RWTH Aachen University

LuFG Informatik 9: Learning Technologies
Chair for Communication Science

Master thesis

Modeling Open Discourse in a Structured Collaborative Discussion System

November 18, 2015

Supervisors:
Prof. Dr. Ulrik Schroeder
Prof. Dr. Martina Ziefle
Advisors:
Dr. André Calero Valdez
Christoph Greven, M.Sc.

Johannes Karoff Matriculation number 291311
johannes.karoff@rwth-aachen.de Informatik, M.Sc.

Eidesstattliche Versicherung

______________________________	______________________________
Name, Vorname	Matrikelnummer

Ich versichere hiermit an Eides Statt, dass ich die vorliegende Arbeit/Bachelorarbeit/ Masterarbeit* mit dem Titel

____

____

____

selbständig und ohne unzulässige fremde Hilfe erbracht habe. Ich habe keine anderen als die angegebenen Quellen und Hilfsmittel benutzt. Für den Fall, dass die Arbeit zusätzlich auf einem Datenträger eingereicht wird, erkläre ich, dass die schriftliche und die elektronische Form vollständig übereinstimmen. Die Arbeit hat in gleicher oder ähnlicher Form noch keiner Prüfungsbehörde vorgelegen.

______________________________	______________________________
Ort, Datum	Unterschrift

*Nichtzutreffendes bitte streichen

Belehrung:

§ 156 StGB: Falsche Versicherung an Eides Statt

Wer vor einer zur Abnahme einer Versicherung an Eides Statt zuständigen Behörde eine solche Versicherung falsch abgibt oder unter Berufung auf eine solche Versicherung falsch aussagt, wird mit Freiheitsstrafe bis zu drei Jahren oder mit Geldstrafe bestraft.

§ 161 StGB: Fahrlässiger Falscheid; fahrlässige falsche Versicherung an Eides Statt

(1) Wenn eine der in den §§ 154 bis 156 bezeichneten Handlungen aus Fahrlässigkeit begangen worden ist, so tritt Freiheitsstrafe bis zu einem Jahr oder Geldstrafe ein.

(2) Straflosigkeit tritt ein, wenn der Täter die falsche Angabe rechtzeitig berichtigt. Die Vorschriften des § 158 Abs. 2 und 3 gelten entsprechend.

Die vorstehende Belehrung habe ich zur Kenntnis genommen:

______________________________	______________________________
Ort, Datum	Unterschrift

Contents

Abstract Popular online collaboration platforms do not scale for complex problems discussed by multiple participants. The lack of structure in traditional forums does not offer a possibility to link posts. Threaded forums and question-answer systems provide a simple structure for solving straightforward problems, but leave no room for collectively developing a solution to more complex problems. Few web-based argument mapping systems exist and they are rarely used in a broader scope. This thesis proposes the hypergraph-based discussion system *Wust*. It is implemented as a website where users can collaboratively discuss issues, ask questions, contribute arguments and solve problems. It allows to flexibly model complex discussions and facilitates reasoning over single statements as well as gathering arguments for and against the connection between elements. The idea is to maintain a certain structure without rigid constraints on the relations which might have a negative impact on the contribution motivation. Relations between posts can be classified to reflect the meaning of the connection. *Wust* was evaluated qualitatively in two user tests and quantitatively with the aid of a questionnaire. The evaluation yields good results for the usability and the discussion model with flexible classifications and hyperrelations.

1. Introduction

"Traditionally, the process of planning and design has been described as a sequence of first understanding, followed by analysis, synthesis and implementation. This is not a realistic model of the act of design. An alternative understanding ('Second Generation') of the reasoning in the design process and the nature of its problems [. . .] demonstrated that designing is more appropriately understood as a process of argumentation." [RN89]

The World Wide Web has enabled mass communication across the globe. The emergence of internet forums, collaboration platforms, social networks, blogs and messaging protocols has enabled users to share information and to collaboratively solve problems. Sharing information is easy but discussing possible prospects and drawing conclusions becomes difficult with the lack of structure in common online discussion systems. There are popular question-answer systems like StackOverflow or Quora that provide a certain degree of structure, while sacrificing flexibility when developing a solution collaboratively. This does not scale well for multiple participants and a broad spectrum of opinions and evidence.

Figure 1 shows the implications of a highly complex discussion with many contributors. When a lot of people collaboratively discuss about a complex issue and explore the space of possibilities, many posts are created. Thereby, it is easy to lose the overview of the whole discussion because of the sheer amount of posts and the lack of focus and entry points. This can discourage new participants, who might not be able to work their way through the discussion to understand the arguments and to finally contribute to the discussion. The mass of new submissions can easily overwhelm the moderators, who are then not capable of deleting spam and filtering out duplicate posts. If the participants cannot easily get a grasp of the current state of the discussion, redundancy will be created as new contributors do not know what has already been mentioned. This contributes to a feedback loop and as a result will lead to more posts being created. It becomes even more difficult to comprehend the essence of the discussion and to evaluate the different arguments – the overview is lost. Discouraged participants and spam further devalue the whole discussion.

In order to structure and visualize arguments for a certain issue, argument mapping systems have emerged, e.g., Compendium, Debategraph, Belvedere and Rationale. These systems provide rich semantics for structuring arguments in a discussion. There are different theoretical models for argument mapping like the Toulmin model [Tou03] or the IBIS model [KR70] [RN89], which serve as a basis for some popular argument mapping systems. Argument visualization is also mentioned in the context of E-Democracy as one technique for information structuring and large-scale deliberation [Hil09]. Argument mapping systems are not widely available and are commonly only applied for

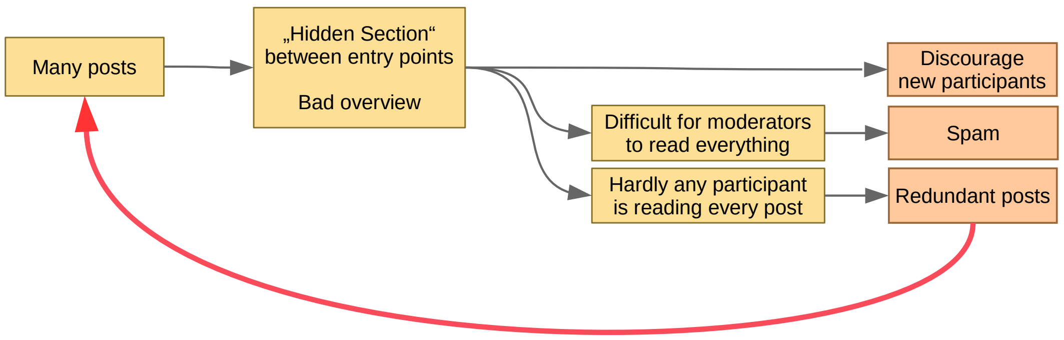

Many posts
„Hidden Section“ between entry points
Bad overview
Discourage new participants
Difficult for moderators to read everything
Spam
Hardly any participant is reading every post
Redundant posts

Figure 1: Implications of a highly complex discussion with many contributors: many posts are created. The lack overview leads to redundancy, spam and might discourage new participants. Redundancy leads to a feedback loop and even more posts are created.

business purposes or in the academic and educational domain. There are only a few web-based systems available and the concept is not widely adopted yet on the internet.

This thesis reports on the conception and the development of a hypergraph-based discussion system named *Wust*. It is implemented as a website to be easily accessible. The main goal is to provide a flexible graph model for capturing structured arguments, collaborative planning and solving complex problems. We also evaluate whether such a system can be usable and whether users can make use of the structure for argumentation. The proposed discussion model aims at providing a better overview of the current state of the discussion and allows to model the argumentation in a more meaningful structure. We expect that the graph representation with explicit relationships between posts makes redundancies more visible and therefore less likely to be created in the first place. Wust was implemented in cooperation with Felix Dietze who reports on quality assurance and the employed collaborative moderation scheme in his master thesis [Die15]. A formative evaluation of the usability and the underlying concepts of the developed discussion system was conducted.

This thesis is organized as follows. In chapter 2, related work is introduced and examined; the presented concepts and discussion systems form the basis of our work. Afterwards, chapter 3 conducts an overview of the approach developed during this thesis. Important concepts of related work are reviewed, and a concept for the discussion system and its underlying discourse model is developed. Section 4 describes the implementation of Wust, which was developed during this thesis and realizes the proposed solution. Thereafter, chapter 5 reports on the formative evaluation of the discussion system, which was tested in two user tests and with the aid of a questionnaire. Prospects for future work and the results of this thesis are discussed in chapter 6.

2. Related work

Internet forums and open collaboration systems have gained increasing popularity for discussing issues online. Furthermore, different theoretical models have been proposed for modeling discussions and mapping arguments, even long before computers were used. Argument mapping systems were implemented for computer-supported modeling of arguments. Popular online collaboration platforms do not provide a structure for arguments, whereas argument mapping systems, that could offer such a structure, are often only used in business or educational contexts.

Section 2.1 reports on the evolution of open collaboration systems and internet forums to introduce the applied concepts. These systems do not provide a proper model to capture the intended structure of the contributions. Therefore, two important theoretical argument models are introduced (section 2.2), which lay the basis for a survey on existing argument mapping tools (section 2.3). Afterwards, in section 2.4, collaborative tagging as a mechanism for indexing content resources is discussed.

2.1. Open Collaboration Systems

This section provides an overview on the evolution of different types of collaboration systems that are popularly used on the internet for discussing issues and gathering information. We take a look at traditional forums and mailing lists as well as more structured forums like threaded discussions or question-answer systems. Furthermore, the form of contributions in Wikipedia and issue trackers for software development are explained.

Traditional Forums

Traditionally, internet forums arrange posts in threads. Users can create new threads by choosing a title and providing a first post. Succeeding posts are then added to the thread in a chronological order.

The strict chronological order makes it difficult to get an overview of large discussions, as the intended structure of the responses is neither captured nor visualized. Most forums offer a possibility to quote text fragments from other posts, in order to reference another post despite its physical distance in the layout. But there is no back link from the quoted post to list the responses. Furthermore, quotes might not capture when the original post was updated and also tend to fill up a lot of space in threads. This structureless collection of posts can work on a small scale, but imagine a thread with a lot of pages: new users seldomly read everything and therefore redundancy will be created, also it is difficult to see how a consensus was reached in the process of the discussion.

Mailing Lists

Mailing lists are commonly used in order to distribute information and discuss questions regarding a certain product or organization. A mailing list is a list of email addresses with the purpose of sending information to a group of recipients. A user can subscribe to a mailing list in order to receive new emails sent to the mailing list. It is possible to respond to incoming emails from the list directly and formatting conventions allow to quote previous emails and inline responses to the statements. At times, people tend to quote the complete message they are responding to and therefore clutter the collection of messages for future readers.

Often a backlog of the correspondence is stored online and can be used as a reference for future readers. This is commonly displayed as a tree of messages, like in threaded forums.

Threaded Forums

A strictly chronological order as in traditional internet forums does not provide an instrument to model how posts are connected. Quoting is an imprecise linking mechanism between posts, as it only works in one direction and the whole spectrum of possibilities distributed across the whole thread is not directly visible. This makes it difficult for new readers to get an overview of the current state of the discussion. Also, there is no way to retroactively understand the rationale for the decisions reached in the discussion.

An alternative to traditional forums are threaded forums, where relationships between posts are visible in a tree of messages. A user can select the post to which he wants to reply, which places the response below the referred post. Commonly, responses are indented to indicate the nesting level in the tree. The tree begins with a thread-opening message and then all responses with their tree of responses are listed – often sorted by quality. This gives too much emphasis to the first response in the list, which will get overpopulated while succeeding responses cannot develop. Social news aggregators often display discussions about linked materials or user contributions in a threaded tree view, e.g., *Reddit*[1] and *HackerNews*[2].

Reddit is a social news and networking platform where users can submit web links and/or text posts. Other users can reply to the submission in a comment section and rate posts by means of up- and downvotes, which influences the order in which they appear. New submissions are categorized under a certain topic or area of interest called *subreddit*. In their study about the evolution of Reddit, Singer et al. describe that the subreddits cover numerous distinct topics and further conclude that "Reddit has transformed itself from a dedicated gateway to the Web content to an increasingly diverse, self-referential community" [Sin+14]. Combined with their finding that self-hosted user contributions typically attract more user comments, this shifts the focus to the comment section of Reddit.

The common visualization of responses via indentation does not work well for deeply nested posts. In order to overcome some limitations of the tree-based structure, Yee and Hearst conducted an evaluation of a column-based tree map visualization with mixed results in [YH05].

In general, the tree representation does not capture the interconnectedness of discussion elements very well, as it only allows one parent per node. Thus, it is impossible to share information throughout multiple isolated discussions or threads. For example, Reddit actually promotes crossposting redundant posts to different subreddits, as it is impossible to group one post under multiple subreddits. This ultimately has the drawback that one thread cannot cooperate knowledge built in another thread, because they cannot be interlinked properly. Furthermore, it is not possible to define the meaning of a relation between posts – reply is a rather vague description.

Question-Answer Systems

With more focus on the encyclopedic value of user contributions in internet forums and with the aim to offer a platform for creating and reading solutions for common problems, question-answer system have attracted a lot of interest. Prominent examples of question-answer websites are *Yahoo! Answers*[3], *StackExchange*[4] and *Quora*[5]

[1]`https://www.reddit.com`
[2]`https://news.ycombinator.com`
[3]`https://answers.yahoo.com/`
[4]`http://stackexchange.com/`
[5]`https://www.quora.com/`

Such systems allow posting of questions and users may then answer a question or comment on a question to request further background information. Each question and each answer has is a list of comments sorted chronologically – like a traditional forum. It is possible to up- or downvote questions and answers to express agreement or to honor quality. Questions on the platform and answers to a question can then be sorted by quality. Furthermore, one of the answers can be marked as the solution for the question. Solutions are highlighted and sorted to the top of the list of answers. This works really well for questions with a straightforward answer, which can easily be tested for validity. But there is no room for structuring arguments, positions and clarifications around answers, which makes it difficult to collaboratively develop a solution for complex problems with a variety of paths that need to be evaluated. The structure-less comment lists for questions and answers do not allow a proper argumentation.

Often, question-answer system employ a community-driven moderation scheme. Users are awarded reputation points for positively perceived contributions and may also lose reputation if that is not the case. Thereby, users can earn different levels of privileges and rights on the website. For example, if the user has a certain amount of reputation, he may edit posts from other users or may be able to delete posts and close questions.

StackExchange is a network of question-answer websites, where each website covers a specific topic. For example, *StackOverflow*[6] is an instance of StackExchange focusing on questions and problems related to software development and serves as a very popular repository of solved problems for developers. The scope of StackOverflow is also stated in their guidelines on what should be asked: Questions should be scoped, not be open-ended and only certain subjective questions are allowed [Sta]. One reason might be that the simple question-answer model does not provide a way to structure deeply nested discussion. In their observations on StackOverflow, Anderson et al. indicate that questions with their whole set of answers including the knowledge-creation process commonly form the inherent and long-lasting value for future readers [And+12]. Vasilescu et al. analyzed contribution in the R community in the form of mailing list and StackExchange activity and see an activity shift towards StackExchange [Vas+14].

Wikipedia

Wikipedia[7] serves as a daily updated encyclopedia backed by a large community. Except for some protected articles which can only be edited by administrators, community members can generally edit and discuss any article to improve its quality. This leads

[6]`https://stackoverflow.com`
[7]`https://www.wikipedia.org/`

to a high demand for coordination and discussion between multiple editors: references need to be checked, invalid conclusions need to be isolated and vandalism as well as outdated or biased information needs to be detected.

Software Development

As collaborative source code and project management services like GitHub[8] or Bit-Bucket[9] get more popular, issues need to be discussed and resolved in collaboration. These web services offer filehosting with version control of software repositories and commonly offer project-related functionality like team management, statistics, a graphical interface and issue tracking.

Issue trackers are a collection of bugs, questions and feature requests for a certain piece of software. Users can open a new issue describing a problem, an idea or pointing out a bug. It is possible to comment on each issue, so collaborators can ask questions to further investigate the issue or to respond to a question or feature request. Issues can be closed if they are resolved or cannot be fixed or are otherwise inappropriate. In order to display the inter- and intra-connectedness of issues in different projects, GitHub always provides a back-link whenever an issue was mentioned – via link – in another issue. Even though, this provides an overview of connected issues, there is no way to specify how the issues relate to each other. An issue might be caused by another one, or a feature request might be prevented by a bug.

2.2. Argument Model

In philosophy, argument models are used for mapping arguments and their structure with the goal of justifying or falsifying a claim with evidence, premises and reasonable conclusions. Different argument models were developed in the past, this section will introduce two important models which form a basis for existing argument mapping systems. In 1958, the British philosopher Stephen Toulmin identified a basic model for laying out and analyzing practical argument, in order to justify claims with evidence and explicit relations [Tou03] (section 2.2). Rittel and Kunz introduced the term Issue-based Information Systems (IBIS) in 1970 [KR70]. The IBIS model is built to capture the design rationale behind decisions reached through an argumentative process including multiple stakeholders (section 2.2). Both models provide a basis for understanding the rationale behind the argument mapping systems introduced in section 2.3.

[8]`https://github.com/`
[9]`https://bitbucket.org/`

Toulmin Model

In 1958, Toulmin published one of his most influential book *The Uses of Argument* which proposes a simple model for practical arguments for the justification of claims [Tou03]. The argument starts with an initial claim or thesis that should be proven; it can be seen as the root node of a tree. It is possible to attach evidence to the claim, which is the information to support and explain the claim. Furthermore, one can define links between the evidence and the claim, which are described by a warrant to indicate how the claim can be concluded from the evidence.

Toulmin identifies three basic elements present in every argument, which consist of the actual debated claim, the evidence and the relation between the two:

- **Claim** The claim, thesis, or conclusion which should be proven in an argumentative process.
- **Ground** Factual information and evidence data which supports the claim that should be made.
- **Warrant** A statement which describes the relation between the ground and the claim. In order to move from the fact to the conclusion the guarantees from the warrant need to explicated.

These three elements make up the basic Toulmin model, that can be used to reason for a specific claim. It describes how evidence justifies the claim which should be proven and defines how one can assume the claim from the evidential ground. In order to give more useful semantics for attaching supportive arguments for a warrant, for mentioning restrictions and for qualifying the conclusion, Toulmin proposes three additional elements:

- **Backing** Adds supporting arguments to an expressed warrant, that is not convincing by itself or does not seem to be relevant.
- **Rebuttal** To add exceptions and restrictions to the claim which might invalidate it.
- **Qualifier** Used to modify and to express the degree of certainty regarding the actual claim.

The overall model with an example can be seen in figure 2. In order to argue for the claim “Ann’s bedroom is on fire”, the existing fact that “Smoke is pouring from Ann’s bedroom” is taken into account. The ground can be linked to the claim “since smoke is a primary sign of fire”, which, if necessary, can further be backed by describing that “fires generally produce smoke”. The qualifier allows to express the degree of certainty

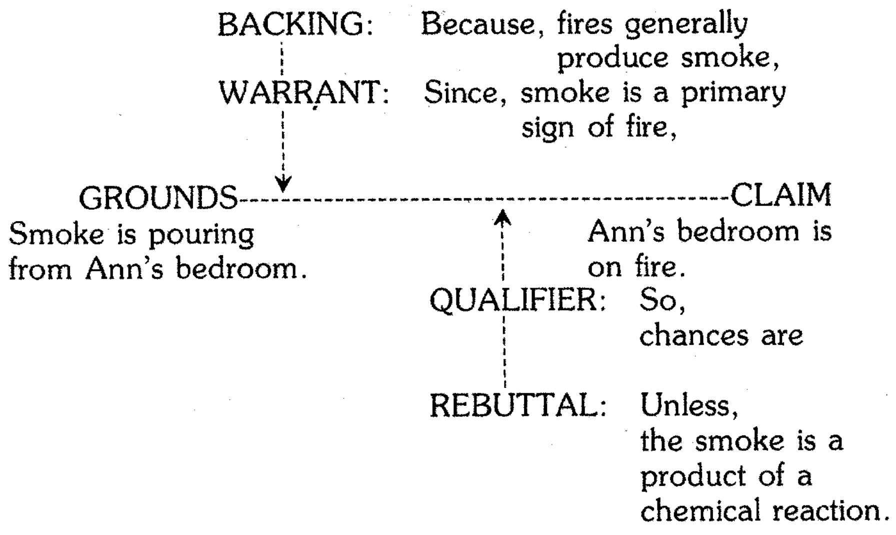

BACKING: Because, fires generally produce smoke,
WARRANT: Since, smoke is a primary sign of fire,
GROUNDS
Smoke is pouring from Ann's bedroom.
CLAIM
Ann's bedroom is on fire.
QUALIFIER: So, chances are
REBUTTAL: Unless, the smoke is a product of a chemical reaction.

Figure 2: The Toulmin model [Kar87]

for the claim and the rebuttal can address that the smoke might also be "a product of a chemical reaction". [Kar87]

The Toulmin model enforces an inherently logical structure, which forces participants to lay out reasons for their claim. Once the claim is identified, evidence needs to be gathered. It helps to explicate the relations between the evidence and the claim; this is helpful for also questioning implicit conclusion, which can be explicated in a warrant if found necessary.

The Toulmin model is targeted at focusing on a single claim, there is no way to link different claims with regards to relatedness or causality.

Issue-based Information Systems

IBIS is a theoretical model with the aim to map dialogues and solve wicked problems including multiple stakeholders. It was developed by Kunz and Rittel in 1970 to provide a scheme for argumentation of design tasks and policy planning: "Issue-Based Information Systems (IBIS) are meant to support coordination and planning of political decision processes. IBIS guides the identification, structuring, and settling of issues raised by problem-solving groups, and provides information pertinent to the discourse" [KR70].

The IBIS model is built for capturing the design rationale, which is the transparent and explicit record of the design decisions reached through argumentation when developing a system. This should ultimately help to solve wicked problems through argumentation. In 1973, Rittel and Webber defined the characteristics of wicked problems in the field of policy planning, which can also be expanded to other domains. Among other characteristics, wicked problems do not have a definitive formulation, there is no stopping rule and solutions cannot be assessed as true or false but rather as good or bad – there is no definite test to check whether a solution is applicable [RW73].

The IBIS model arranges dialogue elements in a graph, where the nodes correspond to the posts of participants and the edges define relations between the posts. It defines the following three types for dialogue elements:

- **Issue/Question** The issues and questions that are open for debate. Typically, issues are formulated as questions about a certain issue or problem.
- **Idea/Position** Ideas are responses to questions. Users can propose solutions for an issue and thereby formulate the space of possibilities.
- **Argument** Arguments can be a pro (supporting) or a contra (opposing) for an idea and can be connected to it.

The basic structure in the IBIS model is a graph. An IBIS discussion commonly starts with an issue or a question about a certain problem that is being debated. Subsequently, it is possible to add ideas or positions to the question that might resolve the issue at hand. It is possible to connect arguments to ideas in order to discuss different viewpoints on the articulated position. One may raise a new issue from any node in the dialogue map. So, the model allows to specify different solutions for an issue and discuss possible arguments for the different solutions. Ideas and issue may only respond to one other node, whereas arguments are allowed to refer to multiple parent nodes. Furthermore, a small set of relations is proposed by IBIS to connect the different dialogue elements, the model is depicted in figure 3. In his report on a ten year long case study, Conklin attests that the IBIS model facilitates collaborative planning while learning the grammar and structure of the IBIS model remains the biggest obstacle for beginners to utilize dialogue mapping [Con03].

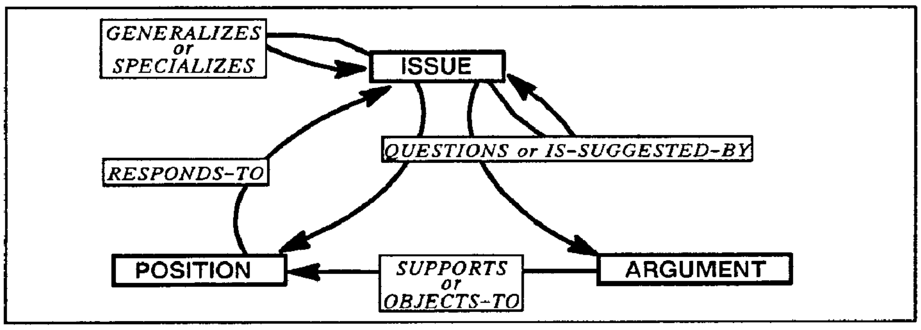


Figure 3: Data model for IBIS [CB87]

Popular question-answer systems like StackOverflow or Quora make use of a similar but less powerful model. As described in the previous section, the question-answer model commonly only structures questions and its answers. Comments under answers and questions are displayed as a list of comments and there is no possibility to link questions across threads. The IBIS model is capable of fully emulating this model. Additionally, it allows to model relations between questions and structuring of arguments for ideas. After publication, paper-based IBIS systems were used until computer programs were developed.

2.3. Argument Mapping Systems

The two presented argument models provide a theoretical foundation for argument mapping systems, which provide an interface for visualizing structured arguments and sometimes involve collaboration capabilities. Some systems aim at improving the users skills in coherent argumentation, others are aimed at planning design tasks. This section provides an overview of available tools and suites that aim for a structured argumentation scheme; most of the freely available tools are standalone software, only a few modern web-based systems exist. Several tools have been developed in this field, e.g., Clip [MT14], Sibyl [Lee90], Cool Modes [Bol+02] [Pin03], Hermes [KP01], Argunet [SVB07], Cohere [Shu08] and Deliberatorium [Kle11]. In the following, six important argument mapping systems and their concepts are described: gIBIS, Compendium, Belvedere, Rationale, Multicentric IBIS and Debategraph.

gIBIS

With advances in technology, gIBIS (graphical-IBIS) was invented, which was the first graphical computer program to model IBIS dialogue maps. In 1987, Conklin and Begeman worked on “a hypertext-based project called Design Journal which is aimed at providing a team of system designers a medium in which all aspects of their work can be computer mediated and supported” [CB87]. This also includes modeling of design decisions and the design rationale. The authors have implemented gIBIS, a graphical tool for collaborative deliberation which is based on the IBIS model: “The IBIS method [. . .] is based on the principle that the design process for complex problems is fundamentally a conversation among stakeholders [. . .] in which they bring their respective expertise and viewpoints to the resolution of design issues” [CB87].

Even though gIBIS is a very early discussion tool, it supports collaboration between different users on a local area network. Users can interactively edit nodes in the discussion and respond to other nodes. Other users are then notified about the changes in the discussion.

The tool further includes a search mechanism to retrieve nodes from the network. Furthermore, gIBIS implemented a linearization of the discussion graph; it is possible to traverse the graph with a structure-aware next button. The graph browser shows the discussion map as a network with nodes and edges. Different node types and relations are differentiated with the help of different shapes and colors. Different zooming levels can be used for displaying different levels of details in the graph – a global and a local view.

Conklin and Begeman observed discussions constructed in gIBIS during their evaluation; it was available for seven months in the MCC Software Technology Program. Within this time frame, 16 people contributed to 21 issue groups with a total amount of 1153 nodes and 1237 links. The authors report that the users were "generally enthusiastic about using the tool, and reported that it imposed a structure on discussions which exposed 'axe grinding, hand waving, and clever rhetoric'." [CB87]

An inherent problem with structured data in a strict model is that ideas need to be fragmented into chunks of information that are connected via relations, therefore gIBIS allows the creation of composite nodes. Non-linear discussion cannot be linearized so there are different paths on which a user might reach a specific node, which is why it is difficult to tell whether the context of the post is explicated sufficiently on all paths. In [CB87], Conklin and Begeman identify three types of collaborative work: (1) substantive (the content of the work), (2) annotative (comments about substance), and (3) procedural (comments about procedures and conventions for use of the medium). A problem when displaying large graphs in a network structure is disorientation. The authors report that the global view of their graph visualization helped to reduce this problem.

Compendium

In the tradition of gIBIS and other graphical tools, Compendium was developed as an IBIS-based system to facilitate solving of design tasks and problems [Shu+06] [Shu07]. Research and development started in the mid-nineties, the system was released as open source in 2009 and development is still ongoing in a new version called Compendium next-generation[10].

[10]`http://www.compendiumng.org/`

Different from gIBIS, it additionally supports user-defined ontologies and also allows using IBIS with its node and link types. A user can create a map which represents a discussion about a certain issue. Furthermore, it supports reuse of nodes by reconnecting, copy pasting and referencing in a new map. Compendium offers all the IBIS node types and some additional types which can be arranged in a graph:

- **Question** A question or issue open for discussion, which usually starts with a key issue for which ideas should be gathered and where arguments should be evaluated.
- **Answer** A possible solution for or answer to a question or issue. This can also be a general idea or a positional statement.
- **Pro** Argument in favor of a position or answer.
- **Con** Argument against a position or answer.
- **Argument** General argument which is neither only supporting nor only opposing a position or an answer.
- **Descision** Mark a decision which was agreed upon. For example, this can be used for marking an idea which was decided as the way to solve an issue.
- **Note** Additional comment on - or a note about - an existing node in the graph.
- **Reference** A link to external material.
- **Map** Allows to add a sub-map to the current map, which is a collapsed discussion that can connect to other nodes in the graph.

An example of a discourse modeled in Compendium can be seen in figure 4. Discussion in Compendium start with a root issue on the left side, from which positions with arguments branch off. From there further issues can be developed.

Compendium makes use of colors and symbols to indicate the node types in the graph. Questions are colored blue and are marked with a question mark, ideas are colored yellow with a light bulb symbol. Pro arguments have a green color with a plus symbol and con arguments have a red color with a minus symbol. This helps to get an overview of the dialogue map at first glance.

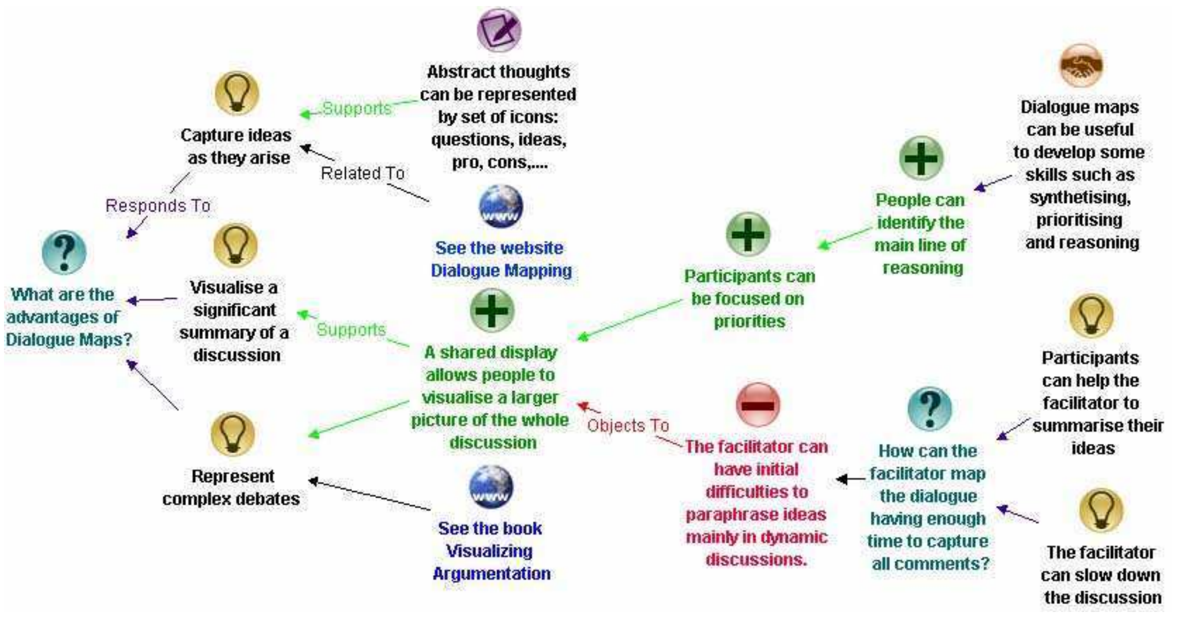

Abstract thoughts can be represented by set of icons: questions, ideas, pro, cons,....
Supports
Capture ideas as they arise
Related To
Responds To
See the website Dialogue Mapping
What are the advantages of Dialogue Maps?
Visualise a significant summary of a discussion
Supports
A shared display allows people to visualise a larger picture of the whole discussion
Represent complex debates
See the book Visualizing Argumentation
Objects To
The facilitator can have initial difficulties to paraphrase ideas mainly in dynamic discussions.
Participants can be focused on priorities
People can identify the main line of reasoning
Dialogue maps can be useful to develop some skills such as synthetising, prioritising and reasoning
How can the facilitator map the dialogue having enough time to capture all comments?
Participants can help the facilitator to summarise their ideas
The facilitator can slow down the discussion

Figure 4: Issue map in Compendium [OSS14]

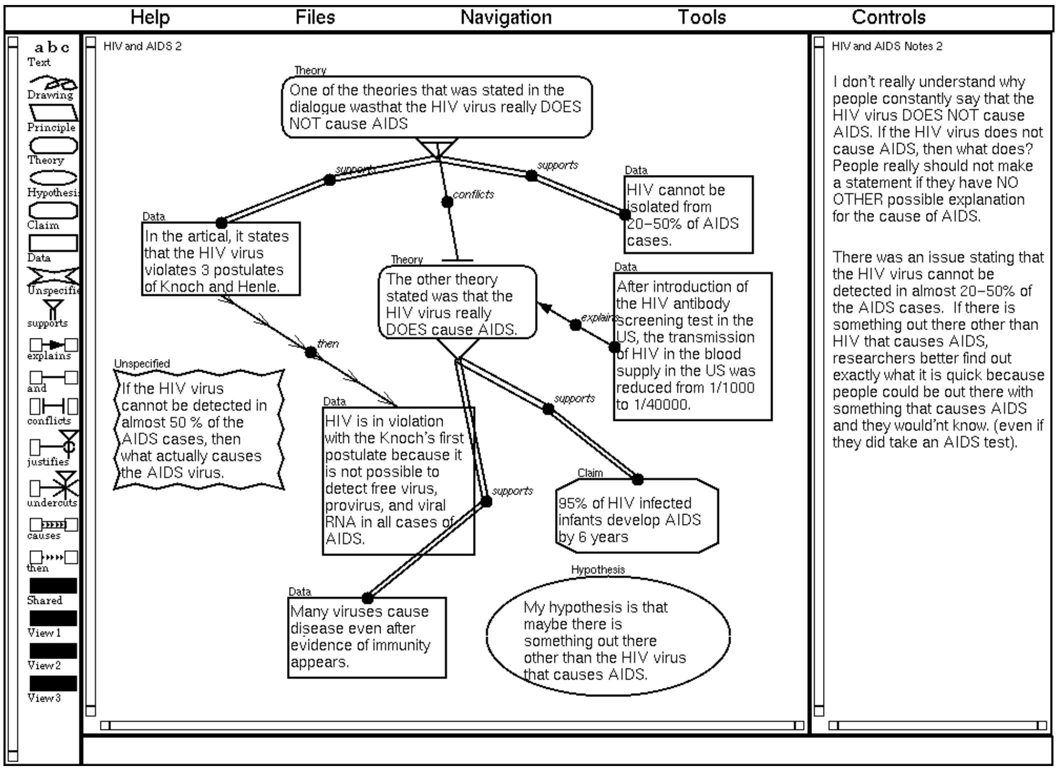
Belvedere

Suthers et al. described the development of Belvedere in [Sut+95] in 1995. Belvedere is a graphical software tool for students to develop argumentation skills and was evaluated with 12-15 year old students discussing issues in the scientific domain.

The authors state that they “have adapted Toulmin’s fundamental representations to scientific argumentation by providing (1) salient graphical representation of different types of argument roles, (2) negative as well as positive links, and (3) enclosure and multiple linkages to accommodate complex arguments” [Sut+95]. Still, the overall diagram leaves room for complex debates which can be interrelated in a more flexible way than in the original Toulmin model.

An example discussion about HIV/AIDS created by Students with Belvedere during the evaluation is depicted in figure 5. Discussions are displayed as a graph of nodes, which represent statements from different users.

The software tool makes use of different shapes in order to visualize the role of the statement (node) in the discussion: theories (ellipses), claims (octagons) and rectangles (empirical observations) make up the basic discussion elements. There is also the possibility to leave a post unclassified when the type of the contribution is not straightforward. Nodes can be linked to model logical and rhetorical relations between them; the shapes and link labels can also be configured. Additionally, it is possible to combine relations into a node in order to represent more complex arguments regarding a certain conclusion or implication. Different viewpoints can be highlighted with the



One of the theories that was stated in the dialogue wasthat the HIV virus really DOES NOT cause AIDS

In the artical, it states that the HIV virus violates 3 postulates of Knoch and Henle.

HIV cannot be isolated from 20–50% of AIDS cases.

The other theory stated was that the HIV virus really DOES cause AIDS.

After introduction of the HIV antibody screening test in the US, the transmission of HIV in the blood supply in the US was reduced from 1/1000 to 1/40000.

If the HIV virus cannot be detected in almost 50 % of the AIDS cases, then what actually causes the AIDS virus.

HIV is in violation with the Knoch's first postulate because it is not possible to detect free virus, provirus, and viral RNA in all cases of AIDS.

95% of HIV infected infants develop AIDS by 6 years

Many viruses cause disease even after evidence of immunity appears.

My hypothesis is that maybe there is something out there other than the HIV virus that causes AIDS.

I don't really understand why people constantly say that the HIV virus DOES NOT cause AIDS. If the HIV virus does not cause AIDS, then what does? People really should not make a statement if they have NO OTHER possible explanation for the cause of AIDS.

There was an issue stating that the HIV virus cannot be detected in almost 20–50% of the AIDS cases. If there is something out there other than HIV that causes AIDS, researchers better find out exactly what it is quick because people could be out there with something that causes AIDS and they would'nt know. (even if they did take an AIDS test).

Figure 5: Argument map in Belvedere [Sut+95]

help of different colors and it is also possible to draw connections with a freehand tool. Furthermore, Belvedere allows users to collaboratively work on a shared diagram from different computers.

Rationale

Rationale is an argument mapping tool, which is built for modeling, editing, displaying and sharing of argument diagrams. In 2007, van Gelder describes Rationale in [Gel07b] and [Gel07a]. It aims at providing a platform for diagramming arguments and reasoning about any topic by means of modeling relationships between propositions from multiple participants and sources. Like Belvedere, Rationale was developed mainly for educational use with the goal to teach reasoning to undergraduate students.

Figure 6 shows an example of a reasoning map in Rationale. It includes a claim or position at the top of the map, where reasons (because) and objections (but) can be added. The reasons and objections can further be reasoned about with the possibility to object to an objection with a rebuttal (however). Additionally, there is the possibility to add a basis as backing to a reason, which can be personal experience or common belief – these elements are leaves in the tree. The items used in the reasoning map in Rationale are similar to the ones used in the Toulmin model and give a similar structure

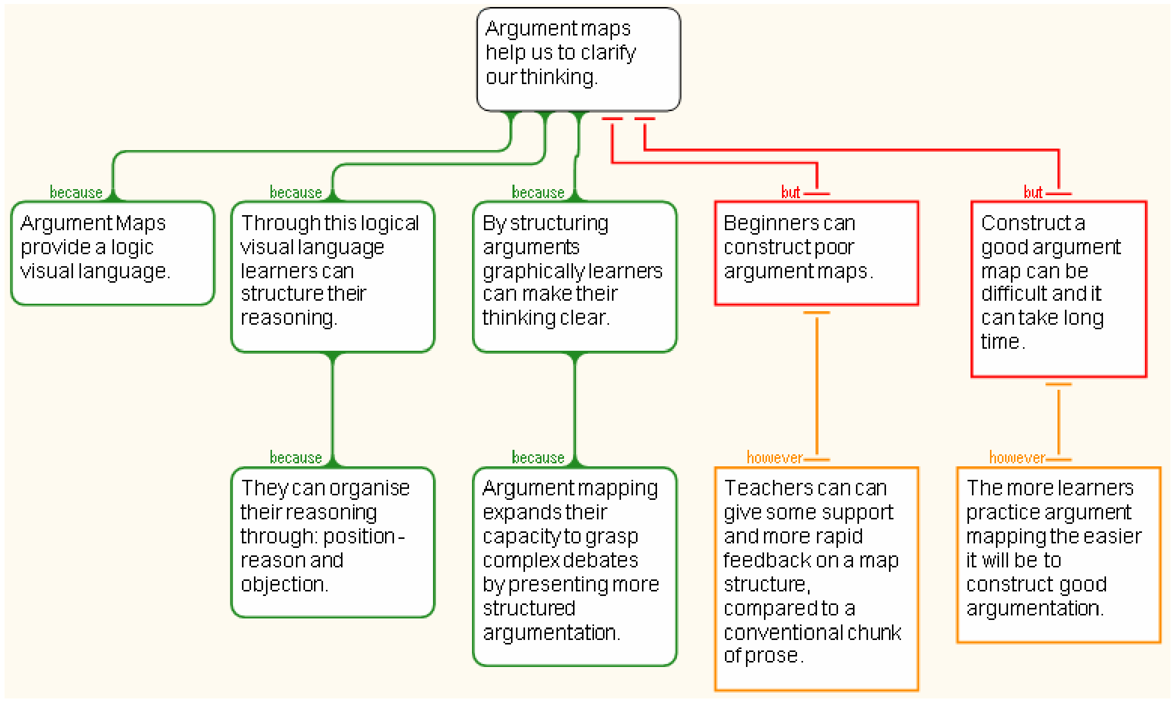

Argument maps help us to clarify our thinking.
because
Argument Maps provide a logic visual language.
because
Through this logical visual language learners can structure their reasoning.
because
By structuring arguments graphically learners can make their thinking clear.
but
Beginners can construct poor argument maps.
but
Construct a good argument map can be difficult and it can take long time.
because
They can organise their reasoning through: position-reason and objection.
because
Argument mapping expands their capacity to grasp complex debates by presenting more structured argumentation.
however
Teachers can can give some support and more rapid feedback on a map structure, compared to a conventional chunk of prose.
however
The more learners practice argument mapping the easier it will be to construct good argumentation.

Figure 6: Reasoning map in Rationale [OSS14]

though it is nested as a tree without reference to an actual relation between evidence and claim. It is also possible to add sticky notes to the diagram for meta discussions and personal annotations.

Another map type supported in Rationale is the analysis map. It allows to model relationships between different claims, which can be grouped together and then form a reason or an objection in conjunction. Groups of claims are displayed in one colored box – green for a combined reason and red for a combined objection. The next children in the tree may either be an argument concerning the whole group of claims or a single claim from the group. These constructs allow to model hyperrelations as one may refer to a whole set of premises and claims.

Multicentric IBIS

Multicentric IBIS (MctIBIS) is a web-based implementation of the IBIS model, it is developed by the Multicentric Technology Sdn Bhd [Mul12]. In contrast to gIBIS and Compendium, it does not focus too much on a graphical representation of the discussion, but allows different perspectives on the underlying graph. A discussion map can be viewed: (1) as a FIFO (first-in-first-out) or LIFO (last-in-last-out) list, (2) as a tree of posts, (3) each post can be focused with all follow-up posts in a threaded list. Even though, a graphical representation exists, it is not possible to interact with it. It is implemented as a Java applet which shows a static map of the arguments. The underlying graph is not visible when editing the discussion; furthermore, the relations of the IBIS model are left out and only the node types are used. The approach defines a different node type for pro and contra arguments This restricts the data model, as it is not possible to append a neutral or balanced argument.

Debategraph

Another prominent web-based example of a graph-based discourse system for collaborative debating is Debategraph[11], which is used by, e.g., the *White House* [Hou09] and the British newspaper *The Independent* [Ind11]. Debategraph is a website where users can open debates and add arguments for and against positions of other participants. The whole discussion is visualized as a graph and the discussion elements are similar to the ones used in the IBIS model: Issues, Positions as well as supporting and opposing arguments. Additionally to the structured discussion, each element has a comment list for open debate. Users can collaboratively edit the nodes in the graph, and reuse them in a different discussion.

2.4. Collaborative Tagging

When content is dynamically created in a broader scope, metadata is needed to index and categorize content by means of tags or keywords, which makes it discoverable for future navigation and searching. Traditionally, e.g., in a library, books and materials are categorized with a set of controlled vocabulary by a single authority, which might either be the librarian or it might emerge from the field of research. But the concept of a single authority does not scale well for collaborative workplaces, where many entities generate content and want to categorize their submissions to be discoverable by themselves and other users. This is where collaborative tagging mechanisms can be used, which do not depend on a central authority to define a legal set of tags, but lets all users collaboratively manage a folksonomy of tags.

Macgregor and McCulloch identify several advantages of controlled vocabularies and taxonomic classifications over a tag folksonomy: The use of synonyms can be handled by agreeing upon a single term, which ensures that participants use and search for the same set of tags. This avoids the confusion between different identifiers for the same entity. It is possible to resolve homonyms, which are words with same spelling but different meaning like *Java* the programming language and *Java* the island. For the controlled vocabulary, a semantic hierarchy of tags can be maintained, defining the

[11]`http://debategraph.org`

subordinate relation between the general concept and the specialization. Furthermore, non-hierarchical, syntactical relationships for modeling collocated tags could be defined. [MM06]

Collaborative tagging cannot guarantee any of these properties, as new tags can be created by anyone. Lack of collocation costs time for the user who has to explore every possible tag, which might be related as there is no common agreement which tags to use. This ultimately leads to a situation where "rules of discovery are not sufficiently predictable nor are they learnable. It therefore becomes a choice between a 'perpetual discovery cost' and a 'one-off cost'." [MM06] But a platform where a huge amount of users can independently create content demands a flexible categorization approach; a single authority is not a sustainable solution, especially considering the mass of possible tags across different areas of interest. Collaborative tagging can help to enable users to freely choose the topics of their submission and still provide a flat category system for discovery of new content. In [FSS13], Font et al. propose a recommendation scheme to assist users when tagging content. It relies on the co-occurrence of existing tags. Their evaluation yields promising results for reducing the signal-to-noise ratio in collaborative tagging systems and may help in using collaborative tagging as an efficient categorization and discovery approach.

3. Wust: a Hypergraph-based Discussion System

Internet forums and online collaboration systems have emerged and gain increasing popularity for asking questions, solving problems and sharing information; but they also serve as repositories of solutions, answers and advice for future readers. The lack of structure in these discussion systems hinders scaling for multiple participants and reuse of factual information. Argument models and argument mapping systems do not seem to get enough traction to be used more widely.

The main goal of this thesis is to provide a flexible and powerful discussion model and interface to discuss wicked problems and plan projects collaboratively. Therefore, the platform should be accessible and flexible enough to be used by many users while still providing a structure for arguments and decision making. Encyclopedic value is inherent with question-answer systems and internet forums in general, which is why we chose our discussion model and visualization method such that the arguments as well as the reached decisions are clearly visible. The IBIS model is a general technique for dialogue mapping which tries to capture the design rationale and handle problem-oriented discourses [KR70]. This is why we considered the IBIS-model as a basis of the presented approach.

Graph structure Posts are arranged as a directed graph with different discussion elements similar to the IBIS model [KR70]. A post may respond to another post or to the connection between posts; if post A responds to post B, this is modeled as $A \to B$. Different from IBIS, this thesis proposes a rather loose model. In order to reduce the barrier for new contributions, the model does not constraint how posts and relations should be connected and the node type is defined dynamically through the outgoing relations – the way a node refers to other nodes. Like Belvedere and Rationale, which are loosely based on the Toulmin model [Tou03], our discussion model is based on hypergraphs, in order to structure arguments regarding a relation between two elements. Thus, a post can respond to another post or to the relation between two discussion elements. This is a specialized form of the mathematical definition of hyperedges, which connect a set of vertices and makes our model a hypergraph. We define discussions as the connected component around a post.

Graph visualization The visualization should give a good overview and provide a good entry point for new users to follow and understand the flow of the discussion. All the information captured within the discussion model can explicitly be visualized in a graph representation. Our visualization method is two-fold: we offer a graph-view which shows the whole graph with all relations and we offer a focus-view where a single submission can be focused with all its neighbors.

Tagging It is possible to group posts under a certain category or topic by tagging them; in the following tags will be called contexts. A post may relate to multiple contexts. Context relations are not shown explicitly in the graph but are displayed as boxes within the node. A post can define how it refers to a context and how it refers to another post.

Collaboration It is important that participants of the discussion can work collaboratively, this includes remodeling and editing of elements in order to shape the discussion and explicate things pointed out in clarifications. We consider discussions with their issues, ideas and arguments to be living documents that adapt to new contributions. This is why it is important to offer a seamless moderation of the discussion for the users.

Reputation and moderation Users obtain karma points in the respective context when contributions are perceived positively by the community. The karma points reflect the moderation weight within a certain context. Every user can edit every post, but the changes are only applied instantly if the user has a sufficient moderation weight in the current context. Otherwise, the edition is transformed into a change request, which can be approved or rejected by other users. Instant changes can be rejected retroactively by the community, the change is then automatically reverted and the author of the edition loses karma points.

Voting In order to model priorities and user affection of posts, a voting mechanism is implemented. As posts might relate to multiple other posts or connections, voting is not done on the post but rather on the relation between two posts respectively between the posts and its contexts. The scoring of the connections enables quality sorting for responses with respect to a specific node and for entry points in a specific context.

Aim Our assumption is that discussions will evolve differently than in traditional tree-boards like Reddit and HackerNews. Furthermore, we expect that the structure helps to solve wicked problems and to discuss open-ended questions, which is not possible in existing question-answer systems. Even though, the first level of responses to a question or problem might lead to a similar structure, arguments for ideas can be discussed and issues can be raised from any post. We further expect that the graph structure is usable for the vast majority of users, even if not familiar with the employed concepts. The structure should help when modeling arguments and for getting an overview. Hyperrelations offer a flexible way to discuss links between information.

3.1. Model Development

In the beginning of this thesis, we investigated different argument models and argument mapping systems introduced in section 2.2 and section 2.3. The IBIS method formalizes a scheme for mapping arguments, which is a powerful model for viewing and elaborating discussions in a structured form. It aims at giving a good overview on the design rationale behind decisions reached within an argumentation. It has been successfully applied in different areas and offers a great basis for modeling discussion and capturing connections between the discussion elements. Other than purely tree-based models, the graph structure allows a single post to respond to multiple different posts. This provides multiple paths in the discussion to reach a certain element but also allows factual information to be reused in a different context.

The possibility to model the argumentation of design decisions enables the reader to get an overview of the explored possibilities and to understand the rationale behind decisions made in the process of the discussion by evaluating position and arguments. This fosters transparency of the decision process and enables collaborative problem solving, as the participants can model issues in their intended structure and therefore make their thinking process visible to other contributors. The visualization of the discussion should always give a good summary of the current state of the discussion by providing the user with an entry point. The graph structure allows a traversal of the discussion from top to bottom.

Therefore, the first idea during model development was based on the IBIS model similar to Compendium. The concept defined four node types: *Problem*, *Idea*, *Goal*, *Argument*. Nodes could be connected through a static set of relations like *solves*, *supports*, *opposes* or *repliesTo*. The relation type followed a similar model like IBIS with the extension of a goal type.

When using the system, we felt that the static types were very restrictive when contributing to the discussion. Selecting one type for each node was especially difficult in the construction phase of the discussion. The need for a classification can make it difficult for users to contribute. For example, it can help to first write down the information in an unstructured manner or as multiple types – stating a goal and a problem simultaneously. It is then possible to classify and structure the posts accordingly as the discussion proceeds. Therefore, we decided to make our model more flexible. We implemented a prototype where every post and relation could be tagged with classifications (node type) and with contexts (area of interests). Every possible combination of nodes can be connected with a relation: nodes can be connected to as many nodes as needed, there is no restriction or legal moves. In the first user test, we then wanted to find out how users would incorporate the tagging mechanism and how they would structure their discussion when using a flexible model.

The first user test showed that users had a tendency to sometimes classify the connection between a post rather than the post itself. The users were confronted with the possibility to freely choose a link type when connecting two posts. Even though, a node was already tagged as an idea, users sometimes wanted to classify the relation as an idea. This corresponds to the assumption that the way how nodes refer to other nodes shapes the type of the node. But the user test also showed that users seemed to be confused with the different tags, which could either be classifications or contexts. This is why the next iteration of the discussion model puts more emphasis on the difference between the two types. The next section describes the final discussion model, which is based on a hypergraph and defines the node type via the classification of its outgoing relations.

3.2. Hypergraph-based Discussion Model

This thesis proposes an IBIS-like hypergraph-based discussion model that allows to flexibly model complex discussions and facilitates reasoning over single statements as well as gathering arguments for and against the connection between elements. The idea is to maintain a certain structure without rigid constraints on the relations, which might have a negative impact on contributions. This is achieved by moving the type of nodes entirely into the relation. So the type of the post is defined by its outgoing relations - by the way it refers to others. This allows posts to refer to other posts in different ways. This is in contrast to the commonly applied IBIS like notation, which restricts relations between certain types of nodes, see section 2.2. Only two kinds of relations are disallowed in our system: self loops and incident relations.

The proposed model is rather loose and does not restrict the possible relations. If a post responds to another post, a relation between the two elements is established. Each relation can be classified, i.e. the way the start node refers to the end node of the relation: one post *is an idea for* a problem or one post *is a pro argument for* an idea. We define a set of classifications, which can be used to define the type of a relation; multiple classifications can be used at once. The sum of all classifications on the outgoing relations of a post shape the type of a post.

An important goal is to provide a good overview of discussions and to capture how messages relate to each other. On the one hand this overview should help the user to explore the space of the mentioned possibilities and to evaluate arguments in favor and against an idea. On the other hand it should also help to reduce redundancy inherent in massive collaborative works, as redundancy can be made more visible when the structure of the argumentation is more obvious to the user. We expect that this makes redundancy less likely to be created. Furthermore, the reuse of discussion elements might help to connect good answers to multiple questions. Also, duplicate questions are a sign for a missing entry point to the discussion; the creator did not find the already existing discussion. If one just connects the duplicate question to said discussion, it gains a differently phrased entry point for the user. This could be a better way to handle duplicate questions instead of closing them, because this just confuses the users where to answer and where to vote. Discussing both in the same graph resolves this problem entirely. One could think of tagged posts as entry points instead of page numbers or scroll levels.

Hypergraph

One problem in Compendium and other traditional IBIS-like systems is the inability to discuss the relation between different posts. Similar structures can be achieved in the Toulmin model [Tou03], when looking at the combination of links in Belvedere or the grouping of premises and claims in Rationale. As nodes may refer to multiple nodes, it is important to not only be able to discuss about the nodes, but also about the relation between two nodes. If a node refers to two different posts, it might be the case that one wanted to discuss the post with regards to one of its parents. If one wanted to discuss this in traditional IBIS, the only way would be to discuss it under the child post. But this is not precise, as the comments actually just refer to the tuple (parent, child) and should not be shown when viewing the other connection. Therefore, our model additionally employs hyperedges, which makes our model a hypergraph.

Consider an example discussion, where two questions are raised: “How to drive to work?” and “How to travel from Germany to Australia?”. One participant responds with the idea “By bike” and connects it to both questions as an idea. If one would want to add the contra argument “Cannot cross ocean by bike” to the discussion, then connecting it to the idea “By bike” would be imprecise, as the problem only refers to the idea relation between the idea and the question “How to travel from Germany to Australia?”. This is why the system models relations as hyperedges and, thereby, uses a hypergraph as the underlying data structure of the discussion model. See figure 7 for a depiction of how one could model the above mentioned example in Wust.

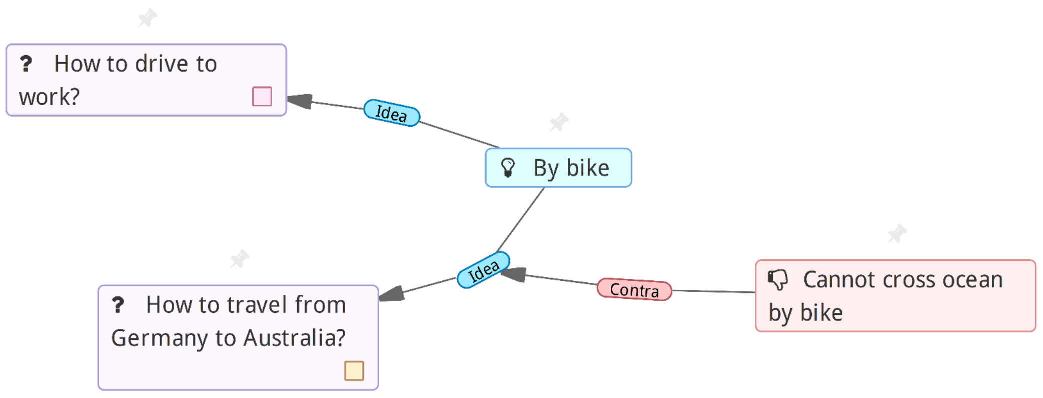

? How to drive to work?
Idea
By bike
Idea
Contra
Cannot cross ocean by bike
? How to travel from Germany to Australia?

Figure 7: Response to a hyperrelation in Wust. The contra argument only refers to the relation between one question and the idea.

Classification of Relations

As explained in section 2.2, the IBIS model defines a set of node types for the discussion elements. Each post has one static type and may relate to other nodes in accordance to a relation model. In contrast, our model requires no classification of the post itself, but only of the relationship – it is defined more dynamically. The type of the node solely depends on the way it refers to other nodes and is defined by the set of classifications on the outgoing relations. Thus, the start node of a relation inherits the classification of the relation. A relation does not need to have a classification and may have more than one. When connecting a post to another post or when responding to another post, the user has to specify, how his post refers to the other one.

In order to mimic the vocabulary and be able to emulate the structure of popular question-answer systems and issue trackers, we do not use the elements from the Toulmin model which comes from the field of argumentation theory [Tou03]. The static set of classifications that used in the discussion model is inspired by the IBIS model and the additions made in Compendium. We define the following classifications for relations:

Problem	Point out a problem, this might a bug in a computer program or it could be used to signal a problem in another post, project or community.
? Question	Ask a question within a context, ask about an issue in a post, or request a clarification in another post.
Idea	Proposes a solution to a question or problem and points out an opinion or just states an idea.

Pro	Argument in favor of its target post.
Contra	Argument against its target post.
Goal	A goal that should be reached within a certain context or a goal that needs to be reached in order to solve a problem, e.g., software requirements.
Task	To denote tasks and Todos.
Cause	Signaling causality between elements, a bug in a computer program might cause an issue in another one.
Done	Mark a decision, a solved problem or a finished task.

As described in [MM05], a rigid structure and a need to define a label for each message might have a negative effect on the contribution. We try to reuse terms commonly used in internet forums, question-answer systems as well as bug tackers and combine them with the common IBIS-like types. Furthermore, this approach reduces the need for labels, because only relations need to be classified and post classifications are implicit. The classifications *problem/question* and *idea* pretty much correspond to the model of StackOverflow, which has questions and answers. The *task* classification offers the possibility to define tasks that need to be done in order to fulfill a certain goal or solve a problem. Using the classification *cause* allows the user to model causation within a discussion. In order to be able to mark things as done, we defined the classification *done*, which is meant to be used in conjunction with another classification: A problem might be solved, a question might be answered, as task might be finished. This is very similar to the *decision* type for ideas in Compendium. In IBIS-like systems, question and problem are often subsumed under the type *issue*. We are aware that both classifications are very similar, but it was chosen in order to distinguish between problems – or bugs in the case of software development – and questions – for asking questions and requesting clarifications in a post. It would be of high interest to investigate the effect of different classifications on the discussion to find a minimal set of needed classifications to leave less room for confusion. Moreover, it might be interesting to define different classification schemes per context to completely emulate other platforms or arguments mapping systems.

Especially the combination of hyperrelations with arguments like *pro* or *contra* classifications allows to model argumentation in a way known from the Toulmin model. It is possible to define supporting or opposing arguments for the relation between two discussion elements and further provide backings and qualifiers for a connection. Otherwise, the discussion can be built up similar to the IBIS model, though the roles of the contributions can be handled more flexibly and hyperrelations allow a more precise reference mechanism for arguments.

Instead of classifying the node, we classify the relation between nodes, this is the way a post relates to other discussion elements. Therefore, it is possible for a post to refer to two parent post in different ways. Figure 8 shows an example discussion about the question “What do you want to do today?”. There are two ideas connected to the question: “Let’s go to the beach” and “Let’s go to the cinema”. Furthermore, someone posted that “It’s raining”, which is connected to both ideas as an argument: obviously it as supporting argument for the suggestion to go to the cinema and opposes the idea to go to the beach. Defining the classification on the relation gives a flexible mechanism to resolve cases where a post relates to multiple other elements in different ways. The model could easily be extended to have different classifications or more domain-specific vocabulary if necessary.

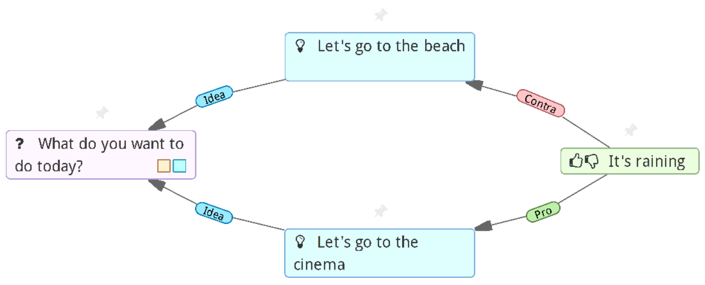


Figure 8: Multiple classifications of one post in Wust. The post refers to two different posts in different ways; it is a pro argument for one idea and a contra argument for another.

The implemented system visualizes the classifications with colors and symbols, so the user can easily detect how discussion elements are related. For each classification, we chose a color and a symbol, which is used throughout the UI to visibly mark posts with their classifications. If a post does not have a classification defined through its outgoing relation, then the background of the posts is white. We hope that this also motivates users in adding classifications to really make use of the structure.

3.3. Context Tags

In order to organize posts under different topics, a collaborative tagging approach is taken. In our system a tag is called a context, as it defines a scope for the discussion and provides contextual information about the topic. A post may be connected to multiple contexts, thereby signaling its membership in the context's domain. The previous section 3.2 describes how the relation between discussion elements can be classified from a set of pre-defined classifications, it describes how the response refers to its parent. Analogously, it is also possible to classify the relation between a post and its context. A user might want to publish a problem, a question or an idea in a certain context, e.g., report a problem in the bug tracker of Wust or ask a question about math.

Multiple contexts may form a hierarchical structure, which is why we implemented multiple inheritance for contexts, such that one context may inherit from multiple other contexts. For example, a *bike* and a *car* are both a *vehicle* – both are hyponyms of the more general concept *vehicle* (which is a hypernym of the *bike* and *car*). If one adds an inherits relation from the *bike* and *car* contexts to the *vehicle*, then posts tagged with either one will be found when searching in the parent context *vehicle*. Similarly, people might use certain words synonymously. For example, *bike* and *bicycle* have the same meaning and can be used interchangeably. This can be modeled as an inherits relation between both contexts in both directions. Being able to model the relations between contexts in the system avoids the need for the user to add synonyms and hypernyms to each post individually.

However, as users can freely create new contexts, redundancy in the form of synonyms will emerge. Furthermore, the tag hierarchy needs to be kept up to date. That is why it could be of interest to investigate the possibility to use Synsets from lexical databases like WordNet, which contain synonyms as well as hypernyms and hyponyms [Fel98]. This could be further facilitated with an approach that tries to automatically create a tag hierarchy from unstructured data categorized by tags [HG06]. Tag recommendation and completion is important for users to make them aware of the existing structure and also to give clues which alternative contexts exist [FSS13].

Contexts are the top level hierarchy of posts in the discussion system. Every post that has a context is defined to be an entry point to its discussion, which is represented by the connected component below the post. Such a post is of interest within the context and can be focused to view the surrounding discussion. If a post does not have a context, it is assumed that the context is built up from its successor nodes and it only implicitly inherits the contexts from its parents. Note that a discussion might have several entry points as multiple posts might have a context. Throughout the UI, we display the contexts as rectangles with different colors in order to be able to easily distinguish between multiple tags on a post.

3.4. IBIS Compatibility

The proposed discussion model is inspired by IBIS, but changes the meaning of relations and nodes. Moreover, it provides richer semantics for the relations between discussion elements and introduces hyperrelations. The similarity between the defined classifications and the IBIS node types allows to create similar maps. IBIS maps can therefore easily be transformed into the Wust model.

For example, the dialogue in Compendium from figure 4 can be modeled in our system (see figure 9). The node types and relation types are combined and transformed into a classification in Wust, e.g., supporting and opposing arguments are transformed into pro and contra classifications. The set of classifications does not have an equivalent for the node type *Note*. This is why we only use the relation type to classify the relation – it becomes a pro argument. Furthermore, using multiple classifications on one relation is, e.g., useful when emulating the decision type from Compendium. We can use the *done* classification in conjunction with the *idea* classification, this allows specifying the initial node type while also stating the decision. In compendium, the intended type of the decision node is not captured. In general, IBIS maps can easily be emulated using our system and need less classification, since only relations and not nodes are classified.

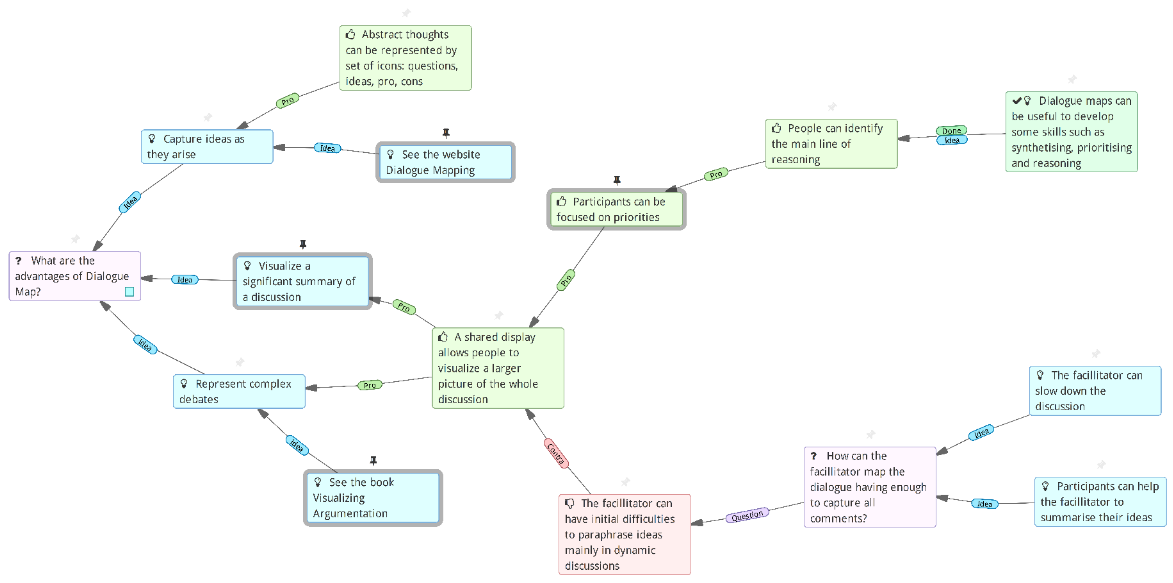

Abstract thoughts can be represented by set of icons: questions, ideas, pro, cons
Pro
Capture ideas as they arise
Idea
See the website Dialogue Mapping
Idea
People can identify the main line of reasoning
Done
Idea
Dialogue maps can be useful to develop some skills such as synthetising, prioritising and reasoning
Pro
Participants can be focused on priorities
What are the advantages of Dialogue Map?
Idea
Visualize a significant summary of a discussion
Pro
Pro
A shared display allows people to visualize a larger picture of the whole discussion
Idea
Represent complex debates
Pro
Idea
See the book Visualizing Argumentation
Contra
The facillitator can slow down the discussion
Idea
How can the facillitator map the dialogue having enough to capture all comments?
Idea
Participants can help the facillitator to summarise their ideas
Question
The facillitator can have initial difficulties to paraphrase ideas mainly in dynamic discussions

Figure 9: Issue map from figure 4 modeled with Wust. Modeling can be done similar to the IBIS model.

3.5. Visualization

The presented approach employs a hypergraph for capturing discussions, which suggests a graph visualization. Graphs are explicit and allow to transport all the information of the underlying discourse model. Thus, it is possible to display posts of different kinds in different shapes and colors and also visualize the kind of relation between two nodes. Also, it makes it easier to visualize that a post might refer to several nodes; or in the case of a hypergraph possibly to a set of nodes. In the case of a directed graph, the direction of the relation plays an important role.

IBIS-like systems most commonly display arguments as a graph with explicit relations, like in Compendium. This gives a good overview of the whole discussion. Debategraph, on the other hand does not show the whole component of a post in their graph visualization. It only displays the direct neighborhood of a node, which has the advantage of small, easily readable graphs, but it hinders exploration as clicking each node to focus it can be tedious. Therefore, we decided to combine both advantages and provide a global and a local view on the discussion. When viewing a post, the whole connected component is retrieved from the server and the user can choose between two views: a graph-view and a focus-view.

The graph-view is similar to the visualization in common IBIS systems like Compendium. It is a graph of the discussion with all nodes and relations and provides an overview of the whole argument. It gives a good overview on direct responses and references. The classifications are used as labels for the edges in the graph and the color of the classifications defines the background of the edge label. The nodes are colored according to their most prevalent classification and the contexts of posts are shown as little squares in the bottom right corner of the posts.

One problem in the graph representation is disorientation in large graphs with a lot of relations and nodes. It becomes difficult to focus a single node and concentrate on the arguments only regarding this node. Therefore, it is possible to filter nodes by classifications and by keywords. If a filter is active, only matching nodes are shown with full opacity, the other nodes within the connected component of a post are grayed out into the background and components without a matching node are hidden. This gives a better overview on certain elements of the discussion, as one can easily search for all questions which occurred in a discussion. For a more fine-grained search, the keyword filter can help to only match questions on a certain topic. However, this might not be enough if the graph is highly nested. In order to explore the direct neighborhood of a post, the focus-view can be used. It displays the focused post in the middle with its successors on the left and its predecessors on the right side. So, the parents of the currently focused post are shown on the left and the responses are shown on the right. The relations are depicted similar to the graph-view to emphasize that both views operate on the same data. If a post responds to a hyperrelation, this is displayed as the set of nodes contained in the hyperrelation. If a post responds to the response relation of a post on the right, it is displayed within the response.

It would be worth to investigate different levels of detail for the graph view. This could incorporate hiding certain kinds of relations and also switching to a different visualization form. For example, this is implemented in gIBIS where zooming triggers a global or a local view on the graph [CB87]. A similar approach was chosen by Riechert et al. in [RQZ15], which makes use of different graph representations which includes a rather compact circle map. Compendium, on the other hand, only relies on a small global map in the bottom right corner that can be activated if needed.

3.6. Title with Optional Description

Apart from the header and the recipient, an email primarily consists of a subject and a body. When responding to an email, most email clients add a common prefix to the initial subject. This enables the client to group conversations about a certain subject. Commonly, internet forums on the other hand only have a title for the thread, which is chosen by the thread opener – who writes the first post. Posts are only viewed in the context of their thread and therefore there is no need for a separate subject for each post. The posts in a thread are always displayed in full length when viewing the thread.

This thesis proposes a concept which treats all posts equally, as there is no distinction between thread-opening posts and responses. Furthermore, our UI concept includes lists of posts in the focus-view as well as a graph-view of the discussion. In both views, posts need to be compactly displayed. Posts might be longer than just a paragraph and displaying the whole content would lead to a cluttered interface. Therefore, we decided to split posts into two parts: the title or subject of the post and the description or further background information about the issue. This is a common distinction with IBIS-like system, because of the upper mentioned requirements. The title is restricted to a maximum of 140 characters – it should be the essence of the post. The description is optional and can be left out for short posts; only if further information needs to be conveyed the description should be used.

3.7. Collaboration

User collaboratively discuss across the internet and new submissions need to be visible immediately for all users. Furthermore, new connections and updates to existing posts should directly be visible for each participant of the discussion. This means that each viewer of the discussion should be notified about changes to the current view and the view should be updated without reloading the page.

These update triggers are important for making contribution and potential interactions of the participants visible. A study about the motivation of contributors in Wikipedia was conducted by Howison et al. in 2011 [How+11]. The authors mined the history of article editions in Wikipedia and analyzed the data for effects of Wikipedia watch lists which makes contributions immediately visible to interested users. They come to the conclusion that making the work of other users directly visible is an important measure for contribution motivation in open collaboration systems. In a collaborative discussion system, all users are participants in the discussion, and therefore need capabilities to react to changes and to update as well as edit the current model.

3.8. Quality Assurance

In the process of the discussion, certain elements might need to be rephrased or extended. Additionally, posts might need to be reconnected. This is why a sensible moderation system is needed to allow users to collaboratively work on the same discussion. Our discussion system deploys a community driven approach, which relies on karma points awarded per context. Every user can issue change requests on existing posts. Other users can vote whether this change request should be approved or rejected. Their voting weight is based on their karma points in the current context. Users receive karma points if their contributions are perceived positively by the community, i.e. their post was upvoted or an authored change request was accepted.

The author of a post gets a boost on his voting weight, so he is able to push changes on his own posts. A change request must reach a certain threshold to be applied, which depends on the popularity of the post. If a change requests is applied instantly because the user has enough karma, then the change request can be voted on afterwards. The community can retroactively validate or revert the edit. So either the change becomes permanent or is reverted. Rejected changes have a negative effect on the karma of the editor; the karma deficit depends on the edit threshold of the post.

We planned to implement a similar moderation system for connections as well, in order to guarantee a certain quality of the edits. For simplicity, we decided to leave this out in the prototype, thus users in the user test were able to freely model the discussion graph to their liking.

In order to enable sorting of posts according to their popularity/acceptance in their context and as a response to another post, a voting mechanism is needed. In this approach, all views of posts from a registered user without an upvote are counted as a downvote. This is based on our hypothesis that users rarely downvote posts. This also avoids the confusion about the down vote button as a disagree button. It is possible to upvote any relation in the graph, this includes relations between posts and/or hyperedges as well as between posts and contexts. Thus, it is possible to sort posts by quality – either within their contexts or as responses to another post.

Further details on quality assurance and the moderation approaches in the presented discussion system can be found in the master thesis of Felix Dietze [Die15].

4. Implementation

Wust is implemented as a single page application, i.e., a website served as a single page where needed resources – HTML templates, CSS styles, images as well as JavaScript code – are dynamically loaded from the server and are then added to the page. The client communicates with the server via a RESTful API. In order to facilitate collaborative modeling and problem solving, connected clients are notified about changes to the currently viewed discussion through a WebSocket.

We investigated different databases as a backing store, which included prominent NoSQL databases as well as relational databases. Because of the requirements regarding consistency and flexible queries over graph structures, we chose a state-of-the-art graph-database neo4j[12]. It supports ACID transactions and offers a declarative query language for powerful graph traversals.

The implementation of the proposed discussion system is flexible with regards to the underlying discussion model. Therefore, we implemented an abstraction layer for neo4j, in order to enable agile development of a discussion schema: renesca and renesca-magic. This offers a comfortable way of traversing the result set of a query as graph or a table. Furthermore, it provides a declarative way to define type-safe schemata over the property-graph model commonly used in graph databases. This allowed to test different discussion models and enables future development of the discussion system.

The discussion model is implemented with renesca-magic and made accessible to the client through a REST API that can be partially generated from the defined database schema. A powerful search API enables the client to search for specific posts or list submissions within a certain context or with a certain classification. The server can notify the client about updates to the currently viewed discussion, which enables real-time collaboration of the participants.

4.1. Architecture

Wust is implemented as a server-client architecture. The server is implemented in the Scala programming language and uses the Play framework[13], which follows the Model-view-controller pattern. As a single page application, there is only a single view served by the server, the remaining resource are loaded dynamically or bundled in the first

[12]`http://neo4j.com/`
[13]`https://www.playframework.com/`

request. For data manipulation and retrieval, the server provides a REST API serving JSON to the client.

The client is implemented in EcmaScript 6, the newest version of the EcmaScript standard – commonly known as JavaScript. For browser compatibility, this is transpiled to EcmaScript 5 on the server using the Babel JavaScript compiler[14]. Parts of the client were implemented in ScalaJs, i.e., Scala compiled to JavaScript. Using ScalaJs enables code reuse between the server and the client. This especially comes in handy for validation code as well as the formulas to calculate moderation rights in Wust. We use AngularJs[15] as a client-side Model-view-controller framework; Bootstrap was chosen as the CSS framework. Stylesheets are written in SASS[16] and are processed with Compass[17] for ease of use and browser compatibility.

4.2. Graph Database Selection

We considered relational databases as well as NoSQL databases for our use case. Popular relational databases like PostgreSQL come to mind as they are well-tested, stable and guarantee ACID transactions. In our experience, handling a flexible graph structure in a relational database turns out to be cumbersome. The edges between two nodes are an $N \times N$-relation and therefore need to be modeled as a separated table. Furthermore, the end node of the relation could be a post or a relation. This lead to complex SQL queries with many joins when, e.g., querying for the connected component of a post.

Common NoSQL solutions only provide weak guarantees regarding consistency and have limited support for transactions. The design is aimed at performance and to scale horizontally, which comes from simplicity. This does not fit well for a collaborative system, which needs consistency to handle a lot of parallel requests and interfering changes. Furthermore, a lot of document-centric database systems exist that do not support joins natively, which puts a heavy performance overhead on the application server. This does not work well for a highly connected database model.

Graph databases offer an alternative approach, which can capture the discussion model natively as a graph. Neo4j represents data as a property graph model, which consists of a set of nodes and a set of edges. Nodes can be labeled to represent their role and may hold multiple attributes as key-value pairs. The edges are directed connections

[14]`https://babeljs.io/`
[15]`https://angularjs.org/`
[16]`http://sass-lang.com/`
[17]`http://compass-style.org/`

between two nodes. A connection may have one relation type to specify its meaning and may also have attributes like nodes.

The database neo4j can either be run as an embedded database in the same process as the application or can be run in its own JVM, which can be queried via a REST API. As an embedded database, it is possible to imperatively traverse the database graph, the REST API exposes routes for nodes and relations as well a declarative way of interacting with the database. Neo4j ships with a declarative query language called Cypher. Similar to SQL, Cypher queries can be composed from various clauses that can be chained. Graph patterns like nodes, relations or whole paths can be expressed for matching certain elements in the database. This makes it easy to construct queries for our graph-based discussion model and, e.g., offers a simple mechanism to query for the connected component of a post.

4.3. Graph Database Abstraction

Existing Scala neo4j REST libraries mostly focused on a relational representation of the data by interpreting results as a table instead of utilizing the graph structure. Furthermore, we needed a possibility to comfortably model hyperrelations. This is why we implemented a library for querying neo4j via the REST API and handle the result as a property graph or as a table: renesca.

The iterative model development employed in this thesis and the need to react to user feedback demanded a flexible implementation. The lack of schema definitions in neo4j makes it difficult to consistently follow a schema across an entire application, as it is only defined implicitly. Furthermore, the missing type information hamper heavy changes to the underlying discussion model, as the lack of type checks makes refactoring error-prone. That is why we implemented an abstraction layer for renesca: renesca-magic, which serves as a type-safe interface to the database.

In their review about argumentation software, Scheuer et al. remark the lack of reusable software components for building educational argumentation systems [Sch+10]. Our graph abstraction layer makes it possible to comfortably model database schemata in a state-of-the-art graph database. We expect this to help for building collaborative systems which apply a certain schema to their discussions.

4.3.1. Renesca

During this thesis, we developed an open source library for neo4j: renesca[18]. It offers functionality to query a neo4j server using Cypher through the REST API and to interpret the result either as a graph or as a table. Nodes and relations can be created, existing ones can easily be manipulated and persisted. The library follows three key design principles:

Work with graphs instead of lists Renesca can interpret query results as a graph, which gives a comfortable interface for traversing the results. The graph data structure can be interpreted natively and there is no need to transform it into lists of entities. If the queried data is not a graph, it is also possible to interpret the result as a table.

Track changes, persist later When creating, modifying or deleting nodes and connecting them with relations, it would be very expensive to submit a REST request for each change. Therefore, renesca tracks changes in the graph and applies all of them at once when persisting the whole graph. This takes fewer REST requests and leaves room for optimization.

No lazy loading There is no further database traversing from a subgraph because there is no need to do so. When working with subgraphs retrieved from a query, the needed data is known in advance and can be fetched with the query before traversing. This approach saves a lot of unnecessary requests.

The database can be queried for specific nodes and relations which are then stored in a graph data structure. The graph consists of a set of nodes and relations, which each have a hash map of properties – neo4j only supports primitive types like String, Boolean and Long. Each relation has one relation type. Each node has a set of labels and methods to explore incoming and outgoing relations. For example, one can query for all posts by their label and get the graph; then, one could, e.g., change the title of the first post to "Wust":

```scala
val graph = db.queryGraph("match (post:POST) return post");
val firstPost = graph.nodes.find(_.labels.contains("POST")).get
firstPost.title = "Wust"
```

In neo4j, nodes and relations can either be created, matched or merged. A merge is a match-or-create operation, which either matches the specified item if it exists or otherwise creates it – merge guarantees uniqueness. When instantiating a node or relation

[18]https://github.com/renesca/renesca

in renesca, one can specify whether it should be created, matched or merged. So, for example, it is possible to create a local node representing a post with a corresponding label and a title; matching and merging nodes works similarly:

```
val newPost = Node.create
newPost.labels += "POST"
newPost.properties("title") = "hello world"
```

When persisting a graph, the tracked changes of all contained nodes and relations are transformed into Cypher queries. This includes matching, creating and merging as well as modifying the properties of the nodes and relations. Cross-dependencies between changes need to be handled, e.g., when matching a hyperrelation and creating a relation pointing to said hyperrelation. In this case the hyperrelation needs to be resolved first, in order to reference it in the query. This is why a dependency graph for the changes is created and the statements are then sent out to the server in order. The result set is then interpreted to update the local graph representation with the newly retrieved information. Persisting changes should be done in a transaction. If any of the queries fails, the transaction is then rolled back and the local graph representation remains unaltered.

For example, it is possible to persist all changes in the initial graph which was queried from the database and it is also possible to individually persist the creation of the new post; this is wrapped in one transaction:

```
db.transaction { tx =>
    tx.persistChanges(graph)
    tx.persistChanges(newPost)
}
```

In mathematical terms, hyperedges are edges that connect a set of vertices. In renesca, this is supported by introducing the concept of paths. A graph may define multiple non-overlapping paths, which are used when persisting the changes. Therefore, it is possible to match all unresolved hyperrelations on their defined path in one query pattern. So, adding hyperrelations to the graph in renesca works by adding the contained nodes and relations as well as a path covering all these nodes.

As nodes and relations in renesca are untyped, the properties need to be casted to the correct type and when visiting neighbors of a node, the nodes need to be filtered for the desired node label. Therefore, it is convenient to write schema classes and factories for each node label and each relation type. These schema classes are wrappers for the raw renesca nodes and relations, which provide proper accessors for the properties and neighbors. The result graph of a query can then directly be wrapped accordingly by mapping the labels and relation types to the corresponding factories. Writing such a wrapper includes specifying a label for the node, defining getters and setters for the properties of said node as well as writing accessors for neighboring nodes and relations. For relations, the wrapper needs to specify the relation type as well as the label of the start and end node. Hyperrelations can be implemented in renesca by creating two relations and one middle node. The first relation starts at the start node and ends in the middle node, the second relation goes from the middle node to the end node.

Renesca provides interfaces for these schema classes as well as their factories, additionally it provides functionality to handle a graph containing these typed nodes and relations. It is possible to override a validation function in each schema node, where properties and neighbors can be checked before persisting. This allows, e.g., to restrict properties to a certain range of values.

4.3.2. Renesca Magic

During the development, we often had to change the underlying schema of the database and writing a wrapper class for each node as well as relation is error-prone and cumbersome when working with complex database models. This is why we implemented renesca-magic[19], an open source Object-Graph-Mapper.

The library provides a domain-specific language (DSL) in Scala for defining a database schema via annotations in the source code. Code generation of the above mentioned boilerplate code for the type-safe wrappers of nodes and relations is done using Scala macros. Scala macros can be written in Scala and work on abstract syntax trees instead of raw strings. They are executed during compile time and allow to generate, transform and type check the abstract syntax trees. Each time the schema is compiled, the generated code is written to a file for debugging purposes.

Nodes and relations are defined as normal Scala classes. Annotations can be used to mark classes for macro expansion, which generates the boilerplate code. Nodes can be defined by just providing a name, relations additionally need a type for the start and end node. The names of the node and relation classes are used as labels and relation types in the property graph model in renesca.

The generated node and relation wrappers can have properties. As neo4j only supports primitive types, the following types are supported: *Boolean*, *Long*, *String*. It is possible to define immutable properties as a *val* or mutable ones as a *var*, which generates getters and setters accordingly. Optional values are expressed as an *Option*. Default values

[19]https://github.com/renesca/renesca-magic

can be defined by specifying an assignment with an expression on the right-hand-side, which is evaluated when calling the factory method for the class. The generated wrappers access the properties through the property map of the wrapped item. Uniqueness constraints for properties in a node class can be specified with an annotation, which generates code for setting up the respecting indices and constraints in the database.

A simple example can be seen in figure 10. It defines two node classes: *Post* and *User*. A post consists of a title and an optional description of type String – both values can be changed and defining the properties as variables generates setters accordingly. Users have a unique name, which cannot be changed – defined as a value. A user may create multiple posts and whenever a post is created an *Authored* relation between the current user and the post is created.

```
@Node class Post {
    var title: String
    var description: Option[String]
}

@Node class User {
    @unique val name: String
}

@Relation class Authored(startNode: User, endNode: Post)
```

Figure 10: Nodes with properties and a relation defined with renesca-magic. This will generate wrapper classes and factories for all the defined node and relation classes.

In order to enable code reuse and querying for base classes, multiple inheritance between classes and traits is supported. Properties and labels of the extended traits are inherited. Additionally, traits define an interface for accessing relations defined for the trait. An example for multiple inheritance between nodes can be seen in figure 11. In order to identify nodes by id, an id property is needed[20]. As multiple node class should have an id, one can define a reusable trait *UuidNode*, which has an uuid property initialized with a random Base64 String. In addition, it is possible to define a *Timestamp* trait for a timestamp property. The post from the previous example can now extend both traits and thereby inherits both properties. Additionally, the *Post* class inherits the labels from its base classes, which makes it possible to match all *UuidNode* subclasses by querying for the according label.

[20]Neo4j has an internal id for nodes and relation. But it is not advised to expose it as an identifier because ids may be reused and are not guaranteed to persist.

```
@Node trait UuidNode {
    @unique val uuid: String = Helpers.uuidBase64
}

@Node trait Timestamp {
    val timestamp: Long = System.currentTimeMillis
}

@Node class Post extends UuidNode with Timestamp {
    var title: String
    var description: Option[String]
}
```

Figure 11: Nodes inheriting from multiple node traits in renesca-magic. This will generate wrapper classes and factories for all the defined node classes and node traits.

Factories for each node and relation are generated, which includes methods to either create, match or merge to instantiate the corresponding wrapper. These methods invoke the respective methods in renesca to create a raw item, which is then wrapped accordingly. In order to create or merge an item, all non-optional properties with a default value need to be supplied. For each property with a uniqueness constraint, a corresponding match function is generated in the factory of the wrapped node.

For example, one can match any subclass of the UuidNode trait with a certain uuid:

```
val uuidNode = UuidNode.matchesByUuid("abcdef")
```

It is also possible to create nodes and merge the relation:

```
val user = User.matchesByUuid("fedcba")
val post = Post.create(title = "hello world")
val relation = Authored.merge(user, post)
```

For local traversal of the graph, the node wrappers include accessors for retrieving neighboring nodes via the defined incoming and outgoing relations. Furthermore, relations have accessors for their start and end node.

The property-graph model does not offer native support for hyperrelations, but they can easily be modeled as a middle node with an incoming and an outgoing edge. In our implementation, a hyperrelation implements the node and relation interface and therefore shares all characteristics with them. Properties are handled equivalent as for normal nodes and are stored on the middle node. Analogously, the label for the middle node and relation type of the two relations is based on the name of hyperrelation class.

Hyperrelations can be instantiated like normal relations, and can be used as a node or relation.

Node and relations can be grouped in different graph wrappers. So it might make sense to define one wrapper for the discussion model, which only includes posts, connections and tags, as well as one wrapper for authentication, containing login information and user data. A graph wrapper is defined by all its contained nodes and relations, for which accessors are generated. A graph can be persisted just like in renesca and it is also possible to persist elements individually.

4.4. Discussion Model

As described in section 3.2, the proposed hypergraph-based discussion model consists of the following elements:

- **Post** A message from a user, which consists of a title and a description.
- **Connects** A directed hyperrelation between two posts or between a post and a connects relation. It describes that the start node of the relation responds to the end node. The start node of a connects relation is defined to be a post.
- **Context** Indicate the topic and scope of a post, a post can be connected to multiple contexts through a tags relation.
- **Classification** Describes the meaning or the type of a relation. Each connects and tags relation may connect to multiple classifications through a classifies relation.

A simplified implementation of our discussion model is shown in figure 12. It uses renesca-magic to define a schema for the database. It reuses the node traits *UuidNode* and *Timestamp* presented in figure 11, which adds a timestamp and an uuid property to the node. First, we define two node traits: *Connectable* which is a common interface for the *Post* and the *Connects* relation and *Reference* which is an interface for the *Connects* and *Tags* relations. The node class *Post* inherits from the *Connectable* trait and is defined by its title and its optional description; as both can be changed they are a variable. Then we define the recursive *Connects* hyperrelation, which implements the *Connectable* as well as the *Reference* trait. The start node of the relation is a *Post* and the end node can again be a *Connectable*, which is either a *Post* or *Connects* relation. As *Classifications* and *Contexts* share a lot of properties, we define a common trait for both of them: *TagLike*. Tags have a unique title, an optional description and an assigned color. A context is a tag. Multiple inheritance between *Contexts* is implemented with an *Inherits* relation, which can be matched when finding connected *Posts* in a *Contexts*. A *Context* may connect to a *Post* with the *Tags* hyperrelation, which implements the *Reference* trait. A *Classification* is also a tag with additional properties: a symbol which is shown in the UI and a precedence, which defines the order of the classification if there are multiple classifications. It can either classify a *Connects* or a *Tags* relation; it is modeled as a normal relation named *Classifies*.

```
@Node trait Connectable extends UuidNode with Timestamp
@Node trait Reference extends UuidNode

@Node class Post extends Connectable {
    var title: String
    var description: Option[String]
}

@HyperRelation class Connects(startNode: Post, endNode: Connectable)
   extends Connectable with Reference

@Node trait TagLike extends UuidNode {
    @unique val title: String
    var description: Option[String]
    var color: Long
}

@Node class Context extends TagLike

@Relation class Inherits(startNode: Context, endNode: Context)

@HyperRelation class Tags(startNode: Context, endNode: Post) extends
    Reference

@Node class Classification extends TagLike {
    val symbol: String
    val precedence: Long = 0
}

@Relation class Classifies(startNode: Classification, endNode: Reference)
```

Figure 12: Simplified implementation of the discourse model in Wust. The schema is defined using renesca-magic, which generates the boilerplate code for the type-safe graph wrapper.

A combined ER- and Venn-diagram of our discussion model is depicted in figure 13. There are three nodes in the graph: *Post*, *Context*, *Classification*. Furthermore, there are two hyperrelations: *Connects* as a response relation between two discussion elements and *Tags* as a tagging relation between a *Context* and its associated *Posts*. There is one simple relation: *Classifies* which links the classifications with the classified relations. As one can either respond to a *Post* node or to a *Connects* relation between two elements, both inherit from the trait *Connectable*. Accordingly, the *Connects* relation goes from the response to the responded *Post* or *Connects* between a *Post* and a *Connectable*. *Connects* and *Tags* relations can both be classified, so they both inherit from the base class *Reference* (blue). The *Classifies* relation goes from the *Classification* to the *Reference*.

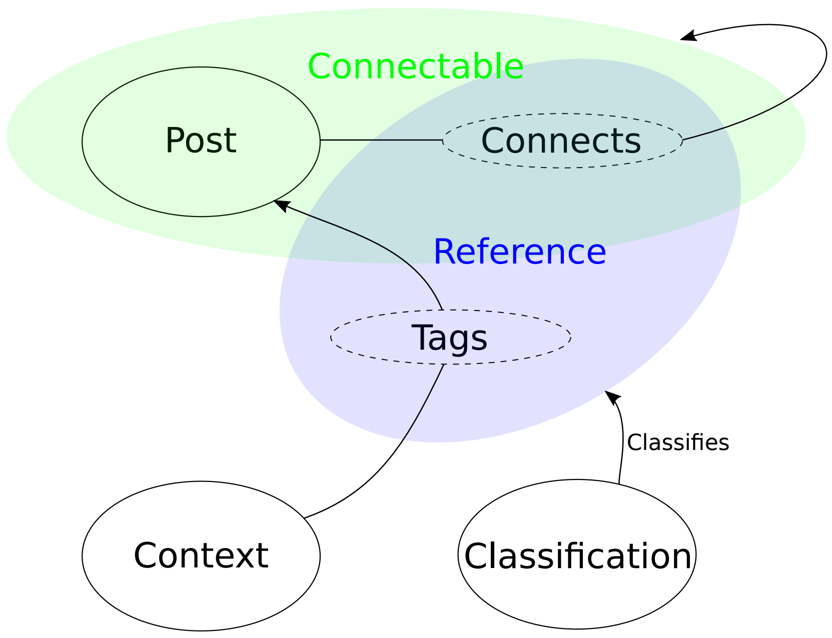


Figure 13: Combined ER- and Venn-diagram of the discussion model in Wust: *Post*, *Context*, *Classification*, *Classifies*, *Connects* (response relation between two elements), *Tags* (tagging relation between context and post). *Post* and *Connects* can be connected and inherit from *Connectable* (green). *Connects* and *Tags* can be classified and inherit from *Reference* (blue).

Due to implementing the website as a single page application, the server is rather thin and a lot of logic is moved to the client. The client needs to be able to retrieve and manipulate a lot of entities stored in the database. Writing controller functions for getting, posting or deleting all of the data can be error-prone and is tedious, especially when prototyping different schemata. Our type-safe schema already defines the model and renesca-magic provides interfaces for generically creating, matching or merging certain nodes and relations. Nodes can be identified via their uuid, which is inherited from the UuidNode trait depicted in figure 11. Therefore, we implemented a small DSL for rapidly defining a REST API based on the database schema, this allows to directly expose certain nodes or relations. The default functionality can easily be overridden for custom behavior which is often necessary for editing of entities and getting connected data around a node. From this DSL, we can generate JavaScript code for using the REST API, so it can directly be used in the client.

As renesca-magic generates classes and factories for the defined schema, we have a graph representation on the server, but the client also needs to be able to handle graphs programmatically. Also, the different views on the discussion, need a different representation of the underlying data: a hypergraph can either be interpreted as a graph where the hyperrelations are nodes or a graph where hyperrelations are edges. This is why we implemented a graph wrapper mechanism in ScalaJs. When retrieving the connected component of a post, the server returns a set of nodes and relations. This data is then wrapped for each view, the view can then freely operate on the graph wrapper and add properties as desired to nodes and relations. Changes can then be committed and thereby be populated to the remaining wrappers in one go. This allows editing and viewing of the same data set in different graph representations.

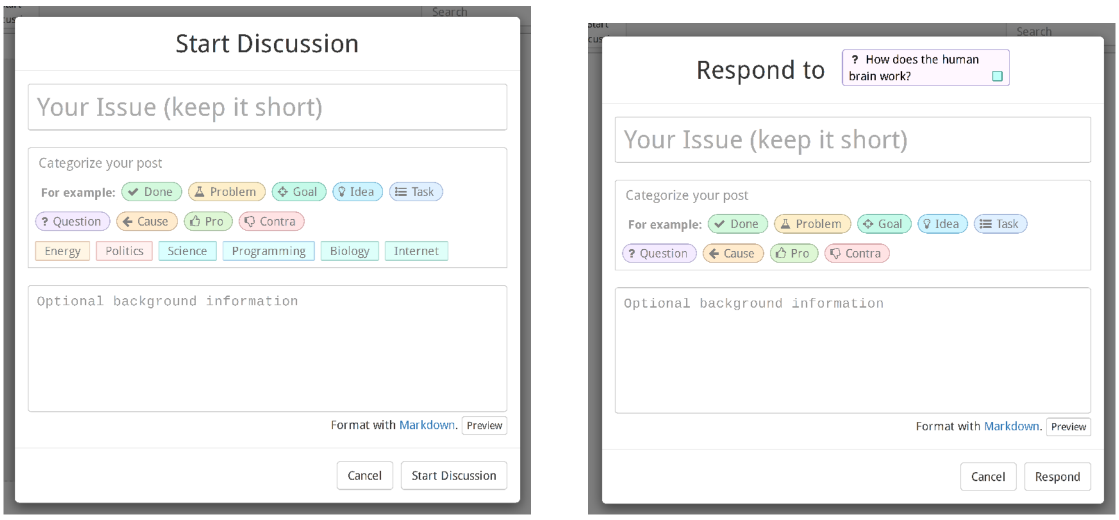
When creating a post or responding to another post, the user is prompted with a form consisting of three parts: (1) an input field for the title, (2) a tag editor for classifications and contexts, (3) an editor for an optional description. Figure 14(a) shows the form for starting a new discussion, figure 14(b) shows the form responding to another post. The title is mandatory and can have a length of at most 140 characters – it represents the essence or summary of the post. The tag editor displays contexts and classifications in a compact way. Each post can be assigned multiple contexts which creates a tagging relation to the post. Each classification – selected in the tag editor – is applied to every tagging relation as well as to responds connection. For a more fine-grained classification, it is possible to define a set of classifications per context. The description can be formatted with Markdown[21], which allows the user to easily format the content and embed links as well as pictures. In order to check the syntax of Markdown we use the Ace editor[22] which gives syntax highlighting of markdown while editing. The Markdown source is then converted to HTML using the JavaScript library marked[23]. Furthermore, a separate preview of the description can be displayed.

4.5. Discovering Content

Users need to be able to easily find posts and get an overview on the trends in different topics as well as recent posts on the platform. This requires a powerful search interface and a way to list existing posts for different topics.

The server provides a REST API for searching by different criteria. One can search by matching for desired keywords or by utilizing the underlying structure of the discussion – especially the classifications and contexts of posts to filter for certain kinds of posts. Therefore, it is possible to specify three filters: (1) a set of contexts and classifications

[21]http://daringfireball.net/projects/markdown/
[22]https://ace.c9.io/
[23]https://github.com/chjj/marked


Start Discussion
Your Issue (keep it short)
Categorize your post
For example: Done Problem Goal Idea Task
Question Cause Pro Contra
Energy Politics Science Programming Biology Internet
Optional background information
Format with Markdown. Preview
Cancel Start Discussion
Respond to
How does the human brain work?
Your Issue (keep it short)
Categorize your post
For example: Done Problem Goal Idea Task
Question Cause Pro Contra
Optional background information
Format with Markdown. Preview
Cancel Respond

(a) Start new discussion (b) Respond to existing post

Figure 14: Form for creating a new post. The form consists of three parts: (1) an input field for the title, (3) a combined tag editor for classifications and contexts, (3) an editor for an optional description.

that need to be present, (2) a set where minimum one needs to be present and (3) a set of ignored ones which may not be present. For example, one might search for all questions or problems in the context Scala which are not yet marked as done. The results can be sorted by time or, if a context is given, by quality within said context. Each search request is transformed into a single Cypher query.

Currently, keyword search is implemented with a regular expression which is expensive, because indices cannot be used for fuzzy string matching. This is not ideal and rather slow, as neo4j needs to traverse the whole database in order to check whether the regular expression matches. In a future version, we would like to use Lucene indices, which are supported by neo4j. This would allow for faster search queries and offer a way to sort the search results by relevance. The constraints for the contexts and classifications can be modeled as required, optional or non-existing relations in the graph. Remember that contexts are defined by a relation between a context and a post and that classifications are defined by a relation between a classification and a hyperrelation. This can be modeled in Cypher and performs efficiently.

The search API provides a convenient access for getting posts with specific criteria. Users can discover new content through three different interfaces: (1) search, (2) browse, (3) streams on the personal start page. The search form is always accessible as an input field in the navigation bar of the website. When typing into the input field, a result pages is shown which lists the results. On the left hand side the options for the search are listed – this responds to the parameters of the search API. The browse view lists all contexts on the left hand side – with a search form in order to search for a context by keyword. When selecting a context, all associated posts are listed on the right hand side; these are the entry points to discussions. The user can create streams on his start page. A stream is set of filters (parameters for the search API) and lists all recent posts that match the given filter. This can be used to be notified about new posts within a certain topic or with a certain classification.

When clicking on a post in the UI, the user is redirected to the focus-view of the post. Every post can be focused and has its own URL to be linked to. The page shows details about the posts and its neighborhood and offers a visualization of the whole discussion connected to the focused post.

4.6. Visualization

The UI should be able to provide the user with different views on a discussion, i.e., it should be possible to read and edit the same discussion in different views. As described in section 3.5, Wust offers a local and a global view on the discussion: focus-view and graph-view. Both views allow the same operations: reading, posts can be edited, one can respond to posts or connect existing posts.

The graph visualization of the discussion is implemented with the help of D3 (Data-Driven Documents), a generic JavaScript library for creating dynamic data visualizations in the browser[24]. It allows to bind a set of input data to elements in the DOM (Document Object Model) and provides a way to dynamically add, remove, and transform the elements depending on the data. The DOM is an expressive tool for visualizing data and D3 offers support for widely used standards like SVG, HTML and CSS. D3 supports visualizing a graph in a force layout. The force layout calculates positions for the defined nodes and edges in the graph. [BOH11]

Nodes are drawn as HTML elements. They are displayed as little boxes with the color of their classification, which also shows the title as well as the contexts. This allows consistent styling of posts throughout the application, as the CSS classes can be reused. Edges between nodes should be displayed as arrows, which is not easily achievable in HTML. Therefore, we added an SVG layer to the drawing area, on which edges are drawn as SVG paths with an arrow as their end marker. Both layers share the same coordinate system and are stacked on top of each other to draw the complete graph.

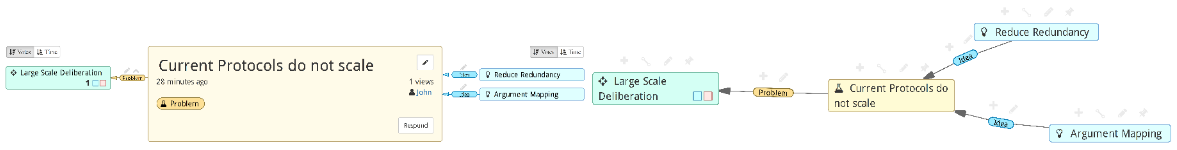
By default, when creating a force layout of some nodes and edges, the nodes are randomly positioned within the target element, which leads to an indeterministic conver-

[24]`http://d3js.org/`

gence. This is why we use the hashed id of the nodes as initial positions. Thus, the force layout convergence always results in the same layout.

The graph visualization is read from left to right (figure 15(b)). The root issue is displayed on the left side with its responses next to it on the right and so forth. There is no unique root issue as there might be multiple entry points to the discussion (posts that have a context). This can be achieved be setting up the appropriate forces which shape the force layout of the graph. A thorough explanation of the calculation on the layout implementation can be found in the Master thesis of Felix Dietze [Die15].

The focus-view only displays successors and predecessors of a post and provides a good overview of the direct neighborhood of the focused post (figure 15(a)). Both visualizations should emphasize the fact that both views operate on the same data and that the focus-view shows parts of a graph. Therefore, neighbors of the focused post are displayed as small boxes just like in the graph, using the same CSS classes. They can be sorted either by time or by quality. Furthermore, the relations between posts are displayed as SVG imitating the design of the SVG in the graph. The direction of the arrows goes from right to left and the parents are displayed on the left-side to have the same reading direction. The focused post in the middle is expanded to a bigger size, which allows to show further information like the time it was created, the author, and the description.



(a) Focus-view (b) Graph-view

Figure 15: Visualization of the same discussion in the focus-view and the graph-view. Both views are designed to share the same layout and use similar graphical elements.

When displaying posts and their relations, the contexts and classifications need to be visualized. In order to help the user to distinguish different discussion elements, distinct colors for contexts and classifications are used. Color spaces like RGB, Hue-Saturation-Lightness (HSL) or Hue-Saturation-Value (HSV) do not take into account how colors are perceived by the human eye – they appear in different intensities even when keeping saturation constant in HSL or HSV. The Hue-Chroma-Luminance (HCL) color space addresses this problem and we can therefore avoid the bias towards certain colors because of their appearance. D3 is used for calculating the color values in the HCL color space. Both, classifications and contexts, have a defined Hue value. For the controlled set of classifications, it is chosen manually. Each context is assigned a Hue value calculated from its hashed name. Posts are colored depending on the classifications defined through their outgoing relations. As a posts might have multiple classifications, the classifications are sorted by their precedence and the first one defines the color of the post. The contexts are shown as colored rectangles within the posts.

It would be interesting to use the quality score from the voting in the graph-view and arrange node depending on their quality or timestamp. This would allow to have the same order of the neighbors in the focus- and graph-view. Furthermore, high quality posts could be displayed more prominently, and it might be of interest to collapse the children of low-quality posts. Similarly, Reddit collapses posts in its threaded interface.

4.7. Real Time Updates

Collaboration software requires instant feedback about new contributions and editions of existing ones. The user should be notified about updates to the currently viewed discussion and should be able to interact with other persons in real time. Therefore, we implemented live updates with the help of WebSockets [FM11]. When viewing a discussion, the client sends a request to the server to be notified about changes to any node in the currently viewed discussion/component. The server then sends an update whenever a post was edited, retagged, got a response or was connected to another post. This also offers the possibility to inform the user about responses to followed posts or to get notifications about new post in the defined streams on his start page. It might be of interest to investigate the effect of a live chat for each discussion. This allows unstructured communication through a second channel, which could help for meta discussions.

4.8. Authentication

Authentication is implemented with the help of the Silhouette library[25]. It uses JSON Web Tokens [BSJ15] for authentication, which is retrieved from the server on login and then added to the header of each succeeding request sent out by the client. This has the advantage over cookie-based authentication that the server can be stateless, as there is no need for storing the session data on the server. This saves a round trip to the database for retrieving the session on each request, because the token is self-contained. The remaining session data is stored on the client in HTML5 Local Storage.

[25] `http://silhouette.mohiva.com/`

Since this approach does not rely on cookies, there is no need to protect against cross site requests: if an attacker would embed our site and generate a POST request to our server, he would not be able to use the authentication cookie as there is none.

4.9. Imports from other Platforms

While developing the discussion model and the graph visualization, we needed to have feedback on how real discussions - especially with a lot of posts - would be visualized in our system. Therefore, we implemented imports for three popular collaboration systems: *reddit*, *HackerNews* and *StackOverflow*. All the mentioned platforms provide a web API for extracting the content. Of those three, only *StackOverflow* has an underlying discussion model with questions, answers and flat comment lists, which could be imported as classifications into our system.

5. Formative Evaluation

The preceding chapters have introduced a web-based discussion system for collaboratively discussing in a structured graph. The model development was accompanied by one user test in order to get feedback on the first design decisions. Furthermore, the final implementation was evaluated qualitatively in a second user test and quantitatively with the aid of a questionnaire. This chapter reports on the evaluation of the presented system named Wust. It begins with a description of the hypotheses on which the presented discussion system is based (section 5.1). Then the user tests and the questionnaire are briefly described and the results regarding the hypotheses are presented (section 5.2). Finally, section 5.3 concludes this chapter with a discussion of the evaluation results.

5.1. Hypotheses

Design decisions in the system implementation are based on several hypotheses. The first hypothesis (**H1**) regards the usability and accessibility of the system, which consists of 5 related hypotheses (**H1a-H1e**) which examine the relationship between usability and other dimensions. The remaining 6 hypotheses (**H2-H8**) investigate the perception of the discussion model and graph representation, in order to test whether the user feels comfortable using a graph as the underlying discussion model. Felix Dietze evaluates further hypotheses regarding moderation and quality assurance in Wust [Die15].

H1: Users find the system usable. An online discussion system needs to be accessible for new as well as experienced users. Multifaceted arguments require a heterogeneous group of participants. Therefore, usability is a key goal of the implemented discussion system. It is important that new users can easily get a grasp of the basic concepts and that interested user find the system usable, regardless of prior knowledge about graph theory, self-efficacy using technology, experience with collaborative online platforms, sex or age.

H1a: Users find the system usable, regardless of their familiarity with graph theory
A graph-based structure might be branded to be only accessible for users already familiar with the basic concepts of graph theory. This hypothesis claims that this is not the case, as commonly users are already familiar with graph structures from drawing mindmaps – without being aware of the graph theory.

H1b: Users find the system usable, regardless of their self-efficacy using technology Some users may be confident at solving technical problems and handling technical devices, while others act carefully when technology is involved. Therefore, the barrier for unexperienced user should be low, which we try to achieve with a simple UI and a tutorial guiding through the system.

H1c: Users find the system usable, regardless of prior experiences with collaborative online platforms Wust makes use of some concepts already known from existing online collaboration systems, which might be an advantage for experienced users. Nevertheless, we claim that experience with concepts like tagging, nesting of responses, community moderation and voting play a minor role when being confronted with the interface of Wust.

H1d: Users find the system usable, regardless of their sex Modeling and understanding discussions in a graph representation should not be influenced by the sex of the user.

H1e: Users find the system usable, regardless of their age The perception of usability should not depend on the age of the user. It is important for balanced discussions to represent a multitude of viewpoints from a diverse audience.

H2: Users find that the graph representation gives a good overview of the discussion The graph representation plays an important role for the visualization of the discussion. It is motivated by the explicitness of graphs which can flexibly visualize relationships between elements and their meaning. Both visualizations, the focus-view and the graph-view, can show all the information which might be overwhelming for some users. We claim that this is not the case and that the explicitness helps for getting an overview of the whole discussion and its arguments.

H3: Users are able to express themselves in the system Because of classifications and different connection points, it might be difficult for users to integrate their submission into the discussion. It is important for discussions to be easily accessible for the contributors, such that new ideas and arguments can quickly be added.

H4: Users find structure in discussion systems more important than chronological order Especially traditional internet forums employ a chronological order to arrange the posts in a thread. For example, comment lists are often sorted by time. In contrast, a graph-based discussion model and visualization naturally emphasizes the structure and context of posts. This hypothesis is a strong reason for a more structured representation of dialogue instead of capturing a timeline of submissions.

H5: Users make use of classifications by themselves The proposed discussion model depends on the user to consistently label relations between discussion elements. Each relation can have multiple classifications, which can be selected from a predefined set of classifications. The user can classify the relation when connecting two elements or when responding to another element, and it is also possible to retroactively change the classifications. The design decisions for the visualization of the discussion are based on the existence of proper classifications and therefore on the motivation of the user to classify relations.

H7: Users see the need for hyperrelations One of the main differences between the proposed discussion model and the IBIS model is the possibility to argument about the relation between two discussion elements. This provides a structure for questioning or supporting the relationship between elements and separates these arguments from the ones only concerning a single node. The model depends on the users to see the need for hyperrelations and to actually utilize them. If only a minority of the users make use of hyperrelations, it could lead to an inconsistent structure where similar arguments are connected to the relation and others are connected to the node.

H8: Users find the technical reading direction in graphs intuitive Both visualization methods, the focus-view and the graph-view, arrange posts from left-to-right. This means that the root issue is placed on the left. From there the responses are located to the right of the node they refer to. The direction of the arrows also goes from left-to-right. If post A responds to another post B, the relation goes from A to B. The used classifications specify how one post relates to another and can be read as, e.g., *A is an idea for B*. This corresponds to the technical reading direction in graphs. The first user test yielded mixed responses to this question and therefore we decided to survey this in the questionnaire in order to reason about an intuitive reading direction. Also, existing argument mapping system make use of both arrow and reading directions.

5.2. Method and Results

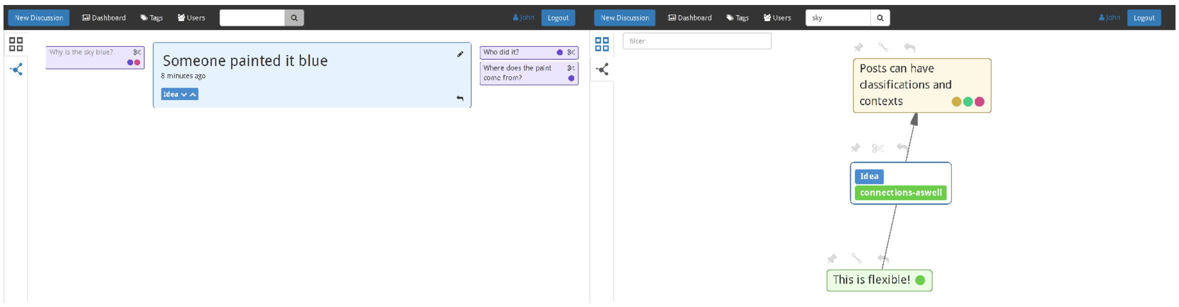
In order to test the above mentioned hypotheses, two user tests and a survey were conducted. The first user test was done to test the overall perception of the graph representation and the usage of classifications as well as to find glitches in the user experience (section 5.2.1). Succeeding design decisions and the final implementation were qualitatively evaluated during a second user test (section 5.2.2). For quantitative evaluation, we have conducted a survey with a questionnaire which is described in section 5.2.3. Wust was equipped with a tutorial which users could read to get a grasp of the basic idea and get familiar with the user interface. The users were asked to follow the steps in the tutorial and explore the system; after finishing the tutorial they were provided with a link to the questionnaire. Additionally, all participants from the second user test were asked to participate in the questionnaire. The results of the evaluation and the test of the hypotheses is discussed in section 5.2.4.

5.2.1. First User Test

During the development of Wust, a first user test was conducted. The goal was to find out whether users are generally capable of understanding discussions modeled as graph. Therefore, the user test covers content creation and navigation in the graph-view as well as in the focus-view. Moreover, the user test helped to find major UI flaws that made the system difficult to use.

The first user test was performed with four students. Each participant in the user test had to perform a set of twelve tasks. He was equipped with a laptop running a browser showing the website of Wust. The screen and the audio was recorded during the whole session. The session was supervised and ended with an open discussion for feedback and questions regarding the system.

State of development During the first user test, a first prototype of Wust was evaluated. In contrast to the presented discussion model, classifications and contexts could be assigned to either posts or relations. The idea was to provide a similar infrastructure as with IBIS like systems, the user specifies a node as an *idea* or *question* and labels the relation with *solves* or *repliesTo*. Discussions were visualized similar to the current version but the focus-view did not show the relation between posts and the layout in the graph-view did not reflect the structure of the arguments (figure 16).



Why is the sky blue?

Someone painted it blue

8 minutes ago

Who did it?

Where does the paint come from?

Posts can have classifications and contexts

This is flexible!

(a) Focus-view

(b) Graph-view

Figure 16: State of development during the first user test

Tasks At the start of the user test, each participant was viewing one specific post in the focus-view. For simplicity, the user was already logged in. The participants were asked to read the currently displayed messages and to describe the view in their own words. Another task was to navigate through the responses in the focus-view, they had to focus one response and then go back to the previously focused post. The list of responses included spam, which they had to remove. As a next step, the participants were instructed to switch to the graph-view in order to get a clearer picture of the whole discussion and its relations. There, the participants were assigned the task to respond to a problem and connect the newly created idea to another, already existing problem. Afterwards, they were asked to remove the previously created relation. As a last step, the participant needed to create a new discussion, and to edit the description after creation.

Results Observations during the first user test helped us in the development of the next iteration of the discussion model. Moreover, we were able to fix some inconsistencies in the UI, which hindered usability. Our key insights from the first user test are the following:

- **Users understand a discussion represented as a graph:** At first sight, users seemed to be confused by the reading order in the focus-view, but quickly got used to having the parents on the left and responses on the right side. Users quickly understood the discussion and the relations between the discussion elements when looking at the graph-view. It made sense to the users to model arguments in a graph, where nodes can respond to multiple parents. Moreover, the participants described that the graph-view reminded them of a mindmap. Some participants suggested to display the graph arrows in the focus-view, which we have implemented in the final version of Wust.

- **Users classify posts when creating them:** The concept of classifying posts in a new discussion or as a response to another node seemed intuitive for the users. Classifications were used by all users and were applied in a meaningful manner. Users seem to be motivated to classify their posts in order to gain more structure. This observation reinforced our aim to develop a structured discussion which relies on users to label their contributions accordingly.
- **Users use post-classifications for relations:** One task required the users to connect posts in the graph-view. After connecting two posts with a free-hand tool, a modal appears to classify the connection between the posts. The tag editor suggested some classifications, which were selected by the users to describe how the two posts relate to another. But they also made use of post-classifications for relations, e.g., using the classification idea for a relation. In our implementation we have adopted this and implement classifications only on relations, which implicitly also defines the type of the start node of said relation.
- **Users confuse tags with buttons and click them instead of the reply button:** In the focus-view of the tested prototype, tags were prominently visualized as colored rectangles with white foreground text. Participants confused the question-classification with a button when they wanted to ask something – instead of using the button with the reply-symbol. The reply-symbol was not intuitive for them, as a reply was strongly associated with an answer and not with, e.g., a question. After the user test we tried different symbols and button labels; in our experience, the button text "Respond" worked best. Furthermore, the final version of Wust now uses lighter background colors with a black foreground text.
- **Users suggest that the graph view should be more organized:** Participants claimed that the graph-view did give a good overview, but were disappointed by the automatic layout of the graph-view. At this point, the start post of the discussion was placed in the middle – with child nodes randomly placed around it and the final positions calculated in a force layout. In contrast, the focus-view shows the parents of a post on the left side and the responses on the right, which seemed to make more sense to the users. This finding motivated us to use the same layout in the focus-view and the graph-view; now, both views list posts and their responses from left-to-right

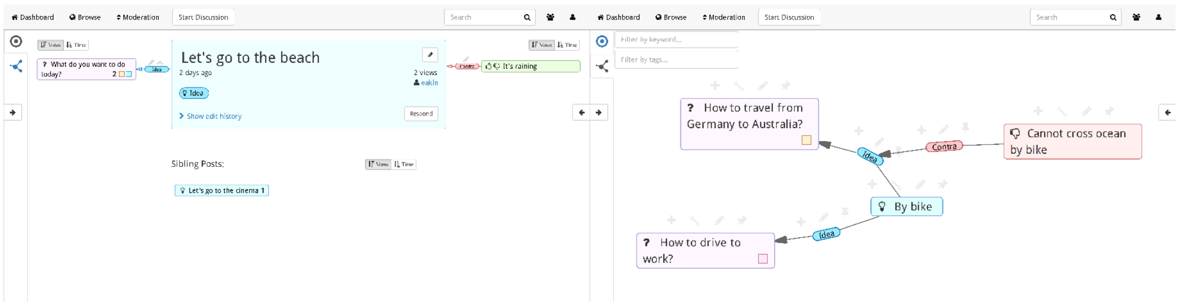
All in all, the first user test showed us that users were capable of understanding the graph structure for discussions. The way how users interacted with the graph gave us insights on how to improve the model to make it more accessible. Moreover, the user test helped in finding inconsistencies and problems in the UI that needed to be made more intuitive.

5.2.2. Second User Test

In order to test the final implementation of Wust and the presented discussion model, we have conducted a second user test. It had four goals: (1) verify the findings from the first user test, (2) observe how users handle hyperrelations as well as relation classifications and whether they would make use of them, (3) evaluate the moderation approach, (4) evaluate the changes in the UI.

The second user test involved eleven participants. The participants were confronted with a set of 19 tasks about the graph representation including the usage of hyperrelations and community moderation.

State of development Except for some bug fixes and UI adaptations, the tested system corresponds to the final version of Wust. For comparison with the state of development in the first user test, figure 17 shows the revised version of the focus-view and graph-view. Both views are designed to look similar and display content in the same order to emphasize that both views operate on the same discussion.



(a) Focus-view (b) Graph-view

Figure 17: State of development during the second user test

Tasks As in the first user test, each participant started in the focus-view of a specific discussion. Again, the participants were first asked to navigate through the responses in the focus-view. They were further instructed to ask a question to the currently focused post. Then they had to switch to the graph-view, where they were asked to describe whether the graph-view provides a good overview. As before, the participants needed to create an idea in the discussion and had to connect it to two different posts. Afterwards, they had to remove one of the newly created relations. As a next step, the participants were instructed to navigate to another discussion with one idea which responses to two questions. There, they needed to add an extra contra argument which only applies to the relation between one question and the idea (figure 17(b)). This was done to test whether they would find the hyperrelation feature and make use of it. In the last part of the user test, the participants had to edit a post from another user, which created a change request. Furthermore, the participants had to accept and reject change requests from other users in the vote-stream, which displays a queue of change requests that need to evaluated. They had the task of rating two change requests. Again, they were confronted with spam and had to delete it. The last task was to cope with a clarification question connected to a post, which they had to edit in order to make the clarification obsolete. They could decide for themselves, how they would like to resolve this redundancy – either mark it as done, disconnect it or delete it.

Results Overall the second user test yields good results. Problems with the UI – that were apparent in the first user test – are fixed. The graph view seems to give a good overview of the whole discussion and the concept of hyperrelations is applicable for most users. We have identified the following key insights from the user test:

- **Users understand the concept of hyperrelations:** Hyperrelations are useful for modeling responses to the linkage of two nodes. Users were able to make use of the concept when they had to attach a contra argument to the idea which refers to two different questions. Without being informed about the feature beforehand, most user attached the argument to the idea; only some discovered this by themselves and made use of the hyperrelation. After being told about the possibility to attach the argument to the relation between question and idea, most of the users reconnected the argument and attached it to the relation. In the discussion, two mathematicians claimed that the concept of hyperrelations might be too difficult for most people.
- **The majority of users can handle classifications on relations:** During the test, users applied classifications to responses through a simple form and were able to edit the classification of a relation. In one task, the participants had to connect an idea to two different parent posts. After the first relation was created with the classification idea, some users seemed to think that the post was thereby marked as an idea. About one third of them saw no need in classifying the second relation as an idea. The majority of the users correctly applied the labels for both relations. Overall, most of the users applied classifications consistently when responding to another post or starting a new discussion in a context. The combined tag editor seems to be intuitive. But some seemed to be confused by

the fact that the classification can only be edited on the relation and not directly on the post. The results indicate that it might be useful if the UI would suggest a classification if derivable from the existing relations, as multiple classifications per node are seldomly applicable.

- **Half of the users eliminate redundancy:** When users were confronted with the redundant clarification question, about half of them decided to delete or disconnect the question and the other half decided to mark the question as done. So, some seem to be in favor of deleting the redundant information and some seemed to be in favor of keeping a history of the discussion with marked decisions.

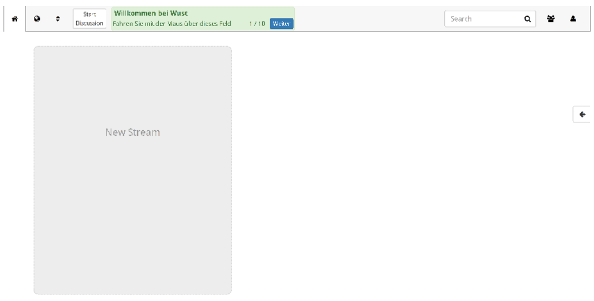
The results of the second user test indicate that the revised discussion model and visualization are more intuitive than in the first user test. The participants were generally comfortable using the system and were able to model discussions without prior introduction. Still, some features like hyperrelations or classifications need more explanation so users can easily get a grasp of the concept. Before handing out the questionnaire, some minor annoyances and bugs – that became apparent during the user test – were fixed Additionally, we added more tooltips and descriptive texts to make the system more user-friendly for beginners.

5.2.3. Questionnaire

To collect data for reasoning about the hypotheses mentioned in section 5.1, a questionnaire was created to evaluate the perception of the system. The questionnaire consists of 17 questions about the usability and the perception of the key concepts of the discussion system. Each participant of this survey was provided with a link to the live system – a single instance of Wust, which was shared by all users.

The live system included a tutorial of ten steps to make the user familiar with the UI and the discussion model. Each step corresponds to a task that the user has to fulfill. For example, the user was asked to click on an existing discussion to inspect the graph-view and the focus-view. Furthermore, he was asked to respond to other contributions and to try out connecting existing posts to other discussion elements. After finishing the tutorial, the user was provided with a link to the questionnaire and was free to further browse through the website and discuss more issues. The tutorial was integrated into the navigation bar to make it visible while browsing the website (see figure 18). On hover, the tutorial expands to show the full description of the current task.

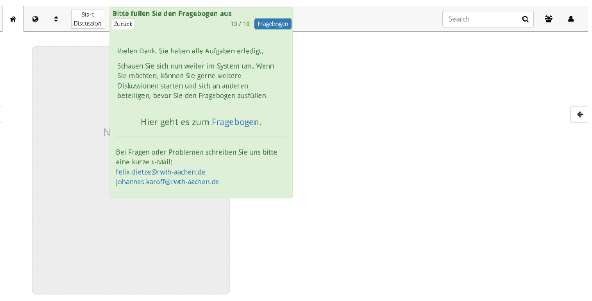
In order to evaluate our hypotheses, t-tests were used and correlations were calculated. We chose $\alpha = 0.05$ as the significance level, which means that in 5% of the cases we find



Willkommen bei Wust

Fahren Sie mit der Maus über dieses Feld

(a) Collapsed tutorial



Bitte füllen Sie den Fragebogen aus

Vielen Dank, Sie haben alle Aufgaben erledigt.

Schauen Sie sich nun weiter im System um. Wenn Sie möchten, können Sie gerne weitere Diskussionen starten und sich an anderen beteiligen, bevor Sie den Fragebogen ausfüllen.

Hier geht es zum Fragebogen.

Bei Fragen oder Problemen schreiben Sie uns bitte eine kurze E-Mail:
felix.dietze@rwth-aachen.de
johannes.karoff@rwth-aachen.de

(b) Finished tutorial

Figure 18: The tutorial is integrated into the navigation bar of the website. After finishing the tutorial, the user is provided with a link to the questionnaire.

a significant effect or relationship, even if there is none. As sensitivity, $\beta = 0.20$ was chosen, i.e. in 20% of the cases we are unable to find a significant effect or relationship, even if it exists. The threshold for the minimal effect size of our sample was calculated with G*Power. All items are collected on a six-point Likert scale, which ranges from 1 (*strongly disagree*) to 6 (*strongly agree*).

We have a sample size of 51 persons, out of which 15 are female and 35 are male – this was missing in one case. The average age is 30.90 ($SD = 12.54$).

5.2.4. Results

The proposed discussion system Wust was qualitatively evaluated in two user tests and was quantitatively evaluated on the basis of a questionnaire. This section reports on the results of the evaluation and reasons about the hypotheses presented in section 5.1. The hypotheses are the basis of the key concepts implemented in Wust, which includes the hypergraph-based discussion model and the graph representation as well as the overall usability.

H1: Users find the system usable The survey evaluates the System Usability Scale (SUS) [Bro96], a six-point Likert scale of ten items for measuring usability with regards to effectiveness, efficiency and satisfaction. For our sample, the SUS scale shows a good reliability ($Cronbach's\ \alpha = 0.867$). The *mean* of the SUS scale is 4.49 ($SD = 0.77$). The calculated SUS score is 70.45 (in a range from 0 to 100), which is above average. Results for the SUS scale are depicted in the box plot in figure 19 and let us conclude that the system is was generally perceived as usable in our sample. In the following, we explore whether there is a correlation for any of the collected characteristics.

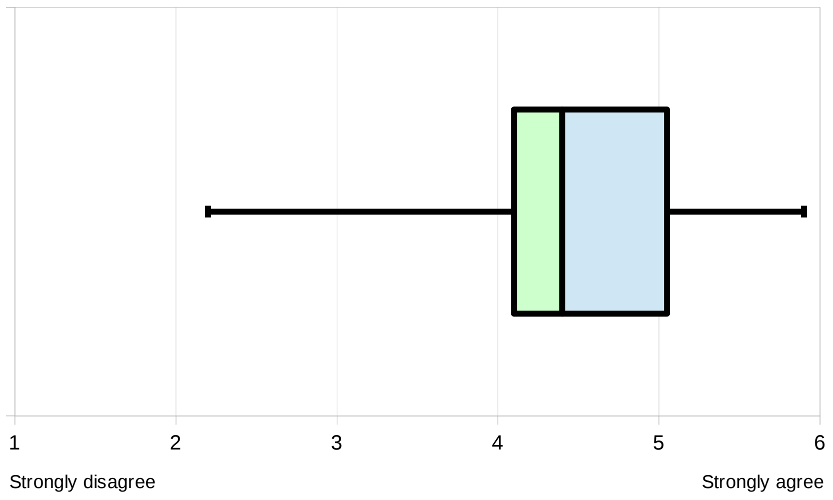
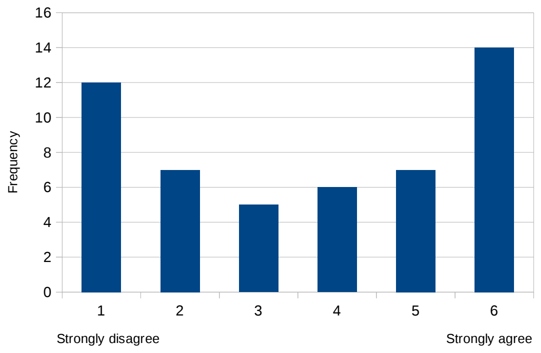


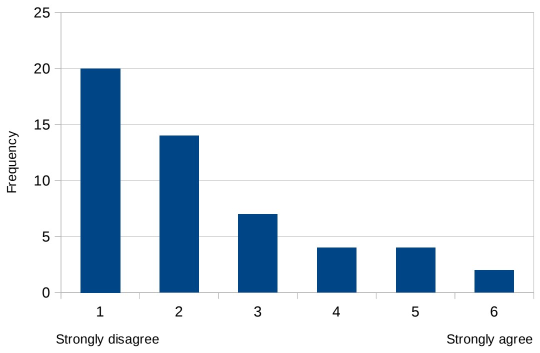
Figure 19: Question 6 - Box plot of the System Usability Scale

H1a: Users find the system usable, regardless of their familiarity with graph theory

In the questionnaire, question 5 gives the participant the opportunity to specify how familiar he is with graph theory. The participants could specify their familiarity with *vertices and edges* as well as their familiarity with *hyperedges*. Figure 20 shows the distribution of the results for both statements. Within our sample, the knowledge about vertices and edges in graph theory has a *mean* of 3.71 ($SD = 1.94$) and follows a bimodal distribution - participants are either very familiar or not familiar with the basic concept of graph theory. Less participants were familiar with hyperedges with a *mean* of 2.32 ($SD = 1.46$). Compared to the total population in Germany, this is probably a rather high prevalence of graph theory knowledge.



(a) Distribution for the item *vertices and edges*



(b) Distribution for the item *hyperedges*

Figure 20: Question 5 - *How familiar are you with the following concepts from graph theory?* Frequency of the level of familiarity with *vertices and edges* and *hyperedges*.

In the context of the usability of Wust, there is neither a significant correlation between the knowledge about vertices and edges and the SUS scale ($\rho = 0.023$, $p = 0.878$) nor between the knowledge about hyperedges and the SUS scale ($\rho = 0.106$, $p = 0.464$). This indicates that there is no relationship between the knowledge of graph theory and the usability of the system.

H1b: Users find the system usable, regardless of their self-efficacy using technology The self-efficacy using technology is evaluated on an eight-item Likert scale (question 16). Participants were asked to specify how much they agree to statements regarding their behavior when confronted with a technical problem or device. The overall *mean* of the Likert scale is rather high with 3.80 ($SD = 0.76$). Out of 51 participants only 41 answered this question – though, one participant only answered nine items. The scale shows a good reliability in our sample (*Cronbach's* $\alpha = 0.873$). We could not find a significant correlation between the SUS scale and the self-efficacy using technology ($r = 0.073$, $p = 0.650$). Therefore, we argue that the system is usable, regardless of the user's self-efficacy using technology.

H1c: Users find the system usable, regardless of prior experiences with collaborative online platforms Question 3 surveys how familiar a user is with different online platforms. The user can select how often he uses popular online collaboration platforms like forums or Facebook, mailing lists, Wikis, StackOverflow, reddit, IRC and issue trackers. For our sample, the scale has an acceptable reliability (*Cronbach's* $\alpha = 0.731$). The seven-item Likert scale for the usage of online platforms has a *mean* of 4.02 ($SD = 1.20$). We could not find a significant correlation between the usage frequency of online platforms and the SUS scale ($r = -0.198$, $p = 0.163$). This lets us conclude that the system is usable, regardless of prior experience with related online platforms.

H1d: Users find the system usable, regardless of their sex The *mean* of SUS for males is 4.55 ($SD = 0.68$) and the *mean* for females is 4.34 ($SD = 0.96$). With a distribution of 15 females and 35 males, the threshold for undetectable differences is $d = 1.14$. The t-test for independent variables did not show any differences between the sexes ($T(48) = 0.891$, $p = 0.377$, Variance equality: $F = 2.179$, $p = 0.146$). Therefore, we can conclude that the system's usability does not depend on the user's sex.

H1e: Users find the system usable, regardless of their age The average age in this survey is 30.90 ($SD = 12.54$). There is no significant correlation between the age and the SUS scale ($r = 0.242$, $p = 0.087$). This indicates that there is no relationship between the age of the participants and how they perceive the usability of the system.

H2: Users find that the graph representation gives a good overview of the discussion Question 7 surveys the perception of some key concepts of Wust, its graph visualization and its discussion model. This includes the graph representation as well as its arrangement for getting an overview and reading a discussion. The subjects were asked how much they would agree to the following statements: *The arrangement of the contributions helps me to comprehend the relationship between contributions*, *The arrangement of the contributions helps me to follow the discussion* and *The arrangement of the contributions helps me to conceive all important arguments*. Figure 21 shows a box plot for the results. The reliability of the three items is acceptable (*Cronbach's* $\alpha = 0.719$). The *mean* of the Likert scale is 4.82 ($SD = 0.72$) with a *median* of 5 – the minimal recorded value is 3. This indicates that users find the arrangement of the elements in the graphical representation useful for getting an overview and to identify important arguments in the discussion.

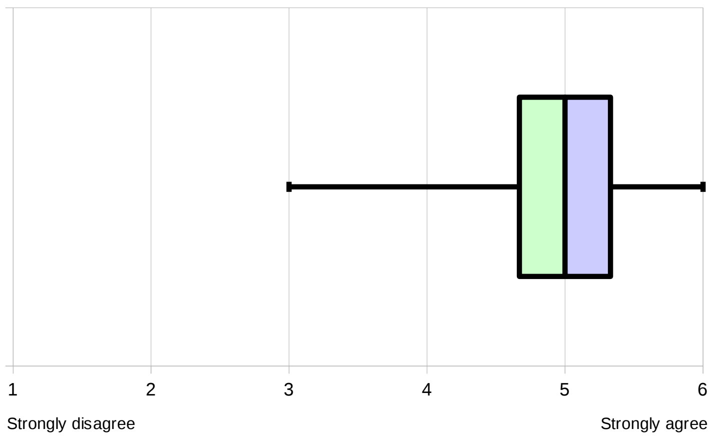


Figure 21: Question 7 - *The arrangement of the contributions provides a good overview of the discussion.* Box plot of the level of agreement.

H3: Users are able to express themselves in the system This survey tries to capture whether users are capable of expressing themselves in the discussion system and feel that their contributions fit into the structure (question 7). They were asked to express their agreement to the following statements: *I can express my thoughts in the system*, *I find it easy to integrate my post into the discussion* and *The system makes it easy for me to contribute to the discussion*. Figure 22 displays a box plot for the results. The three items have an acceptable reliability (*Cronbach's* $\alpha = 0.771$). The Likert scale has a *mean* of 4.76 ($SD = 0.79$) and thereby indicates a good result; the *median* is 4.84 with values ranging from 3 to 6. These results imply that users in our sample are confident that they can contribute to the structured argument map in a constructive way.

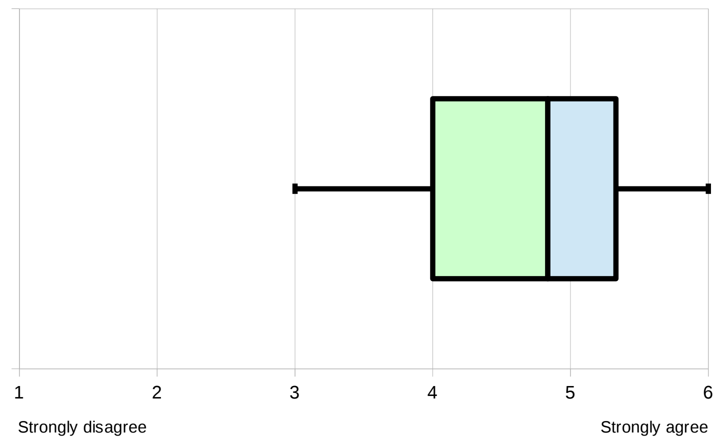


Figure 22: Question 7 - *The system enables the user to express himself.* Box plot of the level of agreement.

H4: Users find structure in discussion systems more important than chronological order In order to find out which sorting order for posts is preferred by the participants, question 8 surveys the importance of the sorting criteria: *It is important to me that posts are sorted according to the following criteria.* There were three options to chose from: (1) Strictly chronological, (2) context/structure, e.g., close to the responded post, (3) quality rating of other users. The results of this question are depicted in figure 23. Chronological order is rated less important ($M = 3.16$, $SD = 1.07$) than the structure ($M = 5.27$, $SD = 0.83$). The comparison between chronological order and structural order yields a big effect ($d = 2.2$). This is a highly significant difference ($T(50) = 10.42$, $p < 0.001$) in the t-test for paired samples and a strong indicator for a more structure-aware discussion system. The quality rating also seems to be very important to the users ($M = 4.76$, $SD = 1.32$), but less than the structure. The t-test for paired samples shows a significant difference between quality and structure ($T(50) = 2.38$, $p = 0.021$).

H5: Users make use of classifications by themselves When creating a post or when responding, the form for adding classifications and contexts is prominently displayed below the title field. The form provides an auto-completion feature for selecting tags by name and recommends suggestions, which can be selected by mouse-click or by pressing the tab key for cycling through the possibilities. This should make it easy for users to add classifications and contexts.

The first and the second user test showed that users had no problem using classifications and most of them consistently classified their contributions. Still, some of them seemed to be confused about the need to classify multiple outgoing relation of one node, which might be an indicator that this should be made more explicit in the UI. In

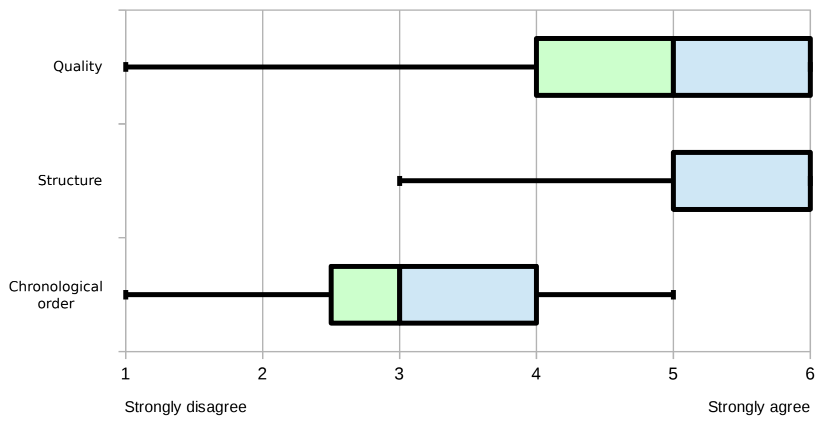


Figure 23: Question 8 - *It is important to me that posts are sorted according to the following criteria.* Box plot of the level of agreement to the possible criteria *strictly chronological*, *context/structure, e.g., close to the responded post* and *quality rating of other users.*

the test system for the participants of the questionnaire a total of 307 nodes and 308 relations were created by 63 users. Out of those, 277 relations had a minimum of one classification, which indicates that users indeed make use of the classifications. But the results from the test system should be interpreted with caution, because the discussions were partially a result of following the tutorial, which explicitly advises users to use classifications. Additionally, some participants claimed that the classifications might be too similar and that they were unsure which one to use; for example the contra and the problem classification. Also, some users requested a classification for expressing loose relatedness between elements. All in all, most participants of the survey made use of classifications and applied them appropriately.

H6: Users see the need for hyperrelations In question 11, we confronted the participants with an already existing discussion with two questions and one idea that was connected to both questions (figure 7). There are two questions: “How to drive to work?” and “How to travel from Germany to Australia?”. The post “By bike” is connected to both questions as an idea, which enables the possibility to make use of hyperrelations. The questionnaire asks where the participant would connect a certain argument, which was specifically designed to only relate to one of the two relations: “Cannot cross ocean by bike”. Participants could select whether the argument should be attached to one of the questions, to the idea or to one of the relations. Multiple answers were allowed.

The vast majority of the participants (80%) were in favor of connecting the argument to the incoming relation of “How to travel from Germany to Australia?”, which corresponds to the hypothesis that there is a need for hyperrelations. Still, one third (33%) of the users also felt that the argument can be connected to the response "By bike", 16% of the users would connect the new argument to the question "How to travel from Germany to Australia?". Interestingly, a few users (4%) would connect it to the incoming relation of the question "How to drive to work?". Nobody considered connecting the argument to the unrelated question "How to drive to work?". The frequency of the responses is plotted in figure 24. In the second user test, the participants were initially not informed about the possibility to connect posts to hyperrelations and therefore most of them did not consider it. Few users discovered the feature by themselves. In contrast, the participants in the questionnaire were aware of the feature because it was mentioned in the tutorial.

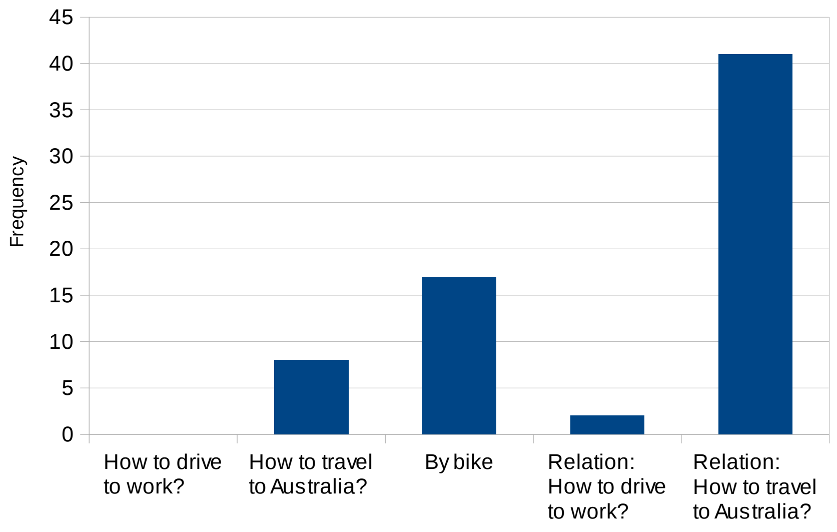


Figure 24: Question 11 - *To which point would you connect the argument "Cannot cross ocean by bike" in the discussion from figure 7?* Frequency of the responses.

H7: Users find the technical reading direction in graphs intuitive Question 10 asks the user which direction of the arrow and reading direction they perceive as correct with four options to choose from: *A is a question about B. Which depiction do you perceive as correct?* Figure 25 shows a box plot of the results for the four options, each measured with a 6-point Likert scale. The first and the last option both represent the usual arrow direction in graphs, A *is a question for* B, whereas the second and third option are in the reverse direction where responses to a node are represented by its outgoing relations. The options represent both arrow direction in both reading directions (left-to-right and right-to-left). Most users prefer the direction *A to B* with a *mean* of 5.50 ($SD = 1.02$). The other option with the technical arrow direction *B from A* has a *mean* of 4.81 ($SD = 1.45$), which is also good. Both depictions which reverse the arrow direction, *B to A* ($M = 1.70$, $SD = 1.02$) and *A from B* ($M = 1.68$,

$SD = 1.07$), are not popular. These results indicate that the participants strongly prefer the technical arrow direction in either reading direction.

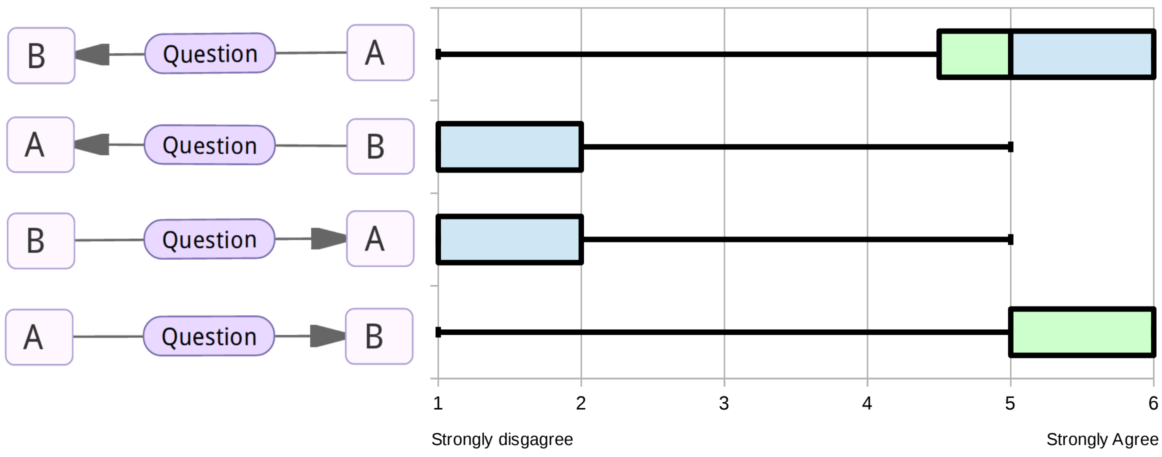


Figure 25: Question 10 - *A is a question about B. Which depiction do you perceive as correct?* Box plot of the level of agreement for the possible options *B from A*, *A from B*, *B to A* and *A to B*.

Even though, this is a strong indication that the technical arrow direction is indeed intuitive, the results should be interpreted with care. The participants first used the discussion system and afterwards filled out the questionnaire. This is why there might be a slight bias towards the employed direction used in the system. It would be worth to evaluate this without previously using a certain system. Both reading direction are perceived positively with a tendency towards having responses of a node on its left side (right-to-left). In contrast, the presented discussion system arranges posts in the opposite reading direction (left-to-right).

5.3. Discussion

Wust was evaluated with regards to usability and acceptance of the employed key concepts. The underlying hypotheses were examined with two user tests and a questionnaire. The overall usability of the system is rated positively by the participants in our sample. There are no significant correlations between the usability and other dimensions: (1) knowledge in graph theory, (2) self-efficacy using technology, (3) prior experience with online platforms, (4) sex, (5) age. In contrast to the usability, the evaluation of the Net-Promoter-Score [Rei03] did not yield such good results. Out of 51 participants, 12 are promoters and 22 are detractors, which constitutes a Net-Promoter-Score of -19.61%.

However, the graph structure seems to help users to comprehend an existing discussion and they feel comfortable contributing to the system. The graph-view can be valuable for getting an overview of a discussion and comprehend arguments for a specific position. Additionally, the structure-based arrangement of contributions shifts the focus away from the timeline of the discussion. The conducted evaluation supports our claim that the structure is of higher importance to the reader than the chronological order. This confirms the decision to model discussions as a graph in order to gain explicit information about relations. Relations can be classified in a flexible manner, which requires the user to consistently label their responses. In the user tests, we found that users generally had no problems assigning classifications and using them appropriately. Modeling arguments for a connection on the relation itself proves to be a valuable feature, most of the user see a need for hyperrelations.

During the survey, all participants worked on the same live system and were able to discuss with each other. But most of the submissions were created during the tutorial which has led to a lot of simple discussions and meaningless responses. In the feedback section of the questionnaire, some participants have mentioned that they would be curious how the system would behave in a discussion about a more complex topic. This is definitely a drawback of this survey, as we did not examine the evolution of complex discussions over a longer time period. Still, the evaluation yields good results for the general concepts and the usability of Wust. It would be very interesting to conduct a more extensive study to investigate the perception of Wust for discussing real world problems.

6. Conclusion and Future Work

This thesis proposes the hypergraph-based discussion system Wust. A post can respond to another post or to the relation between two discussion elements, which is a specialized form of the mathematical definition of hyperedges. The meaning of the relation between two elements can be expressed through a predefined set of classifications, e.g., a post *is an idea for* another post or a post *is a contra argument for* the link between two different posts. Posts can be categorized through a collaborative tagging mechanism. The system is implemented as a website, which makes it easily accessible for a broader public. The model aims at providing a flexible structure for modeling arguments and discussions about difficult problems and for collaboratively planning decisions.

Overall, the key concepts turned out to work and it is possible to collaboratively model structured arguments in a flexible manner. The graph representation generally helps to get an overview of an existing discussion. However, very large discussions with a lot of nodes and relations can be difficult to navigate in the graph-view. The filtering techniques for keywords and tags do not provide enough settings to sufficiently highlight the focused area. For this, the focus-view is very valuable but with a lot of responses, this view degrades to a list of posts. The lack of a more compact graph visualization hinders exploration of large and nested maps. This could be improved with a global graph visualization which reduces the amount of details for better overview. Generally, there should be more views for the user to chose from to fit different reading requirements.

It would be interesting to investigate how classifications and hyperrelations are used in large discussions and whether they are assigned consistently. Moreover, we expect that the tagging mechanism helps to categorize the content for the user, but a more distinct separation of content might be desirable. This is why the discussion system needs to be evaluated in a broader scope to reason how arguments evolve over time when discussing real world problems. Here it would be worth to evaluate different sets of classifications and to explore whether nouns or verbs should be used for classifying relations.

All in all, the formative evaluation of Wust yields good results. The evaluation indicates that the system is usable regardless of the user's prior experience and background. Furthermore, the participants reported that the structure and arrangement of the posts helps to get an overview of the discussion and to comprehend the argumentation. Regarding the contribution of new content, participants felt comfortable expressing themselves and were able to integrate their contributions into the discussion. Towards the end of this thesis, we started using Wust as an issue tracker for itself. This worked really well for keeping track of bugs and ideas. Additionally, it also helped in finding inconsistencies in the user interface and missing classifications. The flexible discussion model and the structure-aware visualization seems to help users in discussions and collaborative planning without creating an excessive barrier for the usability.

References

[And+12] A. Anderson et al. “Discovering Value from Community Activity on Focused Question Answering Sites: A Case Study of Stack Overflow”. In: *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining.* ACM. 2012, pp. 850–858.

[BOH11] M. Bostock, V. Ogievetsky, and J. Heer. “D^3 Data-Driven Documents”. In: *Visualization and Computer Graphics, IEEE Transactions on* 17.12 (2011), pp. 2301–2309.

[Bol+02] L. Bollen et al. “Collaborative Modelling in Group Learning Environments”. In: *Proceedings of the XX International Conference of the System Dynamics Society. Palermo (Italy).* 2002.

[Bro96] J. Brooke. “SUS-A quick and dirty usability scale”. In: *Usability Evaluation in Industry* 189.194 (1996), pp. 4–7.

[BSJ15] J. Bradley, N. Sakimura, and M. Jones. “RFC 7519: JSON Web Token (JWT)”. In: *IETF* (2015).

[CB87] J. Conklin and M. L. Begeman. “gIBIS: A Hypertext Tool for Team Design Deliberation”. In: *Proceedings of the ACM Conference on Hypertext.* Chapel Hill, North Carolina, USA: ACM, 1987, pp. 247–251.

[Con03] J. Conklin. “Dialog Mapping: Reflections on an Industrial Strength Case Study”. In: *isualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making.* Springer, 2003, pp. 117–136.

[Die15] F. Dietze. “Quality Assurance in a Structured Collaborative Discussion System”. MA thesis. RWTH Aachen University, 2015.

[Fel98] C. Fellbaum. *WordNet.* Wiley Online Library, 1998.

[FM11] I. Fette and A. Melnikov. “RFC 6455: The WebSocket Protocol”. In: *IETF* (2011).

[FSS13] F. Font, J. Serra, and X. Serra. “Folksonomy-based tag recommendation for collaborative tagging systems”. In: *International Journal on Semantic Web and Information Systems (IJSWIS)* 9.2 (2013), pp. 1–30.

[Gel07a] T. van Gelder. “Rationale: Making People Smarter Through Argument Mapping”. In: *Law, Probability and Risk* 6.1-4 (2007).

[Gel07b] T. van Gelder. “The rationale for Rationale^{TM}”. In: *Law, Probability and Risk* 6.1-4 (2007), pp. 23–42.

[HG06] P. Heymann and H. Garcia-Molina. “Collaborative Creation of Communal Hierarchical Taxonomies in Social Tagging Systems”. In: *InfoLab Technical Report* (2006).

[Hil09] M. Hilbert. “The Maturing Concept of E-Democracy: From E-Voting and Online Consultations to Democratic Value Out of Jumbled Online Chatter”. In: *Journal of Information Technology & Politics* 6.2 (2009), pp. 87–110.

[Hou09] T. W. House. *Open Government Brainstorm: Collaboration in Action.* [Accessed 17.11.2015]. 2009. URL: `https://www.whitehouse.gov/blog/2009/06/05/open-government-brainstorm-collaboration-action`.

[How+11] J. Howison et al. “Motivation through visibility in open contribution systems”. In: (2011).

[Ind11] T. Independent. *Debategraph: Copenhagen - What’s happening?* [Accessed 17.11.2015]. 2011. URL: `http://www.independent.co.uk/environment/climate-change/debategraph-copenhagen-whatrsquos-happening-1813893.html`.

[Kar87] J. Karbach. “Using Toulmin’s Model of Argumentation”. In: *Journal of Teaching Writing* 6.1 (1987), pp. 81–92.

[Kle11] M. Klein. “How to Harvest Collective Wisdom on Complex Problems: An Introduction to the MIT Deliberatorium”. In: *Center for Collective Intelligence Working Paper* (2011).

[KP01] N. Karacapilidis and D. Papadias. “Computer Supported Argumentation and Collaborative Decision Making: the HERMES System”. In: *Information Systems* 26.4 (2001), pp. 259–277.

[KR70] W. Kunz and H. W. Rittel. *Issues as Elements of Information Systems.* Vol. 131. Institute of Urban and Regional Development, University of California Berkeley, California, 1970.

[Lee90] J. Lee. “SIBYL: A Tool for Managing Group Design Rationale”. In: *Proceedings of the 1990 ACM Conference on Computer-Supported Cooperative Work.* ACM. 1990, pp. 79–92.

[MM05] J. L. Moore and R. M. Marra. “A Comparative Analysis of Online Discussion Participation Protocols”. In: *Journal of Research on Technology in Education* 38.2 (2005), pp. 191–212.

[MM06] G. Macgregor and E. McCulloch. “Collaborative Tagging as a Knowledge Organisation and Resource Discovery Tool”. In: *Library Review* 55.5 (2006), pp. 291–300.

[MT14] N. Mahyar and M. Tory. “Supporting Communication and Coordination in Collaborative Sensemaking”. In: *IEEE Transactions on Visualization and Computer Graphics* 20.12 (2014), pp. 1633–1642.

[Mul12] Multicentric Technology Sdn Bhd. *Tackling Wicked Problems Collaboratively: The Multicentric IBIS.* 2012.

[OSS14] A. Okada, S. B. Shum, and T. Sherborne. *Knowledge Cartography: Software Tools and Mapping Techniques.* Springer, 2014.

[Pin03] N. Pinkwart. “A Plug-In Architecture for Graph Based Collaborative Modeling Systems”. In: *11th Conference on Artificial Intelligence in Education.* SIT. 2003, pp. 89–94.

[Rei03] F. F. Reichheld. “The One Number You Need to Grow”. In: *Harvard Business Review* 81.12 (2003), pp. 46–55.

[RN89] H. Rittel and D. Noble. “Issue-Based Information Systems for Design”. In: *University of California at Berkeley Working Paper* 492 (1989).

[RQZ15] M. Riechert, C. Quix, and R. Zarnekow. “Fostering Transparency in Policy Development Processes - A Development Transparency Framework”. In: *ECIS 2015 Research-in-Progress Papers* Paper 39 (2015).

[RW73] H. W. Rittel and M. M. Webber. “Dilemmas in a General Theory of Planning”. In: *Policy sciences* 4.2 (1973), pp. 155–169.

[Sch+10] O. Scheuer et al. “Computer-Supported Argumentation: A Review of the State of the Art”. In: *International Journal of Computer-Supported Collaborative Learning* 5.1 (2010), pp. 43–102.

[Shu+06] S. J. B. Shum et al. “Hypermedia Support for Argumentation-Based Rationale: 15 Years on from gIBIS and QOC”. In: *Rationale Management in Software Engineering.* Springer, 2006, pp. 111–132.

[Shu07] S. B. Shum. “Hypermedia Discourse: Contesting Networks of Ideas and Arguments”. In: *Conceptual Structures: Knowledge Architectures for Smart Applications.* Ed. by U. Priss, S. Polovina, and R. Hill. Berlin: Springer, 2007, pp. 29–44.

[Shu08] S. B. Shum. “Cohere: Towards Web 2.0 Argumentation”. In: *COMMA* 8 (2008), pp. 97–108.

[Sin+14] P. Singer et al. “Evolution of Reddit: From the Front Page of the Internet to a Self-referential Community?” In: *Proceedings of the Companion Publication of the 23rd International Conference on World Wide Web Companion.* International World Wide Web Conferences Steering Committee. 2014, pp. 517–522.

[Sta] StackOverflow. *What types of questions should I avoid asking?* [Accessed 17.11.2015]. URL: `http://stackoverflow.com/help/dont-ask`.

[Sut+95] D. Suthers et al. “Belvedere: Engaging Students in Critical Discussion of Science and Public Policy Issues”. In: *Proceedings of the 7th World Conference on Artificial Intelligence in Education.* Washington, DC. 1995, pp. 266–273.

[SVB07] D. C. Schneider, C. Voigt, and G. Betz. “Argunet - A software tool for collaborative argumentation analysis and research”. In: *7th Workshop on Computational Models of Natural Argument (CMNA VII).* 2007.

[Tou03] S. E. Toulmin. *The Uses of Argument.* Cambridge University Press, 2003.

[Vas+14] B. Vasilescu et al. “How Social Q&A Sites are Changing Knowledge Sharing in Open Source Software Communities”. In: *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing.* ACM. 2014, pp. 342–354.

[YH05] K.-P. Yee and M. Hearst. “Content-Centered Discussion Mapping”. In: *Online Deliberation* (2005).

A. First User Test: Tasks

Aufgaben

1. Du siehst einen Beitrag zu einer Diskussion. Lies ihn.
2. Beschreibe, was du sonst noch siehst.
3. Navigiere zur Detailansicht einer Antwort.
4. Du findest eine Antwort unpassend. Entferne sie aus der Diskussion.
5. Stelle eine Frage zum Beitrag "Was denkst du über Wäsche?": „Normale oder Unterwäsche?“.
6. Du möchtest mehr Übersicht bekommen, wechsele in eine übersichtlichere Ansicht.
7. Beschreibe was du hier siehst.
8. Füge die Idee "Wäsche waschen" zu dem Problem "Meine Wäsche ist schmutzig" hinzu.
9. Auf den Beitrag "Meine Wäsche stinkt" passt deine Antwort auch. Verbinde Problem und Idee.
10. Du hast dich vertan. Entferne die eben erstellte Verbindung.
11. Du hast das Problem "Ich weiß nicht, wie ich eine neue Diskussion starte.". Starte eine neue Diskussion damit.
12. Du möchtest die eben erstellte Diskussion um eine Beschreibung ergänzen. Editiere den Beitrag.

B. Second User Test: Tasks

Usertest

Dies ist eine Benutzerstudie über ein Graphbasiertes Diskussionssystem. Die Studie soll herausfinden, ob die Benutzeroberfläche einfach zu bedienen ist. Während des Usertests wird Bildschirminhalt und Ton als Video aufgezeichnet. Alle Aufzeichnung werden ausschließlich anonymisiert zur Auswertung verwendet und nicht veröffentlicht.

Bitte erledige die Aufgaben der Reihe nach. Wenn du eine Aufgabe nicht erledigen kannst, ist das nicht dein Fehler, sondern ein Fehler der Benutzeroberfläche, die nicht verständlich genug gestaltet wurde. Genau solche Fehler gilt es aufzudecken.

Bitte sprich deine Gedanken laut aus, so dass noch mehr Missverständnisse erkannt werden können. Zum Beispiel: **"Wenn ich hier klicke, müsste sich die Suche öffnen"** oder **"Ich bin mir gerade nicht sicher, was ich anklicken soll"**.

Aufgaben

Strukturierung einer Diskussion

1. Vor dir siehst du den Beitrag **"Planung des Betriebsausflugs der Firma Wust"**. Beschreibe, wie der Beitrag dargestellt wird. Welche Informationen werden noch dargestellt? Beschreibe auch, was passieren würde, wenn du auf die Buttons und Links klicken würdest.
2. Logge dich als Benutzer **"John"** (großes J) mit dem Passwort **"geheim"** (alles klein) ein.
3. Navigiere zum Task **"Veranstaltungsort finden"**. Navigiere weiter zum Ziel **"Unterbringung von ca. 50 Personen"**. Gehe zurück zum Start der Diskussion.
4. Stelle eine Frage zum Beitrag **"Planung des Betriebsausflugs der Firma Wust"**: Du würdest gerne wissen **"Was wurde letztes Jahr gemacht?"**.
5. Du möchtest mehr Übersicht bekommen. Wechsle in die Graph-Ansicht. Beschreibe, ob diese Ansicht wirklich übersichtlicher ist.
6. Füge die Idee **"Skifahren in den Alpen"** zu dem Task **"Veranstaltungsort finden"** hinzu.
7. Auf den Beitrag **"Zeitrahmen vereinbaren"** passt deine Idee auch. Verbinde Idee und Task.
8. Du hast dich vertan. Entferne die eben erstellte Verbindung.
9. Du würdest gerne Reis kochen und fragst dich **"Wie viel Reis brauche ich pro Person?"**. Starte eine neue Diskussion im Kontext "Haushalt".
10. Du möchtest die eben erstellte Diskussion um eine Beschreibung (z.B. Langkornreis) ergänzen. Editiere den Beitrag.

Fortgeschrittene Strukturierung einer Diskussion

1. Suche eine Diskussion zum Thema **"Urlaub"** und sieh sie dir in der Graph-Ansicht an.
2. Füge das Argument **"Mit dem Fahrrad kann man nicht über Wasser fahren"** hinzu.
3. Füge das Argument **"Fahrradfahren ist umweltfreundlich"** hinzu.

Moderation

1. In der Frage **"Mit welchem Verkehrsmittttel fahre ich am besten zur Arbeit"** ist ein Rechtschreibfehler. Korrigiere ihn. Warum ist die Änderung nicht sofort sichtbar? Wenn dir dies nicht klar ist, öffne nochmal das "Bearbeiten"-Fenster und wirf einen Blick auf die Grüne Leiste unten.

2. Klicke in der Navigationsleiste auf **"Moderation"**. Hier siehst du Änderungen von anderen Benutzern. Du kannst entscheiden, ob diese akzeptiert oder abgelehnt werden. Bewerte zwei Änderungen.

3. Suche nach einer Frage mit dem Stichwort **"reparieren"** und öffne sie.

4. Du findest eine Antwort unpassend. Entferne sie aus der Diskussion. Navigiere zurück zur Frage.

5. Du siehst, dass nach der Art der Schaltung gefragt wurde. Du kennst **"Barbera"** persönlich und weißt natürlich, dass sie ein Fahrrad mit einer Nabenschaltung besitzt. Ergänze die Frage von Barbera. Was bedeutet der Grüne Balken unter dem Beschreibungs-Formular?

6. Die Frage nach der Art der Schaltung ist jetzt beantwortet und somit überflüssig. Was ist zu tun?

C. Questionnaire

Wust - Ein Graphbasiertes Diskussionssystem

Vielen Dank, dass Sie sich entschieden haben an der Benutzerstudie zu "Wust" teilzunehmen.
Die Teilnahme an dieser Studie ist freiwillig und dient der Verbesserung des Software-Systems "Wust". Die erhobenen Daten werden vertraulich behandelt und nur für wissenschaftliche Zwecke verwendet. Sämtliche Informationen werden anonymisiert gespeichert und können auf Ihren Wunsch hin jeder Zeit gelöscht werden.
Es besteht keine Möglichkeit auf Ihre Person Rückschlüsse zu ziehen.

Dieser Fragebogen ist Teil einer Benutzerstudie über Wust - einem graphbasierten Diskussionssystem. Er setzt voraus, dass Sie das System bereits ausprobiert haben. Sollte dies nicht der Fall sein, können Sie das unter folgendem Link nachholen:

Wust ausprobieren

Dies ist ein Test-System, sie können nach Belieben alles ausprobieren.

Nachdem Sie sich mit dem System vertraut gemacht haben, können Sie mit dem Fragebogen fortfahren.

Verantwortlich für die Durchführung dieser Studie:
Prof. Dr. Martina Ziefle
Dr. André Calero Valdez
Lehrstuhl für Kommunikationswissenschaft
RWTH Aachen University

Prof. Dr. Ulrik Schroeder
Christoph Greven, M.Sc. RWTH
Lehr- und Forschungsgebiet Informatik 9
RWTH Aachen University

Wust - Ein Graphbasiertes Diskussionssystem

Zunächst würden wir gerne etwas über Ihre Person erfahren wollen.

* 1. Wie alt sind Sie?

* 2. Geschlecht

○ Männlich ○ Weiblich ○ Keine Angabe

3. Wie häufig nutzen Sie folgende Internet-Plattformen?

	Täglich	2-3x pro Woche	1x pro Woche	1x pro Monat	2-3x pro Jahr	Seltener	Nie
Foren (Bulletin boards) oder Facebook	○	○	○	○	○	○	○
Mailing Listen / Verteiler	○	○	○	○	○	○	○
Wikis (z.B. Wikipedia)	○	○	○	○	○	○	○
Reddit	○	○	○	○	○	○	○
StackOverflow	○	○	○	○	○	○	○
IRC	○	○	○	○	○	○	○
Issue/Bug Tracker	○	○	○	○	○	○	○

4. Wenn Ich eben genannte Plattformen nutze, bin ich ...
(Wenn Sie nicht angemeldet sind, können Sie die Letzten 3 Fragen auslassen)

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
passiv, ich lese nur	○	○	○	○	○	○
zurückhaltend und antworte selten	○	○	○	○	○	○
sehr aktiv und starte neue Diskussionen	○	○	○	○	○	○
sehr hilfsbereit und beantworte offene Fragen	○	○	○	○	○	○

5. Wie vertraut sind Sie mit folgenden Konzepten der Graphentheorie?

	Sehr vertraut	Vertraut	Eher Vertraut	Eher nicht vertraut	nicht vertraut	Gar nicht vertraut
Knoten und Kanten	○	○	○	○	○	○
Hyperkanten	○	○	○	○	○	○

Wust - Ein Graphbasiertes Diskussionssystem

6. Sie haben gerade eine Diskussionsplattform genutzt. Wie sehr treffen folgende Aussagen auf das System, oder den Bereich des Systems den Sie kennen, zu?

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
Ich kann mir sehr gut vorstellen, das Sytem regelmäßig zu nutzen.	○	○	○	○	○	○
Ich empfinde das System als unötig komplex.	○	○	○	○	○	○
Ich finde das System als einfach zu nutzen.	○	○	○	○	○	○
Ich denke, dass ich technischen Support brauchen würde, um das System zu nutzen.	○	○	○	○	○	○
Ich finde, dass die verschiedenen Funktionen des Systems gut integriert sind.	○	○	○	○	○	○
Ich finde, dass es im System zu viele Inkonsistenzen gibt.	○	○	○	○	○	○
Ich kann mir vorstellen, dass die meisten Leute das System schnell zu beherrschen lernen.	○	○	○	○	○	○
Ich empfinde die Bedienung als sehr umständlich.	○	○	○	○	○	○
Ich habe mich bei der Nutzung des Systems sehr sicher gefühlt.	○	○	○	○	○	○
Ich musste eine Menge Dinge lernen, bevor ich mit dem System arbeiten konnte.	○	○	○	○	○	○

7. Diskussion

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
Die Anordnung der Beiträge hilft mir, den Zusammenhang zwischen Beiträgen zu erkennen.	○	○	○	○	○	○
Die Anordnung der Beiträge hilft mir, der Diskussion zu folgen.	○	○	○	○	○	○
Die Anordnung der Beiträge hilft mir, alle wichtigen Argumente zu erfassen.	○	○	○	○	○	○
Ich kann meine Gedanken in dem System verfassen.	○	○	○	○	○	○
Es fällt mir leicht, meinen Beitrag in die Diskussion einzuordnen.	○	○	○	○	○	○
Es ist mir wichtig, dass Beiträge sinnvoll miteinander verbunden sind.	○	○	○	○	○	○
Das System macht es mir leicht, mich an der Diskussion zu beteiligen.	○	○	○	○	○	○
Ich finde es angenehm, dass ich Beiträge aus einer Diskussion mit den Beiträgen aus einer anderen Diskussion verbinden kann.	○	○	○	○	○	○

8. Es ist mir wichtig, dass Beiträge nach folgenden Kritierien angeordnet werden:

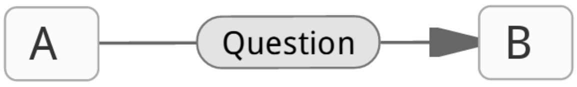
	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
Strikt chronologisch	○	○	○	○	○	○
Kontext, z.B. in der Nähe des beantworteten Beitrags	○	○	○	○	○	○
Qualitätsbewertung anderer Benutzer	○	○	○	○	○	○

Wust - Ein Graphbasiertes Diskussionssystem

Pfeil- und Leserichtung

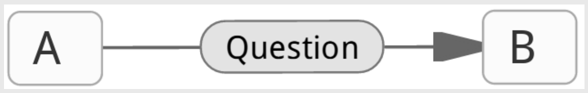
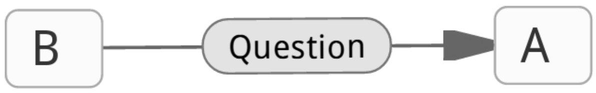
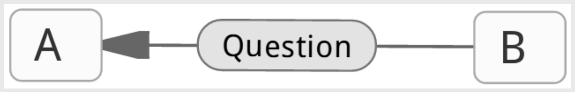
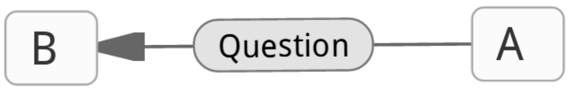
Im folgenden Abschnitt wollen wir heraus finden, wie verschiedene Aspekte in Wust verstanden werden.

9. Wie stehen die Beiträge A und B in Verbindung?

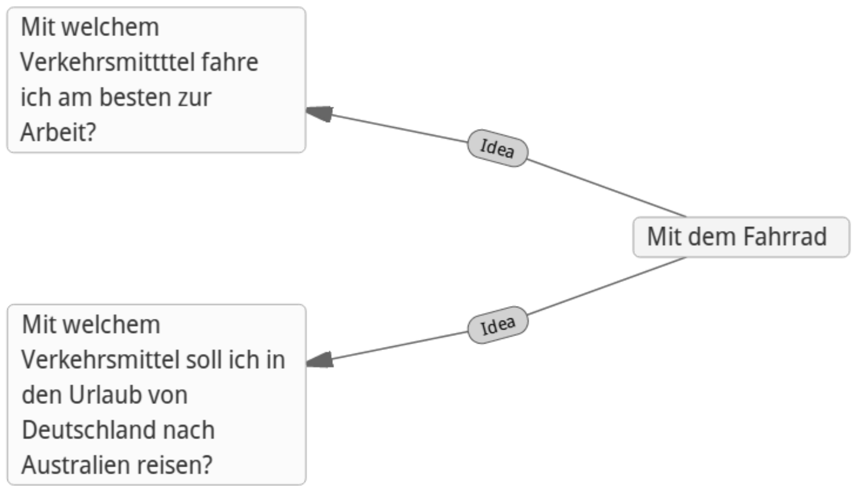


- ◯ A ist eine Frage zu B
- ◯ B ist eine Frage zu A

10. A ist eine Frage zu B. Welche Darstellung empfinden Sie als richtig?

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
	◯	◯	◯	◯	◯	◯
	◯	◯	◯	◯	◯	◯
	◯	◯	◯	◯	◯	◯
	◯	◯	◯	◯	◯	◯

11. An welchen Punkt würden Sie in folgendem Beispiel das Argument "Mit dem Fahrrad kann man nicht über Wasser fahren" anhängen?


Mit welchem Verkehrsmittttel fahre ich am besten zur Arbeit?
Idea
Mit dem Fahrrad
Idea
Mit welchem Verkehrsmittel soll ich in den Urlaub von Deutschland nach Australien reisen?

(Mehrfachantworten sind erlaubt)

☐ An die **Frage** "Mit welchem Verkehrsmittttel fahre ich am besten zur Arbeit?"

☐ An die **Frage** "Mit welchem Verkehrsmittel soll ich in den Urlaub von Deutschland nach Australien reisen?"

☐ An die **Idee** "Mit dem Fahrrad"

☐ An die **Verbindung** zwischen "Mit welchem Verkehrsmittttel fahre ich am besten zur Arbeit?" und "Mit dem Fahrrad"

☐ An die **Verbindung** zwischen "Mit welchem Verkehrsmittel soll ich in den Urlaub von Deutschland nach Australien reisen?" und "Mit dem Fahrrad"

Wust - Ein Graphbasiertes Diskussionssystem

Bewertungen

12. Sie bewerten einen Beitrag positiv. Was sollte passieren?

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
Der Beitrag wird als qualitativ hochwertig eingestuft.	◯	◯	◯	◯	◯	◯
Der Autor des Beitrags wird anerkannter und bekommt mehr Moderationsrechte.	◯	◯	◯	◯	◯	◯
Der Beitrag wird anderen von mir zum Lesen empfohlen.	◯	◯	◯	◯	◯	◯
Ich abonniere alle weiteren Beiträge der Diskussion.	◯	◯	◯	◯	◯	◯
Der Beitrag ist in seinem Kontext besser auffindbar.	◯	◯	◯	◯	◯	◯

13. Ich bewerte...

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
selbstverständlich/direkt beim Lesen	◯	◯	◯	◯	◯	◯
ausschließlich besondere Beiträge	◯	◯	◯	◯	◯	◯
Beiträge lieber positiv	◯	◯	◯	◯	◯	◯
Beiträge lieber negativ	◯	◯	◯	◯	◯	◯
am liebsten gar nicht	◯	◯	◯	◯	◯	◯

Wust - Ein Graphbasiertes Diskussionssystem

Moderation

14. Welche Nutzer sollten Rechte für die Bearbeitung von Beiträgen und Verbindungen erhalten?

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
Nutzer, die lange dabei sind	○	○	○	○	○	○
Nutzer, die vom Seitenbesitzer als Moderator bestimmt wurden	○	○	○	○	○	○
Nutzer, die von der Community als Moderator bestimmt wurden	○	○	○	○	○	○
Nutzer, deren *Beiträge* von der Community positiv bewertet wurden	○	○	○	○	○	○
Nutzer, deren *Korrekturen* von der Community positiv bewertet wurden	○	○	○	○	○	○
Nutzer, die die jeweilige Diskussion gestartet haben	○	○	○	○	○	○
Alle Nutzer	○	○	○	○	○	○

15. Korrekturen

	Stimme vollständig zu	Stimme zu	Stimme eher zu	Lehne eher ab	Lehne ab	Lehne vollständig ab
Wenn ich falsche Verbindungen zwischen Beiträgen sehe, würde ich die Korrekturen selbst vornehmen.	○	○	○	○	○	○
Wenn ich Formfehler (z.B. Rechtschreib-, Kommafehler) in Beiträgen sehe, würde ich sie gerne korrigieren.	○	○	○	○	○	○
Wenn ich inhaltliche Fehler (z.B. falsche Information) in Beiträgen sehe, würde ich sie gerne korrigieren.	○	○	○	○	○	○
Ich finde es sinnvoll, wenn ausgewählte Nutzer (vgl. Frage 14) die Beiträge anderer neu verbinden können.	○	○	○	○	○	○
Ich finde es sinnvoll, wenn ausgewählte Nutzer (vgl. Frage 14) die Beiträge anderer bearbeiten können.	○	○	○	○	○	○

16. Bitte nehmen sie Stellung zu den folgenden Aussagen:

	stimme vollständig zu	stimme zu	stimme eher zu	lehne eher ab	lehne ab	lehne vollständig ab
Ich kann ziemlich viele der technischen Probleme, mit denen ich konfrontiert werde, allein lösen.	○	○	○	○	○	○
Technische Geräte sind oft undurchschaubar und schwer zu beherrschen.	○	○	○	○	○	○
Es macht mir richtig Spaß, ein technisches Problem zu knacken.	○	○	○	○	○	○
Weil ich mit bisherigen technischen Problemen gut zurecht gekommen bin, blicke ich auch künftigen technischen Problemen optimistisch entgegen.	○	○	○	○	○	○
Ich fühle mich technischen Geräten gegenüber so hilflos, dass ich die Finger von ihnen lasse.	○	○	○	○	○	○
Auch wenn Widerstände auftreten, bearbeite ich ein technisches Problem weiter.	○	○	○	○	○	○
Wenn ich ein technisches Problem löse, so geschieht es meistens durch Glück.	○	○	○	○	○	○
Die meisten technischen Probleme sind so kompliziert, dass es wenig Sinn hat, sich mit ihnen auseinander zu setzen.	○	○	○	○	○	○

Wust - Ein Graphbasiertes Diskussionssystem

Feedback

17. Wie wahrscheinlich ist es, dass Sie dieses Diskussionssystem einem Freund oder Kollegen weiterempfehlen werden?

Gar nicht wahrscheinlich — Äußerst wahrscheinlich

0	1	2	3	4	5	6	7	8	9	10

18. Wenn Sie noch Anmerkungen zur Befragung haben, können Sie die hier gerne angeben.

19. Über Feedback zu unserem System würden wir uns freuen.