AUTOMATED ANALYSIS AND FEEDBACK TECHNIQUES TO SUPPORT AND TEACH ARGUMENTATION: A SURVEY

Oliver Scheuer[1], Bruce M. McLaren[1 3], Frank Loll[2], Niels Pinkwart[2]

[1] Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Saarbrücken, Germany
[2] Clausthal University of Technology, Department of Informatics, Clausthal-Zellerfeld, Germany
[3] Carnegie Mellon University, Human-Computer Interaction Institute, Pittsburgh, U.S.A.

Abstract: Argumentation is one of the key competencies in our private and professional lives. However, many people struggle to produce, interpret and evaluate arguments in a systematic and rational fashion. To remedy this situation, a number of computer-based argumentation systems have been developed over the past decades to support or teach argumentation. The use of artificial intelligence techniques holds promise to increase the effectiveness of such systems by automatically analyzing user actions and providing supportive feedback. In this chapter, we review and systemize argumentation analysis approaches with a special focus on the educational uses. We also discuss argument modeling and discussion systems including their analysis approaches, feedback strategies and architectures.

INTRODUCTION

Argumentation skills are central to both our private and professional lives. In a host of everyday and professional situations we are tasked with convincing and persuading others through argumentation, e.g., to agree with a decision, to follow advice, to accept an opinion or just to believe that an assertion is actually true. Argumentation can be informal or more specialized and semi-institutionalized, with agreed-upon patterns, rules, roles and evaluation standards, for instance argumentation in professional domains like the law, science, business, administration and politics. Argumentation is not only a means to persuade others; it is also an essential tool for rational decision making (e.g., rational design [Buckingham Shum, MacLean, Bellotti, & Hammond, 1997]), tightly connected to human reasoning (Andriessen, 2006; Kuhn, 1991) and plays a crucial role as a vehicle for collaborative learning in a process of knowledge co-construction (Weinberger, Stegmann, Fischer, & Mandl, 2006) and collaborative meaning-making (Baker, 2003).

Despite the importance and ubiquity of argumentation, often people struggle to produce, interpret and evaluate arguments in a rational way. This has been shown in a number of studies that document problems with argumentation in informal settings (Kuhn, 1991), as well as in specific professional domains like science (Stark & Krause, 2006). A number of factors contribute to these observed difficulties including the ill-defined nature and inherent complexity of argumentation *per se* (oftentimes so-called "wicked problems" are subject to argumentation [Rittel & Webber, 1973; Buckingham Shum et al., 1997]), psychological biases (e.g., bias towards confirming ones own beliefs with disregard of alternatives [Kuhn, 1991; Easterday, Aleven, Scheines, & Carver, 2009]), social biases (e.g., bias towards avoiding disagreement with others [Nussbaum, Winsor, Aqui, & Poliquin, 2007]) and misconceptions / ignorance of basic argumentation concepts (e.g., inability to distinguish between theory and data [Kuhn, 1991]). As a consequence, it has been claimed that argumentation should be assigned a more central role in our formal educational system (Driver, Newton, & Osborne, 2000; Kuhn, 2005).

In the last few decades digital technologies have emerged to support the learning and practical use of argumentation. Yet, the design of educational argumentation systems is not a trivial matter. Technological, pedagogical, and

human-computer interaction aspects must be taken into account in building tools that effectively support argumentation learning. Throughout the literature we find at least four different design approaches that have been taken to build and deploy argumentation tools:

- **Micro-scripting approaches** try to foster rich and high-quality interaction in collaborative learning situations using special-purpose interfaces that encourage (or sometimes force) a desired mode of interaction. Micro-scripting can take different forms: Some approaches use predefined communication categories such as "claim," "argument" and "question" (Schwarz & Glassner, 2007; Jeong & Juong, 2007). Others use sentence starters such as "Let me explain …" and "I disagree because…" (McManus & Aiken, 1995; Soller, 2001; McAlister, Ravenscroft, & Scanlon, 2004). Others again use form-like interfaces that scaffold the creation of individual arguments and argument sequences (Stegmann, Weinberger, & Fischer; 2007). Positive effects on the quality of argumentation and resultant arguments have been reported, for instance, by Schwarz and Glassner (2007), McAlister et al. (2004), and Stegmann et al. (2007).
- **Representational guidance approaches** (Suthers, 2003; Nussbaum et al., 2007; Pinkwart, Aleven, Ashley, & Lynch, 2006a) try to stimulate and improve individual reasoning, collaboration, and ultimately learning by providing external representations of argumentation structures. A number of studies have shown that such external representations have positive effects both on collaboration (Suthers & Hundhausen, 2003; Suthers, Vatrapu, Medina, Joseph, & Dwyer, 2008; Nussbaum et al., 2007) and individual problem-solving (Easterday, Aleven, & Scheines, 2007).
- **Macro-scripting approaches** (Dillenbourg & Hong, 2008; Lund, Molinari, Séjourné, & Baker, 2007; Munneke, Van Amelsvoort, & Andriessen, 2003; Schellens, Van Keer, De Wever, & Valcke, 2007; Muller Mirza, Tartas, Perret-Clermont, & de Pietro, 2007) are concerned with structuring at the level of predefined phases, roles, and activities. The underlying rationale is to create meaningful contexts that stimulate learning. For instance, Schellens et al. (2007) assigned the roles "moderator," "theoretician," "summarizer," and "source searcher" to students who collaborated in asynchronous e-discussions. The scripting led to higher levels of knowledge construction during the discussions and improved exam scores.
- **Adaptive support approaches** aim at more interactive and dynamic forms of help by providing pedagogical feedback on student actions and solutions (e.g., Suthers et al., 2001; Pinkwart et al., 2006a), hints and recommendations to encourage and guide future activities (e.g., McAlister et al., 2004) or automated evaluation services to indicate whether claims in a current argument are acceptable or not (e.g., Gordon, Prakken, & Walton, 2007; Verheij, 2003; Ranney & Schank, 1998). Here, techniques from the research fields of Artificial Intelligence and Intelligent Tutoring are widely used.

In a previous article (Scheuer, Loll, Pinkwart, & McLaren, 2010) we reviewed computer-supported argumentation systems across a broader set of analysis dimensions. This chapter focuses specifically on techniques for adaptive support in argumentation systems, extending our previous analysis. Roughly following Bell's (1997) distinction between knowledge representation and discussion-based tools, our discussion is structured into

two main sections: argument modeling and discussion / discussion-enhanced systems.

Argument modeling systems support the creation of new arguments and the study of existing ones. Arguments and their constituent components are reified as (oftentimes graphical) objects that can be manipulated by the user. The modeling process can be exploratory and user-driven (e.g., freely creating an argument and testing the acceptability of statements, as in *ArguMed* [Verheij, 2003]) or framed as a problem-solving task (e.g., "translating" a given textual transcript into a diagrammatic argument representation, as in *LARGO* [Pinkwart et al., 2006a]). Argument modeling has been used for various purposes, for instance, to support professional decision making (Buckingham Shum et al., 1997; Van Gelder, 2003), the collaborative writing of texts (Chryssafidou, 2000; Munneke, Andriessen, Kanselaar, & Kirschner, 2007), the sketching of legal arguments (Aleven & Ashley, 1997; Verheij, 2003), the theoretical analysis and evaluation of texts and transcripts (Reed & Rowe, 2004; Pinkwart et al., 2006a; Schneider, Voigt, & Betz, 2007) and the evaluation of alternative hypotheses in science / inquiry learning scenarios (Suthers et al., 2001; Woolf et al., 2005).

On the other hand, **discussion-oriented systems** are collaborative by nature and oftentimes not exclusively focused on argumentation. They also try to foster good discussion practices (e.g., balanced participation, responding to questions, staying on-topic [Hoppe et al., 2008]). Instead of examining the logical and rhetorical properties and relations of arguments, students actually *use* arguments to communicate with one another. In a problem-solving context, discussion facilities are typically employed as a means to coordinate and deliberate on problem-solving actions (McManus & Aiken, 1995; Goodman et al., 2005).

As we will see, argument modeling and discussion systems are quite different in terms of analysis techniques: argument modeling systems tend to have rigorously structured input with well-defined semantics (a user-constructed model). Also, the analysis outputs are typically well-defined (e.g., patterns that indicate modeling errors). On the other hand, discussion-based systems often have to cope with far more ambiguous input formats (e.g., natural language text, user-selected sentence openers). The desired output types are often interpretive in nature or hard to operationalize (e.g., failed attempts to share knowledge). Yet, the distinction between argument modeling and discussion-based systems is not always clear-cut. For instance, the system *Digalo* (Schwarz & Glassner, 2007) can be used equally well to represent and model argumentative structures *and* to conduct graphical e-Discussions. Mediation systems like *Zeno* (Gordon & Karacapilidis, 1997) also target the *process* of argumentation in a joint effort of a group, but the resultant arguments are highly structured and can be formally evaluated.

In the following we take a closer look at representatives of both categories, and discuss analysis techniques for each, before turning to support strategies and system architectures for adaptive support. The focus of this chapter is on educational argumentation systems, although other types of systems and techniques (e.g., systems to support the creation of arguments) will be discussed where appropriate.

ARGUMENT MODELING SYSTEMS

The argument modeling systems discussed in this section are primarily grouped into two major application categories: Systems such as *Belvedere* (Suthers et al., 2001), *Rashi* (Woolf et al., 2005), and *Convince Me* (Ranney & Schank, 1998) train reasoning skills in inquiry / science learning. Systems such as *CATO*

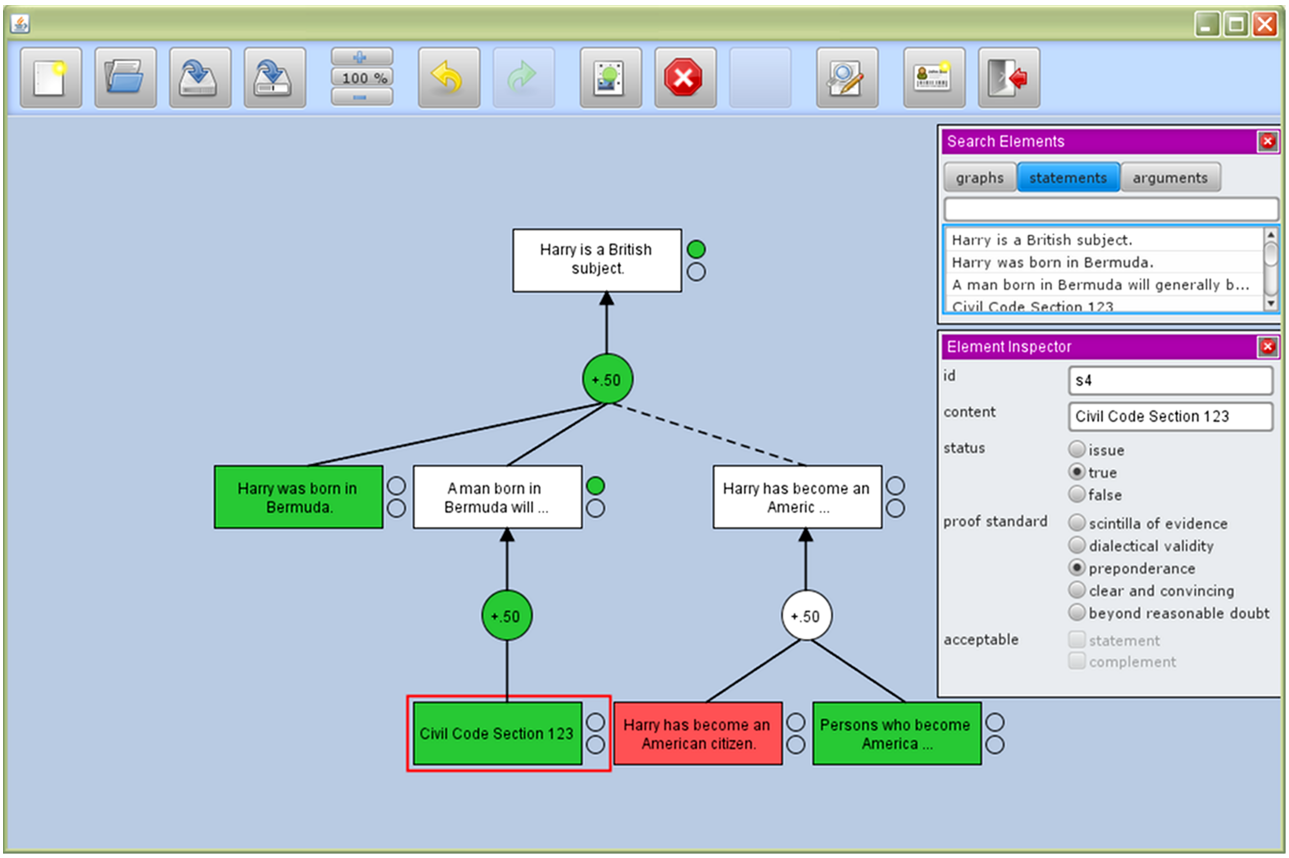

Figure 1: The Carneades argumentation system displaying an argument from Toulmin (1958).

(Aleven & Ashley, 1997), *LARGO* (Pinkwart et al., 2006a), and *Carneades* (Gordon et al., 2007) represent, analyze and evaluate legal arguments.

Belvedere (Suthers et al., 2001) is perhaps the best known educational argumentation systems. Students use *Belvedere* to acquire critical inquiry skills by collaboratively investigating public policy and science problems, e.g. exploring the reasons for the mass extinction of dinosaurs 65 million years ago. Students examine background materials and create *evidence maps* within a shared workspace. The evidence maps are box-and-arrow diagrams that depict the pro and con relations that hold between hypotheses and data elements. The goal is to determine which hypothesis is most likely in light of the given evidence. The process of creating these maps is supported by a computer-based advisor, which provides textual feedback messages upon request.

A more recent approach to intelligent tutoring of critical reasoning and inquiry skills is *Rashi*, which is "a coaching environment that supports structured reasoning within an open-ended, inquiry-learning infrastructure" (Dragon, Woolf, Marshall, & Murray, 2006, p. 144). The *Rashi* framework has been used to implement intelligent tutors in different domains including biology, geology, forestry and art history. Each *Rashi*-based tutor provides an integrated set of tools to support typical inquiry activities, such as formulating questions, collecting data, interpreting data, and evaluating hypotheses. Depending on the domain and specific task, different *data collection tools* are used. For instance, in the *Art History Tutor* (Dragon et al., 2006) students extract critical observations from historical paintings via clickable "hotspots" and relevant text passages from historical texts via markup techniques. In the *Human Biology Tutor* (Woolf et al., 2005) students collect results from patient examinations, lab tests

and patient interviews. Domain-independent *critical reasoning tools* are used across domains to make sense of the collected data: The *Inquiry Notebook* allows organizing and classifying collected data, the *Argument Editor* allows building multi-level arguments in order to evaluate competing hypotheses. The students' arguments are analyzed by an intelligent computer-based coach, which provides feedback upon request. (We discuss *Rashi*'s analysis approach in more detail below in section *Problem-specific analysis*.) While working on an inquiry task the student can switch between activities opportunistically. More recent extensions include software support for collaborative inquiry (Dragon & Woolf, 2006; Dragon, Woolf, & Murray, 2009).

Convince Me (Ranney & Schank, 1998) is an educational tool that aims at improving students' reasoning skills. A typical way the system is used is to analyze and decide between competing theories that are presented as short text paragraphs. Because of *Convince Me*'s generic design a wide range of problems is possible, for instance biology, physics, medicine and ethics problems have been used in the past (Schank, 1995). When using *Convince Me,* students classify statements (either taken from a given text or provided on their own) as hypotheses and evidence, specify the reliability of evidence, generate argumentative structures by laying out how hypotheses and evidence are interrelated (e.g., a set of evidence might explain or contradict a hypothesis) and finally rate how strongly they believe in each statement. To see whether their believability judgments are in line with a theoretical model of coherent reasoning, students run a simulation called *ECHO*, which shows the agreement between *ECHO*'s and the student's believability judgments. (We discuss the *ECHO* simulation in more detail below in section *Simulation of reasoning and decision making processes*.) Based on this feedback, argument structures and believability ratings can be iteratively adjusted to maximize agreement.

CATO (Aleven & Ashley, 1997) is an Intelligent Tutoring System (ITS) for legal case-based argumentation. In this type of argumentation lawyers argue for (or against) a position and corresponding precedents by pointing out similarities (or differences) with the current case under consideration. Cases in *CATO* are described in terms of *factors*, each of which represents a stereotypical collection of facts that are relevant for deciding a case. Factors can be used to compare cases, i.e., two cases might share or differ in certain factors. *CATO* provides a number of tools to assist novice law students in analyzing the current case, in particular, identifying relevant precedents, practicing basic argument moves and formulating complex precedent-based arguments. The *Case Analyzer* helps students identify relevant factors in the current case. The *CATO Database* can be queried to retrieve relevant precedent cases based on a factor profile. Two tools automatically generate example arguments to support students: The *Argument Maker* creates example arguments involving one precedent. The *Issue-Based Argument Window* generates complex example arguments involving multiple precedents, organized by issues. Students can use the generated arguments as learning resources and compare them to arguments they have created themselves.

The Intelligent Tutoring System *LARGO* (Pinkwart et al., 2006a) also trains law students in their legal argumentation skills. However, students who use *LARGO* do not generate their own arguments; rather, they analyze existing expert arguments using a model of *hypothetical reasoning*, which can be described as follows: legal parties propose tests that interpret laws, legal principles or precedent cases in a specific way to decide a current situation at hand; these tests are challenged by posing challenging hypothetical situations (*hypotheticals*) in which the proposed test may lead to an undesired

outcome. This, in turn, may lead to abandoning or modifying the test. Students apply this approach to existing US Supreme Court Oral Arguments. They use *LARGO* to "translate" provided transcripts of oral arguments into graph-based visual representations, supported by an advice-on-request function.

Table 1: Overview of analysis approaches for argument models

Analysis Approach	Description
Syntactic analysis	Rule-based approaches that find syntactic patterns in argument diagrams **Systems:** *Belvedere*, *LARGO*
Problem-specific analysis	Use of a problem-specific knowledge base to analyze student arguments or synthesize new arguments **Systems:** *Belvedere*, *LARGO*, *Rashi*, *CATO*
Simulation of reasoning and decision making processes	Qualitative and quantitative approaches to determine believability / acceptability of statements in argument models **Systems:** *Zeno*, *Hermes*, *ArguMed, Carneades*, *Convince Me,* Yuan et al. (2008)
Assessment of content quality	Collaborative filtering, a technique in which the views of a community of users are evaluated, to assess the quality of the contributions' textual content **Systems:** *LARGO*
Classification of the current modeling phase	Classification of the current phase a student is in according to a predefined process model **Systems:** *Belvedere*, *LARGO*

Finally, we discuss systems that can automatically evaluate the acceptability of statements within a (legal) argument. *Carneades* (Gordon et al., 2007; see Figure 1) "supports a range of argumentation tasks, including argument reconstruction, evaluation and visualization" (Gordon et al., 2007, p. 875). Although it is conceptualized as a tool not restricted to a particular argumentation domain, it is primarily aimed at legal argumentation. With the main claim at the top, the user lays out an argument as a graphical tree. A formal, mathematical model computes and assigns acceptability values to propositions, indicating whether they hold up in the face of other propositions and arguments that have been brought forward. *Carneades* supports multiple *proof standards*, i.e. different procedures to derive the acceptability of a claim and associated arguments. Similar decision procedures are implemented in *ArguMed* (Verheij, 2003), an argument assistant system also focused on the legal domain, as well as in *Zeno* (Gordon & Karacapilidis, 1997) and *Hermes* (Karacapilidis & Papadias, 2001), two systems that are focused on mediating and supporting collaborative decision-making in threaded discussions, which have been used in domains as diverse as public policy deliberation, medicine as well as mechanical and civil engineering.

In the next sections we discuss the different argument analyses approaches taken by these systems. Table 1 provides an overview of the different analysis approaches.

Syntactic analysis

A common approach to argument analysis is to identify syntactical patterns in student argument diagrams, as is done in *Belvedere* (Suthers et al., 2001) and *LARGO* (Pinkwart et al., 2006a). Such patterns are relatively easy to detect: graph structures have, in contrast to natural language texts, unambiguous and explicit semantics. Patterns of interest often indicate a violation of the system's internal model of valid argumentation (e.g., circular arguments, invalidly connected contribution types), correct but not yet complete structures (e.g., a required contribution type is still missing in the argument diagram) or otherwise

pedagogically useful patterns. For instance, in *Belvedere* scientific arguments are examined for conflicting hypotheses that are supported by the same data element. This pattern indicates either a modeling error (e.g., the data element only supports one hypothesis) or the need for further data to better distinguish between alternative explanations. In any case it represents an opportunity for the student to reflect about his model. Another *Belvedere* pattern is the existence of only one hypothesis, which indicates that alternative hypotheses have not yet been considered. *LARGO* analyzes student arguments in the legal domain to find modeling weaknesses, for instance, two hypothetical elements linked by a general relation type, where a more specific relation type would be preferable to better express the structure of the argument flow.

Argument patterns in *Belvedere* and *LARGO* are specified as expert rules; often a single rule per pattern. While *Belvedere* rules are defined in a LISP-based knowledge representation, *LARGO* makes use of graph grammars (Pinkwart, Ashley, Lynch, & Aleven, 2008a), a formalism specifically suited for the analysis of graphical representations. Similar rule-based approaches have been used in constraint-based tutoring systems (Mitrovic, Mayo, Suraweera, & Martin, 2001) to support other types of graphical modeling, for instance, to find patterns in entity relationship (Suraweera & Mitrovic, 2004) or UML diagrams (Baghaei, Mitrovic, & Irwin, 2007). In general, the rule-based approach has several favorable properties: Typically, rules represent local conditions and/or patterns and can be applied more or less independently of one another. The modular nature of rules allows for easy modification of existing rules, addition of new rules, or deletion of rules. Another advantage is that rules can combine fine-grained heuristics from different pedagogical and domain-specific theories within a single analysis framework. For instance, the collaborative problem-solving environment *COLLECT-UML* (Baghaei et al., 2007) uses the same rule-based analysis framework to identify errors in UML diagrams and problems in the students' collaboration.

Problem-specific analysis

A purely syntactic analysis can successfully identify weaknesses and other teaching opportunities in the structure of arguments. However, such an approach is limited to patterns of a general type. For instance, an argument diagram might be well formed in terms of syntactic domain constraints but nevertheless not be an appropriate solution for a given problem instance. Furthermore, a syntactic analysis cannot be used to give *problem-specific hints* to the student because it lacks the required domain knowledge. In answer to this, *Belvedere* (Suthers et al., 2001), *Rashi* (Woolf et al., 2005; Dragon et al., 2006), *LARGO* (Pinkwart et al., 2006a) and *CATO* (Aleven & Ashley, 1997) use a second type of analysis mechanism, one that focuses on characteristics of the *current problem*.

The analysis in *Belvedere*, *Rashi* and *LARGO* is based on the identification of discrepancies between an expert model for a given problem and the corresponding student solution. There are three important aspects to consider: how to represent expert knowledge, how to match elements in the expert model with corresponding elements in the student solution, and how to identify relevant and important differences between the expert model and student solution.

In *Belvedere,* experts provide a complete solution for each problem by constructing prototypical diagrams. Students choose from the same set of predefined knowledge units as the experts when constructing their diagrams, hence matching elements between student and expert solutions is relatively straightforward. *Belvedere* uses a constrained search to find elements in the expert solution that are likely to conflict with the students' solution. For

instance, the algorithm might determine that two elements are *consistent* with one another in a student solution (e.g., a data element that supports a hypothesis) but inconsistent with one another in the expert diagram. Other elements in the expert diagram on the path between the two elements are then candidates to cause the student to rethink his / her way of modeling.

In *Rashi*, experts create a directed graph of propositions for each problem to represent inferential relationships. To provide experts with more control over the coaching process, they can annotate propositions with additional metadata such as relative importance and the minimum amount of evidence required to convincingly support or refute the proposition. Data elements in the student solution can be easily matched against the expert knowledge base because they are generated by *Rashi* itself when students engage in inquiry activities in data collection tools, i.e., *Rashi* "knows" which data elements are used by the student. Other elements are identified using keyword matching. The student argument is essentially an overlay of the expert model and matches a subset of it. *Rashi* analyzes this overlay using a rule-based engine to identify incorrect relationships in the student solution and additional elements of the expert knowledge base worthy of consideration, including not yet considered hypotheses, new data elements to support / refute existing statements (*top-down argumentation*) and new statements that build upon existing elements (*bottom-up argumentation*).

In *LARGO*, expert models consist of annotated transcripts in which passages can be marked as irrelevant or classified according to their function (e.g., "test", "hypothetical"). While constructing an argument diagram, students create diagram elements and mark corresponding transcript passages to indicate that the diagram element represents the marked transcript passage. *LARGO* then checks for diagram elements that refer to irrelevant passages, missing references to important passages and classifications of passages that differ from the annotated ones (e.g., "hypothetical" instead of "test").

CATO takes a different approach in that it does not analyze the students' precedent-based arguments but generates its own example arguments instead. A comparison is then left to the students. The expert knowledge base comprises a *case database*, which contains relevant factors for each case, and a *factor hierarchy*, which represents supporting and opposing relations between factors. Factors are organized hierarchically including basic factors, which can be directly identified in legal cases, as well as top-level factors, which represent normative legal concerns. *CATO* uses these knowledge representations to generate basic argument moves following pre-defined *rhetorical recipes*, for instance to emphasize or downplay a distinction between two cases by comparing their factors. Basic argument moves can be used to generate more complex arguments, which cover multiple issues and analogize / differentiate the current case at hand with multiple precedents to justify a legal decision favorable to the producer of the argument.

Problem-specific analyses enable systems to provide *problem-specific support*, a feature that cannot be achieved using the syntactical approaches. On the downside, considerable effort is needed to define expert models for every problem instance. A possible solution is the provision of authoring tools, which "reduce the development time, effort, and cost" and also "lower the skill barrier" for nontechnical experts (Murray, Woolf, & Marshall, 2004, p. 197). Comparisons with an expert model typically yields only heuristic results since such models do not define the complete space of possible solutions or errors. Furthermore, due to the oftentimes ill-defined nature of argumentation tasks it is in general not possible to unequivocally decide whether

a solution is acceptable or not. Even experts often disagree when evaluating arguments (e.g., appeal court decisions not always follow the decision of a lower court). To facilitate "understanding" of arguments without relying on error-prone natural language processing, systems often constrain the way students can express arguments: In *Belvedere,* students choose from a pre-defined set of knowledge units, in *Rashi* students collect pre-defined data elements in simulated inquiry activities, and in *LARGO* students mark the modeled transcript passage by themselves. In all three of the systems there may also be argument components that are not directly assignable to elements of the expert knowledge base: freely created contributions in *Belvedere*, diagram elements not linked to the transcript in *LARGO,* and hypotheses entered as free text input in *Rashi*. Each system handles this in its own way: *Belvedere* ignores these elements in the analysis, *LARGO* classifies them as a modeling weakness and *Rashi* uses simple keyword matching.

Simulation of reasoning and decision making processes

Syntactical and problem-specific analyses treat argument models as static representations, which can be checked for syntactic patterns or compared to an expert model. We now turn to approaches that employ argument graphs as *executable models* to simulate reasoning and decision-making processes. Such simulations allow the user to see how believable (or acceptable) individual statements are in a given argument constellation when certain evaluation norms or procedures are applied. We discuss two approaches, one relying on qualitative reasoning, the other on quantitative machine reasoning.

From the field of *computational dialectics* (Walton, 2008; Chesñevar, Maguitman, & Loui, 2000), a number of systems have emerged that use formal-logical models of argumentation to evaluate the acceptability of sentences in a given argument. In the systems *Zeno* (Gordon & Karacapilidis, 1997), *Hermes* (Karacapilidis & Papadias, 2001), *ArguMed* (Verheij, 2003) and *Carneades* (Gordon et al., 2007), arguments are modeled by laying out the argument structure (e.g., proposition elements connected via inferential relationships like "support" and "opposes") and specifying operational parameters (e.g., indicating whether propositions are accepted / rejected / open at the current stage of a discussion or the inferential strength of relations). A rule system is then utilized to simulate decision procedures, which determine whether argument elements are acceptable or not. A typical choice for such procedures, even in systems that are not focused on legal argumentation like *Zeno* and *Hermes*, are proof standards from the legal domain like "preponderance of evidence" or "beyond reasonable doubt" (Karacapilidis & Papadias, 2001). The resulting classifications (e.g., a proposition is acceptable or not) are then displayed to the users to help them in drafting and generating arguments (Verheij, 2003) or making decisions (Karacapilidis & Papadias, 2001). The primary purpose of these systems is not educational but rather to support reasoning.

Convince Me (Ranney & Schank, 1998) uses a different approach to determine the acceptability of propositions, based on a connectionist model called *ECHO*. Instead of utilizing a *qualitative* rule mechanism, it bases its acceptability decisions on a *quantitative* mechanism involving the mutual strengthening and weakening of propositions in an undirected network. The input is a network of propositions, which consists of evidence and hypothesis elements that are connected via explanatory and contradictory relations. To evaluate propositions, the input network is first translated into an *artificial neural network* where *units* represent propositions and *synapses* represent consistency and inconsistency relations between propositions. Propositions have an *activation level*, which indicates their

"believability." When the network simulation is started, units increase or decrease the activation of their neighbors and, after a sufficient number of iterations, the activity values in the network stabilize, i.e., the network reaches a "steady state." The system then displays the model's evaluation along with the students' believability assessments for the same propositions to help students restructure their arguments and/or revise believability ratings.

Mathematically, the *ECHO* algorithm can be characterized as a constraint satisfaction problem solver (Thagard & Verbeurgt, 1998). The theoretical foundation of *ECHO* is provided by the *Theory of Explanatory Coherence (TEC*; Thagard, 2006), which describes how humans evaluate competing explanations. The *ECHO* simulator makes use of numeric parameters, which are determined empirically. Supporting evidence for the generality of *ECHO*'s parameters has been gained through successful use across different domains (Schank, 1995). However, it should be noted that such empirically parameterized models are typically not suitable for generating human-understandable explanations, e.g., when results appear counter-intuitive to the user. Pasquier, Rahwan, Dignum and Sonenberg (2006) discuss an approach related to *ECHO* to compute maximally coherent configurations of an agent's private cognitions (e.g., beliefs and intentions) and social commitments based on the *Cognitive Coherence Theory*. This framework can also be used to model the effect of arguments on agents (i.e., a possible persuasion or attitude change). An alternative quantitative approach to argument evaluation is the use of *Bayesian Belief Networks*, as presented by Vreeswijk (2005). The advantage of such an approach is to have a probabilistic framework interpret quantities in a more principled way but on the other hand, *Bayesian Belief Networks* are hard to parameterize and computationally expensive in the general case (Schank, 1995; Vreeswijk, 2005).

The formal-logical and connectionist approaches are similar in that they rely exclusively on the argument structure and element properties as specified by the user. Therefore, the resulting evaluations depend strongly on how skillfully users model an argument. The systems do not test for modeling errors and missing important elements. For instance, contrary to what might be stated in a student-created model, a proposition might actually *not* support a claim. Checking for this would require system knowledge on the content-level as an *external* criterion to assess the validity of the arguments or individual propositions. We already saw examples of this in the previous discussion of problem-specific analysis approaches and will discuss a further example below. Students' argumentation skills can (theoretically) benefit from system-provided acceptability ratings in two ways: First, the simulations might help students understand the mechanics of reasoning by inspecting the acceptability values that emerge from a given argument and by observing how these values change as the argument structure changes. Second, the simulations allow for testing the effects of possible counterarguments and determining whether more evidence for the own position needs to be brought forward.

Finally, we will take a look at another approach based on computational dialectics with a stronger focus on procedural aspects of argumentation (Yuan, Moore, & Grierson, 2008). Here, students are engaged in a structured educational dialogue with a computer agent. The system is based on *Dialogue Game Theory* and models argumentative discourse using a pro and a con position, a fixed set of valid dialogue moves (e.g., assert, question, challenge), commitment rules (determine implicitly and explicitly accepted statements of each participant after each step) and dialogue rules (determine which move can or must be used in a given

situation). The underlying dialogue game was chosen to be especially suitable for educational human-computer debates. To generate appropriate moves, the emulated computer-based arguer employs a planning unit and a topic-specific knowledge base that represents inferential relationships between propositions. This allows the system to determine possible lines of argumentation, a feature not available in the approaches above. The knowledge base enables the system to pursue an informed debate strategy, e.g., attacking student moves with appropriate statements from the knowledge base. On the other hand, users are restricted in expression since they have to choose from a limited set of pre-defined statements.

Assessment of content quality

All of the above approaches rely on well-structured information, amenable to automated processing. However, when students provide natural language contributions to an argument, computational evaluation is much harder and computationally expensive. *LARGO* (Pinkwart et al., 2006a) skirts this issue by letting student peers assess the quality of contributions. After finishing a diagram element and relating it to a transcript passage, students are prompted to provide quality ratings for the diagram elements of their peers that refer to the same transcript passage. These ratings are collected and numerically combined using *collaborative filtering* (Goldberg, Nichols, Oki, & Terry, 1992) yielding an overall quality rating. A prerequisite of this approach is that a sufficient number of peers work on the same problem, which is typically a realistic expectation in educational scenarios.

A critical advantage of collaborative filtering is its relatively low development and online processing cost, especially in contrast to natural language processing approaches. Furthermore, prompting students to rate their peers' contributions may have a learning effect since students have an opportunity to reflect on and assess their own and their peers' contributions. Furthermore, it has been shown that collaborative filtering can lead to reliable and accurate assessments (Loll & Pinkwart, 2009; Cho & Schunn, 2007). A possible limitation of the approach is the need for the presence of peers who are willing and capable of providing high quality assessments. Furthermore, interrupting a student to prompt for such feedback might interfere with the student's learning activities. The challenge is finding ways to elicit feedback without disturbing or annoying students.

Classification of the current modeling phase

Automated analysis may be applied not only to the argument models themselves but also to the process and phases of creating the models. A simple approach is to use fixed time periods to classify process phases; *LARGO* and *Belvedere* are both more flexible in that they determine the current phase based on *dynamic aspects* of the current solution state, allowing students to work at their own pace. In *LARGO* (Pinkwart, Aleven, Ashley, & Lynch, 2006b), the analysis of argument transcripts is perceived as a multi-phase process involving "orientation", "transcript analysis", "relating elements", "error correction" and "reflection phase." *LARGO* determines the current phase through a meta-analysis of domain-specific patterns in the current version of the diagram. Each pattern is associated with a specific phase; hence the aggregate of all patterns provides evidence for the *current* phase. *Belvedere* (Suthers et al., 2001) distinguishes between an "early", "middle" and "late" phase. The classification is based on static conditions of each phase, for instance, the diagram is in the "late phase" if there are at least two hypotheses, four data elements, four evidential relations and the number of data elements and evidential relations exceeds the number of hypotheses.

Of course, knowing the current process phase allows a system to provide more appropriate

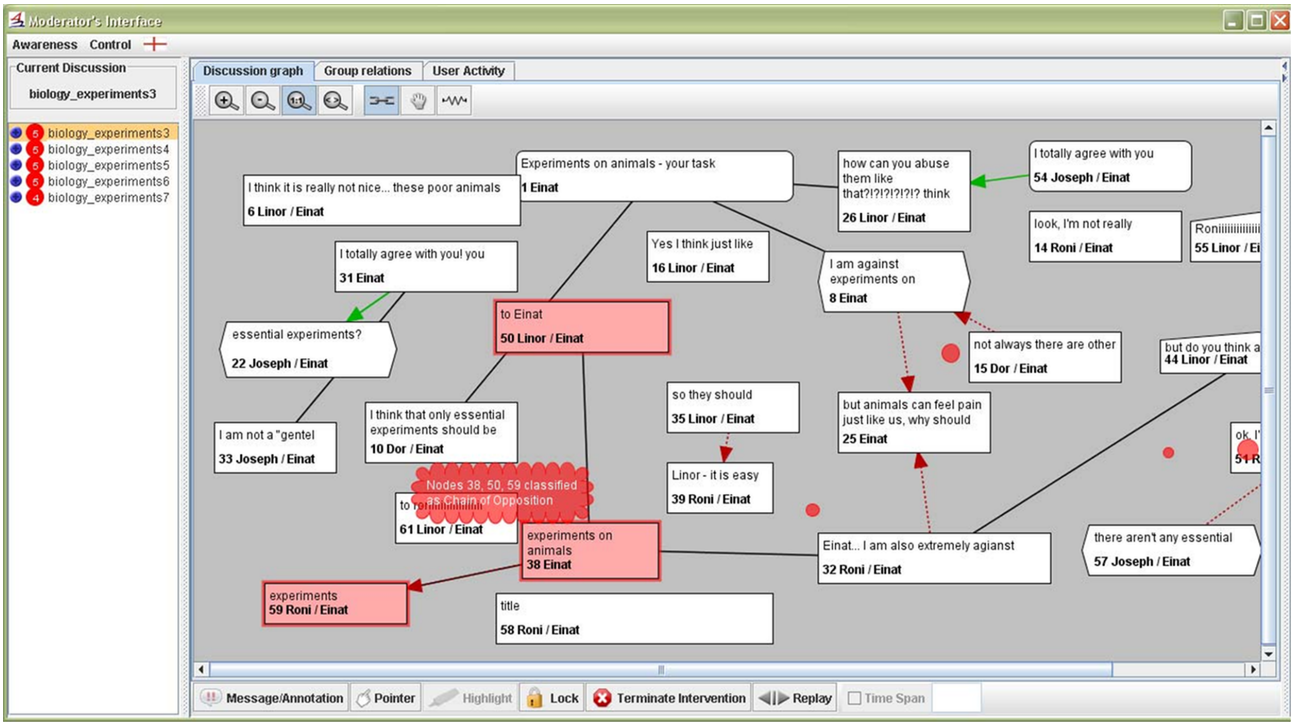

Figure 2: ARGUNAUT's Moderator's Interface displaying awareness information about an authentic student discussion.

feedback on student activities. *LARGO*, for instance, uses feedback messages that encourage reflection at the later stages of diagram construction, when diagrams have reached a sufficient degree of maturity.

DISCUSSION AND DISCUSSION-ENHANCED SYSTEMS

Systems discussed in this section aim at promoting productive discussions and knowledge acquisition via discussions. Before turning to different approaches to analyze discussions, we first give a short overview of five representative systems, which are *ARGUNAUT* (McLaren, Scheuer, & Mikšátko, 2010), *AcademicTalk* (McAlister et al., 2004), *Group Leader Tutor* (Israel & Aiken, 2007), *EPSILON* (Goodman et al., 2005) and *Pedabot* (Kim et al., 2008).

The *ARGUNAUT* system (McLaren et al., 2010) supports teachers in moderating multiple graphical e-Discussions simultaneously. A typical usage scenario is a classroom with groups of three to seven students discussing a controversial topic (e.g., "experiments on animals" – See Figure 2). *ARGUNAUT* provides advanced awareness and intervention tools to enable the teacher to manage the flood of data and provide appropriate guidance on the spot. A *Moderator's Interface* provides visualizations of important discussion aspects and alerts to highlight notable events in the discussions. Visualizations and alerts are based on "awareness indicators": Shallow indicators show, for instance, group relations, number of contributions per student and use of profanity. Deep indicators are based on AI analysis techniques, in particular machine learning and case-based graph matching. They show, for instance, off-topic contributions and chains of argumentation. A *Remote Control* allows teachers to intervene, e.g., by sending messages and highlighting portions of the e-Discussion. Figure 2 shows the *Moderators Interface*. The panel on the left displays five

discussions that are currently going on in a classroom. The panel on the right displays awareness information of one selected discussion. In this specific case uses of "chains of opposition" are marked as red dots in the region where this pattern was detected. The three highlighted contributions indicate one instance of this "chain of opposition" that has been selected by clicking on one of the red dots.

AcademicTalk (McAlister et al., 2004) supports synchronous debates in small distant groups to promote argumentation skills. Students use a forum-like interface to discuss contentious topics (e.g., "The Windows PC owes its success to features like the GUI, that were first developed for the Apple Mac"). *AcademicTalk* is based on the *Dialogue Games Theory*, which defines valid dialogue moves (e.g., assert, question, challenge) and dialogue rules (legal responses to prior contributions). A "Critical Reasoning" game is realized via sentence openers (dialogue moves), which can be highlighted to recommended appropriate ways to respond to prior messages (dialogue rules). The successor of *AcademicTalk* is *InterLoc* (Ravenscroft, Sagar, Baur, & Oriogun, 2008), which offers an improved user interface and an easy configuration of new dialogue games (i.e., available sentence openers and dialogue rules).

Group Leader Tutor (Israel & Aiken, 2007) facilitates small group discussions to help students engage in a more fruitful collaboration and to improve collaboration skills (e.g., creative conflict, leadership, communication). Discussions are synchronous, remote and related to a teacher-assigned task (e.g., designing a database according to a given set of requirements). Similar to *AcademicTalk,* students use a sentence opener interface, here for chat discussions. The *Group Leader Tutor* monitors the group discussions and intervenes when appropriate. For instance, when *Group Leader Tutor* detects that a discussion drifts off-topic, it might send the following message: "Please try to stay on-topic while working with your group." When the discussion is over, *Group Leader Tutor* provides summative feedback about how well the group collaborated.

In *EPSILON* (Encouraging Positive Social Interaction while Learning ON-Line; Goodman et al., 2005), students collaboratively create object-modeling diagrams based on given task descriptions. Because *EPSILON* was designed for remote collaboration it also provides a chat tool for deliberative discussions. Similar to *AcademicTalk* and *Group Leader Tutor*, students select from a pre-defined set of sentence openers to create new chat messages. An intelligent peer agent *Pierce* facilitates discussions and intervenes when collaboration problems occur (e.g., unanswered questions, seemingly confused students). For instance, when *Pierce* detects that student Milhouse is confused, he might send the following message: "Excuse me, Milhouse, but I think you might be confused. You should ask for help on this topic." A second kind of support is given through visual meters that display the "health" of the collaboration (e.g., a student's activity level).

Pedabot (Kim et al., 2008) supports the use of online discussion boards in technical domains, for instance, in courses on operating systems. Typically, such courses are repeated in schools several times over a couple of years. Also, the same topics and questions often recur in the online discussions of following years. *Pedabot* takes advantage of this, using relevant discussion content from previous years to scaffold current discussions. New threads are automatically analyzed to retrieve *relevant* past messages, i.e., messages that cover the same topic. The three most relevant past messages are displayed together with the current discussion thread to provide useful information, promote reflection and stimulate the current discussion.

Table 2: Overview of analysis approaches for discourse

Analysis Approach	Description
Identification of process characteristics	Classifying discourse segments according to process characteristics (e.g., speaker intention, interaction patterns) **Systems:** *Group Leader Tutor*, *EPSILON*, *AcademicTalk*, *ARGUNAUT*, Rosé et al. (2008), Verbree et al. (2006)
Identification of discussion topics	Classifying discourse segments according to discussion topics **Systems:** *EPSILON*, *Pedabot*, Kumar et al. (2007)
Identification of problems in discussions	Identifying interaction problems, e.g., unresponsiveness and failed knowledge sharing **Systems:** *ARGUNAUT*, *EPSILON*, Ravi & Kim (2007)
Assessment of collaboration quality of longer sequences of time	Group and student models to aggregate and evaluate student behavior to assess collaboration quality **Systems:** *Group Leader Tutor*, *EPSILON*
Classification of the current discussion phase	Classifying in which discussion phase a group currently is according to a predefined process model **Systems:** *Group Leader Tutor*

In the next sections, we discuss different approaches of discussion analyses used in these and other systems. Table 2 shows an overview of the discussed analysis approaches.

Identification of process characteristics

Approaches to classifying process characteristics abstract from the concrete textual contents to capture higher-level discourse characteristics such as speaker intentions and patterns of interaction. The approaches apply to different units of analysis ranging from single contributions (e.g., arguing for / against a position, maintaining the dialogue), to adjacency pairs (e.g., question-answer pairs) to larger interaction patterns (e.g., chains of opposition). Such automated classification procedures are typically used to condense complex discourse data, e.g., to support moderators, discourse analysts or subsequent automated analysis procedures.

To make the analysis more tractable, some systems use sentence opener interfaces, which predefine a limited set of phrases to start messages with. Typically there is an underlying model that defines how to interpret a chosen sentence opener. For instance, in an early version of the *Group Leader Tutor* (McManus & Aiken, 1995), each sentence opener is associated with the application of collaboration skills of a three-level hierarchy. For instance, the sentence opener "The advantages of this idea are …" corresponds to the triple (Creative Conflict, Structuring Controversy, Preparing a Pro Position). Their approach was later adopted in the *EPSILON* system (Soller, Goodman, Linton, & Gaimari, 1998). Classifications solely based on sentence openers might suffer from a high misclassification rate, e.g., Israel and Aiken (2007) report a 25% error rate: Sometimes, the text typed in is unrelated to the chosen sentence opener. In other cases, sentence openers can be interpreted in multiple ways. For instance, sentences starting with "I think …" can propose a future action (e.g., "I think we should …") but also express agreement with the group's current course of action (e.g., "I think that's good."). To increase accuracy, the newer version of *Group Leader Tutor* (Israel & Aiken, 2007) also analyzes keywords in the free text sentence closers. *AcademicTalk* (McAlister et al., 2004) uses sentence openers in combination with rules that specify appropriate response types. For instance, *AcademicTalk* highlights the

sentence opener "Let me elaborate …" as an appropriate response to messages that start with "Can you elaborate?…"

Sentence openers allow for easy identification of communicative intentions without natural language processing. However, the discourse is also restricted because students have to select from a predefined set of openers. This can have positive effects, for instance, students might discuss more critically because sentence openers give them the "permission" to directly challenge other contributions, overcoming politeness rules (McAlister et al., 2004, p. 200). On the other hand, students might not find a sentence opener that matches what they want to express, possibly leading to misuse of openers or omitted contributions. The design of sentence opener interfaces therefore requires a careful analysis to achieve desirable effects. For instance, sentence openers in *EPSILON* have been refined to "enable the widest possible range of communication with respect to the learning task" (Soller, 2001, p. 50), and the *Group Leader Tutor* has been enhanced with new sentence openers based on students requesting more choices when discussing aspects related to "working on task" (Israel & Aiken, 2007).

In contrast to sentence opener approaches, which combine interaction design and knowledge engineering to manually build techniques to classify contributions, supervised machine learning (ML) techniques have the potential to automatically derive classifiers from coded data (Witten & Frank, 2005). The following three approaches employ authentic discussion data coded along several dimensions of interest. Rosé et al. (2008) built classifiers for message segments in a threaded discussion according to a coding scheme for argumentative knowledge construction (Weinberger & Fischer, 2006). Their approach aims at supporting analysts in the task of coding collaborative interactions. They developed successful classifiers (Kappa values greater than or close to 0.7) to analyze, among others, the "micro-level of argumentation" (e.g., claims, warrants), the "macro-level of argumentation" (e.g., arguments, counterarguments, integrations), and "social modes of co-construction" (e.g., externalization, elicitation, quick consensus building, conflict-oriented consensus building). *ARGUNAUT*'s *Deep Loop* (McLaren et al., 2010) uses a similar approach to support a moderator's awareness in graphical e-Discussions. Six classifiers were successfully learned, with Kappa values above 0.62. The classifiers detect, among other student activities (both individual and collaborative), off-topic contributions, reasoned claims, question-answer pairs and contribution-counterargument pairs. The most obvious differences between the approaches is that the classifiers from Rosé et al. (2008) are applied to message segments, while the *ARGUNAUT* classifiers are used to analyze complete messages and pairs of related messages. Also, both approaches differ in the set of features that have been used to build classifiers: Both approaches use linguistic features that are extracted from the contributions' texts (e.g., unigrams, bigrams, part-of-speech). However, while Rosé et al. (2008) enhance this basic set with thread structure features (e.g., depth of the message in the thread where it occurs), McLaren et al. (2010) exploit the richer semantics of argumentation graphs (link and contribution labels). Another ML-based approach for classifying discourse data automatically is described by Verbree, Rienks and Heylen (2006). Contributions in meeting discussion protocols are automatically classified according to a coding scheme for meeting deliberations. The classifier distinguishes between six classes, for instance, statements, open issues and issues that require a decision between two alternatives. Here, lexical properties and the labels of the two preceding contributions are used as features. The classifier achieves an accuracy value of 79% (Kappa value of 0.63, recomputed from the given confusion matrix). The long-term goal is

to automatically construct argumentation diagrams as a kind of an "organizational memory" for meeting discussions, i.e., also classifying the types of relations between contributions.

Automatically inducing classifiers using ML is a vital alternative, especially since knowledge engineering approaches (i.e., the hand-crafting of classifiers, the development of expert rules) are often infeasible or highly expensive in terms of time and effort. The identification of process characteristics in natural language discourse is complex and therefore a promising application area for ML, as illustrated by the three approaches described above. However, ML does not completely save the developer from work. Sufficient amounts of data must be collected and coded to train and validate classifiers. For instance, the three approaches discussed above used between 1200 and 5000 coded text segments per category. Other notable efforts include the development and validation of coding schemes and coder training. A related question is a classifier's scope of application. A classifier that was learned with data from one population does not necessarily work well with another population because the use of software and language might differ. Our final issue concerns the unit of analysis. Two of the presented approaches (Rosé et al., 2008; Verbree et al., 2006) rely on pre-segmented data that does not correspond to natural delimitations such as sentence or message boundaries. For instance, each message might contain several "epistemic units," which are currently identified by hand in the approach of Rosé et al. (2008). Extending these approaches to a fully automated online system would require automated and accurate segmentation. *ARGUNAUT*'s ML classifiers do not use manually pre-segmented data. Rather, they use single contributions and pairs of contributions as they occur in the graphical discussions. To capture patterns that go beyond such pre-defined structures, the *DOCE* algorithm (Detection of Clusters by Example; McLaren et al., 2010) has been developed. *DOCE* has shown promising, although preliminary, results, for instance, in identifying chains of opposition that stretch out over an indefinite number of contributions.

Identification of discussion topics

A second important analysis dimension is the identification of *discourse content*. Content can be analyzed on different levels of detail and complexity, ranging from deep semantic to shallow natural language processing. Deep semantic approaches are expensive in terms of development time and effort. Shallow analyses, on the other hand, are often sufficient to provide effective support. Therefore, we focus on representatives of the latter category, in particular approaches to identifying *discussion topics*. The discussed approaches tag messages with topic labels that denote key concepts in technical domains (thermodynamics, object modeling techniques, operating systems). Such topic information can be used to support discourse in many different ways, for instance, to ensure topical coherence, to prompt students with topics that have not been covered yet (e.g., to stimulate stagnant discussions) and to pre-segment discourse according to topic boundaries (e.g., to prepare subsequent analyses [Soller, 2004]). Similar to some of the process analyses discussed before, also topic detection is a *text categorization problem*. We will present different approaches to the problem that use knowledge engineering, ML (Sebastiani, 2002) and Information Retrieval (IR; Baeza-Yates & Ribeiro-Neto, 1999) techniques.

A knowledge engineering approach to topic classification is presented by Goodman et al. (2005). Their topic tracker can identify six topics in deliberative dialogues about the creation of object modeling diagrams. Topics of interest are, for instance, "defining classes" and "creating associations between classes." An intelligent agent uses the topic tracker to ensure a coherent discussion and sufficient coverage of relevant topics. An initial topic

tracker version used a small set of domain-specific keywords, which had been selected based on an analysis of past discussions. This initial version suffered especially from many *errors of omission* (i.e., students talk about a topic not detected by the classifier), which could, among others, be tracked back to a too limited vocabulary (e.g., missing *problem-specific* keywords), unresolved referential statements (e.g., pronominal references) and typing errors. An improved version has been implemented that extends the keyword approach by also considering topic trends and transitions (for instance, referential statements as indicators for the continuation of the previous topic). Kumar, Rosé, Wang, Joshi, & Robinson (2007) developed an approach to identify 16 thermodynamics topics (e.g., "Reheat Cycles") in dyadic dialogues. Seven of these topics are used, when detected, to trigger "knowledge construction dialogues" involving "directed lines of reasoning" with a tutorial agent (Kumar et al, 2007, p. 386). The topic classifier works in two stages. In a first step a ML classifier makes a binary decision whether relevant topics have been raised or not. This serves to filter out messages without any relevant topic, which caused a considerable number of false alarms in preliminary tests. In a second step, the specific topic is selected according to the highest term frequency-inverse document frequency score (TF-IDF). TF-IDF is an information retrieval measure used here to determine how strongly terms are associated with specific topics. *Pedabot* (Kim et al., 2008) retrieves messages from past discussions to scaffold current discussions in the context of undergraduate computer science courses on operation systems. A topic profiler is used to filter out past messages without topical relevance. The topic profiler computes the similarity between messages and topics in a vector space whose dimensions represent task-related and technical terms. A message gets a high score for a topic when it contains many topic-relevant and few topic-irrelevant terms.

Topic identification has many similarities with the identification of process characteristics discussed earlier. In both cases, messages are analyzed and labeled using a pre-defined set of codes. However, there are also important differences in terms of predictive features and scope of application. While topic-specific content words are obviously the most important indicators for topics, they are of limited (or no) use to identifying process characteristics. On the other hand, the use of process scaffolds such as sentence openers and ontology categories are important process indicators but tell little, if anything, about discourse topics. Topic classifiers are restricted to a limited number of a priori determined topics that they were designed for, whereas the process-oriented classifiers capture aspects of human communication that are not tied to a specific topic, leading to a potentially broader scope of applicability. Promising in this respect are approaches like *Pedabot* in which relevant topics and indicative terms for these topics are extracted in fully automated form (Feng, Kim, Shaw, & Hovy, 2006).

Identification of problems in discussions

The approaches that have been discussed so far have been concerned with how students communicate, and about what. However, some of the process characteristics are more than "neutral" descriptors of the process because they also represent aspects of *discourse quality*, some of which can be interpreted positively (e.g., reasoned claims, chains of opposition, conflict-oriented consensus building), others negatively (e.g., off-topic contributions, quick consensus building). This section is devoted in particular to the detection of negative aspects, i.e., *problems* in discussions. We discuss approaches to identify (possibly) problematic patterns in discussions, failed attempts to share knowledge, and lack of responsiveness.

Besides the d*eep alerts* discussed above (e.g., chains of opposition), the *ARGUNAUT* system (Hoppe et al., 2008) also provides *shallow alerts*, which are based on simpler, hand-crafted analysis rules. Moderators use these rules to uncover patterns of problematic student-to-student interaction, for instance, inactivity (e.g., a student has not contributed to a discussion map for *x* minutes), a lack of interaction (e.g., contributions are unconnected for *x* minutes, or users who only link their own contributions with one another), undesired social behavior (e.g., use of profanity, ignored users, indicated by a lack of links to this user's contributions for *x* minutes) or dominant / overly passive users. Via a configuration interface, moderators can set up which rules to run when as well as rule-specific parameters (e.g., keywords to look for). The patterns found by the *Shallow Loop* are simple but effective in indicating possible problems in student-to-student communication. Because they rely on common sense heuristics, they are intuitive and intelligible, which is an advantage over machine-learned models, which are often complex and hard to understand by humans.

It can be a serious problem in discussions when students fail to effectively share knowledge. It is important that participants share common ground in terms of discussion format, values and background knowledge (cf. Van Eemeren & Grootendorst, 2004, p. 58). To maintain (and negotiate) a common knowledge base, students need to exchange knowledge with their fellow students. An approach to detect whether such *knowledge sharing* is successful has been developed within the deliberative discussions of the *EPSILON* system (Soller, 2004). The pieces of shared knowledge correspond to key object modeling concepts. A predictive model has been inferred consisting of two *Hidden Markov Models* (HMM; Rabiner, 1989). One HMM was trained with sequences of dialogue acts and workspace actions that were annotated as successful knowledge sharing. Analogously, a second HMM was trained with failed knowledge sharing attempts. To classify new sequences, both HMMs are applied, the resultant probabilities for success and failure compared and the classification with the highest probability selected. An interesting aspect that distinguishes HMMs from the ML approaches discussed above is their ability to capture *sequential dependencies* between observations. Communication processes are by nature sequential; hence, natively sequential modeling approaches like HMMs may be a good fit for such modeling problems. In this specific case, initial experiments with non-sequential flat representation models did not lead to reliable predictions. However, the HMM results have to be taken with a grain of salt: First, the experiments involved only a very limited number of examples (29), raising the question of reliability and generality. Second, the achieved accuracy of 74% (Kappa of 0.47, recomputed from a given confusion matrix) is clearly better than chance, i.e. the classifier successfully captures some regularities in the students' behavior, But is this enough to build a tutorial model upon? The answer is: We do not know. This question cannot be answered in general terms based on the achieved accuracy or kappa value. It is important to consider the specific usage context and possible (mis-)classification costs. We will come back to this point later in the *Conclusion*. Finally, similar to some approaches discussed earlier, experiments were conducted on manually pre-segmented data. Full automation of analysis would require another component to pre-segment the data.

Productive discussions require participants to be responsive, i.e. to answer questions and to acknowledge the statements of their peers. We discuss two approaches to detect when the norm of responsiveness is violated. In *EPSILON* (Goodman et al., 2005) a classifier consisting of two *Artificial Neural Networks* (ANN) has been developed, one for question detection and the other for answer detection.

Both ANNs are used in tandem, i.e., when a question has been detected, every contribution made within two minutes is tested for being an answer to this question. Immediate prompts are provided when unanswered questions are detected. The ANNs analyze students' contributions in terms of dialogue acts and surface features (e.g., occurrence of a question mark). Ravi and Kim (2007) discuss a similar approach to analyze threads in a discussion board. Two ML classifiers were learned for detecting questions and answers, respectively, using n-gram features (i.e., terms, term pairs, terms triples and term quadruples). The classifiers are intended to bring unanswered questions to the attention of an instructor.

Assessing collaboration quality over longer sequences of time

We discussed approaches to identify positive and negative patterns in discourses, e.g., off-topic contributions, conflict-oriented consensus building, successful, or failed knowledge sharing. However, these patterns only reflect single events and short-term interactions. They provide only limited information about quality and success of the interaction on a *macro-level*, i.e., for extended sequences of time, complete discussions or across multiple discussions. The field of *Intelligent Tutoring Systems* (ITS; VanLehn, 2006) has a long tradition in building student models, which not only analyze current student behavior (e.g., using a cognitive model) but also track the development of (hidden) student traits over longer stretches of time (e.g., assessment of student mastery levels for domain skills). Inspired by classical ITS systems, student and group models have also been developed for discussion systems. These models aggregate evidence from discussions over time, in particular, student actions as they occur in the system (e.g., counting the number of activities), but also results from prior analyses (e.g., counting the number of questions the student has replied to). Such aggregations serve as *indicators* of the current state of the students' interaction (e.g., student *X* contributed *10%* of all messages), which can be compared to a model of desired interaction to diagnose the quality of student interaction (e.g., student *X* has a *low* activity level) (Soller, Monés, Jermann, & Mühlenbrock, 2005).

In *EPSILON* (Goodman et al., 2005), group and student models are used to keep statistics on students' behavior and to determine the general "health" of group collaboration. The group model contains values for group responsiveness (based on the number of answered and unanswered questions), group agreement (based on the number of corresponding dialogue acts) and dialogue speed (based on action counts). Analogously, student models contain values for the student's certainty / confusion and activity level. These indicators are mainly used for the visualization of group health to support student self-reflection. They can also be used for immediate feedback, for instance, to prompt the least active student.

Student and group models of the *Group Leader Tutor* (Israel & Aiken, 2007) count the number of off-topic contributions, initiated new ideas, inappropriate use of sentence openers, and uses of the different collaboration skills, all derived from sentence openers / keywords as described above. When critical values are reached, *Group Leader Tutor* prompts students to improve their collaboration, for instance, when an imbalance of the number of contributions per participant is detected, or when the frequency of off-topic contributions surpasses a certain threshold. Low-level counts are further aggregated to higher-level indicators of the group's collaborative effort. For instance, the indicator "initiating ideas and assuming personal responsibility" is defined as the ratio between initiated ideas and the total number of contributions by that student. A low value indicates that the student rarely initiates new ideas (classification: "low"). Values in the mid

range show a good balance between initiating and responding (classification: "high"). Other values indicate sub-optimal but perhaps not alarming behavior (classification: "medium"). Group and student models are displayed to the participants after the session is closed to stimulate self-reflection processes.

Classification of the current discussion phase

To analyze and evaluate discussions, some researchers have developed *process models* that represent regular patterns observed in past discussions and/or give a normative account of how discussions should proceed. For instance, Van Eemeren & Grootendorst (2004) discuss a model for critical discussions that consists of a "Confrontation," an "Opening," an "Argumentation," and a "Conclusion" stage. Roschelle (1992) describes the process of convergence of meaning in science discussions through iterative cycles of displaying, confirming and repairing shared meaning. This framework has been formalized in the *Group Leader Tutor* (Israel & Aiken, 2007) to track and mediate phases of create conflict in student discussions. A finite state machine (FSM) models the transition through a state space that can be subdivided into the phases "Display," "Confirm/Disconfirm," and "Repair until Convergence." State transitions are triggered based on the selected sentence openers. The FSMs might trigger tutorial interventions when dedicated states are entered or discussion loops are detected, which indicate a lack of progress.

SUPPORT MECHANISMS

Automated argument / discussion analysis is not an end in itself; it typically serves the purpose of supporting feedback: providing messages to the student to assist them in learning or in tackling the task at hand. In this section, we discuss feedback strategies, addressing in particular feedback mode and content, feedback timing, and feedback selection strategies.

Feedback mode and content

In the following, we discuss the different modes of feedback (i.e., textual, highlighting of argument elements, visualizations of behavioral/interaction aspects) and specific strategies used to phrase the textual content of feedback messages.

Textual. The most common form of feedback is textual messages presented to the student. Table 3 shows a selection of feedback messages that are provided by some of the reviewed systems. *Belvedere*'s (Suthers et al., 2001) feedback messages try to foster inquiry skills and the learning of principles of scientific argumentation, for instance, hypotheses should explain observed data, confirming as well as disconfirming evidence should be considered, and new, discriminating evidence should be consulted when two hypotheses have equal support. Feedback is presented as pre-canned text messages, which take the form of suggestions and questions when syntactic patterns are identified, and challenging feedback when differences between the students' diagram and an expert solution have been found. In a similar vein, *Rashi* (Dragon et al., 2006) supports students' inquiry activities by promoting the consideration of multiple hypotheses, top-down arguments (i.e., finding new data to support / refute existing arguments), bottom-up arguments (i.e., inferring propositions from existing elements) and correct usage of relationships between propositions. *LARGO* (Pinkwart et al., 2006b) supports the analysis of a legal transcript by presenting short versions of the five most relevant feedback messages to the student. (We discuss how the most relevant messages are chosen in section *Feedback selection and priority)*. The messages guide the student through the different phases of the learning task (i.e., applying the model of hypothetical reasoning to a legal transcript), starting from an orientation phase, over transcript mark-up,

Table 3: Example feedback messages

System (Domain)	Message Purpose	Message Content
Belvedere (scientific inquiry)	Avoid confirmation bias	"You've done a nice job of finding data that is consistent with this hypothesis. However, in science we must consider whether there is any evidence *against* our hypothesis as well as evidence for it. Otherwise we risk fooling ourselves into believing a false hypothesis. Is there any evidence against this hypothesis?" (*highlighting of Hypothesis element*)
	Discriminate between alternative hypotheses based on (especially negative) evidence	"These hypotheses are supported by the same data. When this happens, scientists look for more data as a 'tie breaker' - especially data that is *against* one hypothesis. Can you produce some data that would 'rule out' one of the hypotheses?" (*highlighting of both Hypothesis elements*)
Rashi (scientific inquiry)	Build bottom-up arguments (i.e., arguments from data to hypotheses)	"Here's a list of possible arguments. Try to pick the one you can support or refute with the data you have already: <*list of arguments*>"
	Repair wrong relationship type between propositions (different to expert model)	"Are you satisfied with the relationship you have established between *P1* and *P2*?" (*student can select between* "Yes, it is correct" *and* "No, help me to fix the relationship")
LARGO (analysis of legal arguments)	Repair modeling weakness: Hypothetical elements should relate to Test elements	"In your solution, the hypotheticals H1 and H2 are distinguished from each other. Yet, hypothetical H2 is not related to any test or the current fact situation. Please explain why you did so, either in free text or by modifying the diagram." (*highlighting of Hypothetical elements H1 and H2*)
	Consider important transcript passage that have not been considered yet	"Please look at this part of the transcript (scroll to line *L*) and explain its role within the argument."
Group Leader Tutor (group deliberation)	Avoid off-topic contributions	"Please try to stay on-topic while working with your group"
	Expressing only one idea per messages to improve the collaboration skill of communication	"You are trying to express several ideas in one sentence. Please re-enter your statements, one idea at a time."
EPSILON / Pierce (group deliberation)	Respond to peer messages that have not been acknowledged or answers yet	"Sarah said '...'. What do you think about that, Jeremy?"
	Eliciting help from peers when something has not been understood	"Excuse me, Milhouse, but I think you might be confused. You should ask for help on this topic."

diagram creation and analysis phases, and finally concluding in a reflection phase. The three systems use suggestions / prompts for self-reflection rather than imperative / corrective formulations to avoid confusion when a diagnosis is a "false alarm" (Pinkwart et al., 2006a) and to foster the development of the students' skills of self and peer critiquing, i.e., the feedback should encourage the student to think for him or herself about the diagram and possible weaknesses (Suthers et al., 2001). While the just discussed systems provide feedback to support the creation of argument diagrams the

next two approaches aim at promoting productive group deliberation. *Group Leader Tutor* (Israel & Aiken, 2007) helps collaborating students become a "high-performing cooperative group with positive interdependence" (Israel & Aiken, 2007, p. 3). It sends messages to the students' chat discussion, which encourage the application of important collaborative skills like communication, trust, leadership and creative conflict skills. In some situations, when *Group Leader Tutor* cannot help by itself anymore, it might suggest to consult a human teacher. The peer agent *Pierce* (Goodman et al., 2005) appears as a learning companion to students and contributes messages to their chat conversation. *Pierce* can consider multiple indicators when generating a message, e.g., he might ask the student with the lowest activity (indicator 1) to comment on a statement that has not been acknowledged / answered (indicator 2). *Pierce*'s interventions are formulated as suggestions and questions.

These approaches pursue, on the whole, simple feedback strategies. They provide feedback messages in response to detected events and consider the history of interaction between student and tutorial agent only to a limited extent. For instance, *Belvedere* (Suthers et al., 2001) gives higher priority to feedback messages that have not been provided yet. *Rashi* (Dragon et al., 2006) supports short feedback sequences: A first message might indicate that support for a knowledge claim is missing. A second message might point the student more specifically to where corresponding evidence can be found. More sophisticated strategies are used by systems that support tutorial dialogues, which require a coherent and goal-directed behavior of an artificial agent. The tutorial agent presented by Yuan et al. (2008) uses his and the student's commitments (i.e., what has been said and accepted during the dialogue) to generate appropriate argument moves. The generated contributions are based on a planning unit and a knowledge base with interrelated propositions. The formulations take a stylized form (e.g., "Is it the case that … is acceptable?"). The dialogue strategy used by Kumar et al. (2007) targets the elicitation of correct conceptual knowledge rather than a proper use of arguments. A tutorial agent takes up a domain concept (or concepts) that have been identified in the students' dialogue and tries to stimulate the students to reflect on the concept(s) in a tutorial dialogue. The dialogues are based on *TuTalk*, a "dialogue system server and authoring tool that supports the rapid development of dialogue systems to be used in learning studies" (Jordan, Hall, Ringenberg, Cue, & Rosé, 2007, p. 43). On the other end of the spectrum are systems that do not generate tutorial feedback at all but rather delegate the task to a human moderator. *ARGUNAUT* (Hoppe et al., 2008), for instance, supports moderators in providing textual feedback in two ways, first as annotations that are embedded in the argument diagrams and, second, as messages that are displayed in pop-up windows. Of course, a human can formulate more appropriate and natural messages, but also must be available and willing to do this task.

Highlighting. A second form of feedback is highlighting of relevant portions of an argument diagram. Such an approach is used in *Belvedere* (Suthers et al., 2001) and *LARGO* (Pinkwart et al., 2006a) when the respective systems automatically find syntactic patterns in the students' diagrams. *ARGUNAUT* (Hoppe et al., 2008) provides a capability to allow moderators to highlight contributions in discussion maps to draw students' attention to salient features of the discussion. Highlighting by these systems is typically accompanied with a textual message. Instead of highlighting elements in an argument diagram, *AcademicTalk* (McAlister et al., 2004) visually emphasizes sentence starters to indicate recommended reply types to previous messages.

Meters. Meters are sometimes used to display group indicators (e.g., dialogue speed, relative amount of statements needing a reply, etc) and student indicators (e.g., certainty level, activity level, etc.). Meters can be used as mirroring tools to reflect students' actions and behaviors (e.g., student *X* provided 10% of all contributions) and as meta-cognitive tools that go further by evaluating students' actions and behaviors and indicate a desired state (e.g., student *X* has a low activity level) (Soller et al., 2005). The peer agent *Pierce* (Goodman et al., 2005) supports students' activity using meters that visualize aspects of the student and group model. The meters indicate whether the current value is in the normal range (green), in borderline range (yellow), or out-of-range (red). The design of *Pierce*'s meters was inspired by research on open student models (Bull, Brna, & Pain, 1995; Bull & Kay, 2007). *Rashi* (Dragon et al., 2006) is another example of meter usage, which is focused on individual reasoning skills (displaying the argument, hypothesis and data collection skill level). *ARGUNAUT* (Hoppe et al., 2008) also provides meters, but to support the discussion moderator rather than students.

Feedback control and timing

Feedback control and timing are crucial design decisions that can strongly affect whether learning is successful or not. In this section we discuss the different actors that trigger feedback (student, system, and moderator) and, in the case of system-triggered feedback, when the feedback is provided (immediate or delayed).

On-demand feedback. On-demand feedback is provided only upon a student's request. In *Belvedere* (Suthers et al., 2001) and *LARGO* (Pinkwart et al., 2006a), for instance, students request feedback to check for possible weaknesses in their argument diagrams and to receive hints on how to proceed. There are several reasons why such a strategy may be beneficial: First, the feedback is provided when the student really wants it, not interrupting naturally occurring activities (Dragon et al., 2006). Second, the student is not flooded with unnecessary messages since he or she decides the feedback frequency. Third, the construction of an argument diagram is a continuous process, with sometimes no clear end or conclusion, hence it makes sense to let the user decide when the process is ready to be checked (Pinkwart et al., 2006b). Fourth, on-demand feedback allows the student to assume more control and the tutoring component less control, possibly leading to more student motivation and less student discouragement (Suthers et al., 2001). On the downside, some students take minimal or no advantage of on-demand feedback, even when they are stuck and obviously need assistance, as observed in *Belvedere* (Suthers et al., 2001) and *LARGO* (Pinkwart, Lynch, Ashley, & Aleven, 2008b).

Immediate system feedback. Immediate system feedback is provided right after a mistake or problem is identified, without a student explicitly requesting help. The *Group Leader Tutor* (Israel & Aiken, 2007) uses this kind of feedback to "repair" communication problems (e.g., when an off-topic contribution has been detected), to mediate phases of creative conflict, to refocus the group when the discussion gets stuck, and to react to changes in the group model (e.g., the relative amount of participation of a student falls below a threshold). The system presented by Kumar et al. (2007) launches a tutorial dialogue whenever a topic profiler identifies a relevant domain topic in the students' contributions. The peer agent *Pierce* (Goodman et al., 2005) provides feedback on unanswered requests / questions, when students appear confused, or when out-of-sequence or missing topics are detected in the *EPSILON* system. Especially when feedback is intended to scaffold and improve the current student activity, it is best provided immediately. For instance, when a discussion is drifting off-topic, immediate feedback can be used to re-focus students again. Furthermore, as mentioned above,

many students do not make use of on-demand feedback and thus miss learning opportunities. On the other hand, if the amount of feedback becomes excessive, it could distract the student (see section *Feedback selection and priority)*.

Summative system feedback. Summative system feedback is provided after a session has finished, typically in order to provide students with the opportunity to reflect on their activities. This kind of feedback is provided by the *Group Leader Tutor* (Israel & Aiken, 2007), which displays the content of the student and group model to students at session end in order to encourage both reflection and self-assessment of the collaboration. A positive aspect is that delayed feedback does not interfere with on-going students' activities. On the other hand, the feedback does not scaffold the student activities in the context in which a problem occurs.

There are mixed results with respect to whether immediate or delayed feedback approaches are more effective (Shute, 2008). In fact, this might depend on the concrete task, situation and student population. Furthermore, using the one does not exclude using the other. Immediate and summative feedback can be seen as complementary approaches, one to provide immediate scaffolding and the other to foster reflection processes when the actual activity is over.

Moderator-driven feedback. Moderator-driven feedback is used in situations in which a human moderator or teachers decides when to provide feedback to students. For instance, in *ARGUNAUT* (Hoppe et al., 2008) a *Moderator's Interface* increases a moderator's awareness and helps him / her decide when to intervene. Moderators can select alerting rules at the push of a button that provide information about, for instance, off-topic contributions and imbalanced participation of group members. If and when interventions are triggered is completely under the control of the moderator. A more extensive treatment of e-moderation within the *ARGUNAUT* system can be found elsewhere in this book (De Groot, 2010).

Feedback selection and priority

It is often helpful to control the frequency and selection of feedback in order to provide the right amount of feedback without flooding students with messages. Interventions of the peer agent *Pierce* (Goodman et al., 2005) have been simulated with existing usage protocols (recorded without the peer agent). If all feedback messages had been sent out, *Pierce* would have interrupted students 75 times simply to react to unacknowledged contributions during a 338 utterance dialogue, and 38 times to hint on out-of-sequence or missing topics during a 328 utterance dialogue. Similarly, the number of identified characteristics sometimes exceeds 100 for a single *LARGO* diagram (Pinkwart et al., 2008a). This amount of intervention by *Pierce* and *LARGO* may lead to more harm than benefit for students.

Belvedere (Suthers et al., 2001) and *LARGO* (Pinkwart et al., 2006a) address this issue by providing the most important and short versions of the five most important feedback messages, respectively, when students request help. *Belvedere* uses a preference-based quick-sort algorithm to make this choice. The prioritization algorithm iterates through a list of criteria, which are ordered from most to the least important. After applying the first criterion, the second one is used to prioritize feedback that received the same priority value in the first iteration and so on (i.e. consecutive criteria are used as "tie breakers" for preceding ones). Some of *Belvedere*'s criteria are: priority of new advice, priority of expert advice over syntactic advice, priority of advice that binds to diagram elements that have been created by the advice-requesting student, priority of certain pattern types over other types, etc. *LARGO* (Pinkwart et al., 2006a) uses, among other sources of data, the

diagnosed usage phases to determine appropriate feedback. Each pattern is associated with one out of five different usage phases. When the student requests a hint message, those that correspond to the current usage phases are preferred. In *ARGUNAUT* (Hoppe et al., 2008), control and regulation of feedback is left to a moderator, typically a classroom teacher, who is assumed to be knowledgeable enough to select the most important feedback. Goodman et al. (2005) propose to tune the activation threshold of *Pierce*'s intervention rules to reduce the number of interventions. They also further suggest the application of more in-depth natural language analyses to improve the accuracy of the indicators and hence, the accuracy of feedback (less "false alarms").

ARCHITECTURAL CONSIDERATIONS

The development of effective argumentation support systems involves considerable time and effort. Thus it is important to base systems on a well-conceived software architecture that combines general good practices in software design with the specific requirements of argumentation support systems (see, for instance, Loll, Pinkwart, Scheuer, & McLaren, 2010 in this book). Figure 3 depicts a conceptual architecture of support engines for collaborative, educational argumentation systems. The architecture combines ideas and adopts terminology from the reviewed argumentation systems, traditional architectural layouts used in ITSs (Poison & Richardson, 1988; Corbett, Koedinger, & Anderson, 1997; Mitrovic et al., 2001) and our current work on the *LASAD* (Learning to Argue: Generalized Support Across Domains) framework (Loll et al., 2010; Scheuer, McLaren, Loll, & Pinkwart, 2009). *LASAD* aims at providing a flexible infrastructure for the development of argumentation systems across domains including dynamic support mechanisms. We use the reference architecture in Figure 3 as a guide through our discussion to highlight the most important considerations in the design of argumentation support. We first address how adaptive support functionality is integrated into the overall system, then discuss possible components and their interplay inside a support engine, and finally conclude with some remarks on the communication between support engine components.

The reference architecture has two main parts: the Argumentation Environment and the Support Engine. The *Argumentation Environment* consists of a central service infrastructure and connected clients, a design appropriate for collaborative systems that support multiple clients in a consistent and synchronized state (as depicted in Figure 3 and used, for instance, in the systems *Belvedere* [Suthers & Jones, 1997], *EPSILON* [Goodman et al., 2005], *ARGUNAUT* [Harrer, Ziebarth, Giemza, & Hoppe, 2008] and *Group Leader Tutor* [Israel & Aiken, 2007]), or alternatively, could just be a single user interface, a design sufficient for single user systems (e.g., *ArguMed* [Verheij, 2003]). The *Support Engine* comprises all the functionality and data structures necessary to analyze, model and support discussions and arguments. The architectural decoupling of intelligent tutoring components from the task environment has shown to be crucial for code maintenance, reusability and extensibility (Linn, Segedy, Jeong, Podgursky, & Biswas, 2009).

The Argumentation Environment and Support Engine can communicate within a single machine (e.g., *ArguMed* [Verheij, 2003]) or over the web, which leads to an even higher degree of decoupling because a physical separation becomes possible, a natural setup in collaborative environments. Especially in web-based settings, a key decision is which technologies and interfaces to use for the integration of both parts. In *Belvedere*, for instance, intelligent "coaches" use the same

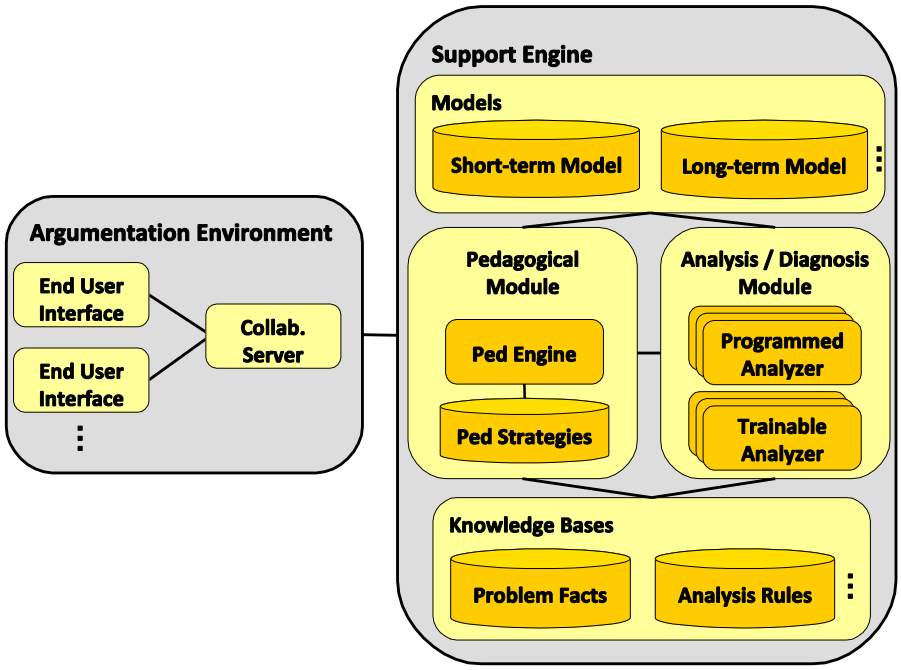

Figure 3: Reference architecture of a support engine for collaborative, educational argumentation systems.

communication protocols as end user clients, i.e., new coaches can be integrated by just replacing the user interface part of an existing end user client by intelligent coaching logic (Suthers & Jones, 1997). In our current work on the *LASAD* project we take a similar approach in that our adaptive support agents will be just another kind of client to the central collaboration server that uses exactly the same communication interfaces. There will be a special role assigned to machine agents in *LASAD*'s user management components, which grants special privileges, e.g. writing to a dedicated tutoring console. The same software interface could then also be used to connect human moderators or tutors.

During runtime, the Argumentation Environment typically sends messages that either represent a complete state (e.g., an entire argumentation graph) or state changes (e.g., a node *n* with properties *p* has been added to the argumentation graph by user *u* at time *t*). Across the different systems a number of state and action-based formats have been utilized. That is, there is no consensus on which format to use, something that clearly hampers the interoperability of systems, but on the other hand also reflects, to some extent, differences in the systems' conceptualizations. To achieve interoperability, additional infrastructure may be needed to unify otherwise incompatible formats (Harrer et al., 2008). Some current research investigates argumentation ontologies to achieve semantic interoperability between systems (Chesñevar et al., 2007; Rahwan & Banihashemi, 2008).

We now turn to the inner workings of the Support Engine. In its simplest form, the Support Engine makes no pedagogical decisions or student modeling (e.g., *Carneades* [Gordon et al., 2007] and *ArguMed* [Verheij, 2003]). Here, the Support Engine consists solely of an *Analysis Module* and a set of *Expert Analysis Rules* to evaluate the dialectical status of propositions; results are then provided back in unfiltered form. *Convince Me* (Ranney & Schank, 1998), although educational, works in a similar manner, but uses the *ECHO* simulator in place of expert rules. More typically, educational support systems make use of an additional *Pedagogical Module*, which processes the provided analysis results and decides on

pedagogically appropriate actions. The architectural separation of diagnostics and pedagogical decision making is a recurring pattern in the design of intelligent learning systems (e.g., Mitrovic et al., 2001; Goguadze, 2009; Linn et al., 2009; Chaudhuri, Kumar, Howley, & Rosé, 2009), which allows easy modification of pedagogical strategies without affecting the diagnosis component. *Belvedere* (Suthers & Jones, 1997), *LARGO* (Pinkwart et al. 2008a), and *EPSILON* (Goodman et al., 2005) use a Pedagogical Module to prioritize and filter diagnoses according to their relative importance and urgency, based on the diagnosis type and other, dynamically evolving factors like current modeling / discussion phase or feedback history. As indicated in Figure 3, pedagogical strategies can be dynamically loaded in declarative form (e.g., XML configuration files), making customization of strategies easy. This approach has been taken in a number of intelligent learning systems (e.g., Linn et al., 2009; Goguadze, 2009) and we also intend to adopt it in *LASAD*. To support more complex instructional interactions it is necessary to monitor and aggregate student and group behavior over a longer period of time. Therefore, systems like *Group Leader Tutor* (Israel & Aiken, 2007) and *EPSILON* (Goodman et al., 2005) maintain student / group models to keep track of relevant parameters (e.g., responsiveness, active participation; off-topic contribution count) for the current session (*Short-term Models*) or across multiple sessions (*Long term Models*). Typically, such models are passive data structures, which have their operational counterpart in the Analysis / Diagnosis Module, which updates the model content based on an analysis of student actions and solutions. The Diagnosis Module itself can take various forms: It might consist of *Programmed Analyzers*, whose behaviors are defined by the system designers (e.g., *Belvedere* [Suthers et al., 2001], *Rashi* [Dragon et al., 2006], *LARGO* [Pinkwart et al. 2008a]) and/or *Trainable Analyzers*, whose behaviors are induced from data (e.g., *EPSILON* [Goodman et al., 2005; Soller, 2004], *ARGUNAUT* [McLaren et al., 2010]). Analyzers are often subdivided into an operational part and declarative expert knowledge. *Rashi* (Dragon et al., 2006), for instance, uses a rule-based engine (the operational part) in concert with domain / problem-independent *Expert Analysis Rules*, which encode general reasoning knowledge to identify relevant patterns, and an *Expert Problem Fact Base*, which keeps formal descriptions of individual problem instances. The expert rules are then used to compare the student's current solution with the corresponding facts in the problem fact base. Defining behavior and problem characteristics in a declarative knowledge representation makes analyzers easily customizable (changing the rule base) and extensible to other problems (adding facts for new problem instances). These processes, especially the creation of a sufficiently large problem fact base, can nevertheless be quite laborious and time consuming, which motivated the development of authoring tools (Murray et al., 2004). It should be noted that both Analysis / Diagnosis and Pedagogical Modules can, in contrast to the examples discussed here, be quite complex and unfold into a number of subcomponents and knowledge resources, for instance, when natural language understanding or generation is involved. Even without natural language processing, the planning and generating of appropriate system responses can go well beyond our previous examples, as illustrated in the human-computer dialogues approach by Yuan et al. (2008).

Finally, we take a look at how Support Engine components are integrated with one another. To facilitate a possible reuse in other contexts, some of the components depicted in Figure 3 might be implemented as independent services. For instance, the AI-based discussion analysis component in *ARGUNAUT*, *Deep Loop*, is deployed as a web service (McLaren et al., 2010), which can be accessed by any client that has the appropriate access rights

and correct API to analyze graphical discussions. A second advantage of independent, web-accessible services is scalability, i.e. computationally expensive (e.g., analysis) tasks can be deployed on a separate server machine. The behavior of a Support Engine might be determined by a considerable number of more or less independent diagnostic and pedagogical tasks. Good software design practice demands encapsulating functionality for different tasks in different software modules, which share their processing results with one another as needed. *Group Leader Tutor* (Israel & Aiken, 2007) uses a software agent design, i.e., tasks are executed by autonomous agents, for instance to classify contributions, update a discussion model, check the discussion model state, update user and group models, etc. These agents communicate with one another in a goal-oriented way by exchanging messages. In *LASAD,* we also intend to adopt the multi-agent paradigm with configurable analysis and feedback agents that can be easily plugged into the system.

CONCLUSIONS

In this chapter, we reviewed a wide variety of analysis mechanisms for computer-supported argumentation. We discussed argument analysis mechanisms that check student-created arguments for syntactic constraints, identify differences with expert models, simulate reasoning processes, assess content quality using collaborative filtering, and identify modeling phases. We then discussed analysis mechanisms for discourse processes that identify process characteristics, discussion topics, collaboration problems and discussion phases, and keep aggregated models of student and group interaction. These approaches cover a wide range of characteristics that can be identified in argument models and discourse without deep semantic processing of natural language text. We also discussed feedback approaches to effectively support argumentation learning. Finally, we addressed architectural / technological aspects relevant to system design and implementation.

Despite the fact that considerable progress has been made, adaptive support mechanisms are still not widely used in practice. The most exciting, yet unrealized, possibility of automated analysis is dynamic adaptation – to the individual, the group, the process and the current situation. To achieve this, there are design goals and research directions that must be pursued. In the following paragraphs we describe important design goals, give recommendations regarding the design and development process and highlight future research directions.

A central question is how to cope with analysis errors. Reasons for errors are manifold: For instance, a number of analysis procedures are induced using machine-learning procedures (e.g., *ARGUNAUT*, *EPSILON*). Because such procedures generalize from a limited set of examples, the resultant classifiers are typically imperfect. A second example is the calculation of differences between student and expert models (e.g., in *Belvedere*, *LARGO*, *Rashi*), which can be misleading because the expert models only represent one of many different ways to solve a given problem. No matter what the actual reason for an analysis error is, the corresponding feedback could potentially confuse and/or discourage students. To address this, some systems provide “soft” feedback, i.e., instead of emphasizing a mistake, the system gently asks the student to reflect about some aspect of his / her solution (e.g., *Belvedere*, *LARGO*). A second approach is to improve the analysis method itself to reduce errors. Often neglected in the design of classifiers are the *relative* error costs. Different types of misclassifications can often be associated with different costs, e.g., incorrectly identifying a non-existent problem might be worse than overlooking an actual instance of a problem. In such cases, the

analysis method should be designed, if possible, to produce a smaller amount of "costly" errors at the expensive of a larger amount of "cheap" errors. In machine learning literature, this approach is known as cost-sensitive learning. Following decision theory, the goal is to maximize a classifier's expected utility, i.e., find the best trade-off between expected costs and benefits. To summarize, it is important that analysis approaches are sufficiently reliable. A cost-benefit analysis might improve a classifier's utility. When designing a feedback approach, error costs and frequencies should be considered.

A second important issue concerns the general design and development approach. Even if a system is based on well-founded theories and principles, its actual success can only be assessed empirically with real users. Iterative, participatory design approaches include important stakeholders and regular experts – from an early stage and on a regular basis. Formative evaluations should drive the development process to identify weaknesses and aspects that need special attention, e.g., to identify analyses that are too error-prone and feedback messages that are not well understood by students. Such evaluations also help to get a sense of possible misclassification costs. The development of adaptive support is costly in terms of time and effort. Before developing an expensive analysis approach it is usually instructive to test the envisioned support in *Wizard-of-Oz* experiments, in which system behavior is simulated by a human operator, i.e. a functional (software) implementation is not needed (Tsovaltzi et al., 2010). A promising design approach might be to adopt a model of human tutoring (i.e., when do humans intervene, what do they say, etc). However, it is important to also consider differences between both modes of tutoring, i.e., an exact adoption of human tutoring might neither be possible nor effective (Mavrikis & Gutierrez-Santos, 2009).

The third issue concerns the software design of technology to support argumentation. Too often analysis and feedback components are designed to be used only in a specific way within a single system. Sometimes we cannot even talk about separate components because the analysis and feedback components are tightly interwoven with code for other system functionality. We advocate a design for reuse, i.e., analysis and feedback functions should be encapsulated in separate modules, should provide well-defined interfaces and be easily extensible and configurable to accommodate for new or changing requirements. This enables not only reuse beyond systems and research projects, but is also a viable approach for an iterative, empirically-driven development process as described in the previous paragraph. For instance, different analysis modules could be easily deployed depending on whether corresponding feedback led to improved learning or not. Feedback strategies could be iteratively optimized by varying and testing corresponding configuration settings. Systems can effectively be both learning tools *and* experimentation platforms that allow comparing different adaptive support approaches. Such experimentation platforms can serve practical and theoretical purposes at the same time: They can bootstrap the development process and help in the acquisition of empirical data to improve our general understanding of computer-supported argumentation learning processes.

In the final paragraphs we discuss future challenges for researching and developing argumentation systems. As pointed out by Kuhn (2005), a dialogical setting might be beneficial for argumentation learning, especially for novices, because they are already familiar with this kind of argumentation from their everyday lives and do not have to anticipate an imaginary opponent. However, fruitful collaboration and discussion – whether computer-mediated or face-to-face – typically does not occur

spontaneously (Weinberger et al., 2006). The mission of the computer-supported collaborative learning (CSCL) community is therefore to research and design computer-based learning tools and contexts that stimulate fruitful forms of collaboration and discussion. The discussion systems in this chapter are in the tradition of CSCL because they provide adaptive support to encourage good collaboration and discussion practices. However, most of them do not support argumentation, in the narrow sense, in any specific way. Argumentation modeling systems, on the other hand, support, by definition, argumentation but are restricted to non-dialogical argumentation activities. Obviously, a combination of both approaches would be beneficial, that is, providing adaptive support for good discussion practices together with explicit support for argumentation. A first step in this direction is *AcademicTalk* (McAlister et al., 2004) with an admittedly simple model of preferred response types. More complex analyses of dialogical *and* argumentation aspects can be conducted by the *ARGUNAUT* system (McLaren et al., 2010). However, *ARGUNAUT* only provides machine models to *analyze* arguments (in support of a human moderator) and is lacking a pedagogical model of *intervention*. Open questions in this respect are (1) *whether* techniques from argumentation modeling systems could also be used in a dialogical context, and (2) *how* to use these analyses then to support argumentation (e.g., feedback strategies).

The second challenge is to research and develop new adaptive mechanisms to enhance current CSCL systems and practices. For instance, micro-scripting approaches (see *Introduction* section) aim at *scaffolding* argumentation processes. As pointed out by Pea (2004), the concept of scaffolding also entails, if understood in its original sense, an ongoing, *dynamic assessment* of student skills and a *fading of the scaffold* to continuously reduce the amount of assistance until students are able to accomplish the task on their own. However, current CSCL systems do not fade the scaffold, lacking thus an important component of the original concept. Stegmann and colleagues (2007, p. 444) therefore remark that using their static micro-scripting approach over longer durations might require "more flexible and dynamic support for learners with increasing self-regulation competencies." To implement such a "scaffolding-with-fading" (Pea, 2004, p. 438) approach a number of research questions must be tackled, including researching and developing fading strategies, computational assessment approaches and ultimately computational realizations of fading.

The third and final research challenge is to make argumentation activities more relevant, engaging and enjoyable to the student to increase motivation. For instance, it has been argued that positive learning results with *LARGO* could not be replicated in a second study due to a lack of motivation and engagement (Pinkwart et al., 2008b). There are a number of approaches to increase motivation, for instance, choosing topics that are interesting and relevant to the student (Kuhn, 2005, pp. 115-116), embedding argumentation in meaningful contexts (e.g., scientific inquiry as in *Belvedere* and *Rashi*), arguing together with peers (e.g., *ARGUNAUT* and *AcademicTalk*) and/or making argumentation more competitive and challenging in a debate scenario (e.g., Yuan et al., 2008). A more recent development holding promise to increase the motivation is the incorporation of game-like and narrative elements (McNamara, Jackson, & Graesser, 2009), for instance, virtual worlds and characters that show emotions and are sensitive to the students' emotions. An example tutoring system for argumentation that uses narrative elements is also presented in this book: *Policy World* (Easterday, 2010) embeds argumentation in the domain of policy deliberations into a narrative scenario. Obviously, new research challenges – many of

them connected to analysis and feedback – arise from the fusion of the "traditional" approaches, as covered in this review, with new game-like and narrative elements, for instance: How to imbue virtual characters with sufficient intelligence in order to be perceived as rational arguers? What is the role of affect in argumentation and can argumentation learning be improved when taking students' affective state into account? Can narrative / gaming elements get in the way of learning? Answering these questions is not only important to build more motivating and engaging argumentation systems. It also helps to build better educational games, which are populated with virtual characters capable of reasoning and arguing more rationally and human-like.

ACKNOWLEDGEMENTS

We want to thank the reviewers for their useful comments. This work is supported by the German Research Foundation (DFG) under the grant "Learning to Argue: Generalized Support Across Domains" (LASAD).

REFERENCES

Andriessen, J. (2006). Arguing to Learn. In K. Sawyer (Ed.), *Handbook of the Learning Sciences* (pp. 443–459). New York: Cambridge University Press.

Aleven, V, & Ashley, K. D. (1997). Teaching Case-Based Argumentation through a Model and Examples Empirical Evaluation of an Intelligent Learning Environment. In B. du Boulay, R. Mizoguchi (Eds.), *Proceedings of the 8th World Conf. on Artificial Intelligence in Education (AIED-97)* (pp. 87–94). Amsterdam: IOS.

Baeza-Yates, R., & Ribeiro-Neto, B. (1999). *Modern information retrieval*, Addison Wesley.

Baghaei, N., Mitrovic, A., & Irwin, W. (2007). Supporting collaborative learning and problem-solving in a constraint-based CSCL environment for UML class diagrams. *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *2*(2–3), 159–190.

Baker, M. (2003). Computer-Mediated Argumentative Interactions for the Co-elaboration of Scientific Notions. In J. Andriessen, M. Baker, D. Suthers (Eds.), *Arguing to Learn: Confronting Cognitions in Computer-Supported Collaborative Learning Environments (*pp. 47–78). Dordrecht: Kluwer Academic.

Bell, P. (1997). Using Argument Representations to Make Thinking Visible for Individuals and Groups. In R. Hall, N. Miyake, N. Enyedy (Eds.), *Proceedings of the 2nd Intl. Conf. on Computer-Supported Collaborative Learning (CSCL-97)* (pp. 10–19). Toronto: University of Toronto Press.

Buckingham Shum, S., MacLean, A., Bellotti, V. M. E., & Hammond, N. V. (1997). Graphical Argumentation and Design Cognition. *Human-Computer Interaction*, *12*(3), 267–300.

Bull, S., Brna, P., & Pain, H. (1995). Extending the scope of the student model. *User Modeling and User-Adapted Interaction*, *5*(1), 45–65.

Bull, S., & Kay, J. (2007). Student Models that Invite the Learner In: The SMILI Open Learner Modelling Framework. *Intl. Journal of Artificial Intelligence in Education (IJAIED), 17*, 89–120.

Chaudhuri, S., Kumar, R., Howley, I., & Rosé, C. P. (2009). Engaging Collaborative Learners with Helping Agents. In V. Dimitrova, R. Mizoguchi, B. du Bulay, A. Graesser (Eds.), *Proceedings of the 14th Intl. Conf. on Artificial Intelligence in Education (AIED 2009)* (pp. 365–372). Amsterdam: IOS Press.

Chesñevar, C., Maguitman, A., & Loui, R. (2000). Logical Models of Argument. *ACM Computing Surveys*, *32*(4), 337–383.

Chesñevar, C., McGinnis, J., Modgil, S., Rahwan, I., Reed, C., Simari, G., South, M., Vreeswijk, G., & Willmott, S. (2007). Towards an Argument Interchange Format. *Knowledge Engineering Review*, *21*(4), 293–316.

Chryssafidou, E. (2000). DIALECTIC: Enhancing essay writing skills with computer-supported formulation of argumentation. In *Proceedings of the ERCIMWG UI4ALL one-day joint workshop with i3 Spring Days 2000 on "Interactive Learning Environments for Children"*.

Cho, K., & Schunn, C. D. (2007). Scaffolded writing and rewriting in the discipline: A webbased reciprocal peer review system. *Computers & Education*, *48*(3), 409–426.

Corbett, A. T., Koedinger, K. R., & Anderson, J. R. (1997). *Intelligent Tutoring Systems*. In M. Helander, T. K. Landauer, P. Prabhu (Eds.), *Handbook of Human-Computer Interaction,* (pp. 849–874). Amsterdam: Elsevier.

De Groot, R. (2010). Teachers' use of the Argunaut system in the classroom. To appear in N. Pinkwart, B. M. McLaren (Eds.), *Educational Technologies for Teaching Argumentation Skills*. Bentham Science Publishers, in press.

Dillenbourg, P., & Hong, F. (2008). The mechanics of CSCL macro scripts. *Intl. Journal of Computer Supported Collaborative Learning (ijCSCL)*, *3*(1), 5–23.

Dragon, T., Woolf, B. P., Marshall, D., & Murray, T. (2006). Coaching Within a Domain Independent Inquiry Environment. In M. Ikeda, K. D. Ashley, T. W. Chan (Eds.), *Proceedings of the 8th Intl. Conf. on Intelligent Tutoring Systems (ITS-06)* (pp. 144–153). Berlin: Springer.

Dragon, T., & Woolf, B. (2006). Guidance and Collaboration Strategies in Ill-defined Domains. In V. Aleven, K. D. Ashley, C. Lynch, N. Pinkwart (Eds.), *Proceedings of the Workshop on Intelligent Tutoring Systems for Ill-Defined Domains at the 8th Intl. Conf. on Intelligent Tutoring Systems* (pp. 65–73).

Dragon, T., Woolf, B., & Murray, T. (2009). Intelligent Coaching for Collaboration in Ill-Defined Domains. In V. Dimitrova, R. Mizoguchi, B. du Bulay, A. Graesser (Eds.), *Proceedings of the 14th Intl. Conf. on Artificial Intelligence in Education (AIED 2009)* (pp. 740–742). Amsterdam: IOS.

Driver, R., Newton, P., & Osborne, J. (2000). Establishing the norms of scientific argumentation in classrooms. *Science Education*, *84*(3), 287–312.

Easterday, M. W., Aleven, V., & Scheines, R. (2007). Tis better to construct than to receive? The effects of diagram tools on causal reasoning In R. Luckin, K. R. Koedinger J. Greer (Eds.), *Proceedings of the 13th Intl. Conf. on Artificial Intelligence in Education (AIED-07)* (pp. 93–100). Amsterdam: IOS.

Easterday, M., Aleven, V., Scheines, R., & Carver, S. (2009). Will Google Destroy Western Democracy? Bias in Policy Problem Solving. In V. Dimitrova, R. Mizoguchi, B. du Bulay, A. Graesser (Eds.), *Proceedings of the 14th Intl. Conf. on Artificial Intelligence in Education (AIED 2009)* (pp. 249–256). Amsterdam: IOS.

Easterday, M. (2010). Policy World: A cognitive game for teaching deliberation. In N. Pinkwart, B. M. McLaren (Eds.), *Educational Technologies for Teaching Argumentation Skills*. Bentham Science Publishers, in press.

Feng, D., Kim, J., Shaw, E., & Hovy, E. (2006). Towards Modeling Threaded Discussions through Ontology-based Analysis. In *Proceedings of the 21st National Conf. on Artificial Intelligence (AAAI-2006)* (pp. 1289–1294).

Goguadze, G. (2009). Representation for interactive exercises. In J. Carette, L. Dixon, C. Coen, S. M. Watt (Eds.), *Proceedings of the 8th Intl. Conf. on Intelligent Computer Mathematics (CICM)* (pp. 294–309). Berlin: Springer.

Goldberg, D., Nichols, D., Oki, B. M., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, *35*(12), 61–70.

Goodman, B. A., Linton, F. N., Gaimari, R. D., Hitzeman, J. M., Ross, H. J., & Zarrella, G. (2005). Using Dialogue Features to Predict Trouble During Collaborative Learning. *User Modeling and User-Adapted Interaction*, *15*(1–2), 85–134.

Gordon, T. F., & Karacapilidis, N. (1997). The Zeno Argumentation Framework. In *Proceedings of the 6th Intl. Conf. on Artificial intelligence and law (ICAIL 1997)* (pp. 10–18). New York: ACM.

Gordon, T.F., Prakken, H., & Walton, D. (2007). The Carneades Model of Argument and Burden of Proof. *Artificial Intelligence*, *171*(10-15), pp. 875–896.

Harrer, A., Ziebarth, S., Giemza, A., & Hoppe, U. (2008). A framework to support monitoring and moderation of e-discussions with heterogeneous discussion tools. *Proceedings of the 8th IEEE Intl. Conf. on Advanced Learning Technologies (ICALT-08)* (pp. 41–45), IEEE.

Hoppe, H. U., Giemza, A., Wichmann, A., Krauß, M., Baurens, B., Rigolleau, B., Scheuer, O., McLaren, B. M., & Hever, R. (2008). *Combined deliverable D3.2b – moderator's interface and D4.2b – The off-line tracer. Part A (documentation)*. Argunaut project deliverable. Available online: http://www.argunaut.org/argunaut-d3-2b-d4-2b-PartA.pdf, last visited: 2009-10-29.

Israel, J., & Aiken, R. (2007). Supporting Collaborative Learning With An Intelligent Web-Based System. *Intl. Journal of Artificial Intelligence in Education*, *17*(1), 3–40.

Jeong, A. & Juong, S. (2007). Scaffolding Collaborative Argumentation in Asynchronous Discussions with Message Constraints and Message Labels. *Computers & Education*, *48*(3), 427–445.

Jordan, P., Hall, B., Ringenberg, M., & Rosé, C. Tools for Authoring a Dialogue Agent that Participates in Learning Studies. (2007). In R. Luckin, K. R. Koedinger, J. Greer (Eds.), *Proceedings of the 13th Intl. Conf. on Artificial Intelligence in Education (AIED 2007)* (pp. 43–50). Amsterdam: IOS.

Karacapilidis, N., & Papadias, D. (2001). Computer Supported Argumentation and Collaborative Decision Making: The HERMES system. *Information Systems*, *26*(4), 259–277.

Kim, J., Shaw, E., Ravi, S., Tavano, E., Arromratana, A., & Sarda, P. (2008). Scaffolding On-line Discussions with Past Discussions: An Analysis and Pilot Study of PedaBot. In B. Woolf, E. Aimeur E, R. Nkambou, S. Lajoie (Eds.), *Proceedings of the 9th Intl. Conf. on Intelligent Tutoring Systems Conf. (ITS-08)* (pp. 343–352). Berlin: Springer.

Kuhn, D. (1991). *The Skills of Argument*, New York: Cambridge University Press.

Kuhn, D. (2005). *Education for Thinking*. Cambridge, MA: Harvard University Press.

Kumar, R., Rosé, C., Wang, Y. C., Joshi, M., & Robinson, A. (2007). Tutorial Dialogue as Adaptive Collaborative Learning Support. In R. Luckin, K. R. Koedinger, J. Greer (Eds.), *Proceedings of the 13th Intl. Conf. on Artificial Intelligence in Education (AIED 2007)* (pp. 383–390). Amsterdam: IOS.

Linn, J. G., Segedy, J. R., Jeong, H., Podgursky, B., & Biswas, G. (2009). An Reconfigurable Architecture for Building Intelligent Learning Environments. In V. Dimitrova, R. Mizoguchi, B. du Bulay, A. Graesser (Eds.), *Proceedings of the 14th Intl. Conf. on Artificial Intelligence in Education (AIED 2009)* (pp. 115–122). Amsterdam: IOS.

Loll, F., & Pinkwart, N. (2009). Using collaborative filtering algorithms as eLearning tools. In R. H. Sprague (Ed.), *Proceedings of the 42nd Hawaii Intl. Conf. on System Sciences (HICSS 2009)* (pp. 1–10), IEEE Computer Soc. Press.

Loll, F., Pinkwart, N., Scheuer, O., & McLaren, B. M. (2010). Simplifying the Development of Argumentation Systems using a Configurable Platform. To appear in N. Pinkwart, B. M. McLaren (Eds.), *Educational Technologies for Teaching Argumentation Skills*. Bentham Science Publishers, in press.

Lund, K., Molinari, G., Séjourné, A., & Baker, M. (2007). How do argumentation diagrams compare when student pairs use them as a means for debate or as a tool for representing debate? *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *2*(2–3), 273–295.

Mavrikis, M., & Gutierrez-Santos, S. (2009). Informing the Design of Intelligent Support for ELE by Communication Capacity Tapering. In U. Cress, V. Dimitrova, M. Specht (Eds.), *Proceedings of the 4th European Conf. on Technology Enhanced Learning (EC-TEL 2009)* (pp. 556–571). Berlin: Springer.

McAlister, S., Ravenscroft, A., & Scanlon, E. (2004). Combining Interaction and Context Design to Support Collaborative Argumentation Using a Tool for Synchronous CMC. *Journal of Computer Assisted Learning*, *20*(3), 194–204.

McLaren, B. M., Scheuer, O., & Mikšátko, J. (2010). Supporting Collaborative Learning and e-Discussions Using Artificial Intelligence Techniques. *Intl. Journal of Artificial Intelligence in Education*, in press.

McManus, M. M., & Aiken, R. M. (1995). Monitoring computer based collaborative problem solving. *Intl. Journal of Artificial Intelligence in Education*, *6*(4), 308–336.

McNamara, D. S., Jackson, G. T., & Graesser, A. (2009). Intelligent Tutoring and Games (ITaG). In H. C. Lane, A. Ogan, V. Shute (Eds.), *Proceedings of the Workshop on Intelligent Educational Games at the 14th Annual Conf. on Artificial Intelligence in Education (AI-ED)* (pp. 1–10).

Mitrovic, A., Mayo, M., Suraweera, P., & Martin, B. (2001). Constraint-based tutors: a success story. In L. Monostori, J. Váncza, M. Ali (Eds.), *Proceedings of the 14th Intl. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems (IEA/AIE-2001)* (pp. 931–940). Berlin: Springer.

Muller Mirza, N., Tartas, V., Perret-Clermont, A. N., & de Pietro, J. F. (2007). Using graphical tools in a phased activity for enhancing dialogical skills: An example with Digalo. *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *2*(2–3), 247–272.

Munneke, L., Van Amelsvoort, M., & Andriessen, J. (2003). The role of diagrams in collaborative argumentation-based learning. *Intl. Journal of Educational Research*, *39*(1–2), 113–131.

Munneke, L., Andriessen, J., Kanselaar, G., & Kirschner, P. (2007). Supporting interactive argumentation: Influence of representational tools on discussing a wicked problem. *Computers in Human Behavior*, *23*(3), 1072–1088.

Murray, T., Woolf, B., & Marshall, D. (2004). Lessons Learned from Authoring for Inquiry Learning: A Tale of Authoring Tool Evolution. In J. C. Lester, R. M. Vicari, F. Paraguaçu (Eds.), *Proceedings of the 7th Intl. Conf. on Intelligent Tutoring Systems (ITS 2004)* (pp. 197–206). Berlin: Springer.

Nussbaum, M. E., Winsor, D. L., Aqui, Y. M., & Poliquin, A. M. (2007). Putting the pieces together: Online argumentation vee diagrams enhance thinking during discussions. *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *2*(4), 479–500.

Pasquier, P., Rahwan, F., Dignum, F., & Sonenberg, L. (2006). Argumentation and Persuasion in the Cognitive Coherence Theory. In P. Dunne, T. Bench-Capon (Eds.), *Proceedings of the 1st Intl. Coference on Computational Models of Argument (COMMA 2006)* (pp. 223–234). Amsterdam: IOS.

Pea, R. D. (2004). The Social and Technological Dimensions of Scaffolding and Related Theoretical Concepts for Learning, Education, and Human Activity. *The Journal of the Learning Sciences*, *13*(3), 423–451.

Pinkwart, N., Aleven, V., Ashley, K., & Lynch, C. (2006a). Toward Legal Argument Instruction with Graph Grammars and Collaborative Filtering Techniques. In M. Ikeda, K. Ashley, T. W. Chan (Eds.), *Proceedings of the 8th Intl. Conf. on Intelligent Tutoring Systems (ITS-06)* (pp. 227–236). Berlin: Springer.

Pinkwart, N., Aleven, V., Ashley, K., & Lynch, C. (2006b). Schwachstellenermittlung und Rückmeldungsprinzipen in einem intelligenten Tutorensystem für juristische Argumentation. In M. Mühlhäuser, G. Rößling, R. Steinmetz (Eds.), *GI Lecture Notes in Informatics - Tagungsband der 4. e-Learning Fachtagung Informatik* (pp. 75–86).

Pinkwart, N., Ashley, K., Lynch, C., & Aleven, V. (2008a). Graph Grammars: An ITS Technology for Diagram Representations. In D. Wilson, H. C. Lane (Eds.), *Proceedings of 21st Intl. FLAIRS Conf. (FLAIRS-21)* (pp. 433–438).

Pinkwart, N., Lynch, C., Ashley, K., & Aleven, V. (2008b). Re-evaluating LARGO in the Classroom: Are Diagrams Better than Text for Teaching Argumentation Skills? In B. Woolf, E. Aimeur, R. Nkambou, S. Lajoie (Eds.), *Proceedings of the 9th Intl. Conf. on Intelligent Tutoring Systems (ITS-08)* (pp. 90–100). Berlin: Springer.

Poison, M. C., & Richardson, J. J. (1988). *Foundations of Intelligent Tutoring System*, Hillsdale, NJ: Erlbaum.

Rabiner, L. R. (1989). A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, *77*(2), 257–286.

Rahwan, I., & Banihashemi, B. (2008). Arguments in OWL: A Progress Report. In A. Hunter (Ed.), *Proceedings of the 2nd Intl. Conf. on Computational Models of Argument (COMMA-2008)* (pp. 297–310). Amsterdam: IOS.

Ranney, M., & Schank, P. (1998). Toward an integration of the social and the scientific: Observing, modeling, and promoting the explanatory coherence of reasoning. In S. Read, L. Miller (Eds.), *Connectionist models of social reasoning and social behavior* (pp. 245–274). Mahwah: Erlbaum.

Ravenscroft, A., Sagar, M., Baur, E., & Oriogun, P. (2008). Ambient pedagogies, meaningful learning and social software. In S. Hatzipanagos, S. Warburton (Eds.), *Social Software and Developing Community Ontologies* (pp. 432–450). Hershey: IGI Global.

Ravi, S., & Kim, J. (2007). Profiling Student Interactions in Threaded Discussions with Speech Act Classifiers. In R. Luckin, K. R. Koedinger, J. Greer (Eds.), *Proceedings of the 13th Intl. Conf.*

on Artificial Intelligence in Education (AIED-07) (pp. 357–364). Amsterdam: IOS.

Reed, C., & Rowe, G. (2004). Araucaria: Software for Argument Analysis, Diagramming and Representation. *Intl. Journal of AI Tools*, *14*(3–4), 961–980.

Rittel, H., & Webber, M. (1973). Dilemmas in a General Theory of Planning. *Policy Sciences*, *4*, 155–169.

Roschelle, J. (1992). Learning by Collaborating: Convergent Conceptual Change. *Journal of the Learning Sciences*, *2*(3) 235–276.

Rosé, C., Wang, Y. C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., & Fischer, F. (2008). Analyzing Collaborative Learning Processes Automatically: Exploiting the Advances of Computational Linguistics in Computer-Supported Collaborative Learning Intl.. *Journal of Computer-Supported Collaborative Learning (ijCSCL*), 3(3), 237–271.

Schank, P. (1995). *Computational Tools for Modeling and Aiding Reasoning: Assessing and Applying the Theory of Explanatory Coherence*. Dissertation, University of California, Berkeley.

Schellens, T., Van Keer, H., De Wever, B., & Valcke, M. (2007). Scripting by assigning roles: Does it improve knowledge construction in asynchronous discussion groups? *Intl.. Journal of Computer Supported Collaborative Learning (ijcscl)*, *2*(2–3), 225–246.

Scheuer, O., McLaren, B. M., Loll, F., & Pinkwart, N. (2009). An Analysis and Feedback Infrastructure for Argumentation Learning Systems. In V. Dimitrova, R. Mizoguchi, B. du Bulay, A. Graesser (Eds.), *Proceedings of the 14th Intl. Conf. on Artificial Intelligence in Education (AIED 2009)* (pp. 629–631). Amsterdam: IOS.

Scheuer, O., Loll, F., Pinkwart, N., & McLaren, B. M. (2010). Computer-Supported Argumentation: A Review of the State of the Art. *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *5*(1), 43–102.

Schneider, D. C., Voigt, C., & Betz, G. (2007). Argunet - A Software Tool for Collaborative Argumentation Analysis and Research. In R. Kibble, C. Reed, F. (Eds.), *Working Notes of the 7th Workshop on Computational Models of Natural Argument (CMNA VII)*.

Schwarz, B., & Glassner, A. (2007). The Role of Floor Control and of Ontology in Argumentative Activities with Discussion-Based Tools. *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *2*(4), 449–478.

Sebastiani, F. (2002). Machine learning in automated text categorization. *ACM Computing Surveys,34*(1), 1–47.

Shute, V. J. (2008). Focus on Formative Feedback. *Review of Educational Research, 78*(1), 153–189.

Soller, A., Goodman, B., Linton, F., & Gaimari, R. (1998). Promoting Effective Peer Interaction in an Intelligent Collaborative Learning System. In B. P. Goettl, H. M. Halff, C. L. Redfield, V. J. Shute (Eds.), *Proceedings of the Intl. Conf. on Intelligent Tutoring Systems (ITS-98)* (pp. 186–195). Berlin: Springer.

Soller, A. (2001). Supporting Social Interaction in an Intelligent Collaborative Learning System. *Intl. Journal of Artificial Intelligence in Education*, *12*, 40–62.

Soller, A. (2004). Computational Modeling and Analysis of Knowledge Sharing in Collaborative Distance Learning. *User Modeling and User-Adapted Interaction: The Journal of Personalization Research*, *14*(4), 351–381.

Soller, A., Monés, A. M., Jermann, P., & Mühlenbrock, M. (2005). From Mirroring to Guiding: A Review of State of the Art Technology for Supporting Collaborative Learning. *Intl. Journal on Artificial Intelligence in Education*, *15*(5), 261–290.

Stark, R., & Krause, U. M. (2006). Konzeption einer computerbasierten Lernumgebung zur Förderung von Kompetenzen zum wissenschaftlichen Argumentieren. In G. Krampen, H. Zayer (Eds.), *Didaktik und Evaluation in der Psychologie* (pp. 218–230). Göttingen: Hogrefe.

Stegmann, K., Weinberger, A., & Fischer, F. (2007). Facilitating argumentative knowledge construction with computer-supported collaboration scripts. *Intl. Journal of Computer-Supported Collaborative Learning (ijCSCL)*, *2*(4), 421–447.

Suraweera, P., & Mitrovic, A. (2004). An Intelligent Tutoring System for Entity Relationship Modelling. *Intl. Journal of Artificial Intelligence in Education (ijAIED)*, *14*(3–4), 375–417.

Suthers, D., & Jones, D. (1997). An Architecture for Intelligent Collaborative Educational Systems. In B. du Boulay, R. Mizoguchi (Eds.), *Proceedings of the 8th World Conf. on Artificial Intelligence in Education (AIED-97)* (pp. 55–62). Amsterdam: IOS Press.

Suthers, D., Connelly, J., Lesgold, A., Paolucci, M., Toth, E., Toth, J, & Weiner, A. (2001). Representational and Advisory Guidance for Students Learning Scientific Inquiry. In K. D. Forbus, P. J. Feltovich (Eds.), *Smart machines in education: The coming revolution in educational technology* (pp. 7–35). Menlo Park: AAAI/MIT Press.

Suthers, D. (2003). Representational Guidance for Collaborative Inquiry. In J. Andriessen, M. Baker, D. Suthers (Eds), *Arguing to Learn: Confronting Cognitions in Computer-Supported Collaborative Learning Environments* (pp. 27–46). Dordrecht: Kluwer Academic.

Suthers, D., & Hundhausen, C. (2003). An Experimental Study of the Effects of Representational Guidance on Collaborative Learning Processes. *Journal of the Learning Sciences*, *12*(2), 183–219.

Suthers, D. D., Vatrapu, R., Medina, R., Joseph, S., & Dwyer, N. (2008). Beyond Threaded Discussion: Representational Guidance in Asynchronous Collaborative Learning Environments. *Computers & Education*, *50*(4), 1103–1127.

Thagard, P., & Verbeurgt, K. (1998). Coherence as constraint satisfaction. *Cognitive Science*, *22*, 1–24.

Thagard, P. (2006). Evaluating Explanations in Law, Science, and Everyday Life. *Current Directions in Psychological Science*, *15*(3), 141–145.

Toulmin, S. E. (1958). *The Uses of Argument*. New York: Cambridge University Press.

Tsovaltzi, D., Rummel, N., McLaren, B. M., Pinkwart, N., Scheuer, O., Harrer, A., & Braun, I. (2010). Extending a Virtual Chemistry Laboratory with a Collaboration Script to Promote Conceptual Learning. *Intl. Journal of Technology Enhanced Learning (IJTEL)*, *2*(1–2), 91–110.

Van Eemeren, F. H., & Grootendorst, R. (2004). *A Systematic Theory of Argumentation: the Pragma-Dialectical Approach*, Cambridge: Cambridge University Press.

Van Gelder, T. (2003). Enhancing Deliberation Through Computer-Supported Argument Visualization. In P. A. Kirschner, S. Buckingham Shum, C. Carr (Eds.), *Visualizing Argumentation: Software Tools for Collaborative and Educational Sense-Making (pp. 97–116)*. London: Springer.

VanLehn, K. (2006). The behavior of tutoring systems. *Intl. Journal of Artificial Intelligence in Education,16*(3), 227–265.

Verbree, A., Rienks, R., & Heylen, D. (2006). First Steps Towards the Automatic Construction of Argument-Diagrams from Real Discussions. In P. Dunne, T. Bench-Capon (Eds.), *Proceedings of the in 1st Intl. Conf. on Computational Models of Argument (COMMA 2006)* (pp. 183–194). Amsterdam: IOS.

Verheij, B. (2003). Artificial argument assistants for defeasible argumentation. *Artificial Intelligence*, *150*(1–2), 291–324.

Vreeswijk, G. A. W. (2005). Argumentation in Bayesian Belief Networks. In I. Rahwan, P. Moraitis, C. Reed (Eds.), *Argumentation in Multi-Agent Systems (ArgMAS 2004)* (pp. 111–129). Berlin: Springer.

Walton, D. (2008). Computational dialectics. In D. Walton, *Witness Testimony Evidence: Argumentation, Artificial Intelligence and the Law*. (pp. 151–193). Cambridge: Cambridge University Press.

Weinberger, A., & Fischer, F. (2006). A Framework to Analyze Argumentative Knowledge Construction in Computer-Supported Collaborative Learning. *Computers & Education*, *46*(1), 71–95.

Weinberger, A., Stegmann, K., Fischer, F., & Mandl, H. (2006). Scripting argumentative knowledge construction in computer-supported learning environments. In F. Fischer, I. Kollar, H. Mandl, J. M. Haake (Eds.), *Scripting Computer-Supported Collaborative Learning Cognitive, Computational and Educational Perspectives* (pp. 191–211). New York: Springer.

Witten, I. H., & Frank, E. (2005). *Data Mining: Practical machine learning tools and techniques*, 2nd ed., San Francisco: Morgan Kaufmann.

Woolf, B., Murray, T., Marshall, D., Dragon, T., Kohler, K., Mattingly, M., Bruno, M., Murray, D., & Sammons, J. (2005). Critical Thinking Environments for Science Education. In C. K. Looi, G. McCalla, B. Bredeweg, J. Breuker (Eds.), *Proceedings of the 12th Intl. Conf. on AI and Education* (pp. 702–709). Amsterdam: IOS.

Yuan, T., Moore, D., & Grierson, A. (2008). A Human-Computer Dialogue System for Educational Debate: A Computational Dialectics Approach. *Intl. Journal of Artificial Intelligence in Education (ijAIED)*, *18*(1), 3–26.