A Common Ground for Virtual Humans:
Using an Ontology in a Natural Language Oriented Virtual Human Architecture

Arno Hartholt (ICT), Thomas Russ (ISI), David Traum (ICT), Eduard Hovy (ISI), Susan Robinson (ICT)
05/30/2008
Overview

- Virtual Humans Project
- Challenge
- Virtual Humans Architecture
- Ontology
- Conclusion
- Future Work
Virtual Humans Project
Virtual Humans Project

- Natural Language Dialogue
- Reasoning and Emotion
- Knowledge Representation
- Conversational Minor Characters
- Natural Language Generation
- Behavior Animation
- Speech Processing
- Non-Verbal Behavior Sensing
- Non-Verbal Behavior Understanding
- Non-Verbal Behavior Generation

Virtual Human Architecture
Challenge – Knowledge Representation

- **Module-specific representation**
  - How to communicate between modules?
  - How to make sure translations are correct?

- **Uniform representation**
  - Common understanding
  - Reuse
  - Impoverished or rich?
Our Process

- Multi-phase project life cycle
- First, choose individual representations, suitable for the state of the art in that area
- Next, bring languages closer together
Solution - Old and New Architecture

Old architecture

- Interface 1: Ontology 1
- Interface 2: Ontology 2
- Interface 3: Ontology 3
- Interface 4: Ontology 4

New architecture

- Central Interface
- Common Ontology

Processes:
- Emotion Reasoning
- Action Planning
- Dialogue Management
- NL Generation
- Speech Synthesis
- Gesture Generation
- NL Understanding
- Speech Recognition

- Emotion Reasoning
- Action Planning
- Dialogue Management
- NL Generation
- Speech Synthesis
- Gesture Generation
- NL Understanding
- Speech Recognition
Knowledge Representation

- **Static knowledge:**
  - Offline authoring
  - Social / psychological behavior
  - Domain knowledge

- **Dynamic knowledge:**
  - Runtime
  - Current state / events
  - Communication through message protocol
Virtual Humans Architecture

“I want to move the clinic”

Speech-act: statement
Event: move
Agent: captain
Theme: clinic

“We have no medical supplies”

Speech-act: statement
Polarity: negative
Object: market
Attribute: resource

Domain Specific Knowledge
Domain Independent Knowledge
World State
Protocol Knowledge Management
Intelligent Cognitive Agent
Task Planner
Emotion Model
Dialog and Discourse Management

Speech Generation
Dialog and Discourse Management
Emotion Model
Task Planner

Body Planning

Visual Game Engine

Virtual Human

Human Trainee

Real Environment

Vision Recognition

Speech Recognition

Body

Mind

Natural Language Understanding

Vision Understanding

Environment

Human

“IKnow to move the clinic”

Smartbody Procedural Animation Planner
Focus

- Task model and NLU
- Domain dependent knowledge
- Strong interaction
- Big authoring effort
Task Model - Plans
Task Model - Whole
defTask doctor-moves {
    :agent doctor
    :theme clinic
    :event move
    :source here
    :destination there
    :instrument locals
    :pre {have-transport clinic-here}
    :add {clinic-there}
    :del {clinic-here}
}
defState clinic-here  

:belief true 
:initialize here 
:probability 0.8 
:concern {doctor 20} 
:sim-object *none*
Natural Language Understanding (NLU)

<i want to move the clinic</i>

<i>.meta.id mcsw</i>
<i>.mood declarative</i>
<i>.sem.speechact.type statement</i>
<i>.sem.modal.desire want</i>
<i>.sem.type event</i>
<i>.sem.event move</i>
<i>.sem.theme clinic</i>
<i>.sem.source here</i>
<i>.sem.destination there</i>
Natural Language Understanding (NLU)

- Framebank: sets of semantic frame / utterance tuples
- NLU trains on framebank
- Either building frames or retrieving frames
Frames Ontology

Core Concepts

Module-specific knowledge

Module-specific knowledge

Exporters

Module 1

Module 2
Frames Ontology – Core Concepts

Task:  
- event move
- agent captain-kirk
- theme clinic
- source market
- destination downtown

State:  
- object clinic
- attribute location
- value downtown
Frames Ontology – Task Model

Task: event move
agent captain-kirk
theme clinic
source market
destination downtown
pre: clinic-location-market
del: clinic-location-market
add: clinic-location-downtown

State: object clinic
attribute location
value downtown
belief false
concern {doctor-perez 10}
Frames Ontology – Natural Language

mood declarative
sem.speechact.type statement
sem.modality.deontic must
sem.polarity positive
sem.type event
sem.event move
sem.agent captain-kirk
sem.theme clinic
sem.source market
sem.destination downtown
OWL

- Hierarchy
- Assertions on all levels
- Automatic classification
OWL Ontology

- Scenario-independent Ontology

- Scenario family 1
  - Scenario 1-1
  - Configuration file

- Scenario 2-1
  - Configuration file

- Scenario 3-1

- General world knowledge
  - (generic actions, objects…)
- Linguistic structures
- Generic dialogue items

- Scenario actor classes
- Scenario action templates
  - (generic preconds, effects…)
- Propositions for scenario

- Specific actors, locations, etc.
- Sentences for scenario

- Actor positions, attitudes, etc.
- Simulation initialization
Ontology Action Templates

Templates define semantics of action:
Ex: One effect of a move action is that the destination will contain the thing being moved (the ‘theme’)

Hierarchy allows inheritance of semantics
Creating a new *move* action

1. Select class
2. Create instance
3. Add details
   - agent
   - theme
   - source
   - destination
4. Inherit details
   Preconditions, effects, etc. are added automatically (and can be edited)
Creating a new *move* utterance

1. Select class
2. Create instance
3. Add details
   - semantic object
   - speech act
   - modality
   - addressee
4. Add utterance(s)
Conclusion

- Multiple phase life cycle allows quick prototyping
- **Ontology pros**
  - Module synchronization
  - Formal specification
  - Reuse of knowledge
  - Reference to data, rather than copy
  - Common user interface
- **Ontology cons**
  - Extra learning curve
  - Some changes are more difficult
  - User interface not ideal
Future Work

- Extend natural language capabilities (NLG, Lexicon)
- Integrating other modules
- Completing consistency and integrity testing routines
- Exploring using existing resources
- We want to enable non-Computer Science people to author new scenarios...so we’re investigating the optimal point in the trade-off between:
  - Programming for Non-Experts:
    - Reduced, simple, scripting languages
    - BUT limited in functionality
  - Powertools for Experts:
    - High functionality
    - BUT need considerable expertise
Questions

Thank you