions with respect to the list of activities answered at question 2 (the principle of chaining). The control structure in the original architecture was a fixed one. We now allow the interviewer to navigate on the control structure as well as to ask questions not in the agenda.

In order to make the navigation possible we used a stack to store the last state of the questioning. As such, we may advance to new questions or revisit questions previously answered. We anticipated that such a facility would only be effective for those familiar with the questioning structure of FAES. As we can see in Figure 5, the new control strategy gives more freedom to the interviewer.

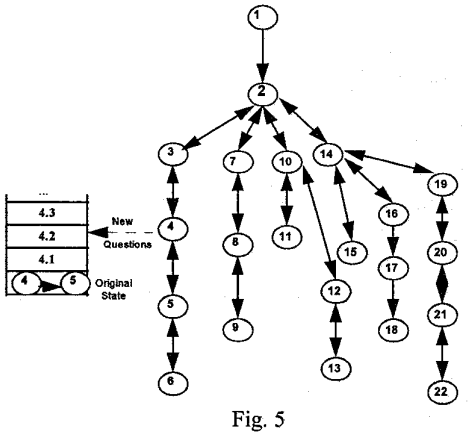


Fig. 5

Another feature that we added to FAES is the possibility that at any point in the interview process the interviewer makes a non planned question. This new feature also uses the stack mechanism showed above. In that case, we create a sub-tree, where the root is the last question of the prefixed agenda. Each question asked has to be annotated by the interviewer and will be stored together with the answer as a sub-node of the node that would hold the answer for the last questionnaire question asked. So in terms of the conceptual model, see Figure 2, we are creating a network of sub-nodes in a freely manner, but with the constraint that each node does have a link to the original node in the conceptual model. Figure 6 gives an example of such sub-network.

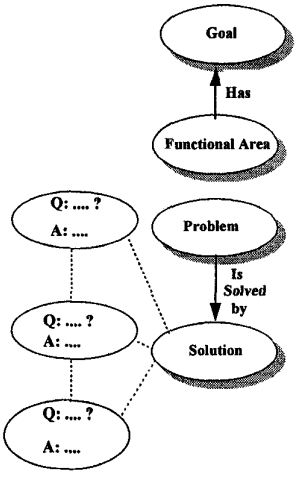


Fig. 6

It is important to note that when creating the sub-network, we still use the chaining technique, that is the next question may use the answer of the previous question.

4. Viewpoints

The heuristics used in the original version of FAES were simple and were basically driven to find out possible links in the conceptual model. It is important to note that the simplicity of the heuristics was due to the non usage of domain knowledge. One possibility of increasing the quality of the critique provided by the assistant, without relying on domain knowledge, is to follow the idea of previous answers as pointed out in [4] (see the original screen of FAES, Figure 5). We have developed this idea and used the ideas of viewpoints [7] to enhance the possibility of an on-line critique based on previous interviews.

In [7] we devised a process and a technique to compare very early requirements expressions. The technique proposed encompassed an automatic comparison of pairs of viewpoints that were expressed in a language, VWPl. The language was built on top of the production system paradigm, and basically was a typed manner of expressing rules about the problem being addressed. The language was designed to make it easier its use. Facts about the problem were described by production rules and the behavior was modeled by adding and deleting facts from the working memory.

Automatic comparison of viewpoints was performed by an AI based program that used both pattern matching and semantic information to analyze a pair of views expressed in VWPl. The comparison was driven by several heuristics which were classified according to an analogy framework described by Hall [5]. This framework is composed of the following phases: recognition, elaboration, evaluation and consolidation.

Leite's original analyzer was able to point out three types of discrepancies between views:
a) wrong information, contradiction between the facts of the different rule sets, **b)** missing information, incomplete hierarchies with respect to rule facts, missing rules and missing facts, and **c)** inconsistency, contradiction between a fact and the hierarchy and redundancy in the same rule set. The analyzer was implemented as a Scheme program that analyzes the given VWPl descriptions.

FAES was not developed with the idea of viewpoints, although Gilvaz and Leite had discussed the possible links with Leite's previous work. As mentioned before, the comments at Case'95 were a motivation for coming back to FAES and looking at how it could be enhanced with the viewpoint ideas. In studying in more detail the relationship between the two works we found out that: **a)** the viewpoint analyzer was very dependent on VWPl, **b)** it was not reasonable to ask the interviewer to express the answers of the interviewee on VWPl, **c)** the analogy framework used by the viewpoint analyzer could be applied and d) some of the heuristics geared to VWPl could be restated, if we would consider FAES conceptual model as the base representation.

With these first observations, we decided to adopt the following general strategy:

The comparison strategy would take in consideration all the models available in the FAES knowledge base, and would use three possible perspectives: goals, functional areas and clients.

In order to make the statement above more clear we have to define what do we mean by models available and the notion of perspective. First, a model is an instantiated conceptual model, that is, the result of an interview. As such, a model has always attached to it the identification of the respondent. Second, we call a perspective the following nodes of the conceptual

model: goals, functional areas and clients, since we believe each of them to be representative of a different perspective of the model. These perspectives will be used to find out similarities between models.

Using this as the basis, we have devised two group of heuristics to help the detection of wrong information and one group to detect incompleteness. First of all, we show the fine matching algorithm used in all groups. The purpose of this algorithm is to find similar text strings in two different sets. At the end of the section we present an example of the use of those heuristics.

4.1 The Matching Strategy

The matching strategy is based on the fine-matching algorithm used in [7] augmented by the use of FAES dictionary.

```
Fine-matching-with-dictionary
    Filter the answers by getting rid of articles and
    prepositions
    Find the shortest answer (measured by the
    number of words)
    member-count <-- 0
    For each word in the shortest answer
        If a word is a member of the longest
        answer
        then
            add 1 to member-count.
        else
            If word is in dictionary and its
            synonym is a member of longest
            answer
            then
                add 1 to member-count
    End_For
    Score <-- member-count / number words in
    the shortest answer
    /* OBS:. the member function is sensitive to
    /* regular verb tenses,
    /* so talk is a member in the sentence "She talked
    /* all night"
End_Fine-matching-with-dictionary
```

4.2 Group I, Finding Wrong Information

The group of heuristics presented below have the objective of finding out, at the time a question is answered and annotated by the system analyst, if there are relevant previous questions that are worth to be presented to the system analyst. The main rationale for this group is that by showing possible discrepancies between "related questions" we may alert the interviewer of a problem with that question, in the sense that there is a possible conflict of answers between two similar models. It is up to the interviewer to use the information which will be presented to him/her. The justification is that a 60% of complete different terms is a reasonable indication of a possible different answer (less than .4 fine matching score, scores are computed by matching the actual perspective with the models built by previous interviews).

1. Order the previous models by the highest score (maximum score is 3, that is 1 for each of the three perspectives).
2. Select the top two models
3. For each question do:
    - 3.1. Compute the fine-matching between the models' answers and the answer for the question at hand.
    - 3.2. If the best score is below .4 then
        - 3.2.1. Show the previous answers not matched to the interviewer
    - 3.3. If the questions matched have links to a sub-network, then
        - 3.3.1 Show these auxiliaries questions and answers to the interviewer.

4.3 Group II, Finding Wrong Information

The group of heuristics presented below have the objective of finding out, in three types of questions, if there is a possibility of wrong information at the time a question is answered by the respondent and annotated by the system analyst. The main rationale for this group is that answers to the "almost the same question" (the fixed part is the same and the variable part has a 70% fine matching) on problems, clients and sources for the same functional area must be "similar". In our case, we use the 70% fine matching as a measure for "almost the same question", and consider a "not at all similar" answers if there is less than 40% fine matching. Note that this group of heuristics issues a message about the possibility of wrong information, so the elaboration aspect of the analogy is more relevant in this group.

1. Given a matching functional area
    1.1 If there is a match, fine-matching with score > .7, on problem then
        1.1.1 compare fine-matching, solutions of the matched problem
            1.1.1.1 if score < .4 then
                1.1.1.1.1 give a msg reflecting a possible wrong information
    1.2 If there is a match, fine-matching with score > .7, on product/service then
        1.2.1 compare, fine-matching, clients
            1.2.1.1 if score < .4 then
                1.2.1.1.1 give a msg reflecting a possible wrong information
    1.3 If there is a match, fine-matching with score > .7, on information then
        1.3.1 compare, fine-matching, sources
            1.3.1.1 if score < .4 then
                1.3.1.1.1 give a msg reflecting a possible wrong information

4.4 Group III, Finding Incompleteness

The group of heuristics presented below have the objective of finding out if a given model is incomplete with respect to others. These heuristics are only applied to goals and clients. The rationale behind the heuristics is: given a similar model (main strategy), if the number of goals and clients are reasonable lower between the present model and the previous ones, then there is a great possibility of incompleteness in the present model. The heuristic for comparing the numbers uses a weight average (max, med, min) for the top two models.

1. Order the models by the highest score (maximum score is 3, that is 1 for each of the three perspectives).
2. Select the top two models
3. Compute the max, med, and min numbers of goals
5. Compare number of goals with the numbers computed at 3. Give a proper msg, in case of incompleteness
6. Compute the max, med, and min numbers of clients
7. Compare number of clients with the numbers computed at 6. Give a msg in case of Incompleteness

4.5 The Analogy Process

Our use of viewpoints in FAES is less powerful than the one used in [7], since there are several aspects that the conceptual model given by FAES is not able to capture, and does not provide the opportunity for analysis. We basically detected possibilities of wrong information and missing information, but not of inconsistency. Contrary to the schema used in [7] we use the results of the analogy analysis to take actions (first group). In our case the action is showing to the interviewer answers and questions of a previous interview.

4.5 An Example

At [4] we reported on the case study we conducted with the FAES tool in July of 1994 at Johnson's Wax information support center. At that time one member of the information center was interviewed with the assistance of the tool and the results were very positive with respect to our proposal, we managed to acquire more reliable information in a structured way. In order to exemplify our extension to FAES, one of us interviewed in December of 1995 another member of the Johnson's Wax information support center. Although this interview was performed partially, that is not all the branches of the questioning scheme were instantiated, due to time constraints, we managed to get enough data to run our example.

Once we had the interview data, we applied, by hand, the proposed heuristics using the new data and the original data collected in 1994. We will list below parts of the hand simulation we had performed for each group of proposed heuristics. The first and second group of heuristics are to be applied on the fly, that is at the moment when the interviewer enters the response to a given question. The third group are heuristics activated at the end of the interview, which may direct the interviewer to come back to a problematic question.

4.5.1 Finding Wrong Information, I

94 Interview: Question 2. *What are the activities of <information support center>?*
Answers: 1) Support PC users.
2) Manage the installed equipment.

95 Interview: Question 2. *What are the activities of <information support center>?*
Answers: 1) Give support to PC users.
2) Give support to the PC environment.

In this case, we will find that for the first answer of the actual interview (95) nothing should be shown (the score is 1 if we match answer 1 with 94 answer 1), for answer 2 there is also no reason to show the previous answers (the score .66 with 94 answer 1).

94 Interview: Question 3. *What are the problems of <Support PC users>?*
Answers: 1) Lack of user training.
2) Lack of human resources in certain areas of the support center.
3) Need to provide support to activities not belonging to the area.

95 Interview: Question 3. *What are the problems of <Give support to PC users>?*
Answers: 1) Lack of resources in the support center.
2) Little knowledge of the users.

In this case, for the first answer nothing should be shown (there is a .66 score with 94 answer 2), for answer 2, 94 answers 1 and 3 will be shown (since the best score with 94 answer 1 is below .4)

94 Interview: Question 10. *What are the critical success factors of <Support PC users>?*
Answers: 1) Good knowledge of the tools used by users.
2) Availability of human resources.

95 Interview: Question 10 *What are the critical success factors of <Give support to PC users>?*
Answers: 1) Availability of the support center employee.

In this case, for the first answer the two 94 answers will be shown (the score with 94 answer 2 is .33, that is one hit (*availability*) divided by the total number of words, less articles or prepositions, of the shorter sentence - 94 answer 2 with 3 words).

4.5.2 Finding Wrong Information, II

94 Interview: Partial list of "Information". 1) Experience. 2) Time of the task. 3) Understanding of the problem. 4) Locale of equipment. 5) Software description. 6) Know how of the training company. 7) Technical knowledge of the specialist. 8) Employee work load.

95 Interview: Question 17. *Who provides <execution time of a task>?*
Answer: Support center employee.

95 Interview: Question 5. *Who provides <employee experience>?*
Answer: Data processing manager.

Question 17 is of the type *Who provides <information>?* (see 1.3 of 4.3), as such we have to find a matching information on the previous models and compare the answers (sources). For instance, comparing the answer (source of information) of question 17 with the answer of the corresponding 94 question (*Who provides <time of a task>?* -- **Answer:** Analyst), we get a non match, thus issuing a message of a possible wrong information. If we compare the sources for question 5 (Data processing manager) and the 94 corresponding answer (Analyst) we also get a non match and the proper message.

4.5.3 Finding Incompleteness

The 94 interview had one goal and the 95 interview also had detected just one goal., so no message is issued here. In the 94 interview two clients were identified and in the 95 interview just one, in this case a message indicating a possible incompleteness will be issued.

4.5.4 Comments on the Example

Our intention with the example was to induce the reader to follow the heuristics and to come to their own conclusions. Nonetheless, we would like to point out some of the facts observed. First of all, we believe that the example reinforced our hypothesis that the viewpoint matching approach is a sensible way of providing automated support validation. If we examine group I we will observe that not all the questions in an interview will be candidates for comparison, since only questions derived of a previous match will be analyzed, in that sense this is positive because these heuristics will be only applied when appropriated.

If we examine question 3 of group I above, we will note that for 95 answer 2 the 94 answers 1 and 3 will be shown to the interviewer. Showing 94 answer 3 to the interviewer detects that the respondent may have failed to observe a problem pointed out by a previous respondent, on the other hand, the fact that 94 answer 1 is shown points out the limitation of syntax matching, but the interviewer may filter the information and use it as a confirmation of the answer given. With respect to question 10 the interviewer may brought to light the question of user training.

If we examine the results got from group II we believe is more evident the kind of support that this approach can provide to the interviewer. For the question of type 5 is clear the difference in viewpoints between the respondents, probably the 95 respondent did a better job in identifying the source of the information. For question type 17 the answer is a generalization of the answer previously given, the approach does not detect this (VWPI hierarchies [7] would detect this type of match) so it complains of a possible wrong information.

Group III provides a critique at the end of the interview, and the result may lead the interviewer to came back to a previously answered question of type 15 (*Who are the clients of < products/services>)*. In the example we have found out that the 95 interview may have failed to identify a client.

5. Conclusion

This article elaborates on the result of previous research. We [4] showed an architecture for an interview assistant and gave data of its use in a case study. Considering that work and encouraged by the discussions of FAES at Case'95 we have proposed in this article an improvement on the assistant heuristics based on Leite and Freeman's work on viewpoints. Here as in [4] the main focus is on very early elicitation. In this article we have shown how we can easily include flexibility in our questioning scheme as well how viewpoint analysis can improve the feedback provided by the assistant.

Your contribution is well focused. We managed to show how a simple and non domain oriented elicitation strategy could be improved by established results in the field of viewpoint software engineering.

Although we did not conducted a complete case study with the new FAES architecture, the example showed that we have solid grounds to hypothesize the improvement in performance by adding viewpoint analysis as well as free questioning.

With respect to viewpoint analysis our previous experience shows that the kind of heuristics we have included in FAES are very effective. On the other hand our experience with the original FAES has shown, by the use of the OBS bottom (see Figure 5), that allowing free questions and answers would help the software engineer.

Reubenstein [9] and Drake [3] also dealt with interview automation. Reubenstein has developed a general assistant to gather information in the process of knowledge acquisition, but his strategy is based on a previous encoded knowledge base, which will serve as an oracle for the acquisition of requirements. Drake proposed an assistant to guide the client in answering questions anchored on a general model geared towards input/output. As we stressed before, our approach is not dependent on previous encoded knowledge, but the viewpoint analysis strategy will work better if our knowledge base is populated with interviews models (previous interviews). Contrary to the original proposal in [7], which used a special language for expressing viewpoints, we have developed analysis heuristics for an existing representation scheme. For that matter, this approach is similar to Finkelstein et al [1], since they have written heuristics to compare different instances of well-known software engineering representation schemes.

Future work should be geared towards an efficient implementation of the new FAES and its use in other case studies. The major problem regarding implementation will be how to integrate the original heuristic application with the necessity of dealing with more than one instantiated conceptual model at once.

Acknowledgments

We would like to acknowledge the contributions at those at the Case'95 session, whom by asking questions prompted us to extend FAES capabilities. In special we thank Jeff Kramer, since his question on viewpoints was the main reason for this article. The review process improved our paper, we are grateful to the referees for their comments.

References

[1] Finkelstein, A., Gabbay, D., Hunter, A., Kramer, J., Nuseibeh, B.;., Nuseibeh, B.; Inconsistency Handling in Multi-Perspective Specifications; *IEEE Trans. on Software Eng.*, Vol. 20, n. 8, Aug. 1994, pp. 569--578.

[2] Feather, M.S., Requirements Reconnoitering at the Juncture of Domain and Instance, *RE93, Proceedings of the Intl. Symposium on Requirements Engineering, IEEE Computer Society Press;* Jan. 1994, pp. 73--76.

[3] Drake J. M., Xie W. W., Tsai W. T. A Case Study of Requirement Analysis where End Users take an Active Role *Proceedings of the 15th International Conference on Software Engineering, IEEE Computer Society Press*, 1993, pp. 177-186.

[4] Gilvaz, Ana Paula, Leite Julio Cesar S. P. FAES: A Case Tool for Information Acquisition; *Case'95, Proceedings of the Seventh Intl Work. on Computer-Aided Soft. Eng. IEEE Computer Society Press;* Jul. 1995, pp. 260--269.

[5] Hall, R.; Computational approaches to analogical reasoning: a comparative analysis; *Artificial Intell.;* vol. 21, no. 1, Jan. 1988, pp. 241--250.

[6] *Information Systems Planning Guide, Application Manual*, GE20-0527-3, Third Edition, IBM Corporation (July 1981)

[7] Leite, Julio Cesar S P. and Freeman, P.A. Requirements Validation Through Viewpoint Resolution, *IEEE Trans. on Soft. Eng.*, Vol. 17, No. 12, Dec. 1991, pp-1253--1269.

[8] Loucopoulos, Pericles and Karakostas, Vassilios *System Requirements Engineering,* McGraw Hill International Series in Software Engineering, 1995.

[9] Reubenstein Howard B., Waters Richard C. The Requirements Apprentice: Automated Assistance for Requirements Acquisition
IEEE Transactions on Software Engineering,, vol 17, (3) 1991, pp. 226-240.

[10] Rich, Charles, Waters Richard C. Knowledge Intensive Software Engineering Tools, *IEEE Transactions on Knowledge and Data Engineering, 1992, vol 4, (5),* , pp. 424 -- 430.

[11] Rockart, J. F. Critical Success Factors *Harvard Business Review, 1979, (March-April), vol 57, (2)* pp. 81 -- 91.

[12] Wetherbe James C. *Systems Analysis and Design,* West Publishing Company, St.Paul, MN, 1988.

[13] Wetherbe James C. Executive Information Requirements: Getting It Right *MIS Quarterly, March* 1991 pp. 50 --65.