Coordinating Multi-dimensional Support in Collaborative Conversational Agents

David Adamson and Carolyn P. Rosé

Language Technologies Institute
Carnegie Mellon University
dadamson|cprose@cs.cmu.edu

Abstract. The field of computer supported collaborative learning has evolved an ontology of types of support for group learning. In recent years, conversational agents have been used successfully to realize forms of dynamic micro and macro level script based support for group learning. However, using existing architectures for managing the coordination of these agent-based behaviors (which can vary widely in scope, timing, and constraints), infelicitous "collisions" of behaviors have been observed. In this paper, we introduce a new architecture that facilitates the development, coordination, and co-performance of multiple agent-based support behaviors.

Keywords: collaborative learning, intelligent agents, multi-party conversational agents, conversational scripting, dynamic support

1 Introduction

This paper describes a new architecture for intelligent support of collaborative learning, motivated by recent work in dynamic scripting. A *script* in CSCL is a method for structuring collaboration [1]. A script can provide structure at a macro-level, or it can scaffold a participant's contributions at a micro-level. Such scripts can be implemented statically, providing the same support in all cases, or dynamically, responding to the students and their context to deliver an appropriate level of support at opportune times.

The *Basilica* agent architecture [5] pioneered dynamic collaborative support alongside traditional static macro- and micro-scripts. Agents were defined as a collection of modular components, any of which could influence the agents' user-facing behavior. Despite its design innovations, tutors built with Basilica tended to be cumbersome to develop and fragile in deployment.

The contribution of this paper is an illustration of the design space of multi-dimensional support for collaborative learning as enabled through *Bazaar*, a successor architecture to Basilica, designed to simplify the coordination of multiple dynamic supportive behaviors. In Section 3, we describe Bazaar in detail. In particular, Section 3.2 describes a feature of this architecture that allows for the graceful resolution of conflict between proposed system actions. Finally, Section 4 showcases a number of agents that were developed with this architecture, and locates them within the space of multi-dimensional support.

2 Collaborative Scripting and Support

A script can describe any of a wide range of features of collaborative activities, including task, timing, roles, and the patterns of interaction between the participants. A number of models have been proposed to aid the design and analysis of collaborative scripts [3] [4] [9]. Scripts can been classified as either macro-scripts or micro-scripts [2]. Macro-scripts are pedagogical models that describe coarse-grained features of a collaborative setting, such as the sequence and structure of an activity. Micro-scripts, in contrast, are models of dialogue and argumentation that are embedded in the environment, and are intended to be adopted and progressively internalized by the participants. Examples of macro-scripts include the classic Jigsaw activity, as well as specialized scripts like ArgueGraph and ConceptGrid [3]. Micro-scripting can be implemented by offering prompts or hints to the user to guide their contributions [8], which may depend on the current phase of the macro-script.

Early approaches to scripting have been static, offering the same script or supports for every group in every context. Such non-adaptive approaches can lead to over scripting [1], or to the interference between different types of scripts [10]. A more dynamic approach that triggered micro-scripted supports or the appropriate phases of macro-scripts in response to the automatic analysis of participant activity [7] would be preferable. Such analysis could can at a macro-discourse level, following the state of the activity as a whole, or it can be based on isolated user events. Such dynamic awareness might allow minimal scripting to be used to greater effect, with greater hopes of the users internalizing the support's intended interaction patterns. Further, the benefits of fading the support over time [8] could be more fully realized, as the timing and degree of such fading could be dynamically tuned to the group's level of internalization. The collaborative tutoring agents described by Kumar [5] were among the first to implement dynamic scripting in a CSCL environment.

Student	1:03	I think it has to do with the flow through the membrane.
Tutor	1:05	That's interesting, Student - can you say more about permeability?
Tutor	1:06	Let's move on to the next problem.
Student	1:09	What about my answer? :-(

Table 1. Sample of Agent Self-Collision

2.1 Coordinated Multi-dimensional Support

Participants in a collaborative session aren't just completing the assigned task. They're involved in numerous simultaneous processes including social bonding, idea formation, argumentation, and time management. To allow for rich, holistic interactive support, a tutor must be able to express several differently-scoped behaviors concurrently - it can be considered to be working through several

overlapping macro- and micro-scripts at once. However, the tutor has to remain effective while doing so. As illustrated in Table 2, a tutor managing several scripts at once can "step on its own toes". When multiple responses from the tutor interfere with, or interrupt each other, the students' belief in the tutor's competence can be shattered. Although several approaches have been described to address some of these concerns [5], it remains an actively-pursued grail [6].

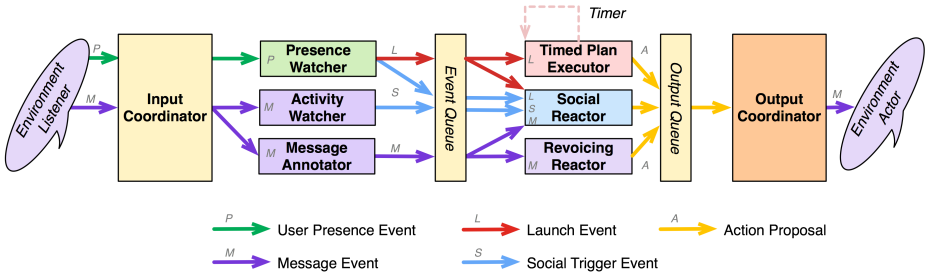

Fig. 1. A sample Bazaar configuration.

3 The Bazaar Architecture

The *Bazaar* architecture builds upon Basilica [5], a modular framework for designing multi-party collaborative agents. Both are event-driven systems where independent components receive and respond to user-, environment-, and system-generated actions, and present the unified output of these components to the user. We adapt the Basilica architecture to accommodate competing sources of agent behavior, to streamline agent development. Both architectures are able to interact with the same varied set of collaborative environments, which include text chatrooms and shared whiteboards, as well as more novel environments like the virtual world of SecondLife.

3.1 Events and Components

An Bazaar *Event*, like its Basilica counterpart, is an object representing something interesting that has happened in the world of the agent. An Event might represent an incoming student message or a user entering a chat room. Events can also result from the analysis of other events, or changes in system state. Events such as these are used to launch phases of macro-scripts, or to dynamically initiate suitable support behavior. A *Component* is a modular representation of related behavior and state-knowledge, and often corresponds to a single method of scripting or support. Basilica components were arranged in an agent-specific graph of relationships, frequently defining a custom event for each inter-component connection. This led to an undesirable degree of coupling, especially

among components that sought to mediate or suppress the behavior of their neighbors. Bazaar replaces the web of components with a two-step event flow, dividing component responsibility between *Preprocessor* and *Reactor* interfaces. When a new event is received by the system, all Preprocessors registered for the event's type are given the opportunity to respond to it, either generating new events (perhaps to indicate a shift in the conversation's focus) or modifying the original (like adding a conceptual annotation to a user message). All preprocessed events are subsequently delivered to the Reactors registered for the resulting event types. Reactors have the opportunity to respond to events (and thus dynamically enact sub-scripts or supports) by proposing actions to the *Output Coordinator*. Figure 1 illustrates a typical Bazaar configuration.

3.2 Output Coordinator: Prioritizing Proposed Actions

Proposals for agent action are queued in the Output Coordinator with a time-window of relevance and an initial priority value assigned by the originating Reactor. The Output Coordinator will periodically re-evaluate the priority of each remaining proposal, rejecting those that are no longer relevant and accepting and enacting the one with the highest priority.

As a solution to the multi-source management problem described in Section 2.1, we employ a generalization of the "concurrent mode" approach described by Lison [6]. A previously-accepted agent action can leave a lingering presence in the Output Coordinator, a *Proposal Source*, which can re-prioritize (or entirely suppress) incoming proposals until its influence expires. Each action proposal is constructed with a timeout-window after which it is no longer relevant - if a queued proposal has not been accepted when its timeout expires, it's removed from the queue. When a message is accepted or rejected, a callback-method is invoked, allowing the originating Component to update its state accordingly.

4 Case Studies in Multi-dimensional Support

The tutors in the following case studies highlight the capabilities of the Bazaar architecture, notably the coordination of multiple sources of behavior and support. Table 4 illustrates their dimensions (macro or micro, static or dynamic, as discussed in Section 2) along which each system offered scripting or support. The first two were developed in-house, and have been used in recent studies. The third is one of a set of conversational agents developed by small teams of undergraduate students as part of a two-week CSCL workshop at IIIT Delhi.

4.1 Dynamic Feedback: Revoicing in Chemistry and Biology

How does tutor revoicing affect the quality of student explanations? In a college chemistry lesson on intermolecular forces, we deployed an agent that matched student input against a list of target concepts, and offered the matched concept as a rephrasing of their contribution. In addition, the tutor followed a

Support	Revoicing	CycleTalk	Devil & Guardian
Static Macro	X	X	
Dynamic Macro	X		X
Dynamic Micro	X	X	X

Table 2. Dimensions of Support in Bazaar Agents

macro-script to deliver the problem sets and background material that framed the discussions, and also employed dynamic macro-level social strategies as first implemented by Kumar [5]. The revoicing and social behaviors operated in tandem - higher-priority revoicing responses softly blocked any social prompts that were triggered until several seconds after the revoicing move had completed. The macro-script's timing was similarly softened - where previous Basilica tutors would drop everything and interrupt themselves for a macro-level timeout, in this tutor a prompt for the next macro-phase would be delayed long enough for the current move-sequence to play out. The same arrangement of behavioral components has since been re-deployed in a high-school biology domain [**?**]. Only the lesson's macro-script and the targeted-concept list for the revoicing behavior had to be modified. This study, showed a significant effect from the dynamic revoicing behavior on the quality of student discussion and explanation.

4.2 Multiple Agent Scripts: CycleTalk

How can we manipulate the self-efficacy of group members? This Bazaar tutor employed two chat-room user presences to present both an authoritative and non-authoritative face to the human users. The "Doctor Bob" presence delivered the lesson content, while additional questions were posed to the targeted student in each group by "Jimmy", portrayed as a clueless student. Results from this study indicate that this sort of targeting may be detrimental to students groups with low-self-efficacy.

4.3 Dynamic Macro-Scripting: Devil and Guardian

Will a balanced debate lead to greater mutual understanding? Devil and Guardian employed a topic-classification model to classify the recent history of a conversation (as a rolling window over past participant turns) by topic and by "side" (i.e., a Gun Control discussion dominated by Conservatives), and used this classification to select and insert talking-points from the opposing side on the same topic. In addition, the rate of per-user contributions was monitored, triggering events to encourage participation by the less vocal user.

5 Conclusions and Future Work

Bazaar is a powerful tool for facilitating research in collaborative learning. Its flexibility and simplicity mean it can be used to very rapidly develop platforms

for investigating a wide range of important questions within the design space of dynamic support for collaborative learning. We have developed a number of such research platforms, and actively employ them in our learning studies. As we continue to do so, we expect to discover ways in which the Bazaar architecture can be extended and refined. We look forward to sharing Bazaar with other researchers exploring dynamic supports for collaboration, and to continue to improve the architecture and make it accessible to this target audience.

Acknowledgments. Thanks, everyone!

References

[1] Dillenbourg, P.: Over-scripting CSCL : The risks of blending collaborative learning with instructional design . Three worlds of CSCL Can we support CSCL pp. 61–91 (2002)
[2] Dillenbourg, P., Hong, F.: The mechanics of CSCL macro scripts. The International Journal of Computer-Supported Collaborative Learning 3(1), 5–23 (2008)
[3] Kobbe, L., Weinberger, A., Dillenbourg, P., Harrer, A., Hämäläinen, R., Häkkinen, P., Fischer, F.: Specifying computer-supported collaboration scripts. The International Journal of Computer-Supported Collaborative Learning 2(2-3), 211–224 (2007)
[4] Kollar, I., Fischer, F., Hesse, F.W.: Collaborative scripts - a conceptual analysis. Educational Psychology Review 18(2), 159–185 (2006)
[5] Kumar, R., Rosé, C.P.: Architecture for Building Conversational Agents that Support Collaborative Learning. IEEE Transactions on Learning Technologies 4(1), 1–1 (2011)
[6] Lison, P.: Multi-Policy Dialogue Management. In: Proceedings of the SIGDIAL 2011 Conference. pp. 294–300. Association for Computational Linguistics (2011)
[7] Rosé, C., Wang, Y.C., Cui, Y., Arguello, J., Stegmann, K., Weinberger, A., Fischer, F.: Analyzing collaborative learning processes automatically: Exploiting the advances of computational linguistics in computer-supported collaborative learning. The International Journal of Computer-Supported Collaborative Learning 3(3), 237–271 (2008)
[8] Wecker, C., Fischer, F.: Fading scripts in computer-supported collaborative learning: The role of distributed monitoring. Proceedings of the 8th iternational conference pp. 764–772 (2007)
[9] Weinberger, A., Fischer, F.: A framework to analyze argumentative knowledge construction in computer-supported collaborative learning. Computers & Education 46(1), 71–95 (2006)
[10] Weinberger, A., Stegmann, K., Fischer, F., Mandl, H.: Scripting argumentative knowledge construction in computer-supported learning environments. Environments 6(6), 191–211 (2007)