An Integrated Authoring Tool for Tactical Questioning Dialogue Systems

Sudeep Gandhe and **Nicolle Whitman** and **David Traum** and **Ron Artstein**
Institute for Creative Technologies
13274 Fiji way, Marina del Rey, CA 90292.
<*lastname*>@ict.usc.edu

Abstract

We present an integrated authoring tool for rapid prototyping of dialogue systems for virtual humans taking part in tactical questioning simulations. The tool helps domain experts, who may have little or no knowledge of linguistics or computer science, to build virtual characters that can play the role of the interviewee. Working in a top-down fashion, the authoring process begins with specifying a domain of knowledge for the character; the authoring tool generates all relevant dialogue acts and allows authors to assign the language that will be used to refer to the domain elements. The authoring tool can also be used to manipulate some aspects of the dialogue strategies employed by the virtual characters, and it also supports re-using some of the authored content across different characters.

Introduction

Tactical Questioning dialogues are those in which small-unit military personnel, usually on patrol, hold conversations with individuals to produce information of military value (Army 2006). Building Tactical Questioning characters that can play the role of a person being questioned has been an on-going project at Institute for Createive Technologies. The simulation training environment can be used to train military personnel in how to conduct such dialogues. The project has evolved through many different architectures for dialogue systems (Traum et al. 2008). Gandhe et al. (2008) provide description of the latest architecture for this tactical questioning dialogue system.

These tactical questioning dialogues are different from typical question answering dialogues in that they can be non-cooperative at times. The character may answer some of the questions by the interviewer in a cooperative manner but some other questions which are of a more sensitive nature may need more coercion from the interviewer. Some of the strategies used by interviewers include building rapport with the character, addressing their concerns, promising to do certain actions in their favor or pointing out the effects of non-cooperation.

Traditionally, one step in the development life cycle for a dialogue system is to build a corpus of in-domain human-human dialogues through roleplays or Wizard of Oz sessions; this is the starting point for specifying the domain. Corpus collection can be costly and time-consuming. If the domain of interaction is relatively simple and can be authored *consistently* and *completely* by a scenario designer, the collection of dialogue corpora can be bypassed. Here *consistency* refers to generating only the valid dialogue acts that can be correctly handled by the dialogue manager and *completeness* refers to generating all dialogue acts that are relevant with respect to the character's domain knowledge and associating all of these to corresponding surface text.

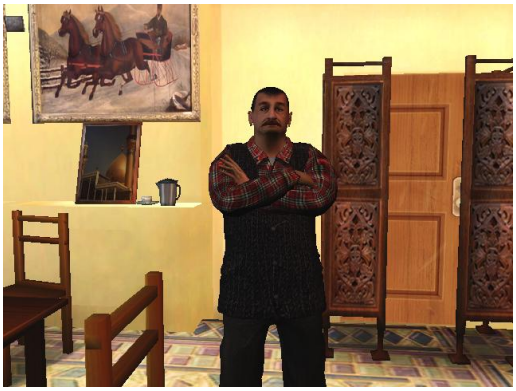


Figure 1: Hassan – A virtual human for Tactical Questioning

We have implemented a character named Hassan (see Figure 1), who is being questioned about illegal tax collections at a local marketplace. We will use this domain for most of the examples in this paper.

In the next section, we review some of the existing authoring tools for building dialogue systems. We then list the required features an authoring tool should provide and explain our design decisions for the tool. Next, we describe the tool starting with how the domain knowledge is specified. The subsequent section explains how dialogue acts are automatically generated based on the domain knowledge and how the dialogue manager functions at the dialogue act level. It is followed by the discussion of surface text authoring and how authored content can be re-used across multiple characters. We present a preliminary evaluation of the tool and conclude by discussing avenues for future improvements.

Copyright © 2009, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

Related Work

Many toolkits and authoring environments have been developed for building dialogue systems. Rapid Application Developer from CSLU toolkit (Sutton et al. 1998) allowed designers to build dialogue systems employing finite state dialogue models. The authoring environment was accessible by non-experts and allowed building systems that could conduct simple directed dialogues. Our tactical questioning dialogue system is mainly reactive but allows for some initiative for simple negotiations. It can be cast into finite state models augmented with information state. Since our virtual human system engages the trainee in a natural conversation, the input from the user is free-form and is more challenging for the NLU.

There have been several commercial dialogue building solutions based on VoiceXML, which allows for a form-based dialogue management. RavenClaw (Bohus and Rudnicky 2003) is another dialogue architecture where designers can specify hierarchical domain task specification. The dialogue management builds on top of agenda based dialogue management technique (Xu and Rudnicky 2000). Although this architecture has been successfully used for building multiple dialogue systems, it is most suited for task-oriented dialogue systems and using it requires considerable expertise in programming and design of dialogue systems. Other dialogue system architectures such as TrindiKit (Larsson et al. 2004) or Midiki (MITRE 2005), which use information state based dialogue modeling (Traum and Larsson 2003) have the same issue. These systems require considerable knowledge of the dialogue theories and software development.

There have been some efforts in the area of tutorial dialogue systems that concentrate on building authoring tools which can be used by non-experts for rapidly building a dialogue system. TuTalk (Jordan et al. 2007) is one such system. TuTalk authoring tool allows tutorial system researchers who may not have expertise in the dialogue system design to rapidly prototype dialogue systems and experiment with different ideas. Our tactical questioning project has a similar requirement. The TuTalk authoring tool allows authoring of initiation-response pairs along with many features suitable for tutorial dialogue systems.

Our initial efforts in providing authoring tools for tactical questioning were along the same lines. Designers were allowed to author questions and the corresponding answers (Leuski et al. 2006). Although this works very well for simple question answering systems, it suffers from the inability to maintain coherence over sequences of utterances greater in length than two. We need this ability to engage in simple negotiation dialogues. We follow an approach similar to TuTalk in designing an authoring tool that is specialized for a specific genre of dialogue viz. tactical questioning and allows authoring by non-experts.

Requirements

One of the requirements for the tactical questioning project is to allow subject matter experts to rapidly build different scenarios within the same tactical questioning framework. Moreover, these authors should not require any expertise in linguistics or computer science. For these reasons, we designed a simple schema for specifying the domain knowledge which is easily understandable. The authoring process starts with the domain knowledge construction which is done with the help of our authoring tool (see Figure 2). The authoring tool automatically constructs all relevant dialogue acts that are used by the dialogue manager. The tool also allows direct linking of these acts to surface text of the utterances for training NLU and NLG.

Although Tactical Questioning dialogues are mainly question-answering dialogues, we need the ability to model simple negotiations over when to release certain sensitive information. The dialogue manager maintains a model of emotions and compliance which are updated as the dialogue proceeds. In compliant mode, the character may elicit certain offers from the interviewer before answering questions regarding the sensitive information. Whereas in adversarial mode, the character may choose to lie in response to these questions. We need to allow the scenario authors to mark certain information elements as sensitive and modify some of the policies regarding when to release this information.

There are cases where we would like to build several characters that can be questioned about the same incident. E.g. Multiple witnesses of a shooting incident at the marketplace will have a considerable overlap in their domain knowledge. One of the requirements for the authoring tool is the ability to re-use the existing authored content across different characters. Our tool allows for such re-use of the domain knowledge along with all the dialogue acts and the language associated with it.

Figure 2 shows a screenshot of our authoring tool. It has three horizontal panels. The topmost panel is used for editing the domain knowledge level. The middle one allows authors to view all dialogue acts and select one of them. The bottom panel allows editing of the surface text corresponding to the chosen dialogue act.

Domain Knowledge Level

Domain knowledge is created as a four level hierarchy. The highest level is the *characters*, the conversational participants in the domain, who can be speakers and addressees of utterances and dialogue acts. In the Hassan domain there are two characters viz. the trainee (called player) and Hassan. Each character knows about a set of *objects*. These objects can be of different types such as person (imam), location (market) or abstract concept (tax). Each object can be further described by *attributes*. Finally, attributes can take on *values*, some of which can be marked false – to be used as lies. A basic proposition is a triple <object,attribute,value>. Queries for the value field of a such propositions form the basis for questions. Objects of type person can also have representations of the actions they can perform (e.g. offers, threats, admissions), their *goals*, and their *attitudes* toward other objects. Actions and *goals* are not further specified with values. *Attitudes* are used in a similar fashion to attributes. Currently *attitudes* and *goals* are used as talking points only. In future, we plan to connect *goals* with actions and other domain knowledge. These additional aspects are

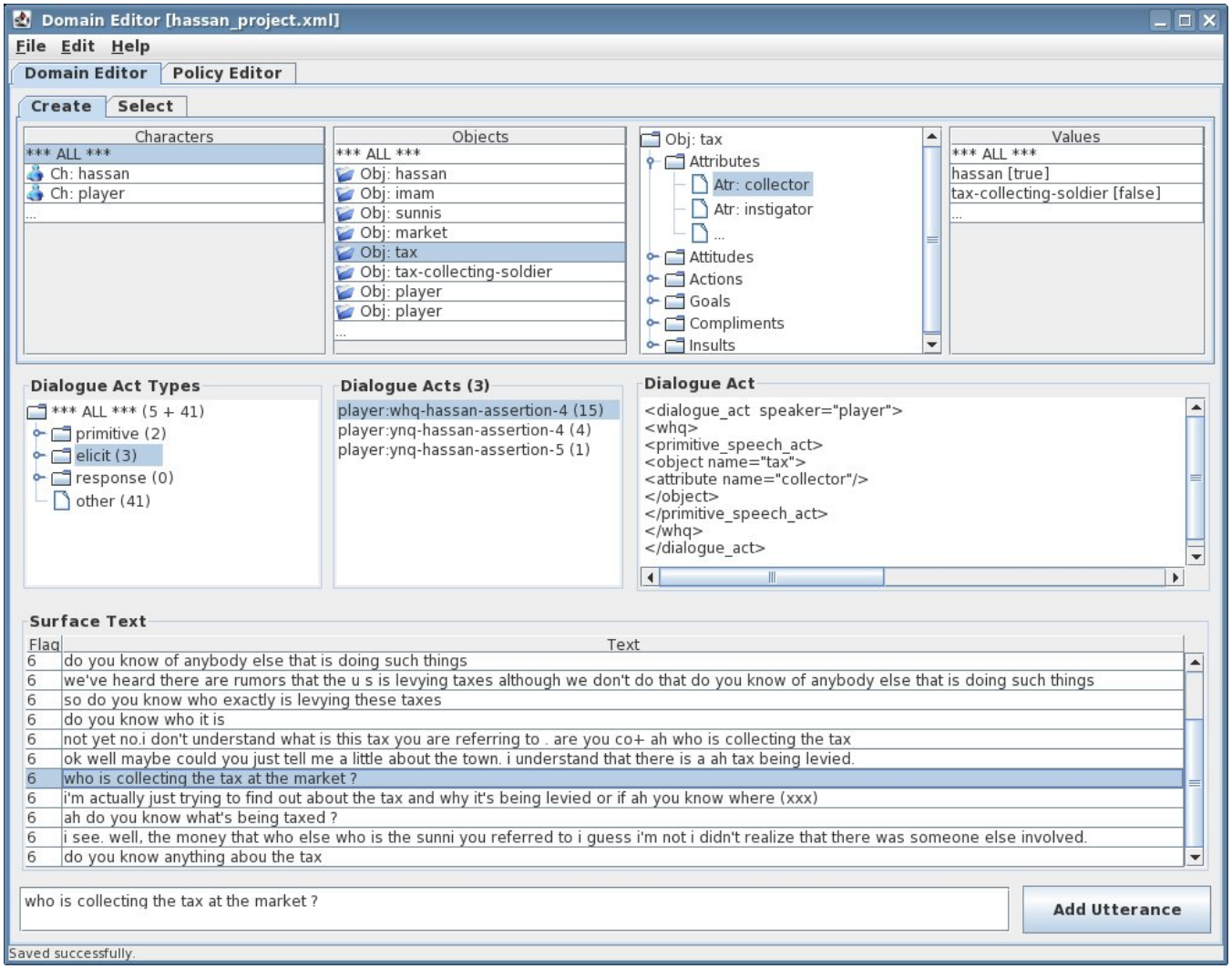


Figure 2: A tool for designing the domain, dialogue acts and the utterances that map to those dialogue acts.

used to create policies for how to engage in questioning dialogues. Another aspect crucial for tactical questioning is social behavior for building rapport, oriented more generally to the characters rather than specific objects. We have sets of these kinds of content, including *compliments* and *insults*.

We use a simple XML representation of these domain knowledge aspects, for ease of use across modules of the system at both domain specification and run-time. Figure 3 shows parts of the domain specification used for the Hassan scenario, including an attribute and an action for Hassan, and both a true and false (i.e. lie) value for an attribute about the tax.

The topmost panel of the authoring tool (see Figure 2) shows the domain creation aspects, where, moving from left to right, authors can add or delete characters, objects, attributes (or other object contents) and values. The tool automatically constructs XML like in Figure 3. This top most section is also used to filter the set of dialogue acts shown in the middle panel of the GUI.

Dialogue Level

Once the domain is defined, it needs to be linked up with the language that will be used to refer to it. Dialogue acts form the middle level in this link, having domain aspects as their contents and being identified directly as the interpretations of language utterances.

Dialogue Acts

Our dialogue manager reasons about several standard types of dialogue acts, including *assertions*, *yn-questions*, *wh-questions*, *offers*, *threats*, *compliments* and *insults*. Following Core and Allen (1997) we have dialogue acts with forward-function – *elicitations* and with backward-function – *responses* for most of the acts. Figure 4 shows our XML representation of some of these acts, which contain a speaker (one of the characters), an act-type, and contents.

All dialogue acts are automatically created from the domain representation as per Algorithm 1. E.g. all <object,attribute,value> triples known by a character can

```
<domain name="hassan">
  <character name="hassan">
    <object name="hassan" type="person">
      <attribute name="role">
        <value>middle-man</value>
      </attribute>
      <actions>
        <offer name="cooporate"/>
      </actions>
    </object>
    <object name="tax" type="abstract">
      <attribute name="collector">
        <value>hassan</value>
        <value isTruth="false">
          tax-collecting-soldier
        </value>
      </attribute>
      ...
```

Figure 3: Aspects of the *Hassan* domain

hassan.assert

```
<dialogue_act speaker="hassan">
  <primitive_speech_act>
    <assertion>
      <object name="tax">
        <attribute name="collector">
          <value>hassan</value>
        </attribute>
      </object>
    </assertion>
  </primitive_speech_act>
</dialogue_act>
```

Indeed, you might say that I collect the taxes.

player.offer

```
<dialogue_act speaker="player">
  <primitive_speech_act>
    <offer name="give-money"/>
  </primitive_speech_act>
</dialogue_act>
```

We can offer you financial reward.

hassan.elicit-offer

```
<dialogue_act speaker="hassan">
  <elicit>
    <primitive_speech_act>
      <offer name="give-money"/>
    </primitive_speech_act>
  </elicit>
</dialogue_act>
```

I might tell you what you want if there was something in it for me.

Figure 4: Sample dialogue acts automatically generated from the *Hassan* domain along with example utterances.

serve as the contents of an *assert* with that character as the speaker. Likewise, any <object,attribute> pair known by another character can be queried with a *wh-question* addressed to that character. We also generate some generic

Algorithm 1 Generation of dialogue acts from domain

for all *speaker* $\in$ characters **do**
  /* Primitive dialogue acts */
  for all *obj* $\in$ objects under *speaker* **do**
    ADD assertions (*speaker*, *obj*, *atr*, *val*)
    ADD attitudes (*speaker*, *obj*, *atd*, *val*)
    ADD actions (*speaker*, *obj*, *act*)
    ADD goals (*speaker*, *obj*, *goal*)
    ADD compliments (*speaker*, *obj*, *compl*)
    ADD insults (*speaker*, *obj*, *insult*)
    ADD groundingDAs (*speaker*, *obj*)
  end for
  /* Dialogue acts that relate to other characters */
  for all *char*$'$ $\in$ (characters $\setminus$ *speaker*) **do**
    for all *obj*$'$ $\in$ objects under *char*$'$ **do**
      /* Forward-looking dialogue acts */
      ADD whq (*speaker*, *obj*$'$, *atr*$'$)
      ADD ynq (*speaker*, *obj*$'$, *atr*$'$, *val*$'$)
      ADD elicit-action (*speaker*, *obj*$'$, *act*$'$)
      /* Backward-looking dialogue acts */
      ADD response-action (*speaker*, *obj*$'$, *act*$'$)
      ADD response-compl (*speaker*, *obj*$'$, *compl*$'$)
      ADD response-insult (*speaker*, *obj*$'$, *insult*$'$)
      ADD groundingDAs (*speaker*, *obj*$'$)
    end for
  end for
  /* Generic dialogue acts */
  ADD greetings, closings, accept, reject, refuse-answer, ack, offtopic, ...
end for

dialogue acts that are customary in human-human conversations like *greeting* and *closing*, that are not tied to any specific domain content. Grounding acts like *repeat-back*, *request-repair* are also generated. *Offtopic* is a special dialogue act specifically designed to handle out-of-domain dialogue acts from the player. The Hassan domain has 102 dialogue acts with Hassan as speaker and 108 dialogue acts with player as the speaker.

The middle panel of the authoring tool shown in Figure 2 allows selection from among the full set of dialogue acts. The left pane allows selection of the type of dialogue act; the middle pane lets one select individual dialogue acts; the right pane shows the full XML content of the dialogue act. In future, instead of showing a dialogue act with XML representation, we plan to use pseudo-natural language – possibly generated using templates. E.g. A template like "*Attribute* of *Object* is *Value*" for *assert* dialogue act type.

Dialogue Manager

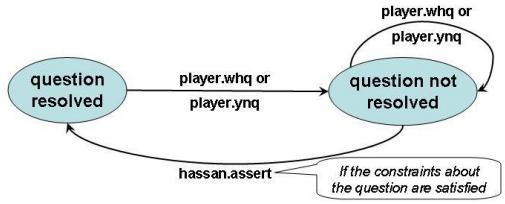
Previous dialogue manager for tactical questioning characters like Hassan (Roque and Traum 2007) made use of hand-authored rules for tracking affective variables and offers and threats made. It used these to compute a *compliance level*, which would dictate how the character would respond. There were three possible compliance levels – adversarial, reticent and compliant. The system's response

was determined using text-to-text mappings. That architecture required developers to specify complete input-text to output-text mappings for all three compliance levels. But this architecture could not handle the dependencies between utterances that go beyond just the adjacent ones.

In order to handle such dependencies and to reason at a more abstract level, the new dialogue architecture makes use of dialogue acts and the domain content. The dialogue manager used in this architecture is based on the information state model (Traum and Larsson 2003). The information-state is in part based on conversational game theory (Lewin 2000). The main responsibilities of the dialogue manager are to update the information state of the dialogue and use it to select the contents of the response. The dialogue manager gets input dialogue acts from the NLU and outputs dialogue acts to the NLG. It decomposes the dialogue acts in order to update the information state.

The information state update rules describe grammars for conversational game structure and are written as state charts. We are using State Chart XML (SCXML), a W3C working draft (Barnett et al. 2008), for describing the state charts. SCXML allows for explicit data models that can be manipulated by executable code. This code can be triggered on entry or exit from a state or during a transition. As pointed out by Kronlid and Lager (2007), all these features make it viable to implement the information-state based dialogue model with SCXML.[1]

(a) offer subdialogues



(b) question-answer subdialogues

Figure 5: State charts for *Hassan* domain.

We have defined a set of networks for each type of game/subdialogue. Following Traum and Allen (1994), we model the character's conversational obligations using these networks. Each node indicates the state of the obligations and outgoing arcs with the character as the speaker indicate ways to address these obligations. Figure 5(a) shows a sample network that handles dialogue acts for the offer subdialogue. The outgoing arcs from the currently active states denote all possible dialogue acts that can be generated as a response by the system or can be handled as input from the user. Some of these transitions can be conditional and depend on the data model configuration (i.e. information-state). Although the network structures themselves are currently hand-authored as SCXML documents, some of the constraints for these networks can be authored using the authoring tool as shown in Figure 6.

Figure 6 shows the policy editing pane. The leftmost pane lists domain elements that are marked as sensitive information. This can be marked at the level of an object or a specific attribute of an object. For every sensitive information the author can provide a constraint. The constraint can be any boolean expression formed by using the information state elements which can be chosen from a drop-down list. A question about this sensitive information will not be answered till corresponding constraint is satisfied. In case the constraint is not satisfied then actions specified in the rightmost pane are executed. E.g. Any *yn-question* or *wh-question* about the *object* "Tax" will not be answered unless the player has extended the *offer* of "give-money". In case, such a question is asked and the required constraint is not met, then Hassan will set a preference for the offer "give-money", which in turn will result in the next move from Hassan being an *elicit-offer*.

		question resolved, offer not elicited	
1	P	whq	Ok I'm trying to understand where the local taxation is coming from?
		question not resolved, offer not elicited	
2.1	H	grounding	So you want to talk about the taxes.
2.2	H	elicit-offer	I might tell you what you want if there was something in it for me.
		question not resolved, offer elicited	
3	P	offer	We can offer you financial reward.
		question not resolved, offer given	
4.1	H	response-offer	That is very generous of you.
		question not resolved, offer not elicited	
4.2	H	assert	Please understand, I collect taxes for my Imam. All in service to Allah.
		question resolved, offer not elicited	
5	P	whq	And what is his name?
		question not resolved, offer not elicited	
6	H	elicit-offer	My friend, if people find out that I tell you this, it would be a problem for me.

Figure 7: Example dialogue showing the currently active states for the networks in Figure 5. P is the player (human trainee) and H is Hassan.

As an example, in the dialogue from Figure 7, the player asks a sensitive question (utterance 1), the constraints for which are not yet satisfied. At this point as per the authored policy (see Figure 6), Hassan sets the preference for "give-money" offer and chooses to start the offer subdialogue by eliciting that offer (utterance 2.2). After utterance 3 the constraints are met. Hassan can then respond to the offer (*hassan.response-offer* – utterance 4.1) thus completing the offer subdialogue and answer the question (*hassan.assert* – utterance 4.2) thus resolving the question under discussion

[1]We used the apache commons SCXML implementation. [http://commons.apache.org/scxml]

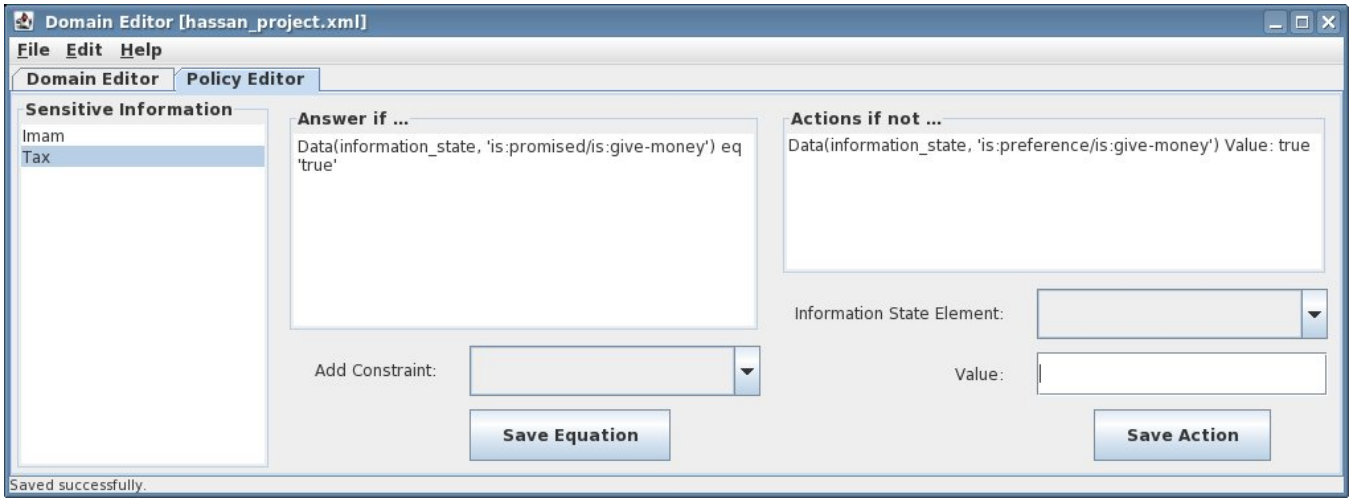


Figure 6: Authoring tool can be used to specify the conditions for question-answering network.

and completing the question-answer subdialogue.

We have authored subdialogue networks for greeting, compliment, insult, question-answering, offer, threat, preclosing, closing and grounding subdialogues. Consistent with our design approach of allowing non-experts to rapidly build the dialogue systems, the scenario developer is expected to select from such a set of subdialogues/games for a given domain. A finite set of games can be identified that would cover most of the dialogue phenomena. Still the user is allowed to author subdialogue networks from first principles if needed.

As part of the information state, the dialogue manager maintains which offers or threats have been given. Apart from these each subdialogue maintains appropriate information to conduct that subdialogue. e.g. The question-answer network remembers the last question asked. The dialogue manager also keeps track of the emotional state of Hassan which is composed of emotions like feels-respected, respects-interviewer, social-bonding and fear (Roque and Traum 2007). The transition networks inside the dialogue manager update these emotions based on the incoming dialogue acts. Based on these emotions the character's compliance level is determined as adversarial, reticent or compliant. This compliance level influences what kind of reply will be given. E.g., when adversarial, the character may choose to lie in response to questions, if a lie is available. Apart from emotional state the dialog manager also manages grounding with help of separate set of networks (Roque and Traum 2008).

Textual Level

Natural language understanding and generation converts from surface text to dialogue acts and back again respectively. The authoring tool shown in Figure 2 supports this via links between natural language texts in the bottom pane, and dialogue acts in the middle pane. For each dialogue act from the character, the author can add one or more options for the character to realize this act. Likewise, for the player dialogue acts, the author can link possible ways for the player to produce this act. The Hassan domain has its 102 dialogue acts with Hassan as the speaker connected to 129 surface text utterances. Its 108 dialogue acts with player as speaker are connected to 187 utterances.

The NLU uses a statistical language modeling text classification technique (Leuski and Traum 2008) to map the text produced by the speech recognition to dialogue acts. In case the closest dialogue act match falls below a threshold an *unknown* dialogue act is passed on to dialogue manager. The NLG works in a similar fashion but in reverse direction. Both NLU and NLG require a training corpus of sample utterances linked to dialogue acts, which can be produced using the authoring tool as described above. The task of generating a training corpus for NLU and NLG can be time consuming. It is mitigated by allowing utterances to be linked only to a dialogue act drawn from a specific set of automatically generated dialogue acts. It is easier to choose a dialogue act for an utterance rather than construct one from scratch. As an example consider the dialogue act as shown in Figure 8. Some of these utterances have multiple functions and can be marked up with multiple dialogue acts. But for simplicity, we annotate only the most salient part of the utterance that can be handled by our dialogue manager. Consider utterance 2 in Figure 8, the clause "so that you could do other things that will better benefit allah." does not have any representation in the dialogue act. By avoiding the construction of the dialogue act from scratch and focusing on the most salient part, we can facilitate and speed up the annotation process of such utterances. This produces *consistent* annotations which by design will be handled correctly by the dialogue manager. Some of the utterances shown in Figure 8 are a result of corpus collection through user testing of the virtual human dialogue system. If available, roleplays or WoZ sessions can also be annotated in a similar fashion. If the non-represented parts of these utterances are deemed important, then the domain specification can be expanded to include those using the tool shown in Figure 2. The tool

will also automatically generate dialogue acts which will be appropriate elicitations/responses to the new additions, thus ensuring *completeness*.

```
<speech_act speaker="player">
  <primitive_speech_act>
    <offer name="protect-hassan"/>
  </primitive_speech_act>
</speech_act>
```

1 ***I promise you that you will not receive any harm*** *for giving me this information.*

2 *Well I can also help you in other ways and* ***we can protect you*** *so that you could do other things that will better benefit allah.*

3 *Well, if you could help us, the perhaps* ***we could put you in protection. and offer you protective custody*** *because if your people are being taxed unfairly, then you're being taxed unfairly as well too and perhaps we can help.*

4 *Sure I understand, as I said,* ***I can make you safe*** *ah if you're able to share information with me. but ah hopefully that will be enough.*

Figure 8: A sample dialogue act along with the corresponding surface text utterances. The most salient part of these utterances which matches with the dialogue act is highlighted.

Evaluation

Two new characters were built using the authoring tool within a period of a few weeks by subject matter experts who did not have any experience in building dialogue systems. One of these characters is named Amani (see Figure 9), who has witnessed a recent shooting in the marketplace. The trainee is to question her to find out the identity, location and description of the shooter (see Figure 10 for a sample interaction). This Amani domain has 89 dialogue acts with Amani as the speaker and these are connected to 98 utterances which are used in the NLG. The domain also has 113 dialogue acts with player as the speaker linked to 681 utterances which are used in the NLU. We have also built a character named Assad, a local shopkeeper in this marketplace. Since then we have also started to build Mohammed, Amani's brother who will share some domain knowledge with Amani. We expect to use the ability of our tool to re-use the authored content from Amani character. Domain knowledge can be re-used at the object level. All the dialogue acts and the corresponding surface text associated with the object can be re-used. Although some of the surface text may need extra processing for things like indexicals.

Our authoring tool allows to annotate an utterance only with a dialogue act that has been automatically generated from the domain knowledge using a simple dialogue act scheme. To verify the coverage of the scheme, we conducted a dialogue act annotation study for one of our characters, Amani (Artstein et al. 2009). A total of 224 unique players utterances which were collected during system testing were matched by 3 annotators to the closest dialogue

Figure 9: Amani – A virtual human for Tactical Questioning build by using the new authoring tool. The man sitting in the chair is Amani's brother, Mohammed.

act; utterances which did not match an appropriate existing dialogue act were marked with the special *unknown* dialogue act. Overall, 53 of the possible 113 player dialogue acts were selected by at least one annotator as matching at least one player utterance. Inter-annotator agreement was substantially above chance, but fairly low compared to accepted standards: $\alpha = 0.489$ when calculated on individual dialogue acts, and $\alpha = 0.502$ when these were collapsed into dialogue act types indicating illocutionary force alone.[2] However, a detailed analysis of the annotations suggested that some of the disagreements were due to unclear guidelines that do not have an impact on system performance, for example whether a question of the form *Do you know...* or *Can you tell...* should be treated as a *yn-question* or *wh-question*. The analysis also revealed some gaps in the coverage of our dialogue acts scheme, such as the absence of questions which ask about an object without specifying an attribute, as in *Tell me more about the sniper*. Since such questions are very common, constituting nearly 12% of our corpus, we added corresponding dialogue acts to the generation algorithm (Algorithm 1). Overall, the analysis shows that with improved guidelines and extensions, our dialogue act scheme can adequately represent around 80% of actual player utterances. The reader is referred to (Artstein et al. 2009) for further details.

Even with the extended dialogue act scheme and improved guidelines, some of the player's utterances will still be marked with the *unknown* dialogue act. Preliminary analysis suggests that it is difficult for annotators to decide whether an utterance can be coerced into one of the existing dialogue acts or whether a new dialogue act needs to be created by extending the domain knowledge of the character. We are currently developing guidelines about when and how to extend the domain to increase the coverage of

[2] Krippendorffs α (Krippendorff 2004) is a generalized measure of interrater agreement, similar to the more familiar K. For a detailed discussion of inter-rater agreement coefficients, see (Artstein and Poesio 2008).

player's utterances. Besides this the dialogue manager has a special network to handle *unknown* dialogue acts which can be caused by out-of-domain utterances or ASR/NLU errors. The dialogue manager attempts to confirm the topic of the conversation and then asks the user to repeat or rephrase. Other strategies to handle *unknown* include taking initiative providing related information about the current topic of conversation if in *compliant* mode or give an *offtopic* response.

Recently, we conducted field testing of Amani at U.S. Military Academy, Westpoint. A total of 33 participants interviewed Amani. These are the users from our target population. In response to the question "In general, Amani responded appropriately to what I was saying." Amani scored 3.24 (mean) on a scale of 1 to 7. For the question "Taken as a whole, Amani was a human-like conversation partner" the score was 3.09. These figures are comparable to third generation Hassan who scored 4.0 and 3.55 respectively (Roque and Traum 2009). Hassan is a character which has been build by several experts over a period of years and through different architectures.

Conclusion

We have described an integrated authoring tool and the accompanying dialogue manger which is used to build several virtual characters for Tactical Questioning. One of the goals is to enable scenario designers to build a dialogue system without the need of expertise in computational linguistics. Our success in building new characters in short amount of time validates the usefulness of the tool and overall architecture.

Acknowledgments

We wish to thank Kevin He and Sarrah Ali for authoring the initial Assad and Amani scenarios, and Michael Rushforth and Jordan Peterson for participating in the annotation study. We also thank two anonymous reviewers for their thoughtful comments and suggestions.

The project or effort described here has been sponsored by the U.S. Army Research, Development, and Engineering Command (RDECOM). Statements and opinions expressed do not necessarily reflect the position or the policy of the United States Government, and no official endorsement should be inferred.

References

Army. 2006. Police intelligence operations. Technical Report FM 3-19.50, Department of the Army. Appendix D: Tactical Questioning.

Artstein, R., and Poesio, M. 2008. Inter-coder agreement for computational linguistics. *Computational Linguistics* 34(4):555–596.

Artstein, R.; Gandhe, S.; Rushforth, M.; and Traum, D. 2009. Viability of a simple dialogue act scheme for a tactical questioning dialogue system. In *DiaHolmia, 13th Workshop on the Semantics and Pragmatics of Dialogue*.

Barnett, J.; Akolkar, R.; Auburn, R. J.; Bodell, M.; Burnett, D. C.; Carter, J.; McGlashan, S.; Helbing, T. L. M.; Hosn, R.; Raman, T.; and Reifenrath, K. 2008. State Chart XML (SCXML) : State machine notation for control abstraction. http://www.w3.org/TR/scxml/.

Bohus, D., and Rudnicky, A. 2003. Ravenclaw: Dialog management using hierarchical task decomposition and an expectation agenda. In *proccedings of Eurospeech-2003*.

Core, M. G., and Allen, J. F. 1997. Coding dialogs with the damsl annotation scheme. In *In Proceedings of AAAI97 Fall Symposium on Communicative Action in Humans and Machines, AAAI*.

Gandhe, S.; DeVault, D.; Roque, A.; Martinovski, B.; Artstein, R.; Leuski, A.; Gerten, J.; and Traum, D. 2008. From domain specification to virtual humans: An integrated approach to authoring tactical questioning characters. In *Interspeech 2008*.

Jordan, P.; Hall, B.; Ringenberg, M.; Cue, Y.; and Rose, C. 2007. Tools for authoring a dialogue agent that participates in learning studies. In *proceedings of AIED 2007*, 43–50.

Krippendorff, K. 2004. *Content Analysis, An Introduction to Its Methodology 2nd Edition*. Sage Publications.

Kronlid, F., and Lager, T. 2007. Implementing the information-state update approach to dialogue management in a slightly extended SCXML. In *Proceedings of the SEMDIAL*.

Larsson, S.; Berman, A.; Hallenborg, J.; and Hjelm, D. 2004. Trindikit 3.1 manual. Technical report, Department of Linguistics, Goteborg University.

Leuski, A., and Traum, D. 2008. A statistical approach for text processing in virtual humans. In *Proccedings of 26th Army Science Conference*.

Leuski, A.; Patel, R.; Traum, D.; and Kennedy, B. 2006. Building effective question answering characters. In *Proceedings of the 7th SIGdial Workshop on Discourse and Dialogue*, 18–27.

Lewin, I. 2000. A formal model of conversational game theory. In *Fourth SemDial Workshop: Gotalog 2000*, 115–122.

MITRE, C. 2005. Midiki: MITRE dialogue kit user's manual. Technical report, The MITRE Corporation.

Roque, A., and Traum, D. 2007. A model of compliance and emotion for potentially adversarial dialogue agents. In *The 8th SIGdial Workshop on Discourse and Dialogue*.

Roque, A., and Traum, D. 2008. Degrees of grounding based on evidence of understanding. In *Proceedings of the 9th SIGdial Workshop on Discourse and Dialogue*, 54–63.

Roque, A., and Traum, D. 2009. Improving a virtual human using a model of degrees of grounding. In *Proceedings of IJCAI-09*.

Sutton, S.; Cole, R.; Villiers, J. D.; Schalkwyk, J.; Vermeulen, P.; Macon, M.; Yan, Y.; Rundle, B.; Shobaki, K.; Hosom, P.; Kain, A.; Wouters, J.; Massaro, D.; and Cohen, M. 1998. Universal speech tools: The CSLU toolkit. In *In Proceedings of ICSLP*.

Traum, D. R., and Allen, J. F. 1994. Discourse obligations in dialogue processing. In *proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*.

Traum, D., and Larsson, S. 2003. The information state approach to dialogue management. In van Kuppevelt, J., and Smith, R., eds., *Current and New Directions in Discourse and Dialogue*. Kluwer.

Traum, D.; Leuksi, A.; Roque, A.; Gandhe, S.; DeVault, D.; Gerten, J.; Robinson, S.; and Martinovski, B. 2008. Natural language dialogue architectures for tactical questioning characters. In *Proceedings of 26th Army Science Conference*.

Xu, W., and Rudnicky, A. I. 2000. Task-based dialog management using an agenda. In *ANLP/NAACL 2000 Workshop on Conversational systems*, 42–47.

			both not greeted, compliment replied, question resolved, offer not elicited, unknown responded, track topic
1	P	greeting	hello
			amani greeted
2	A	greeting	Hello.
			both greeted
3	P	compliment	you have a wonderful house
			compliment given
4	A	response-compliment	This house used to be much nicer before.
			compliment replied
5	P	whq	can i ask you a few questions about the shooting incident yesterday
			question not resolved
6	A	assert	Go on if you can be quick about it.
			question resolved
7	P	unknown	i would like to assure you that you will be protected
			do request repair
8	A	request-repair	You were not clear in what you said. Say it again.
			unknown responded
9	P	offer	i promise to keep you safe
			offer given
10	A	response-offer	You promise me safety.
			offer not elicited
11	P	whq	did you know who did the shooting
			question not resolved, repeat back
12.1	A	repeat-back	So you want to talk about the man, who did the shooting?
			question not resolved, track topic
12.2	A	assert	I have heard that he is called Saif.
			question resolved
13	P	whq	do you know where he lives
14	A	elicit-offer	You are asking for a lot of information. if you expect me to tell you that. You must offer me secrecy.
15	P	offer	i will keep this a secret
16.1	A	response-offer	You have assured me of secrecy.
16.2	A	assert	I believe he hides on the second floor of Assad's shop. I know he is in there. but when i go in the shop i don't see him. And i have no idea where he is now.
17	P	whq	do you know anything about the shooter's daily activities
18.1	A	repeat-back	So you want to talk about the man, who did the shooting?
18.2	A	elicit-offer	As you know, I am a teacher. We are not provided the luxuries of great wealth. One hundred of your dollars would go a long way to help my family make it through these hard times.
			question not resolved, offer elicited
19	P	reject	sorry i'm not authorized to give you any money
			question not resolved, offer not given
20	A	refuse-answer	There is not much i wish to say about that.
			question resolved, offer not elicited

Figure 10: A sample dialogue with Amani. The first line shows currently active states (shown in **bold**) for greeting, compliment, question-answer, offer, unknown and grounding subdialogue networks. In subsequent lines only the change in the active state is shown. P is the player (human trainee) and A is Amani. In Amani domain *Shooter's name, location* and *daily routine* are marked as sensitive information. The *name* will be released only if *safety* is promised. Since *safety* is already offered in utterance 9, it is not elicited after utterance 11. Information regarding *daily routine* will be released only if *give-money* is promised. In utterance 19 player rejects the elicitation of this offer which results in Amani refusing to answer (utterance 20).