Collaborative Problem Solving

Lisa Beinborn

Seminar: Advanced dialogue modeling for practical applications

Outline

- Background
  - Understanding of terms and assumptions
- Model
  - Problem Solving
  - Collaborative Problem Solving
- Applications
  - A sample architecture (TRIPS)
  - Some sample rules (Sammy)

Articles

- Nate Blaylock, James Allen, and George Ferguson

  Managing communicative intentions with collaborative problem solving (2003)

- Nate Blaylock

  Towards flexible, domain-independent dialogue management using collaborative problem solving (2007)

- James Allen, Donna Byron, Myroslava Dzikovska, George Ferguson, Lucian Galescu, and Amanda Stent

  Towards Conversational Human-Computer Interaction (2001)

Collaborative Problem Solving

- The process of agents jointly choosing goals, planning and acting in order to accomplish them.
- No single-agent decisions possible
- Model the range between system-control and user control

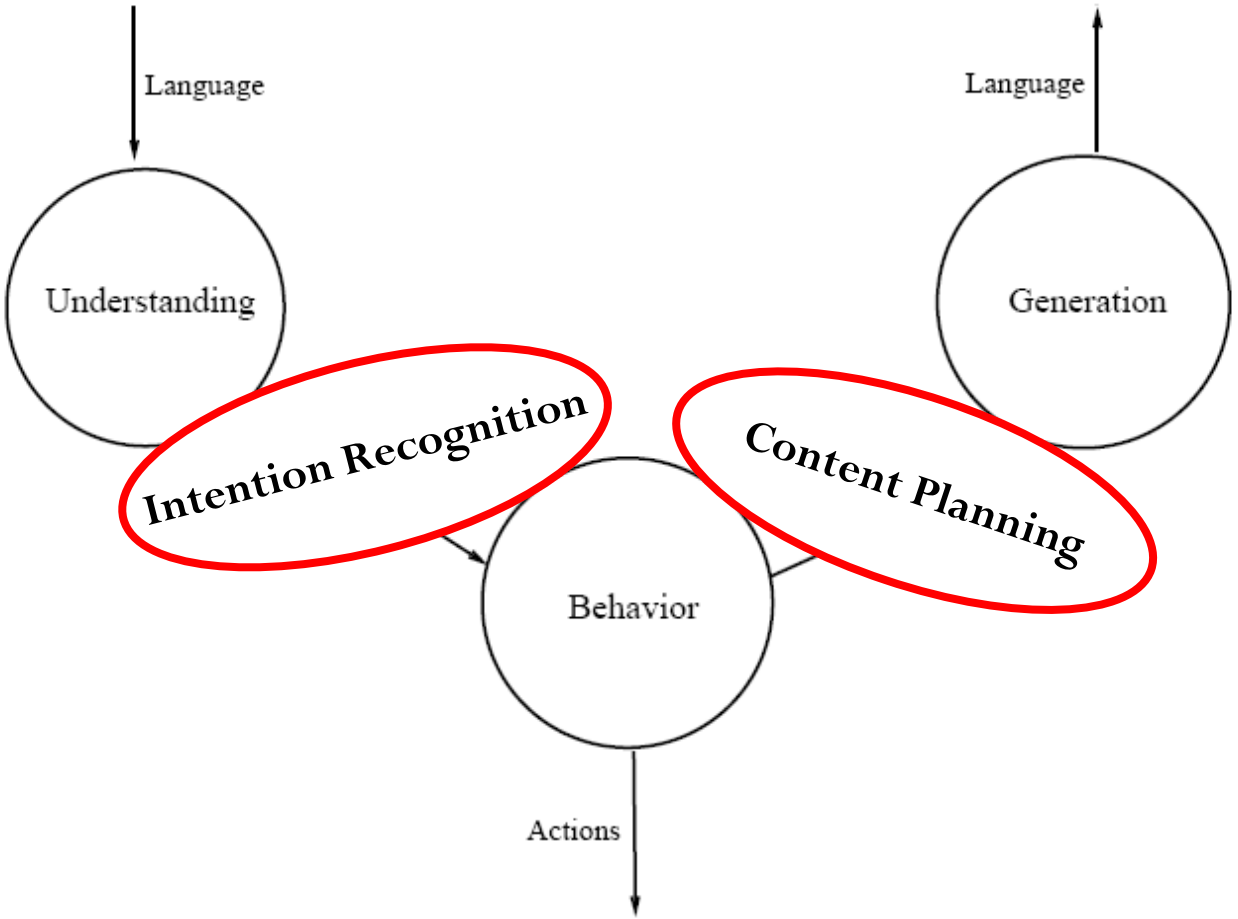
Assumptions

- Practical Dialogue Hypothesis
  - The conversational competence required for practical dialogues, while still complex, is significantly simpler to achieve than general human conversational competence.
- Domain Independence Hypothesis
  - Within the genre of practical dialogue, the bulk of the complexity in the language interpretation and dialogue management is independent of the task being performed.

Goals

- Domain Portability
- Abstracting from linguistic surface
- Create Intention/Language Interface
  - Intention Recognition
  - Content planning
- Collaboration in planning *and* acting

Building Conversational Agents



Abstract Objects

- Objectives
- Recipes / Solutions
- Resources
- Situations
- Atomic Actions
- (Evaluations)
- (Constraints)

Task Model

- A task model provides domain-specific instantiations for the abstract objects
  - Implemented as typed feature structures
  - All objects descend from the abstract types
- The problem solving methods are domain-independent

Single Agent Problem Solving (PS)

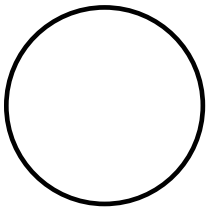
- Determining Objectives
  - ♦ What is the goal?
- Determining and Instantiating Recipes for Objectives
  - ♦ How can it be achieved?
- Executing Recipes and Monitoring Success
  - ♦ Acting and Reporting

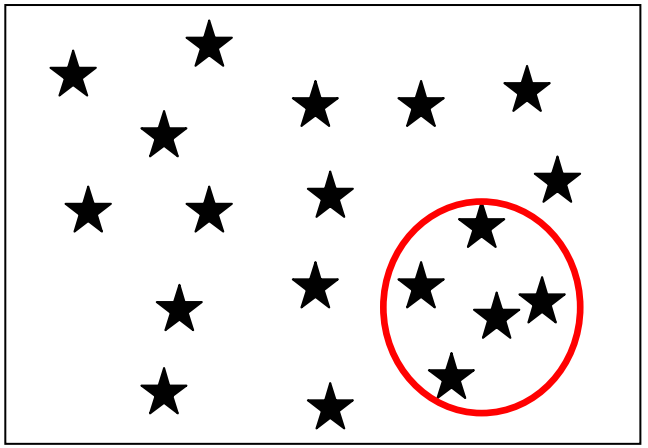
Commitment Acts

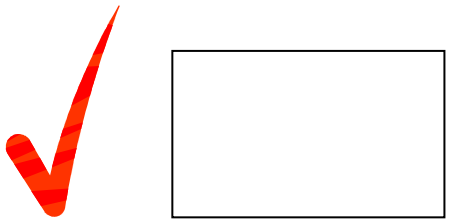
- Adopt
- Select
- Defer
- Abandon
- Release

Commitment Acts

- Adopt
- Select
- Defer
- Abandon
- Release

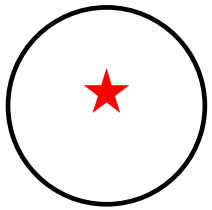


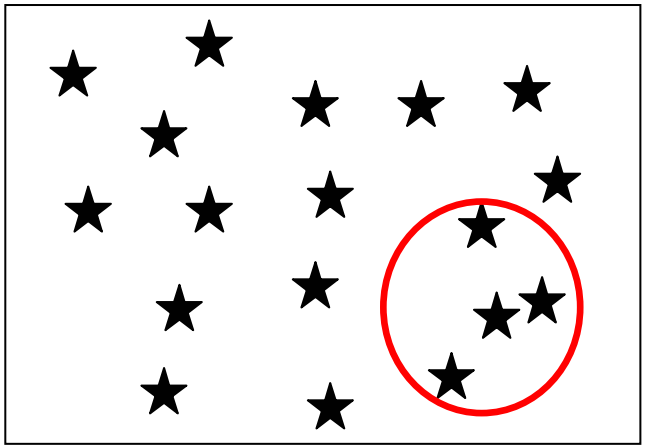


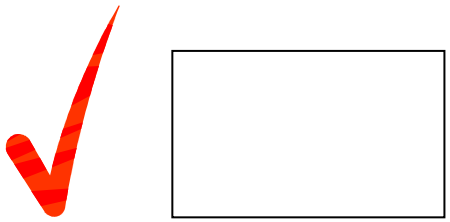


Commitment Acts

- Adopt
- Select
- Defer
- Abandon
- Release







Application of Commitment Acts

	Adopt	Select	Defer	Abandon	Release
Objective	X	X	X	X	X
Recipe	X			X	X
Action	X	X	X	X	X
Resource	X			X	X
Situation	X			X	

Reasoning Acts

- Identify
  - Which options are available?
- Evaluate
  - Which options make sense?
- Modify
  - Any parameters have to be changed?

Application of Reasoning Acts

	Identify	Evaluate	Modify
Objective	X	X	X
Recipe	X	X	X
Action	X	X	X
Resource	X	X	
Situation	X	X	X

Collaborative Problem Solving (CPS)

- Same Acts as in single agent PS
  - ♦ c-adopt, c-select, c-defer,…
  - ♦ c-identify, c-evaluate
- Can only be performed after interaction
- Cooperation and Coordination of both agents
- Different grades of collaboration

Interaction Acts

- Initiate
- Continue
- Complete
- Reject

- Take CPS-acts as argument
  - ♦ (**initiate** (**c-adopt** (**objective** (rescue person1))))

Dialogue Types

- Planning
- Acting
- Interleaved Planning and Acting
- Rejection
- Negotiation

Interleaved Planning and Acting

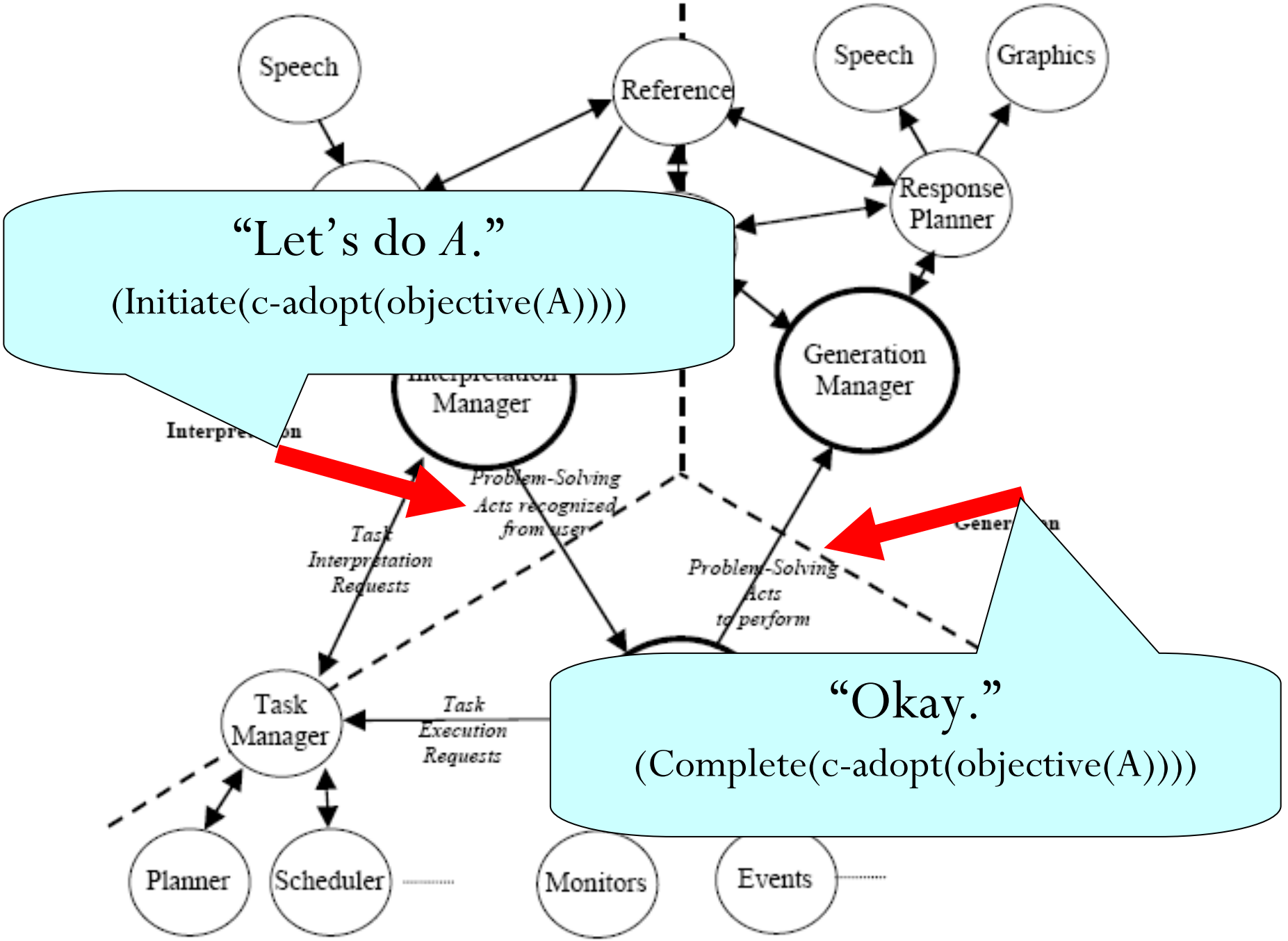
User: **Send ambulance one to Parma right away.**
(initiate (c-adopt (action (send amb1 Parma))))
(initiate (c-select (action (send amb1 Parma))))
System: **OK.** [sends ambulance] Acting!
(complete (c-adopt (action (send amb1 Parma))))
(complete (c-select (action (send amb1 Parma))))
System: **Where should we take the victim once we pick them up?**
(initiate (c-adopt (resource (hospital ?x))))
User: **Rochester General Hospital**.
(continue (c-adopt (resource (hospital RocGen))))
System: **OK.**
(complete (c-adopt (resource (hospital RocGen))))

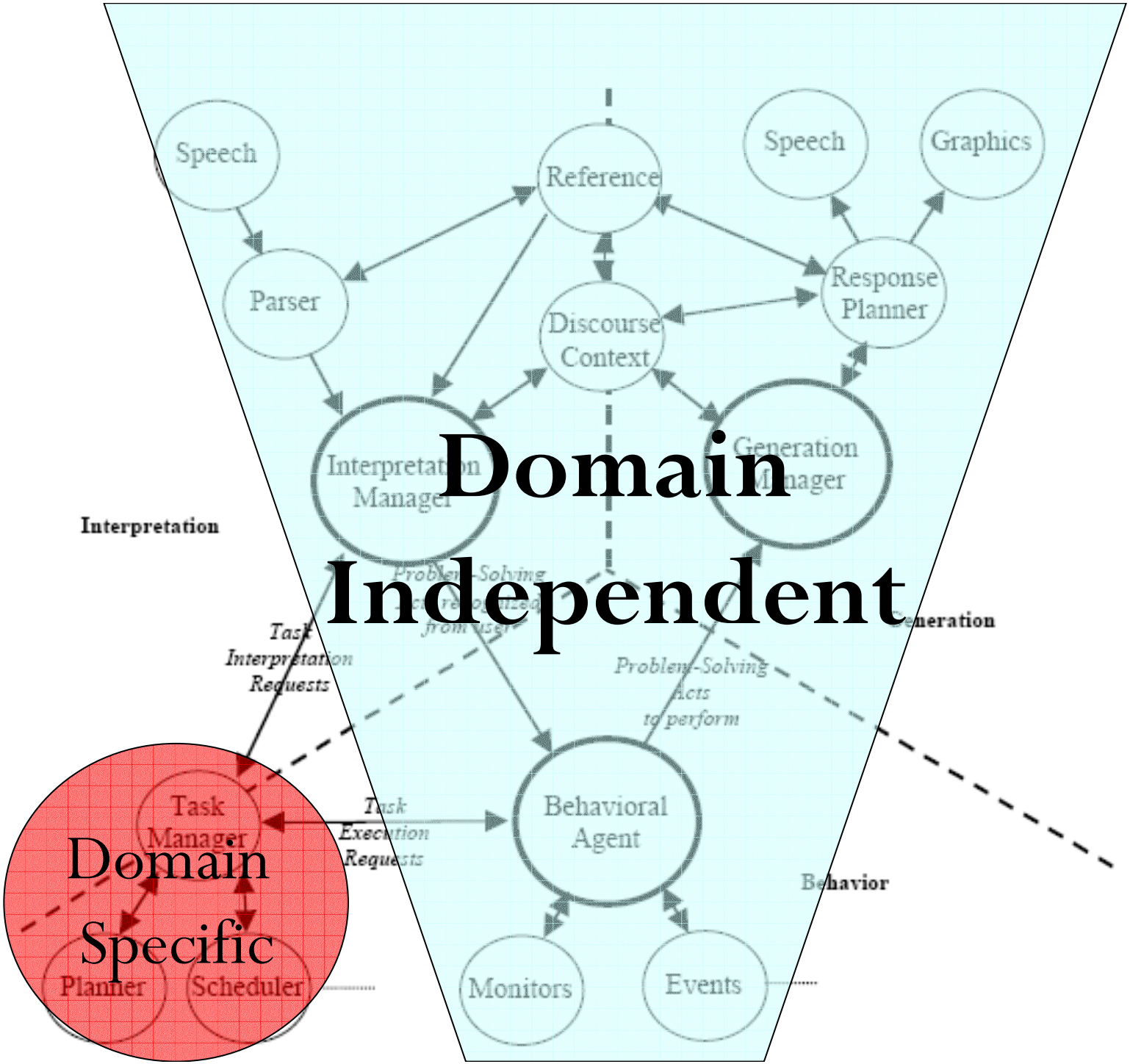
Negotiation

User: **Let's use Ambulance one.**
(initiate (c-adopt (resource (amb1))))
System: **Ambulance one is too far away.**
(reject (c-adopt (resource (amb1))))
(initiate (c-evaluate (resource (amb1) bad too-far)))
System: **Ambulance two is closer.**
(initiate (c-evaluate (resource (amb2) good closer)))
System: **Let's use it.**
(initiate (c-adopt (resource (amb2))))
User: **OK, we'll do that instead**.
(complete (c-evaluate (resource (amb2) good closer)))
(complete (c-evaluate (resource (amb1) bad too-far)))
(complete (c-adopt (resource (amb2))))

TRIPS

- The Rochester Interactive Planning System
- Intelligent Planning Assistant
- Architecture implements CPS methods


Speech
Reference
Speech
Graphics
Response Planner
"Let's do A."
(Initiate(c-adopt(objective(A))))
Generation Manager
Manager
Problem-Solving Acts recognized from user
Task Interpretation Requests
Problem-Solving Acts to perform
Task Manager
Task Execution Requests
"Okay."
(Complete(c-adopt(objective(A))))
Planner
Scheduler
Monitors
Events


Speech
Reference
Speech
Graphics
Parser
Response Planner
Discourse Context
Interpretation Manager
Generation Manager
Interpretation
Task Interpretation Requests
Problem-Solving Acts to perform
Task Manager
Task Execution Requests
Behavioral Agent
Behavior
Monitors
Events
Planner
Scheduler
Domain Independent
Domain Specific

SAMMIE-05

- Multimodal dialogue system for MP3-player control
- Collaboration only in slot-filling

- Part of TALK-project

Domain-independent rules

IF *(initiate(c-identify(resource(x))))*

AND *resource A can be uniquely identified*

THEN *(continue(c-identify(resource(A))))*

- U: Which Beatles album do you have?
  (initiate….
- S: I have “Magical Mystery Tour”.
  (continue…

Domain-independent rules

IF *(complete(c-select(objective(A))))*

THEN *put A on internal stack for execution*

- U: Play “Happy Birthday”!
- S: OK. *(complete(…..*
- *Execute*?

Using domain-specific knowledge

- Invoke grounded recipe
  - Points to object methods
- Some domain-specific rules necessary
  - Execution actions which return value
  - e.g. Create playlist

Questions?

- Considering human as agent (Fabian)
- Choice of Objects (Miro)
- Domain-Portability (Alexander, Zhenghan)
- Modifying a completed CPS-act (Faisal)
  - initiate(abandon(resource((RH))))/initiate(adopt…(NYH))
  - initiate(modify….
- External knowledge in rule formulation (Elahi)

- State-of-the-art? (Milos)
- Differences to SharedPlans (Raveesh)

Thank you!

TRIPS demo video

Conversational Interaction and Spoken Dialogue Research Group