Architecture of a Cooperative Discussion Environment based on Visual Languages

Katrin Gaßner

University of Duisburg, Germany
Dept. of Mathematics and Computer Science
kati@informatik.uni-duisburg.de

Abstract

A discussion support environment is introduced which is based on the use of visual languages as communication media. In the intended face-to-face scenario, a central big interactive display is a common focus. The discussion members can work cooperatively in joint workspaces based on a technically distributed system. Visual languages are the means to build external representations for increasing the plausibility and the sustainability of a discussion. They support the collection, reflection and summarisation of contributions as well as discussion phases such as brainstorming or the exploration of certain subjects. System triggered mapping of cards between workspaces is to support the smooth and flexible integration of the different discussion phases. Different visual languages allow for taking multiple perspectives on topics at convenient points of time. The dependencies of mapped cards are maintained in node clusters.

Keywords Discussion, Architecture, Visualisation, Visual Languages, Cooperation, Documentation, Knowledge Management

1 Discussion Support

Discussions can be viewed as integral parts of learning because they integrate several important learning strategies in a natural manner. For example, explanations one has to offer to others are comparable to "self explanation" (VanLehn, Jones & Chi, 1992). Critical inquiry is an important strategy to construct new knowledge (Scardamelia, Bereiter, Brett, Burtis, Calhoun & Smith Lea, 1992) for which argumentation in general and particularly scientific argumentation (Suthers, Toth & Weiner, 1997) are relevant methods. Tools that support thinking like the construction of knowledge are called cognitive tools by Jonassen (1992) and Lajoie (1993). The need to express or explain complex or complicated circumstances makes it useful to represent and restructure contributions externally. Therefore, visualisation obtains an important role during the discussion process (Suthers, 1999; Zhang, 1997, Carlson, 1995).

The acceptance and therefore the relevance of discussion results depends strongly on a good quality of the discussion process and its results. This quality can hardly be measured objectively. How the discussion results are accepted subjectively by the members depends e.g. on the depth of argumentation and the attribution of expertise to people. Also social behaviour is important. Dominance only based on power probably plays a negative role whereas an equal participation of the members is rather positive. These aspects are subsumed here by the term *plausibility of a discussion.*

Viewing discussions as knowledge construction leads to another problem which is the *sustainability* of the discussion results. So far discussions are only conducted verbally, the results are non-permanent and hence volatile. Discussion support brings added value by storing the written discussion contributions in order to make them accessible for later use. Thus, together with the appropriateness of discussions to elicit implicit knowledge, discussion support can be a means of a knowledge management process. Generally speaking, to reach acceptance by the users, the advantage of increased plausibility and sustainability has to compensate for the additional effort for the external representation of contributions.

The discussion support system is intended to fill the gap between speech and documentation. Visual languages bridge that gap by supporting the discussion flow on one hand (plausibility) and using the externalised contributions themselves as documentation (sustainability).

The enriched discussion scenario emanates from a big interactive computer display that shows a reference application as a focus for the whole group. Members can work with their individual computers that run also a discussion support application. This setting includes the advantage to use joint and private workspaces in parallel. Because of the joint workspaces, this is not restricted to the face-to-face situation. Working face-to-face, it is also possible to work directly with the board application and to hand round a wireless keyboard as remote input device for the board. The intended face-to-face scenario does not invent a new working situation but enriches discussions as they typically take place.

2 Visual languages

Visual language are mainly used for four purposes: 1) Visual programming and visualisation of programs (Myers, 1987) 2) Learning support (Mandl & Fischer, 2000) 3) Knowledge elicitation (Gaines, 1993) 4) Communication support by externalisation of discourse contributions and structures (Streitz et al., 1992; Scardamelia et al., 1992; Buckingham Shum & Hammond, 1994). Independent of the field of application, visual languages are in the majority of cases used to clarify structures and thus coherences. Such desire for clarifications contains the goal to understand something new, to create knowledge or meaning.

But what are visual languages? Kremer (1998) says „A visual language, ..., is any form of communication that relies on two- or three-dimensional graphics ...“. There is no restriction on the kind of representation but at least the demand for a communication process. Myers (1987) takes another view: „‘Visual Languages‘ refer to all systems that use graphics,...“. This view seems very much influenced by his working field which is visual programming.

The view that is represented in this paper is that the term visual language refers to two central aspects: On one hand the characteristic to be a language and on the other hand the visualisation as a decisive feature. To call them languages makes sense so far they can be used to build expressions. Consistent to Lakin (1990) visual languages are primarily communication media. They enrich and influence discussions by external representations (external memory) that are stimuli for the discussion. In our approach, the means for such external representations are visual languages that are based on cards (Hoppe, Gaßner, Mühlenbrock & Tewissen, 2000). Two general types of cards exist:

- Content cards as container for contributions.
- Connector cards for relations.
  Connector cards potentially offer labelled links to connect content cards.

The visual languages (Figure 1) themselves are not used to express content, but types of contributions and their relations. Card types (content cards and connector cards), visualised via shapes, symbols and colours are the means for expressions that are graphs and the formal part of the representation. The written contributions are the informal part. A reasonable use of the card types can not be checked by the system. But the symbols shall induce useful categorisation of the contributions whereby the structure of the contributions is increased. Because these kinds of visual languages usually have a restricted expressiveness, they standardise the possible expressions (Gaßner & Hoppe, 2000). This allows for combining methods with the visualisation again.

Visual languages are used in workspaces. The workspaces reflect the collaboration that takes place outside the system. For example, because not every contribution should be noted it has to be co-ordinated in the group which contributions are relevant. A pre-test (Gaßner & Hoppe, 2000) produced evidence that it is important to trigger the point of time when collection, reflection and summarisation phases should be passed. That seems to be even central for an efficient integration of the external representation into the discussion.

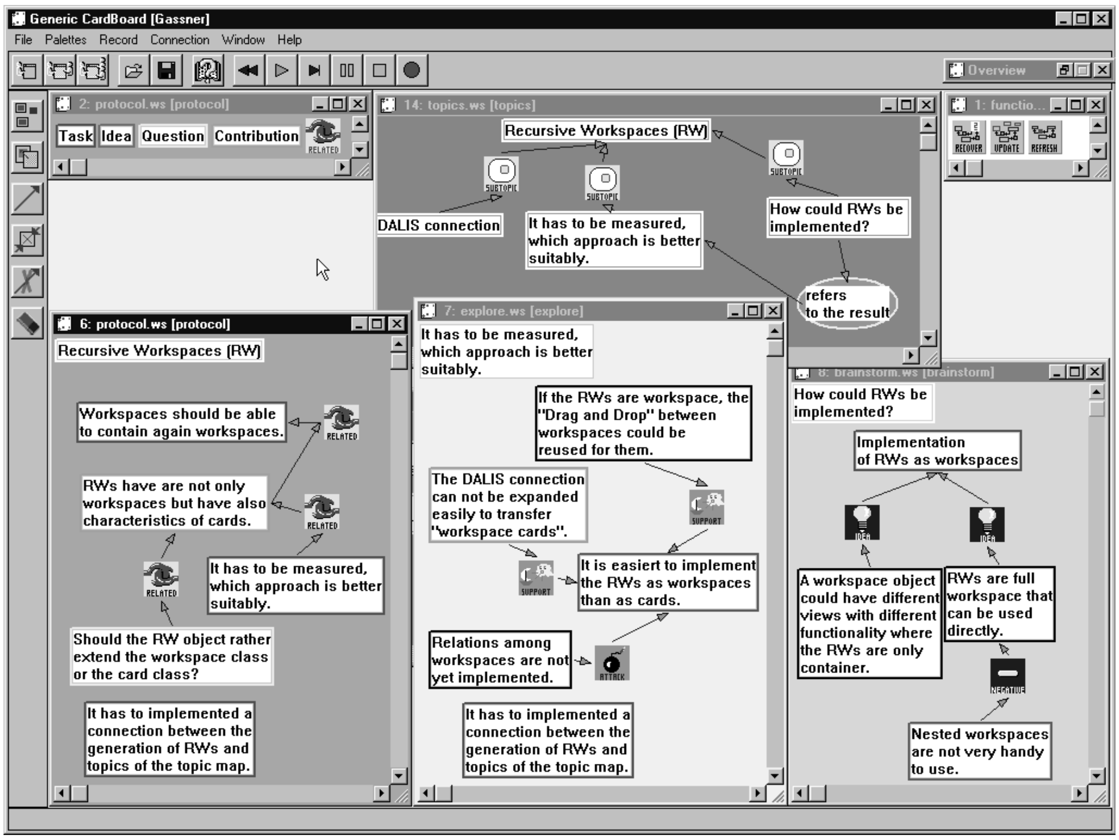
3 Discussion model

A goal oriented communication that includes external representations typically passes collection, reflection and summarisation phases (Gaßner, 2000). This hypothesis is based on a pre-test (Gaßner & Hoppe, 2000), on results concerning scientific argumentation (Carlson, 1995), it refers to the SEPIA system that supports cooperative writing (Streitz et. al., 1992), and to methods like Meta-Plan.

Summarisation can be viewed as one aspect of the reflection phase but here, under the precondition of using visual languages, they are distinguished: Reflection rather signifies restructuring cards and adding relations. That can be interpreted as the construction of the discussion space. Differing from that, summarisation rather means abstraction or to draw conclusions.

For the presented discussion support system, these phases are at first supported by the work with the digital card representation. In principle, creation, deletion, modification of cards and relations as well as restructuring them are means for reflecting and collecting information.

Besides these phases of collection, reflection and summarisation, it has to be considered that a discussion also follows different methods and goals. This could be for example argumentation, brainstorming, elicitation, planning or decision making. The visual languages should supplement each other: argumentation is needed in order to make a decision or to brainstorm to get an alternative for a decision. For example, several ideas have occurred during a brainstorm. One of these ideas is to be explored to get a more detailed understanding of problems that might be caused by that idea. In order to support the discussion, the system should generate input for related workspaces to *prepare* further work. Such preparation demands for further discussions but the appropriate time can be chosen by the user and the aspect is not lost.



Recursive Workspaces (RW)

DALIS connection

It has to be measured, which approach is better suitably.

How could RWs be implemented?

refers to the result

Recursive Workspaces (RW)

Workspaces should be able to contain again workspaces.

RWs have are not only workspaces but have also characteristics of cards.

It has to be measured, which approach is better suitably.

Should the RW object rather extend the workspace class or the card class?

It has to implemented a connection between the generation of RWs and topics of the topic map.

It has to be measured, which approach is better suitably.

If the RWs are workspace, the "Drag and Drop" between workspaces could be reused for them.

The DALIS connection can not be expanded easily to transfer "workspace cards".

It is easiert to implement the RWs as workspaces than as cards.

Relations among workspaces are not yet implemented.

It has to implemented a connection between the generation of RWs and topics of the topic map.

How could RWs be implemented?

Implementation of RWs as workspaces

A workspace object could have different views with different functionality where the RWs are only container.

RWs are full workspace that can be used directly.

Nested workspaces are not very handy to use.

Figure 1: The discussion support interface contains in this example one protocol (left), one exploration (mid) and one brainstorming (right) workspace at the bottom. Left on the top a protocol palette is visible for the currently used workspace type. Top on the right the function palette is visible. Top in the middle a topic workspace is shown.

It is assumed that the discussion phases are *not* passed one after the other. That means that every workspace type can be used as a starting point and that there is no assumption about a sequence and frequency of the used visual languages. The possibility to work with different visual languages allows for taking *multiple perspective* on a subject. In order to implement a general discussion support system, a number of such perspectives has been developed (Gaßner, 2000):

Protocol: This visual language supports the development of a commented agenda. On one hand it pre-structures the discussion whereby contributions are collected an structured. On the other hand it is to include results of other phases. This is supported by an automatism that will be introduced in the following chapter. Adding results from other workspaces supports the summarisation of a discussion. Overall, the protocol shall emphasise the summarisation. It contains *Task*, *Question*, *Contribution* as content cards and the general connector card: *related*.

Exploration: This visual language is designed to explore topics. It offers the biggest set of cards (*Subject*, *Fact*, *Estimation*, *Personal context*, *Result*, *Idea*) and relations (*Support*, *Solve*, *Constrain*, *Cause*, *Explain*, *Attack*). The exploration language is designed to construct and explore the discussion space. Thus it is both a collection of ideas, interests, facts, etc. and their reflection.

Brainstorming: With the brainstorming language AND/OR-like graphs can be built. Single options can be ranked and assessed. The brainstorming is similar to gIBIS (Conklin & Begemann, 1987). It can be used on one hand to brainstorming options or ideas. Additionally, the ranking can lead to decisions. Thus, the brainstorming connects all three phases: the collection, the reflection and the summarisation.

Figure 1 shows the discussion support interface. At the bottom there are three workspaces, left the protocol workspace, mid an exploration workspace and right a brainstorming workspace. They contain a prepared example concerning the question how recursive workspace can be introduced to the discussion support. Each of these three workspaces contains exactly one topic card that is always the card on the top left of the workspace. In this figure the cards with text input are content cards where the default text together with the colour mark the card type. The symbols are used as connector cards.

Additional visual languages are to be implemented which interpret structures and processes that are otherwise difficult to recognise with the means of the previous visual languages. These perspectives are generated automatically by the DALIS system (cf. chapter 4).

Topics: The topic workspace represents the topics of all other workspaces. Each workspace contains exactly one topic card (*topic*). The topic view collects all topic cards from the overall discussion and combines them as a topic map. The *subtopic* relation should be generated automatically based on a recursive workspace structure. Other relations can be freely named as appropriate. The topic workspace produces an abstract view on the discussion and thus a summarisation.

Arguments: Argument workspaces filter arguments concerning one topic from the exploration and the brainstorming workspaces and restructure them in a way to make the conclusions and the premises of the argument visible. The argument language is a subset of the exploration language. Due to the re-structuring it is a kind of reflection and supports elicitation of full arguments. Moreover, it documents the argumentation for further use.

Pro/Contra: The pro/contra workspace filters the exploration workspace concerning the attack and support relation and re-structure the contributions in tables. This representation enables seeing gaps in the argumentation. Hence it is a means for reflection in terms of elicitation.

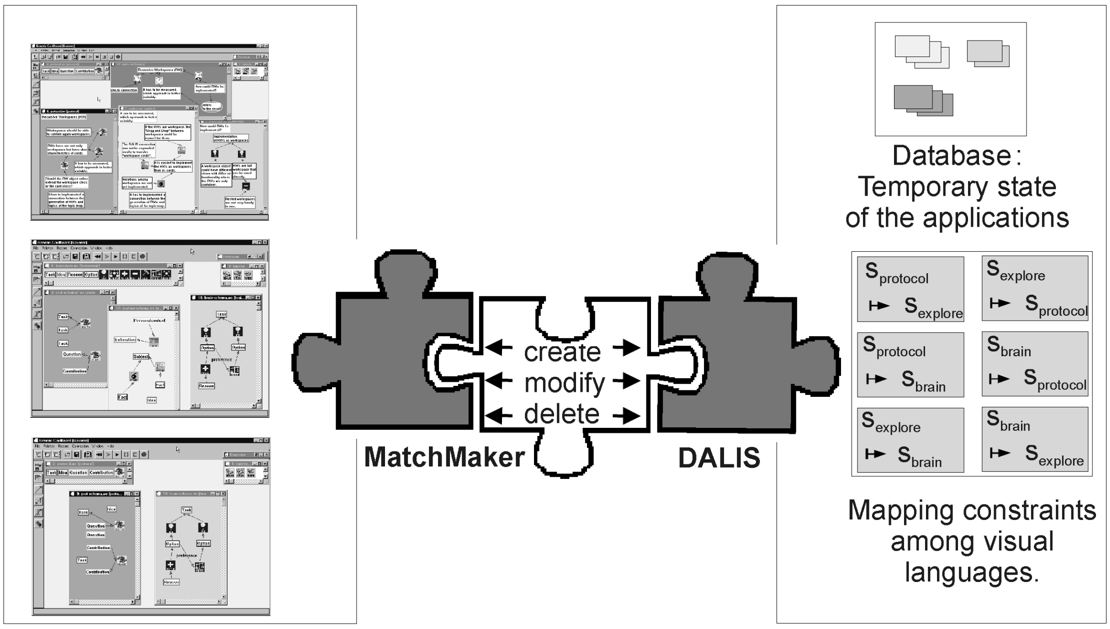
4 The architecture to integrate multiple perspectives on a discussion

The basic platform for the discussion support system is combined of three subsystems:

1. The *CardBoard environment* offers the functions to handle visual languages (Tewissen, 1996; Gaßner, Tewissen, Mühlenbrock, Loesch & Hoppe, 1998; Hoppe et al., 2000) The CardBoard organises the cooperative work of a discussion group in workspaces. Figure 1 shows a screenshot of the CardBoard interface where different workspaces have been generated. Workspaces are either private (only the owner has access) or joint (input media for the whole group). Each workspace is assigned to exactly one visual language. Palettes give an overview on the available card and link types.
2. The MatchMaker server (Tewissen, Baloian, Hoppe & Reimberg, 2000) synchronises (couple) applications temporarily and objectwise. User interface events are distributed among coupled applications. Because of this replication each coupled application keeps the current data of the communication process after leaving a coupled session. Furthermore, MatchMaker is the connection between the CardBoard and the DALIS system.
3. With the DALIS system it is possible to add an operational semantics to the visual languages (Mühlenbrock, Tewissen & Hoppe, 1997; Gaßner et al., 1998; Mühlenbrock & Hoppe, 1999). DALIS monitors the temporay state of cards and relations and reacts on constellations that have to be defined in the operational model. In order to implement a bi-directional communication between DALIS and the MatchMaker server, an interface realises a communication via *create*, *modify* and *delete* primitives which describe the changes that have been performed (from MatchMaker to DALIS) or which have to be performed (from DALIS to MatchMaker) in the application.

Figure 2 shows a schema of the subsystems. On the left, three applications with different workspaces are symbolised. They communicate via the MatchMaker server during the work in joint workspaces. Additionally, MatchMaker sends all creation, modification and deletion actions concerning cards and relations to DALIS. The small rectangles on the right symbolise the temporary state of each visual workspace of any application. The rectangles underneath represent the operational models. For each language it can be defined which reaction is caused by the creation of cards or links. Based on these models, DALIS can trigger changes in the applications.

The basic DALIS architecture only monitors the temporary state of workspaces. But this is not sufficient for the discussion support, because workspaces are maybe closed that would cause the loss of data because of the temporary state description. This is solved by maintaining data that is persistent for the whole discussion. In order to re-open closed workspaces by the users, a handy mechanism has to be implemented that allows for selecting workspaces. Such a mechanism will be given by the topic workspace. When a topic card is selected, all workspaces concerning this topic will be opened.



Database: Temporary state of the applications
$S_{protocol} \mapsto S_{explore}$
$S_{explore} \mapsto S_{protocol}$
$S_{protocol} \mapsto S_{brain}$
$S_{brain} \mapsto S_{protocol}$
$S_{explore} \mapsto S_{brain}$
$S_{brain} \mapsto S_{explore}$
Mapping constraints among visual languages.
create
modify
delete
MatchMaker
DALIS

Figure 2:DALIS and MatchMaker communicate via the *create*, *modify* and *delete* primitives in both directions. DALIS maintains the temporary states of the applications and the operational models that can trigger the communication. Left, the applications of multiple users interact directly via MatchMaker when joint workspaces are used.

Based on the discussion model in chapter 3, the architecture for the discussion support follows two assumptions:

1. Discussions integrate different phases.
2. Using a couple of visual languages increases the clarity of each single language and has the advantage to use methods at all and to focus on one method at a time.

Consequentially, two mechanisms have to be implemented for the overall system. Because discussion phases directly migrate to others that re-use some already given input under a different perspective, one mechanism has to realise the *mapping* among visual language concerning constraints. Additionally, because the reused contributions still belong together, the second mechanism are *node clusters* which maintain the connections among related cards.

Mapping

Mapping means that the creation of a new card in one workspace potentially causes the automatic generation of corresponding cards for other visual languages. The same holds for relations but the mapping for relations is not implemented yet. Such mapping is not restricted to single cards but can also be used for structures.

Figure 3 shows concrete mapping constraints as implemented for the discussion support. In the left column the card types of the protocol language are represented, in the right column the card types of the exploration language. The arrows represent between which card types a relation exists and which is the direction of the mapping.

Assumptions about the flow of a discussion are the background of the mapping. Even though during a protocol phase a task is expected to be more general than a question, both could be explored in the exploration space. Vice versa the subject card is mapped only to the task card because no assumption can be made that the subject is a question. Contributions are intended to be used for general statements. It is assumed that they are in most cases "on the fly" add-ons for further discussions, no results or facts. Therefore, they are mapped to estimations which could be for example opinions. Only the result card is mapped back to the protocol. Facts and personal context cards reside parts of the exploration. Idea cards are not mapped anywhere so far. They are used as containers for later use.

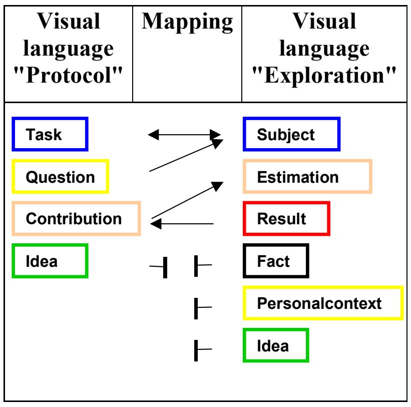


Figure 3: Mapping constraints between the protocol and the exploration language.

translate(protocol,task,explore,subject,content).
translate(protocol,question,explore, subject,content).
translate(protocol,contribution,explore,estimation,content).

translate(explore,subject,protocol,task,content).
translate(explore,result,protocol,contribution,content).

Figure 4: In DALIS the constraints are represented in terms of Prolog facts.

In order to prevent mapping cycles there are no generation chains implemented. That means the generation of one card only causes the generation of these cards which directly derive from the mapping rules in one step. Hence, that mapping is in a logical sense not sound. But since the aim is to support a discussion not to determine it, this incompleteness is accepted. Whenever a user wants to get more contributions from one workspace to another it can be done simply by copy and paste actions.

Mapping is triggered by the DALIS system based on Prolog facts as mentioned in Figure 4. The facts have five parameters: 1) The type of the visual language where a card has been generated. 2) The card type that has been generated. 3) The goal language. 4) The new card type. 5) The last parameter represents if the original card was a content card or a connector card. One visual language can have mapping constraint to every other visual language.

Node Clusters

In case that one contribution is automatically mapped to other workspaces, these cards still belong together. There are physically several cards with the same content but of a different type. These physically different cards are maintained by one node cluster. A node cluster becomes important if one of the these cards is modified. For example, the protocol workspace contains a question. This card is mapped to the appropriate exploration workspace. Whenever it is decided during the discussion that the question is maybe not posed in a suitable manner, it might be changed in the exploration space.

Such change should be reflected for the other cards in the node cluster. Therefore, a node cluster contains a state information about deletion or modification for each card of a cluster. This is internally solved by states for each card of a node cluster. These internal states are visible for the users with the aid of temporary state marker cards. Based on these markers, the user can decide separately for each node of the cluster if he or she wants to upgrade the content or not. If it is decided to deny the upgrade, the changed node is taken out of the cluster. Therefore, the clusters become only smaller during the discussion. In case a cluster still only contains one node the cluster information is deleted. The background idea for that kind of clusters is that the mapping shall only support the starting point of further work with other visual languages. The more the discussion precedes, the more the content of the corresponding workspaces diverge.

Figure 5 symbolises three visual languages as horizontal planes. Each of them contains a different graph structure where nodes that belong to a cluster are connected by a vertical line. The bottom plane symbolises the node cluster.

In DALIS, a cluster is a database entry as *db(cluster(Topic, ClusterId, ClNodeList, Content))*. Each cluster is associated with a topic and an id. The content is the same for all nodes of the node cluster. In case a modification is accepted by a user, this node will be removed from the cluster. Each node of the node cluster is represented in the *ClNodeList* in terms of list entries as *spec(Language, SemCardType, State)*. *State* contains the information if one of the cluster nodes has been changed. Two state markers exist: changed and deleted.

The screenshot in Figure 1 has shown one palette, above on the right, that contains function cards. They are used for the handling of the state markers. Moving them into one workspace forces a DALIS reaction. Three function cards, *recover*, *update* and *refresh* have been implemented. Recover removes all state markers in the respective workspace. The contents of the cards stay as they are. Resulting, the concerned nodes are erased from the corresponding node clusters. Update removes the state markers, too. But in this case, the content is adapted to the change or the card is deleted. Of course, recover cards and update cards can be moved on the state marker that should be effected and do not only work for the whole workspace. The refresh function is needed because of a former design decision wherein it has been favoured not to show system-generated cards immediately. This would cause sudden and on the first sight unmotivated changes for the user. Therefore, in this version the user is responsible to do a refresh independently and at a convenient point of time.

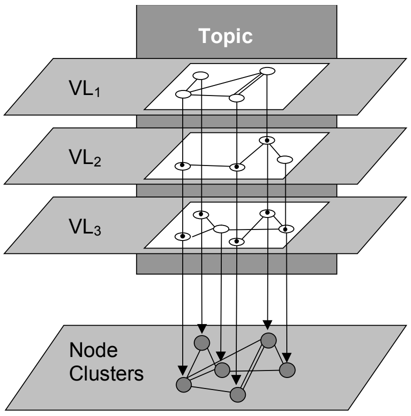


Figure 5: Node clusters maintain related cards of different visual languages.

5 Summary

The aim of this paper is to present an approach to support discussions during multiple phases in terms of structured visualisations of contributions. The protocoling, brainstorming and exploration language has been designed to support discussion phases. Some other visual languages interpret (pro/contra), filter (argumentation) or summarise (topics) the available information based on the card and relation types of the previous mentioned languages. Each language offers a certain set of content cards and relation types to support a specific "intended" semantics which constrains the structure of the developed graphs.

It is assumed that the discussion phases are *not* passed in a pre-defined sequence. This means that each workspace type can be used as a starting point and that any visual language can be used at any time. To discus existing contributions under another perspective, they are re-used in different workspaces. In order to prepare this kind of re-use, the system generates corresponding cards for the related workspaces automatically. Such a preparation, on one hand, simply reduces work but, on the other hand, it is a means to decrease cognitive load in a certain discussion situation: Prepared elements stand as reminders for open discussion points.

A discussion model specifies constraints for the card and relation dependencies among perspectives. These constraints trigger the generation of several cards in different perspectives and hence different types. After their generation, these cards have the same content and belong together. To maintain the dependencies, node clusters have been implemented. Future work will include the implementation of recursive workspaces and an extension for structure transformation between workspaces.

Literatur

Buckingham Shum, S. & Hammond, N. (1994). Argumentation-based design rationale: What use at what cost?. *International Journal Human-Computer Studies*, *40*, 603-652.

Carlson, P. A. (1995). BROCA: A computerized environment for mediating scientific reasoning through writing. In *Journal of Universal Computer Science*, *1(8)*, 571-590.

Conklin, J. & Begemann, M. L. (1987). gIBIS: A hypertext tool for team design deliberation. In *Proceedings of Hypertext'87* (pp. 247-251). Chapel Hill (North Carolina).

Gaines, B. R. & Shaw, M. L. G.(1993). Knowledge acquisition tools based on personal construct psychology. In *The knowledge engineering review*, *8*, 49-85.

GAßNER, K. (2000). to appear: Entwicklung einer kooperativen Diskussionsumgebung mit multiplen Sichten.Workshop Intelligente Lehr - und Lernsystem of the German society of computer sciences. (http://lls.informatik.uni-oldenburg.de/Fachgruppe_ILLS/)

GAßNER, K. & HOPPE, H. U. (2000). Visuelle Sprachen als Grundlage kooperativer Diskussionsprozesse (Visual languages as a base for cooperative discussion processes). In Mandl, H. & Fischer, F. (Eds.), *Wissen sichtbar machen, Wissensmanagement mit Mappingtechniken* (pp. 93–118). Göttingen: Hogrefe.

GAßNER, K., TEWISSEN, F., MÜHLENBROCK, M., LOESCH, A. & HOPPE, H. U. (1998). Intelligently supported collaborative learning environments based on visual languages: a generic approach. In Darses, F. & Zaraté, P. (Eds..), *Proceedings of 3rd International Conference on the Design of Cooperative Systems* (pp. 47-55). Cannes (Frankreich).

HOPPE, H. U., GAßNER, K., MÜHLENBROCK, M. & TEWISSEN, F. (2000). Distributed visual language environments for cooperation and learning: Applications and intelligent support. In *Group Decision and Negotiation*, *9(3)*, 205-220.

JONASSEN, D. H. (1992). What are cognitive tools?. In Kommers, P. A. M., Jonassen, D. H. & Mayes, J. T. (Eds.), *Cognitive tools for learning*. Berlin: Springer Verlag.

KREMER, R. (1998). Visual Languages for Knowledge Representation. Eleventh Workshop on Knowledge Acquisition, Modeling and Management KAW-98. Banff, Canada. *http://ksi.cpsc.ucalgary.ca/KAW/KAW98/kremer*

LAKIN, F. (1990). Visual languages for cooperation: A performing medium approach to systems for cooperative work. In Galegher, J. & Kraut, R. & Egido, C. (Eds.), *Intellectual teamwork* (pp. 453-488). Hillsdale (NJ): Lawrence Erlbaum Associates.

LAJOIE, S. P. (1993). Computer environments as cognitive tools for enhancing learning. In Lajoie, S.P. & Derry, S.J. *Computers as cognitive tools*. Hillsdale: Lawrence Erlbaum Associates.

MANDL, H. & FISCHER, F. (2000). *Wissen sichtbar machen. Wissensmanagement mit Mappping-Techniken.* Göttingen: Hogrefe.

MÜHLENBROCK, M. & HOPPE, H. U. (1999). Computer supported interaction analysis of group problem solving. In J. Roschelle & Hoadley, C. (Eds.), *Proceedings of the Conference on Computer Supported Collaborative Learning CSCL-99* (pp. 398-405). Palo Alto (CA).

MÜHLENBROCK, M., TEWISSEN, F. & HOPPE, H. U. (1997). A framework system for intelligent support in open distributed learning environments. In du Boulay, B. & Mizoguchi, R. (Eds.), *Artificial intelligence in education: Knowledge and media in learning systems* (pp. 191-198). Amsterdam (The Netherlands): IOS Press.

MYERS, B. A. (1987). Taxonomies of Visual Programming and Program Visualization. In *Journal of Visual Languages and Computing 1*, 97-123.

SCARDAMELIA, M., BEREITER, C., BRETT, C., BURTIS, P. J., CALHOUN, C. & SMITH LEA, N. (1992). Educational applications of a networked communal database. *Interactive Learning Environments, 2(1)*, 45-71.

STREITZ, N., HAAKE, J., HANNEMANN, J., LEMKE, A., SCHULER, W., SCHÜTT, H. & THÜRING, M. (1992). SEPIA: A cooperative hypermedia authoring environment. In *Proceedings of the 4th ACM Conference on Hypertext* (pp. 11-22). Milan (Italien).

SUTHERS, D. D. (1999). Representational bias as guidance for learning interactions: A research agenda. In Lajoie, S. P. & Vivet, M. (Eds.), *Artificial intelligence in education* (pp. 121-128). Amsterdam: IOS Press.

SUTHERS, D., TOTH, E. E. & WEINER, A. (1997). An integrated approach to implementing collaborative inquiry in the classroom. In *Proceedings of Computer Supported Collaborative Learning*. Toronto (Canada).

TEWISSEN, F. (1996). *Begriffsnetze als Basis für ein System zur kooperativen Lösung physikalischer Aufgabenstellungen*. Master Thesis, University of Duisburg, Dept. of Mathematics / Computer Science.

TEWISSEN, F, BALOIAN, N., HOPPE, H. U. & REIMBERG, E. (2000). TO APPEAR. „MatchMaker" - Synchronising objects in replicated software-architectures. In *Proceedings of Cyted-Ritos International Workshop on Groupware (CRIWG) 2000.* Madeira (Portugal).

VANLEHN, K., JONES, R. M. & CHI, M. T. H. (1992). A model of the self-explanation effect. *The Journal of the Learning Sciences, 2*, 1-59.

ZHANG, J. (1997). The nature of external presentations in problem solving. In *Cognitive Science, 21(2)*, 179-217.