Dealing with conflict in CSCW Model for Writing Formal Software specification Document

Mohamed Ahmed Sullabi , Mustafa Bashir Abugharsa , AlhusainMohamed Taher
Faculty of information technology
sullabi@yahoo.com,maughasra@yahoo.com,hmat1976@yahoo.com

Abstract— For studying how a group of people can be work to preparing and produce a correct formal software specification, and how can they communicate with each other. One of main issues that should be taken into consideration when we propose a solution is the conflict among the group members and how we can manage this conflict. We cannot engage in group activities without encountering conflict. It is an inherent part of the collaborative process. Conflict can be either destructive or constructive, depending on how it is handled. Misunderstanding of roles can lead to conflicts within a group. Collaborators must reach some degree of consensus or they will never get anything accomplished. In this paper, we present our model that provides software developers with web environment that supports them to collaborate and to help them to produce correct software formal specifications. We discuss some approaches that were applied in our model to help in managing any conflicts could be arising during the rectifying processes.

Keywords — Collaborative writing; Conflict; Formal specifications

I.  INTRODUCTION

In this study, we concern with how a group of rectifiers can communicate with each other and work to prepare and produce a correct formal software specification. To address these issues, we come into view of a theoretical framework that categorized into two; collaborative issues and formal specification issues. Each of these issues are taken into consideration when we propose the solution of the subject that is a web-based model of Computer Supported Cooperative Work (CSCW) for preparing and writing formal software specifications document [1]. This model provides software developers with a web environment that supports them to collaborate and to help them to produce correct formal software specifications. A group of rectifiers and a drafter who co-ordinates and supervises the whole process and initiates each stage, are the team work in the model. They collaborate to perform all the activity of writing and rectifying the specification constructs in Z formal notation that are used to modularize system state and behaviour. The model uses a spiral approach for resolving conflicts that may arises through the rectifying process between the writer and the reviewers. The model introduces a new technique to produce a formal specification called SNL2Z [2]. It provides the rectifiers with the facility of writing the operation schemas in structured natural language statements, then it automatically translates these statements into formal specification form. A report with the translated operation schemas is produced and many other reports for each stage of the process and a team over all progress report for the management purpose is also produces.

Based on the proposed model, a web-based prototype application system has been implemented and developed [1]. A multiple method approach is used at the laboratory test to evaluate the developed system. In the laboratory test, two main types of data have been collected. The first is objective includes observations from the interaction between users through their use of the system and the other type is subjective, resulting from the survey we have carried out with users.

II.  COMPONENTS OF THE MODEL

A.  People

The team work involved in this model for preparing software formal specifications consists: a drafter who initiates each stage of the process, writes the draft of documents, and supervises the whole process. The rest of the team will be a group of rectifiers who work on the shared document, write specifications, write corrections, and comment on corrections. Beside his/her main job as co-ordinator for the team, the drafter could be also worked as a rectifier in a setting where the team members are not located in the same room. Through the Internet, the team members collaborate to write, edit and correct the shared specifications document.

B.  Processes

In Z formal notation, specification constructs are used to modularize system’s state and behaviour. Among these constructs, schema is the most important tool to encapsulate specification chunks. Schema construct is used to model both system state (as state schema) and behaviour (as operation schema) [3]. There are two main processes. In the first stage, the main characteristic is that it allows users to collaborate to correct and comment on shared formal specification document of basic types and state schemas. In the second stage, the system is providing the users all the facilities to collaborate to write the operation schemas, and to write corrections and comments on these operation schemas. The specifications of the operations schema will be written in structured natural language then the system will translate these natural specifications into formal specification in Z.

1. Rectifying of Basic Types and State Schemas.

The drafter initiates the process by passing a document through the system which contains either the introduction and details of the problem and the a draft of basic types and state schemas specifications to the team or only the
The draft document will be automatically broken into small parts. Each part represents either a single basic type or a single state schema.

By breaking-up the document into small parts, every rectifier will be able to get one part of the document to work on. Each part of the document will be edited by one rectifier at a time, and all parts will be locked for one rectifier at a time. When the rectifier finishes editing a certain part of the document, this part will be unlocked to be available for editing to the rest of rectifiers. By this way all parts of the document can be edited by all rectifiers one by one and all rectifiers can edit all parts of the document as well.

The team will examine the different parts of the document for a necessary action. Every member of the team has the right to rectify the document and write down his/her correction on any part of the document. Every written correction will be connected to the relevant part of the document. Any one in the team can read the all corrections when he/she chooses its relevant parts. Every one in the team can correct other’s corrections, and then he/she can evaluate and write his/her comments on each correction. Every correction will be connected with the all comments written by the team on it. Any one in the team can modify his/her correction in the light of the other’s comments on that correction. Then the other rectifiers can update their comments on the modified correction, and so on, until every one accepts and agrees about that correction, if not the correction is not deleted by the writer, it will be leaved to the drafter either he/she will accept or just ignore that correction.

After all correction processes have been completed on the basic types and state schemas, and both drafter and rectifiers are agreeing about it, the rectified document parts will be combined into one final document. The drafter will apply the necessary changes to the main document of basic types and state schemas, taking in his/her account the rectifiers’ corrections and comments. Then he/she will launch the process of writing and rectifying the operation schemas of the system.

2. Writing and Rectifying of Operation Schemas

As mentioned above, after the team have finished the writing, correcting and editing the first stage of the process on the basic types and state schemas, and the drafter have rectified the document, the drafter will launch the second stage of the process of writing and rectifying the operation schemas of the system. Through the launching this stage, the structure and all the relations in the basic types and state schemas document will be automatically studied to provide the necessary help in writing the second stage. In the second stage, there are two main tasks should be performed. First task is to write the main operation names list, while the other task is to write the specification of the operation schemas.

Main operation names: The drafter may (not) proposes a list of main operations names and he/she passes this list to the team. Rectifiers can also suggest a new main operation names by add them on the list. Every one in the team can add new main operation names, edit or delete any main operation name was written by him/her, or write his/her comments on the main operations names that written by others.

Operation schemas: The structure of the operation specification in our proposed model has been adopted and modified from the table proposed by Bottaci and Jones [4]. They suggest that it is helpful for the specifiers to summarize the decision about the operations in a table form. This will free the specifiers from the details of mathematical formulas at the first round of preparing the formal specification. Normally, one operation of a system consists of several basic operations. So as Z language, one operation might be represented by several schemas. Each schema handles different type of pre-condition of the operation. The table proposed by Bottaci and Jones is designed in such a way that: a row represents a schema whilst five columns are used to record the schema name, the inputs of the operation, the pre-conditions, the changes to the system state, and the outputs of the operation. In this study, we add a new column which is used to record the other schemas’ name that will be included in the currently specified schema. Each main operation name mentioned in the previous section (section A), may has a several operation schemas. Every member in the team can add a new operation schema after he/she chooses a main operation name from the list. When the rectifier plans to write specifications of an operation schema, he/she will be provided with all the necessary details that he/she needed to help him/her to write these specifications. For example; when the rectifier intends to specify the included state schema part, he/she will provided with a list of all the state schemas included in the main document to choose from. Then the rectifier also supplied a list with all the variables that chosen state schema contains. The list of the state schema variables will help the rectifier when he/she intends to write the pre-condition and post-condition of the entire operation schema by choosing the write variable from the list.

By using this method in writing the specifications of the operation schemas, the writing of specifications will be under control of the system. Most of the important operation schema specifications will be written under a certain procedure, for example, state schemas and variables will be chosen from a list provided by the system. In addition, a particular structure is also required when writing of the specifications of the operation schema. This procedure and structure that we use here will produce specifications in a natural language but in a structured form. This is what we call structured natural language.

When specifications of the operation schemas is specified and written, they will be available as list to the all of team member. Every one in the team can chooses any operation schema from the list. The chosen operation schema will be locked for him/her. Then he/she can either edit or delete the chosen operation schema (if he/she is the writer of the entire operation schema), or review the
chosen operation schema and write his/her comments after evaluating it (if he/she is not the writer of the entire operation schema). When the rectifier finishes editing, evaluating, commenting the entire operation schema that he/she was working on, this particular operation schema will be unlocked to be available for editing to the rest of rectifiers. By this way all operation schemas can be edited by all rectifiers one by one. Every operation schema will be connected with the all comments written by the team on it. Any one in the team can modify his/her operation schema in the light of the other’s comments on this operation schema. Then the other rectifiers can update their comments on the modified operation schema, and so on, as shown in “Fig. 1,” until every one accepts and agrees about this operation schema, if not and the operation schema is not deleted by the writer, it will be leaved to the drafter either he/she will accept or just ignore that operation schema.

![Diagram](image)

**Figure 1. Rectifying processes of a operation schema**

### III. MANAGING CONFLICTS

Conflict is a natural disagreement resulting from individuals or groups that differ in attitudes, beliefs, values or needs [5]. It can also originate from trying to negotiate before the timing is right or before needed information is available [6]. Some approaches are applied in our model to help in managing any conflicts could be arising during the rectifying processes.

**A spiral approach:** when a member writes his/her correction on any part of the basic types and state schemas draft document, this correction will be attached to that part of the document. Every one in the team can review this correction, after he/she chooses the relevant part, and then he/she can evaluate and write his/her comments on the correction. Evaluation of the correction is to determine whether the reviewer is agree, not sure, or disagree with writer, while writing comments is to get an explanation why the reviewer was agree, not sure, or disagree about the correction. In the comments area the reviewer also can give an advices and/or ideas to the writer about how the correction should be, what the good and bad points in his/her correction, and what should to do and should not …etc. All these comments and evaluation results written by the group will be attached with that a particular correction.

Every correction’s writer can review all the comments written by others and their evaluations, and then he/she can modify his/her correction in the light of these comments. In the second round, the comment writers, can update their comments on the modified correction, and so on, until every one accepts and agrees, if not and the correction is not deleted by the writer, it will be leaved to the drafter either he/she will accept or just ignore that correction.

**Majority voting:** As we mentioned, every written correction on any part of the draft document or any added operation schema received by the drafter will be attached with the evaluation of the team. This evaluation is a voting from the team members on that correction or that operation schema. The drafter will get statistical values of the team voting on each correction. These values show the level of the agreement of the group on the each correction. Voting is the best-known and most commonly used method for making final decisions. For many issues, it can be the easiest and quickest way to decide a matter [7]. Therefore, it will help the drafter to decide which correction he/she should accept or ignore.

**Explicit of updates:** In the rectifying process scenario of basic types and state schemas, and the operation schemas, the explicit updates must be carried out after each modification has been specified. In each cycle of the rectifying process, the correction’s writer needs to be aware of the updated comments to modify his/her correction. Likewise, the commenter needs to be aware whether the correction had modified after his/her comments were written or not. Because of rectifying is an ongoing process, the showing message <Updated> or <Not Updated> together with the comments, it may speed up the rectifying process and resolve conflict via negotiation between the writer and the reviewers. It allows the writer to mind only the updated comments and he/she can modify his/her correction in the light of these comments. In the same way, it will be easy for the commenter to modify his/her comments if it shown to him/her that the particular correction has been modified after the date of the writing of these comments.

**Veto:** In our model, the drafter is the team leader and supervising the whole process. At the end of each stage of project, the drafter is the main person who has the right to accept or ignore any correction, comment, or idea from any one in the team. The drafter with this privilege can put a limit for any conflict that could not be solved through the negotiation among the team member. The drafter can build his/her decision based on the comments and evaluations of the team through the negotiation on that a particular correction.
IV. CONCLUSION

Our model presented in this paper and the approaches that are applied in the model provide a stepping-stone to progress the software development field. It supports software developers to help in managing any conflicts that could be arising during the rectifying processes and to produce correct software specifications by helping them to collaborate on preparing formal software specifications document and it gives the way for them to communicate, edit and correct the shared formal specification document.

Writing a correct specification is very difficult, but by applying a model that melting away the difficulties which prevent software developers from having formal specification, it will encourage them to collaborating to write, rectify, and produce correct formal software specifications. The successful construction of this prototype system demonstrated that the model was implementable and usable. The evaluation processes of the system and all the case studies have been carried out successfully in an educational setting. This means that we have had the possibility to introduce a Web-based system for writing and teaching formal specification in the classroom.

REFERENCES


