Introduction: Dialogue Management

Seminar „Multimodal Ontology-based Dialogue Systems“

Markus Löckelt
loeckelt@dfki.de
Content

• What is Dialogue Management?
• Different interaction types
• Essential Techniques
  – Information State and Interaction Representation
  – Action Planning Paradigms
  – Interaction with and support of other modules
• Important Issues
  – Managing Multi-Party Interaction
  – Embodiment Issues
  – Dialogue Design
Content

- **What is Dialogue Management?**
- Different interaction types
- **Essential Techniques**
  - Building blocks
  - Information State and Interaction Representation
  - Action Planning Paradigms
  - Interaction with and support of other modules
- **Important Issues**
  - Managing Multi-Party Interaction
  - Embodiment Issues
  - Dialogue Design
The Role of Dialogue Management

Decide what to do
...and do it!

<table>
<thead>
<tr>
<th>Speech</th>
<th>Gesture</th>
<th>Facial Expression</th>
<th>Mouse Input</th>
<th>input modalities</th>
</tr>
</thead>
<tbody>
<tr>
<td>Recognition</td>
<td></td>
<td></td>
<td></td>
<td>attending</td>
</tr>
<tr>
<td>Interpretation / Fusion</td>
<td></td>
<td></td>
<td>Mouse Input</td>
<td>identifying</td>
</tr>
<tr>
<td>Discourse Modelling</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Action Planning</td>
<td></td>
<td></td>
<td></td>
<td>comprehending / conceptualizing</td>
</tr>
<tr>
<td>Fission</td>
<td></td>
<td></td>
<td>Mouse Input</td>
<td></td>
</tr>
<tr>
<td>Generation</td>
<td></td>
<td></td>
<td></td>
<td>formulating</td>
</tr>
<tr>
<td>Synthesis</td>
<td></td>
<td></td>
<td></td>
<td>vocalizing</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Speech</th>
<th>Gesture</th>
<th>Facial Expression</th>
<th>Graphical Events</th>
<th>output modalities</th>
</tr>
</thead>
</table>
The DM module in the System Context
Example: SmartKom
The DM module in the System Context
...if something goes wrong
Tasks of the Dialogue Manager

• Recognize what the user wants (not what s/he said)
  – the utterance is (hopefully) understood and (linguistically) disambiguated
  – Pragmatics: what to do now?
• Devise how and when to do it
  – Action planning and triggering / coordinating actions with application back-end
  – Support other modules with task knowledge
• Provide an answer / reaction
• Manage the interaction
  – Who is talking / provides information
  – System initiative / user initiative / mixed initiative
  – Floor management / Turn taking
• (Do it fast enough)
Content

• What is Dialogue Management?
• **Different interaction types**
• Essential Techniques
  – Building blocks
  – Information State and Interaction Representation
  – Action Planning Paradigms
  – Interaction with and support of other modules
• Important Issues
  – Managing Multi-Party Interaction
  – Embodiment Issues
  – Dialogue Design
Interaction types

- Task-oriented Systems
  - Goal: (collaborative) solution to some task
  - Criteria: Efficiency and effectivity – shortest or cheapest path to the goal
  - Rational / logical approach
  - Cooperation can be assumed
  - Participants usually in fixed roles (e.g., customer and vendor)
  - Frequently used in research systems
    - Restricted domain, clearly defined goals
    - Also very relevant in practical applications
    - ...however, common dialogue phenomena are often ignored for „idealised“ interactions
      - Barge-in / barge-before
      - Offtalk
      - Deliberate topic shifts
Task-oriented dialogue
Task hierarchy and task decomposition
Task-oriented dialogue
Subtask ordering

Diagram:
- Sell train ticket
  - Establish travel time
    - Set departure time
    - Set arrival time
  - Select from list of connections
  - Confirm travel data
- Dominance
- Satisfaction-precedence
Interaction types

- Interactive Narratives / Instructional interaction
  - Goals may be not explicitly pre-determined and/or known to the user
  - Criteria: provide interesting interactive experience, entertainment and atmosphere
  - Characters should act believably with regard to personal attributes and motivations
  - Participants take on different roles, possibly involving uncooperative and/or irrational behaviour
  - Agents are frequently embodied, exhibit affective state
Interactive Narrative Scenario
Example *VirtualHuman*

**Virtual Characters with different roles**

- **Moderator**
- **Expert Kaiser**
- **Expert Herzog**

**Multimodal Interaction**
Speech, Gesture, Facial Expression, Body Language

**Human Quiz Contestants**
Content

• What is Dialogue Management?
• Different interaction types

**Essential Techniques**
  – Building blocks
  – Information State and Interaction Representation
  – Action Planning Paradigms
  – Interaction with and support of other modules

• Important Issues
  – Managing Multi-Party Interaction
  – Embodiment Issues
  – Dialogue Design
Use of knowledge for the DM

- Representing communication in terms of communicative acts that are exchanged, and dialogue history
- Representing the state of the world and the task
- Providing rules for communication between DPs and the modules of the system
- Providing a way to reason about the state of the world and determining courses of action
- Agnostic to modality?
Actions levels – building blocks
Hierarchical depiction

Activities
- Sale
  - Greeting
  - Order

Interactions
- Question/Answer
  - Counter-Question

Dialogue acts
- Request
  - Utterance
- Request
  - Utterance
- Inform
  - Utterance
  - Gesture

Realisation as multimodal Utterances
DSPs

1. A: Good morning.
2. B: Hi.
3. B: Have you seen the coffee machine?
4. A: The one on our floor?
5. B: Yes.
6. A: I think it is broken.
7. A: They took it away yesterday for repairs.

<table>
<thead>
<tr>
<th>Game</th>
<th>Subtree</th>
<th>DSP</th>
</tr>
</thead>
<tbody>
<tr>
<td>$G$</td>
<td>(1)-(2)</td>
<td>Greeting</td>
</tr>
<tr>
<td>$I_1$</td>
<td>(3)-(7)</td>
<td>Information Seeking (“where is the coffee machine?”)</td>
</tr>
<tr>
<td>$I_2$</td>
<td>(4)-(5)</td>
<td>Information Seeking (subgame of $I_1$; “which coffee machine?”)</td>
</tr>
</tbody>
</table>
Dialogue Act Types
(Partial) Example
Dialogue Act Instances

Inform
HAS_INITIATOR
HAS_NAME “User1”
HAS_ADDRESSEE
Character
HAS_GENDER “female”

Goal
HAS_AGENT
FootballPlayer
HAS_NAME “Michel Platini”
HAS_DIRECTION BottomRight
HAS_STYLE RightFootShoot
Information State

- Many modules can be stateless, the DM usually is not
- Necessary to provide interaction / task context for reasoning and determining what to do next
- Different representations, e.g.
  - State in a FSA
  - Belief/Desire/Intention (BDI / BOID)
  - SharedPlans
  - SmartKom et al. (Ontology Instance Store)
Information State Update

Information State
(Collection of assertions about the state of the world and the interaction)

User Utterance

Backend Interpretation

Pragmatic Interpretation

Determining Reaction

System Utterance

Backend Request

Turn
Information State
A definition (TRINDI)

(1) a description of the informational components of the dialogue model, including context and motivating factors such as participants, user models, obligations, beliefs, etc.,

(2) formal representations of the informational components in terms of, e.g., lists, discourse representation structures, or logic expressions,

(3) a set of dialogue moves that realize natural language utterances and trigger information state update,

(4) a set of update rules that are applicable depending on the current information state and performed dialogue moves,

(5) an update strategy for deciding which rule to select from a set of applicable ones, which can be a simple “pick the first one” strategy, or use more sophisticated arbitration mechanisms, such as game theory, utility theory, or statistical methods.
# Action motivation

<table>
<thead>
<tr>
<th>Type</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>automatic</td>
<td>(1) Peter is breathing.</td>
</tr>
<tr>
<td></td>
<td>(2) Peter winces at the sound of the explosion.</td>
</tr>
<tr>
<td>reflexive</td>
<td>(3) Peter looks at Mary while addressing her.</td>
</tr>
<tr>
<td></td>
<td>(4) Peter and Mary shake hands.</td>
</tr>
<tr>
<td>deliberative</td>
<td>(5) Peter answers Mary’s question.</td>
</tr>
<tr>
<td></td>
<td>(6) Peter calls Mary so they can lift the heavy box together.</td>
</tr>
</tbody>
</table>

- Not all interactions / reactions require explicit planning or expensive deliberation.
## Action Planning Paradigms

### Some Examples

<table>
<thead>
<tr>
<th>Technique</th>
<th>Example task</th>
<th>Dialogue phenomena</th>
</tr>
</thead>
<tbody>
<tr>
<td>Finite state scripts</td>
<td>Ticket sale</td>
<td>User answers questions</td>
</tr>
<tr>
<td>Frame-based</td>
<td>Getting train arrival and departure information</td>
<td>User asks questions, simple clarifications by the system</td>
</tr>
<tr>
<td>Set of contexts</td>
<td>Travel booking agent</td>
<td>Shifts between predetermined topics</td>
</tr>
<tr>
<td>Plan-based</td>
<td>Kitchen design consultant</td>
<td>Dynamically generated topic structures, collaborative negotiation dialogues</td>
</tr>
<tr>
<td>Agent-based</td>
<td>Disaster relief management</td>
<td>Multi-party dialogue</td>
</tr>
</tbody>
</table>
Action planning with FSA

- Commonplace in different versions (e.g. VoiceXML)
- Suitable for less complex and "linear" tasks
- Relatively easy to specify
- Not very powerful, but fast and "simple"
- Example: the infamous automated service phone call
Action planning with logic and full planning

BOOK-FLIGHT($A, C, F$):

- **Constraints:** agent($A$) $\land$ flight($F$) $\land$ client($C$)
- **Precondition:** know($A$, depart-date($F$)) $\land$ has-seats($F$) $\land$ want($C$, BOOK-FLIGHT($A, C, F$))
- **Effect:** flight-booked($A, C, F$)
- **Body:** make-reservation($A, F, C$)

INFORM($S, H, P$):

- **Constraints:** speaker($S$) $\land$ hearer($H$) $\land$ proposition($P$)
- **Precondition:** know($S, P$) $\land$ want($S$, INFORM($S, H, P$))
- **Effect:** know($H, P$)
- **Body:** believe($H$, want($S$, know($H, P$)))

- Theoretically very powerful
- Can cover domains by logical descriptions and operators
- Difficult to specify and maintain
- Example: TRIPS
Intermediate approach: using partial plans („recipes for joint action“)

- Uses precomputed „plan library“
- Construction of applications by building blocks
- Less expensive computationally
- Example: dialogue games
# Dialogue Games

<table>
<thead>
<tr>
<th>Game Name</th>
<th>Joint Goal</th>
<th>Goal of Initiator ($I$)</th>
<th>Goal of Responder ($R$)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Information Seeking</td>
<td>$I$ knows the information that is sought</td>
<td>$I$ has identified to $R$ the information that is sought</td>
<td>$R$ has provided the information that is sought</td>
</tr>
<tr>
<td>Socratic Challenge</td>
<td>$I$ knows whether $R$ can construct particular information $i$ which $R$ plausibly is able to construct based on prior experience</td>
<td>$I$ has identified $i$ to $R$</td>
<td>$R$ has exhibited $R$'s own knowledge of $i$</td>
</tr>
<tr>
<td>Permission Seeking</td>
<td>Determine whether $R$ gives permission to $I$ to do a particular action $a$ or seek a particular outcome $o$ of action</td>
<td>$I$ has identified to $R$ the action $a$ or outcome $o$ for which permission is sought</td>
<td>$R$ has decided whether $R$ grants permission to $I$ to do $a$ or seek to achieve $o$</td>
</tr>
</tbody>
</table>
Support for other modules

• The DM often has a good idea of what to expect during the next (few) turns

• Sharing this information: Publishing *expectations* and task state
  - Information attributes likely to be addressed
    • Asked-for, salient from context, already known
    - Lexicon entries
    - Used modalities

• Can benefit, e.g.,
  - Dynamic lexicon
  - Reference resolution
  - Presentation planning
## Using Expectations

<table>
<thead>
<tr>
<th>Initiative</th>
<th>User Initiative</th>
<th>System Initiative</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Expected slot</strong></td>
<td>N A</td>
<td>expected, unify, plan continues</td>
</tr>
<tr>
<td><strong>Possible slot</strong></td>
<td>plausible, unify, replanning</td>
<td>plausible, unify, replanning</td>
</tr>
<tr>
<td><strong>Filled slot</strong></td>
<td>plausible, overlay, replanning</td>
<td>plausible, overlay, replanning</td>
</tr>
<tr>
<td><strong>Other</strong></td>
<td>implausible – recover strategies</td>
<td></td>
</tr>
</tbody>
</table>
Content

- What is Dialogue Management?
- Different interaction types
- Essential Techniques
  - Building blocks
  - Information State and Interaction Representation
  - Action Planning Paradigms
  - Interaction with and support of other modules

- Important Issues
  - Managing Multi-Party Interaction
  - Embodiment Issues
  - Dialogue Design
From Monologue to Multi-Party Interaction

1. System Monologue
2. Interactive System
3. Simulated Conversation
4. Interactive Performance
5. Dialogue between virtual and human agents
**Multi-Party Interaction**

- New and complex phenomena not encountered in simple dialogue, e.g.
  - More intricate management of the right to talk
  - Many different roles possible (cf. Court scene)
  - Social Factors
  - Addressee and speaker recognition
  - Parallel threads of conversation (related or not)

- Agent-based approach most suitable for MPI
  - Goal: *Emergent* Multi-Party Interaction
  - Behaviour is generated independently
  - "Natural" approach with MPI problems; e.g., individual and active negotiation for the floor as an alternative to central coordination
Embodiment Example: Time Coordination

Moderator

„Put that player in the Midfield“
Affirmative Gesture

Silence

Demand Floor (Gesture)

„Yes, he’s strong there“

Processing
Waiting/Hesitation

Realisation time
Grab Turn

Time

Expert
The Interaction triangle

User

Virtual Environment

Characters

Dialogue system

Influence / Demands

Dialogue Designer

Interaction

Definition

Coordination
games
game
  name: comeAgainSpec
  gameType: comeAgainGame
states
state
  name: s2
  body
    setSlot(slotName: lastNotUnderstood, getABoxObject(name: lastNotUnderstood))
    select
      template
        conditions
          condition
            slotEqualsSlot(slotName: lastNotUnderstood, slotName: consumedAct)
        body() <!-- same as last time, do nothing -->
      template
        conditions() <!-- reset counter -->
        body
          replaceSlotAtPath(pathToSlot: notUnderstoodCounter:has_name, slotValue: counter)
    assertObject(name: lastNotUnderstood, slotValue: consumedAct)
    select
      template
        conditions
          condition
            slotIsSet(slotName: content)
        body
          addSlotAtPath(pathToSlot: comeAgainContent:has_content, slotValue: content)
      template
        conditions()
        body()

addSlotAtPath(pathToSlot: comeAgainContent:has_count, getABoxObject(name: notUnderstoodCounter:has_name))
setParameter(name: addressee, string: GUIGen)
setParameter(name: content, slotValue: comeAgainContent)
increment(slotName: counter)
replaceSlotAtPath(pathToSlot: notUnderstoodCounter:has_name, slotValue: counter)

[...]
Dialogue Design

• Very complicated for non-trivial domains
  – „You can‘t see the forest for the trees“
  – Interdependencies
  – „The user can say anything at any time“
  – Debugging and Testing

• Alleviate by
  – Graphical, hierarchical design tools
  – Machine Learning of Interaction Patterns
  – Automated Testing and Verification