Experiences on the Belief-theoretic Integration of Para-consistent Conceptual Models

Ebrahim Bagheri and Ali A. Ghorbani
Faculty of Computer Science,
University of New Brunswick, Fredericton, Canada
{e.bagher,ghorbani}@unb.ca

Abstract

Viewpoint-based conceptual modeling is concerned with the identification of a complete and coherent set of software models that have been developed with the involvement of various analysts. The contribution of multiple analysts in this process will provide a rich and comprehensive final product. One of the major concerns in any process requiring the direct involvement of human analysts is the introduction of uncertainty and inconsistency. In this paper, we employ a formal model based on belief theory that attempts to capture the degree of analysts’ uncertainty towards their specifications and builds on these information to create a unique integrated model. The model is employed in the process of developing a conceptual model for the Pet Store application. The results show that the formal framework provides suitable tools for formal negotiation, belief revision, consensus building, belief recommendation and expert reliability evaluation.

1 Introduction

Requirement engineering is an important branch of software engineering which is involved with the identification and formalization of the goals, functions and constraints of a software entity [18]. Many researchers believe that requirement engineering is a process composed of two elements namely requirement elicitation, and requirement modeling [9]. In requirement elicitation, analysts are concerned with the revelation, communication, and validation of the facts related to the software entity, while in the modeling phase, they attempt to represent and organize the identified facts in some agreed format usually known as conceptual models. Conceptual models are therefore representational abstractions of the possible model of the software entity [2]. Conceptual modeling is a challenging aspect of the requirement engineering process.

Various attempts have been made to create a firm foundation for conceptual modeling. Formal specifications such as the Z notation or models based on logical formalisms, category theory, requirement templates, state transition diagrams, conceptual graphs, and object oriented models are among the most widely used [5]. Besides the problem of developing a unified model for software conceptual models is the issue of developing and instantiating a proper conceptual model for a software entity. The principle of employing more information sources for getting a better insight into a given problem has been widely used in court investigations for many years [9]. The intuition behind this practice is that various information sources, more specifically human evidences in this case, have different areas and amounts of knowledge that may help in better analyzing the problem. They may also each use different styles of knowledge expression that can itself be helpful in beating the concern [2].

Requirement analysts have been interested in using information from multiple sources to create a concrete, consistent and complete compilation of software requirements. Sources of information are mainly known as viewpoints. Nuseibeh defines viewpoints as “loosely coupled, locally managed distributed objects which encapsulate partial knowledge about a system and its domain, specified in a particular, suitable representation scheme, and partial knowledge of the process of development” [12]. Although not all models of viewpoint-based requirement engineering conform to this definition, most of them roughly agree on this basis. For example, Greenspan and Feblowitz [7], have identified four viewpoints that in their understanding are useful for deriving software systems requirements. These viewpoints are service viewpoint, service workflow viewpoint, organizational model viewpoint, and capabilities and resources viewpoint.

The incorporation of information from multiple sources for creating a complete conceptual model of a software entity seems to be intuitively appealing; however, there are various issues that need to be addressed before viewpoint-based models can be correctly employed. First, humans
usually make conception errors due to risk aversion, short-
term memory or even framing and perceptual problems [13]. This implies that not all asserted information from the sources are correct or equally reliable. Furthermore, epistemic uncertainty (also known as partial ignorance) is an indispensable element of human judgments that makes them even more susceptible to inaccuracy and imprecision [15]. There is no guarantee that the received information coming from various human sources be consonant or consistent. They may be in many cases arbitrarily (with very few common elements) or disjointedly distributed (no common elements) [15]. Hence models that incorporate human expert judgment into their analysis need to consider the significant role of uncertainty and imprecision.

There are three major theoretical frameworks through which uncertainty can be handled and manipulated, namely: subjective probabilities, possibility theory, and Dempster-Shafer theory of evidence [15]. In this paper, we intend to employ and evaluate an existing formalization of viewpoints [3]. This framework should allow us to develop a unified framework for integrating the beliefs of different viewpoints regarding various aspects of a software entity through directed collaboration.

1.1 The Employed Framework Overview

The integration model that we have employed in this paper is founded on the degree of experts’ uncertainty towards their expressed specifications [4, 3]. To capture the degree of uncertainty, a three dimensional belief structure based on Subjective logic is employed. This belief structure allows opinions to be expressed with varying degrees of belief, disbelief, and uncertainty. The exploitation of this underlying logic allow us to reason about different specifications that are coming from multiple sources.

One of the important phases of a collaborative conceptual model development is viewpoint merging. In the employed framework, this is performed on the basis of belief merging operators such as Dempster’s rule of combination. The only restriction that is applied on the expressed specifications is the use of a common application vocabulary. Similar assumption has been made in the related literature [11, 6]. In this approach, the terminology repository is incrementally developed and shared as the viewpoints define and express their models. The common set of vocabulary is required so that specification overlaps are detected. Three metrics are used for analyzing the effectiveness of the merging process and the quality of the obtained model. These metrics can assist the analysts in the process of decision making and planning. The proposed framework also develops and guides a formal negotiation process between the viewpoints so that a stable consensus is faster developed. Through the process of viewpoint merging and consensus building, the reliability of the experts and information sources are calculated with regards to various domains of expertise and will be used to discount the expressed information of the viewpoints.

It is important to notice that the merged views are not always syntactically or semantically consistent. Although we agree with Richards et al. [14] that lazy consistency allows the emergence of a richer final specification, it is required that the analysts perform a consistency check on the developed specifications at some point of time. The employed framework provides the basis for automatic analysis of the specifications and performs the pruning of the specifications that are not significant from a collective perspective. Figure 1 shows the flow of the processes in the proposed framework. Interested readers are recommended to read [4, 3] for further details of the employed framework.

1.2 Consensus Effectiveness Metrics

The process of belief revision and combination in collaborative conceptual model development is usually repeated iteratively until a desirable degree of stability is reached. Measuring the degree of stability requires some quantitative measure for evaluating the resulting model. Stephanou and Lu have proposed a quantitative metric called the generalized entropy criterion which measures the degree of consensus effectiveness obtained as a result of combining evidence coming from multiple sources through the application of the Dempster’s rule of combination [17].

It has been shown that the generalized entropy criterion decreases with consensus; therefore, the amount by which it decreases as a result of consensus can be considered as the degree of consensus effectiveness. There are various obstacles to the direct application of the generalized entropy criterion in our application domain. Firstly, it seems that the generalized entropy criterion is most suitable for large frames of discernment, and hence some of its metrics such as partial ignorance do not make sense in a binary frame with singleton elements. Furthermore, Since the entropy model is designed for measuring consensus effectiveness based on Dempster’s rule of combination, it has assumed that the conflict mass has been normalized out in combination, which is not a correct assumption in our case. It also does not directly address uncertainty and disbelief dimensions that are present in our belief formalization. For these reasons and inspired by the generalized entropy criteria, we propose three similar metrics for measuring consensus effectiveness, namely ambiguity ($\zeta$), indecisiveness ($\psi$), and conflict ($\delta$).

Each of the metrics are calculated separately for all of the present concerns in the analysis process. The metric values for each concern are superscripted with the corresponding concern indicator. For instance, $\zeta_k$ depicts the ambiguity
value with regards to concern \( k \). The overall metric values are calculated by a weighted aggregation scheme shown in Eq. 4. Ambiguity is defined as:

\[
\zeta^k = - \sum_{i=1}^{n_k} \left( \frac{1}{e^{b_x^k i + d_x^k i}} - 1 \right).
\] (1)

Ambiguity is similar to belief entropy in that it provides the basis to calculate the degree of confusion in the overall viewpoints knowledge about the exact fraction of belief that they should assign to each hypothesis. The indecisiveness metric is a measure of the ability of the viewpoints to firmly state a given proposition. For this purpose, the further away the degree of belief and disbelief of a given hypothesis are, the stronger and more decisive the hypothesis is:

\[
\psi^k = \sum_{i=1}^{n_k} \left( \frac{1}{e^{b_x^k i - d_x^k i}} - 1 \right).
\] (2)

The third metric that we define for consensus stability analysis is the conflict metric. Conflict defines the degree of inconsistency between the beliefs of the different viewpoints. Therefore, analogous to the previous two metrics, conflict should also be minimized for reaching a more stable conceptual model as a result of consensus.

\[
\delta^k = \sum_{j=1}^{n_k} (b_j^\text{consensus} + d_j^\text{consensus}).
\] (3)

where \( n_k \) is the number of propositions in viewpoint \( k \). To create a unified view for each of the three metrics, the value of the metrics in each concern should be aggregated. The aggregation process is a weighted sum of each metric. The applied weight is proportional to the number of propositions in the corresponding concern. In cases where the analysts can assign an importance value to each concern \( (\gamma^k) \), that is also taken into consideration, else all concerns are considered equally important \( (\gamma^k = 1) \). Let \( m, |\text{proposition}|, |\text{proposition}| \) be the number of concerns, number of expressed propositions in concern \( k \), and the overall number of expressed concerns, the overall conflict metric will be:

\[
\delta = \sum_{k=1}^{m} \frac{|\text{proposition}|}{|\text{proposition}|} \times \gamma^k \times \delta^k.
\] (4)

Ambiguity and indecisiveness metrics are also similarly aggregated. As mentioned earlier, the process of belief revi-
mission and integration is repeated until a stable and agreeable model is reached. In this process, each iteration should be analyzed based on the consensus effectiveness metrics. The desirable state is when the value of all the three metrics has decreased significantly enough as a result of the application of the consensus operator.

2 An Example

Here we introduce a sample application of the framework. Consider a simple example where three analysts John, Bob, and Mary representing three different viewpoints collaborate to create a complete model of a transportation system. We partially show how these viewpoints can create a unified conceptual model. For simplicity purposes, the analysts consider the system from a single shared perspective (single concern) and we further assume that the analysts are equally reliable. Figure 2 shows a part of the conceptual models designed by each of the perspectives. The letters in the gray ovals represent viewpoint opinions in linguistic terms. As it can be seen the models are annotated with opinion values that help the integration process.

In this example, Bob and John are concerned with the design of the car itself, while Mary is aiming to design the external relationships of the car with the other elements. Before the integration of the models, we have to transform them using the underlying construct. While transforming the conceptual models through the underlying construct, we can infer that there is an inconsistency between the models designed by Bob and John which is the result of the difference in the definitions of the concept of tire. John has modeled the tires of a car as its attributes, while Bob has defined them as a separate stand-alone class.

After the models have been turned into belief representation they can be merged by first discounting each of the propositions of the models with the reliability of the viewpoint and then combining them using the consensus operator. The models are not yet checked for inconsistency, and hence the existence of conflict is permitted. At this stage, the process of proposition recommendation for consensus formation and viewpoint reliability re-calculation can take place. For example, since Mary has disagreed with the definition of tire as an attribute and Bob has defined a tire as a class, John will be informed (proposition recommendation) that the definition of a tire being an attribute is not a suitable choice. Furthermore, based on the assertions of Bob, a suitable amount of belief for assignment to the tire attribute proposition is recommended to John (belief recommendation). The viewpoints can gradually adjust their belief values and use the belief and proposition recommendation procedures to form a formal negotiation process and finally settle on an agreed set of specifications.

As it can be seen in Figure 3, the merged model can contain partial inconsistency that needs to be resolved. Based on the potency metric [3], we can infer that the model that has defined the tire of a car as a stand-alone class has a higher degree of believability (potency) and therefore, the notion that has modeled tire as a class attribute is removed from the final conceptual model.

It is worth noting that each of the viewpoints had the possibility to insist on its opinion by explicitly adding the conflicting proposition stated by the other viewpoints to its belief base and assigning it a very high degree of disbelief. For example, John could have added a tire class to
its model and assigned it an ‘Absolutely Certain/Firm Disagreement’ opinion (Mary has done this for the concept of tire). This would have significantly affected the final merged model. On the contrary, Since s/he had already expressed some doubt on his first proposition by assigning an ‘Either Way/Slightly Certain’ opinion to the tire attribute, s/he could have overly softened his opinion (or even removed his initial proposition) so that consensus could have been achieved more easily and the degree of confidence in the final product would have been higher.

3 Tool Support

To support the collaborative conceptual model design process, we have employed two separate Eclipse plug-ins, namely Integration Client, and Integration Server. The plug-ins are developed on top of the Eclipse Modeling Framework (EMF). The viewpoints can install the integration client plug-in into Eclipse and develop their conceptual models. The required information regarding the various types of recommendations, available concerns, common application vocabulary, etc. is sent to the clients from the Integration Server that needs to be set up for the viewpoints to connect to.

The Integration Client plug-in provides the viewpoints with the functionality to assign linguistic beliefs to their conceptual model elements (see Figure 4), check for any element without an explicit belief value assigned to it, assert self assessment of the viewpoint’s own reliability, visually observe the change of the integration effectiveness metrics, view and employ the common terminology that is incrementally developed by all viewpoints, and also receive belief and proposition recommendations. Using the Integration Server plug-in the moderator of the conceptual modeling process can define the participating viewpoints and their network location, the set of possible concerns and their significance weights, perform versioning on the integrated conceptual models, and visually track the reliability values of the participating viewpoints.

The belief and proposition recommender modules of the plug-ins provide the means for the viewpoints to formally negotiate over their expressed specifications. Besides this type of facility for the viewpoints to communicate, there are also cases where the viewpoints require direct informal communication with the other viewpoints in the form of natural language texts. In such a case, the viewpoints can send textual messages to other viewpoints to clarify their intentions. These messages are attached to model elements so that the track of the issues discussed in the modeling process can be kept. Furthermore, when a viewpoint receives a recommendation from the Integration Server, it can appeal to the proposed recommendation by sending a message to the server. The server will then re-route the rebuttal to those viewpoints that had an influence in the recommendation. With this feature the viewpoints can both formally and informally collaborate through the proposed framework towards building a unified conceptual model.

The tool also provides features for tracking the origins of the elements of the final conceptual model. The viewpoints can observe the history of changes, discussions, and belief adjustments that have been made on the elements of the conceptual model. This feature is useful for understanding the intuition behind the existence of each model element in the integrated conceptual model.

4 Performance Evaluation

The proposed framework was evaluated from two perspectives: effectiveness, and usability. In the set of experiments which were carried out to evaluate the effectiveness of the framework, a group of Computer Science graduate students were asked to design two conceptual models for the Pet store application from a given textual description. The development process of these conceptual models was facilitated by the proposed framework. Furthermore, the participating students were also asked to complete the Computer System Usability Questionnaire (CSUQ) whose analysis was employed for examining the usability aspects of the proposed framework and its supporting tool. In the next two subsections the effectiveness and usability facets of the proposed framework are evaluated and discussed.
4.1 Framework Effectiveness

Sommerville and Sawyer [16] have proposed three main viewpoints, namely domain, interactor, and indirect stakeholder viewpoints that in their belief can cover most of the aspects of a software design process. The domain viewpoint is the advocate for the set of requirements and design needs imposed from the application domain. The interactor viewpoint is the representative of the things that directly interact with the target system and finally, the indirect stakeholder viewpoint is the understanding and requirements of the parties that do not have direct interaction with the target system, but have some sort of interest in the system. They also propose four major concerns that need to be considered which are safety, availability, functionality, and maintainability. Following the same setting, we selected different viewpoints that represented these three types of viewpoints. Each viewpoint was given the same textual definition of the Pet Store application. The viewpoints were required to identify the requirements of the Pet Store application from the textual information. According to their own requirement definitions, each viewpoint then designed two different conceptual models in the form of UML class and state diagrams. They were also asked to initially specify the degree of their own reliability in each of the four concerns.

To support the negotiation, and model integration process, the developed client-side plug-in was provided to the viewpoints. The server-side plug-in was also employed to manage the processing of information that were gathered from each individual viewpoint. The conceptual models developed by each viewpoint were stored on the server-side plug-in in Ecore format. Using these tools, each of the viewpoints started to specify their requirements and then design their conceptual models. In this process, belief and proposition recommendations were constantly provided to the viewpoints. Furthermore, based on the agreement of the viewpoints, the developed conceptual models were merged automatically and stored on the server and the individual models were versioned and stored. This process was repeated until the final result of the collaborative development of the conceptual models was considered acceptable by the viewpoints. Epoches were periodic forty-five minute long working sessions. The development of the class diagram and state chart required ten and eight epoches, respectively. In each epoch, the final merged model was compared with a reference conceptual models that were developed by a group of experienced modelers for the Pet Store application. These reference models were considered as sound models designed based on the given textual description to be compared to.

In the two experiments the behavior of the three metrics were also observed. Figures 5 and 6 show the average value of these metrics in every epoch for all the viewpoints. The Conflict metric increases until the middle of the experiment and then gradually decreases towards the end. This can be interpreted as a result of the behavior of the modelers. In the collaborative design procedure, the viewpoints start off by adding those elements of the conceptual model that they think is correct from their own stand point. This causes conflicts and discrepancies between their designs. Gradually as the viewpoints observe proposition and belief recommendations, they consider adjusting their expressions in order to reach a final conclusion. Throughout this process, the degree of conflict will gradually decrease. Similarly, in the other two metrics the Ambiguity and Indecisiveness of the opinions of the viewpoints decreased as a result of their collaboration.

In each epoch, the individual conceptual models were merged and stored. The comparison of the merged model of each epoch with the reference conceptual model yielded the competency of the proposed model. For the UML class diagram, two main metrics were observed in each epoch: Precision and Recall. To calculate precision and recall, four base metrics were measured, namely True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). TP represents the number of design entities that existed in the reference model and also did exist in the merged model. FP denotes the number of model elements that did not exist in the reference model, but did exist in the merged model. On the other hand, TN shows the number of model elements that did not exist in either the merged model or the reference model, and FN depicts the number of elements that did exist in the reference model but were omitted (or did not appear) in the merged model. Using these four metrics, Precision and Recall can be calculated.

Precision (quality of the merged model) is the ratio of correctly labeled elements in the merged model over all of the elements in the merged model, and recall (coverage of the merged model) is the average number of correctly classified elements over all elements. These metrics are determined as follows:

\[
\text{Precision} = \frac{TP + TN}{TP + TN + FN + FP}. \tag{5}
\]

\[
\text{Recall} = \frac{TP + TN}{2FN + FP}. \tag{6}
\]

The weighted harmonic mean of precision and recall (F-score) was also measured. In addition to precision and recall that were calculated for the UML class diagram conceptual model, an additional metric was required for the state diagrams. This metric is needed so that not only the correct addition or removal of an element from the final conceptual model is calculated, but also the correct transition from one state to the other is also considered. This is because in some cases the correct state exists in the final conceptual model, but the transitions from/to that element are not proper. The
The metric used for this purpose is named Accuracy and is measured by calculating the total number of correct transitions over all of the correct (CT), incorrect (IT) and missing transitions (MT):

$$\text{Accuracy} = \frac{CT}{IT + CT + MT}.$$  \hfill (7)

The experiments showed that the degree of conflict between the viewpoints has a direct influence on precision, recall and accuracy of the final conceptual models. As it can be seen in Figure 7, the precision and recall in the first experiment decreases as the degree of conflict between the viewpoints increases; however, as the degree of conflict decreases, these two measures increase. Analogously in the second experiment, accuracy decreases as a consequence of conflict between the viewpoints, which gradually rises as conflict is resolved. Furthermore, Figure 7 shows that in the final epoch of both experiments, the viewpoints were able to reach a consensus through the employment of the proposed framework and develop an acceptable final conceptual model (viewpoint convergence).

The other aspect of the system that requires evaluation is the belief and proposition recommender modules. In order to evaluate these two modules, the number of correct recommendations over the total number of recommendations made by each recommender in each epoch was considered as a benchmark for the performance of the modules; therefore, the definition of correctness is important. In the proposition recommender module, if the proposed element existed in the reference conceptual model, the recommendation was considered as correct. In the belief recommendation module if the degree of belief recommended to an item conveyed the correct decision (it being either disbelief or belief), it was considered as correct. Furthermore, if the recommended belief consisted of more belief/disbelief values as opposed to uncertainty it was also considered to be strong. Figures 8 and 9 show the performance of the belief and proposition recommender modules, respectively. The results show that both recommenders have a positive performance and effect on the overall process. An interesting observation can be made in the behavior of the proposition recommender module in the second experiment around the fifth epoch where the percentage of correct recommendations drop in the fifth epoch. To explain this, if we overlay Figures 9 and 6, it can be seen that as the degree of conflict rises, the accuracy of the recommender module decreases. This effect is not significant in the first experiment since the development and integration of class diagrams does not incorporate the notion of sequence or time; therefore, the complexity involved in detecting the correct proposition for recommendation is reduced. In the second experiment where state diagrams are developed, due to the involvement of sequence and time, a mere recommendation of the existence of a state is not enough and the recommen-
4.2 Framework Usability

In order to evaluate the usability/applicability of the proposed framework under real world scenarios, the participating students were asked to complete the computer system usability questionnaire. CSUQ [10] is an instrument for measuring user satisfaction with computer system usability in the context of scenario based usability studies. This questionnaire has been developed and evaluated by the IBM Corporation. CSUQ is made up of four sub-scales, each consisting of items ranked on a 7-point scale: the overall satisfaction score (OVERALL: all 19 Questions), the system usefulness score (SYSUSE: Questions 1-8), the information quality score (INFOQUAL: Questions 9-15), and the interface quality score (INTERQUAL: Questions 16-18). This questionnaire has been chosen because of its acceptable reliability: a coefficient alpha ($\alpha$) exceeding 0.89 for all of its sub-scales has been proved. Seven-point rating scales (1=totally disagree, 7=totally agree) were used in the questionnaire because they allow three levels of either positive or negative ratings.

The initial results obtained from the evaluators showed that the proposed framework received a score over the average of the scale score (3.5) in all 19 questions (See Figure 10). Only in two of the questions (9 and 11), the system received a value of less than 4. However, these questions are not directly related to the performance of the proposed framework and address the lack of online help and error messaging in the integration client plug-in tool. Further analysis shows that the values assigned to each question in the range of [5, 7] dominate the other ranges and for some questions the percentage is similar to the [3, 4] range, which is also acceptable. From the perspective of the sub-scales, the framework received a significant score. The score of all sub-scales were higher than 4.5. The proposed framework received 4.804, 4.633, 4.745, and 5.277 in the OVERALL, SYSUSE, INFOQUAL, and INTERQUAL sub-scales, respectively.

Other than the 19 questions, the evaluators were asked to point to three of the positive and negative aspects of the system. One of the aspects that was highly liked by the analysts was the possibility of assigning belief to the model elements; however, there were suggestions that a group selection feature be added to the integration client plug-in so that several elements can be selected at once and a similar belief be assigned to them. The two proposed recommender modules were also appreciated by the evaluators. Here, the only drawback reported by the evaluators was that since the

![Figure 7. The Overall Performance of the Proposed Framework.](image)

![Figure 8. The Performance of the Belief Recommender Module.](image)
framework is reliant on a common terminology some times different wordings are used for the same concepts, which causes problems for the recommender modules. To overcome this deficiency a thesauri, WordNet, or even a unique application vocabulary can be incorporated into the system.

Besides the positive and negative aspects of the system, two of the evaluators had specific recommendations for the improvement of the current work. One of the suggestions was to provide all the viewpoints with the possibility to view the merged model after the end of each epoch, and ask them to continue their design on the merged model and file their own conceptual model to speed up convergence. We are currently not sure of the performance of this suggestion, but it can be anticipated that many of the useful design elements of the individual conceptual models that did not appear in the merged model but had the chance of being a part of the final model in the later epochs will be lost in this approach.

The second recommendation was to adopt a similar approach to the constrain and move strategy [1] widely used in robotics. Looking at mechanical systems, it can be observed that the undesirable movement of the moving parts of the system is constrained by a set of bearings arranged in a proper way. Having the constraining parts properly arranged, the actuators can move the system in their desired path. In a mechanical system, the actuators are not concerned with constraining undesirable movements of the objects and are only forcing agents; therefore, a mechanical system can be thought of as constrain (constrain part) and actuators (move part). The idea here is to split the modelers
into two groups. The first group of designers with less experience can act as actuators, while the second group with more experience receive the recommendations from the system and if they deem appropriate adjust and enforce them to the first group. We are quite interested to analyze the suitability of these two suggestions in future work.

5 Concluding Remarks

In this paper, we have evaluated the performance of a belief-theoretic collaborative conceptual model development framework. The model is founded on concepts from belief theory and mainly derives its operators from Subjective logic [8]. It provides features for merging different conceptual models developed collaboratively by various viewpoints, and offers various tools for reasoning and negotiating over the models and hence formally building and measuring consensus among the viewpoints. The proposed framework has been evaluated from two perspectives: effectiveness and usability. The evaluations show that the various aspects of the framework make it suitable for performing the intended tasks. Furthermore, the experts participating in the experiments felt comfortable using and evaluating the system and believed the framework to be well suited for its intentions and expressed their interest in using it in the future.

Our immediate future work is to incorporate a module into the framework so that different viewpoints be free to use dissimilar vocabulary. The module should then enable the viewpoints to identify the possible matches between the employed vocabulary and employ these correspondences in the integration process. Moreover, we are interested in performing controlled experiments to evaluate the suggestions of the two evaluators regarding the replacement of the individual viewpoints specifications with the merged model after each epoch and the adoption of the constrain and move strategy.

6 Acknowledgements

The authors graciously acknowledge the funding from the Atlantic Canada Opportunity Agency (ACOA) through the Atlantic Innovation Fund (AIF) and through grant RGPN 227441 from the National Science and Engineering Research Council of Canada (NSERC) to Dr. Ghorbani.

References