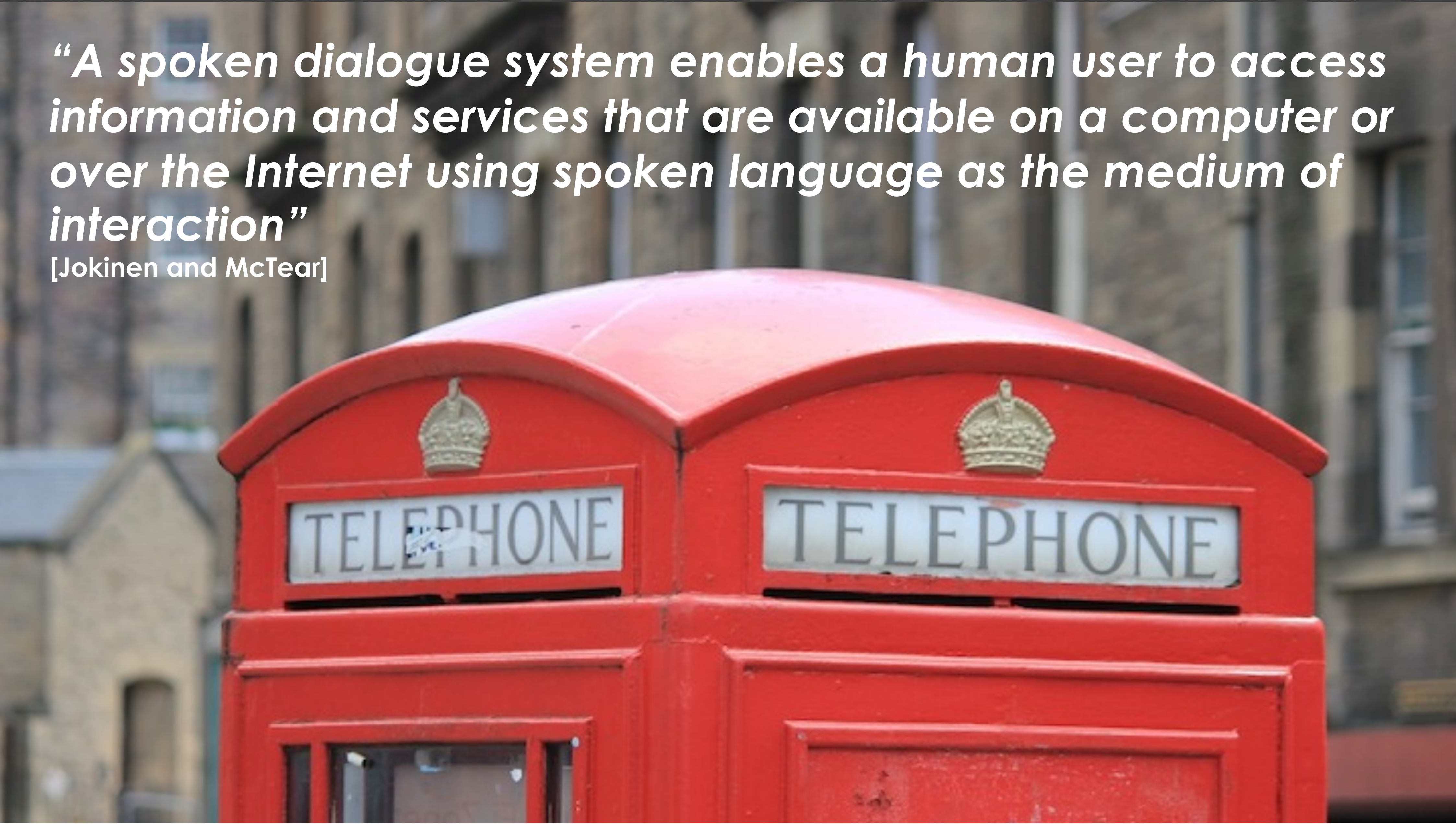
A Simplified Approach Towards the Development of **Natural Dialogue Systems**

NLDB 2015

Passau, 17 - 19 June

Markus M. Berg
University of Kiel, Germany
Institute of Computer Science
http://mmberg.net

Context



"A spoken dialogue system enables a human user to access information and services that are available on a computer or over the Internet using spoken language as the medium of interaction"

[Jokinen and McTear]

What we are used to...

"...If you'd like to hear all of your options again, press 49. If you've forgotten why you called in the first place, press 50."

© 2007 by Randy Glasbergen, www.glasbergen.com

- ...
- Where do you want to start?
- *I'd like to go from Belfast to San Francisco*
- Ok, you start in San Francisco. Where do you want to go?
- *No, Belfast!*
- When do you want to leave from San Francisco?
- *I want to leave Belfast next Monday!*
- Please tell me the date when you want to leave San Francisco!
- ...

What we observe...

“...current dialogue systems show only ***limited capabilities*** *with regard to natural dialogue“* [Pfleger]

*Need “****more powerful*** *dialogue systems so that users do not need to adapt [to the system]“* [Pfleger]

Not usable enough
Not natural enough
Not cheap enough [Bringert]

“... Building a system which interacts competently with users [...] is a ***significant challenge****“* [Glass and Weinstein]

What we want... (and do)

- An easy approach to develop dialogue systems
- Overcome problems with SDS:
  - Endless list of choices
  - Users rather want to actively tell their concern
    - Instead of answering lots of questions
  - Users prefer human-like and polite formulations
- Solution
  - Mixed initiative
  - Over-informative answers
  - Sub dialogues
  - Adaptive formulation
  - Open-ended questions

Idea & Features

- We present a system that addresses these demands and...
  - … separates the dialogue engine from the dialogue specification
    - reusability of the engine
    - model allows the construction of IDEs
  - … focusses on the definition of information units
    - combine description of questions and their possible answers
    - define behaviour in a declarative way
  - … uses language generation methods
    - adaptive and multilingual systems
    - reduce effort for prompt specification
  - … uses predefined natural language understanding modules
    - no need to create grammars and understanding algorithms

Dialogue Model

- **Dialogue**
  - Container element
  - Global settings (e.g. politeness, dialogue strategy, language)
  - List of tasks

Comprises all the capabilities of the interaction

- **Task**
  - Selector that activates the task
  - List of ITOs
  - **Action** to be executed (media control, database, web service...)

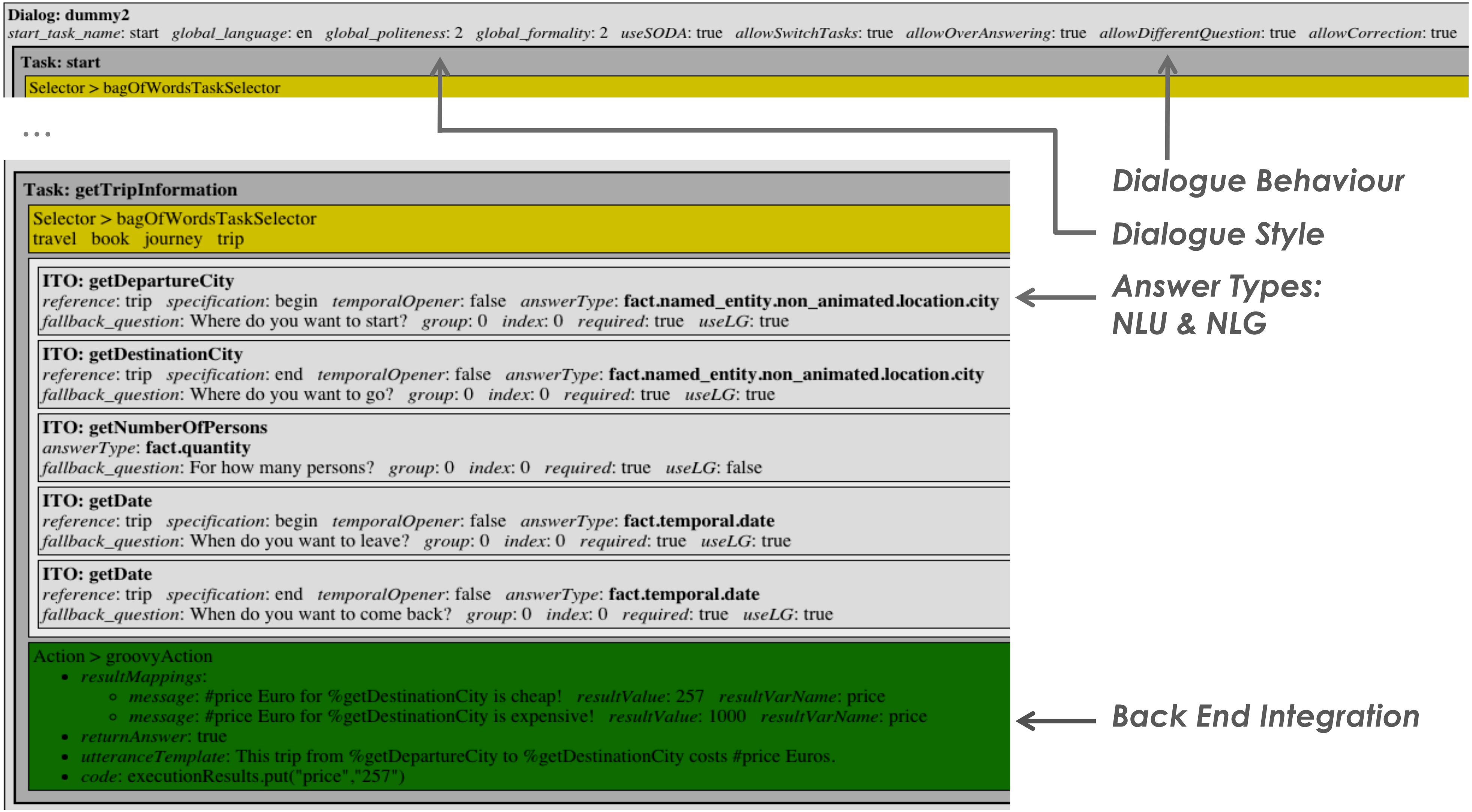
A specific subdialogue, e.g. get weather information

- **Information Transfer Object (ITO)**
  - Define system question and possible answers within one unit

Specific information unit of the task, e.g. the destination city

Visualisation: Dialogue Description

- A Dialogue consists of many tasks



Dialog: dummy2
start_task_name: start *global_language*: en *global_politeness*: 2 *global_formality*: 2 *useSODA*: true *allowSwitchTasks*: true *allowOverAnswering*: true *allowDifferentQuestion*: true *allowCorrection*: true

Task: start

Selector > bagOfWordsTaskSelector

…

Task: getTripInformation

Selector > bagOfWordsTaskSelector
travel book journey trip

ITO: getDepartureCity
reference: trip *specification*: begin *temporalOpener*: false *answerType*: **fact.named_entity.non_animated.location.city**
fallback_question: Where do you want to start? *group*: 0 *index*: 0 *required*: true *useLG*: true

ITO: getDestinationCity
reference: trip *specification*: end *temporalOpener*: false *answerType*: **fact.named_entity.non_animated.location.city**
fallback_question: Where do you want to go? *group*: 0 *index*: 0 *required*: true *useLG*: true

ITO: getNumberOfPersons
answerType: **fact.quantity**
fallback_question: For how many persons? *group*: 0 *index*: 0 *required*: true *useLG*: false

ITO: getDate
reference: trip *specification*: begin *temporalOpener*: false *answerType*: **fact.temporal.date**
fallback_question: When do you want to leave? *group*: 0 *index*: 0 *required*: true *useLG*: true

ITO: getDate
reference: trip *specification*: end *temporalOpener*: false *answerType*: **fact.temporal.date**
fallback_question: When do you want to come back? *group*: 0 *index*: 0 *required*: true *useLG*: true

Action > groovyAction
- *resultMappings*:
  - *message*: #price Euro for %getDestinationCity is cheap! *resultValue*: 257 *resultVarName*: price
  - *message*: #price Euro for %getDestinationCity is expensive! *resultValue*: 1000 *resultVarName*: price
- *returnAnswer*: true
- *utteranceTemplate*: This trip from %getDepartureCity to %getDestinationCity costs #price Euros.
- *code*: executionResults.put("price","257")

Dialogue Behaviour

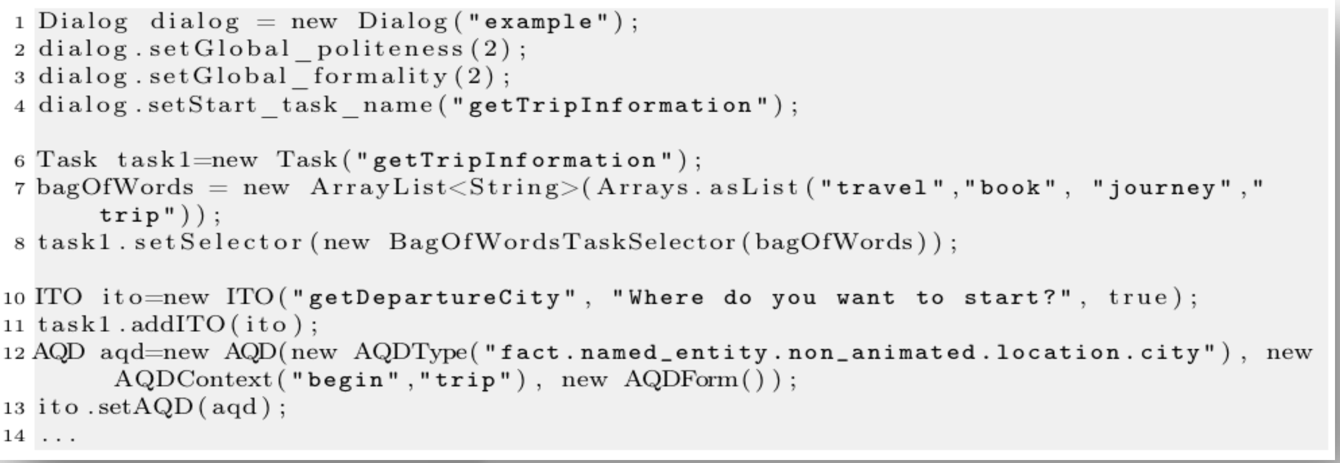
Dialogue Style

Answer Types: NLU & NLG

Back End Integration

Dialogue Description: XML, Java, IDE

- The XML dialogue description can be directly created with a text editor, with the help of a Java library, or with an IDE

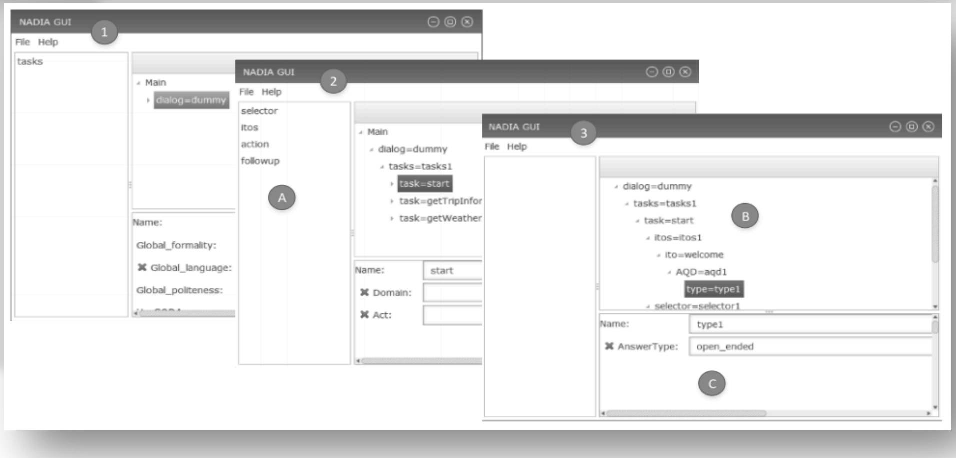


```java
Dialog dialog = new Dialog("example");
dialog.setGlobal_politeness(2);
dialog.setGlobal_formality(2);
dialog.setStart_task_name("getTripInformation");

Task task1=new Task("getTripInformation");
bagOfWords = new ArrayList<String>(Arrays.asList("travel","book", "journey","
     trip"));
task1.setSelector(new BagOfWordsTaskSelector(bagOfWords));

ITO ito=new ITO("getDepartureCity", "Where do you want to start?", true);
task1.addITO(ito);
AQD aqd=new AQD(new AQDType("fact.named_entity.non_animated.location.city"), new
      AQDContext("begin","trip"), new AQDForm());
ito.setAQD(aqd);
...
```

```xml
<task name="getTripInformation">
         <selector>
            <bagOfWordsTaskSelector>
               <word>travel</word>
               <word>book</word>
               <word>journey</word>
               <word>trip</word>
            </bagOfWordsTaskSelector>
         </selector>
         <itos>
            <ito name="getDepartureCity">
               <AQD>
                  <context>
                     <reference>trip</reference>
                     <specification>begin</specification>
                  </context>
                  <form>
                     <temporalOpener>false</temporalOpener>
                  </form>
                  <type>
                     <answerType>fact.named_entity.non_animated.location.city</answerType>
                  </type>
               </AQD>
               <fallback_question>Where do you want to start?</fallback_question>
                   <required>true</required>
               <useLG>true</useLG>
            </ito>
                 ...
```



Selected Aspect: Question Generation

- Goal: Prevent static formulation
- Abstract Question Description (AQD)
  - **Type** (Linguistic Datatype): ***What the question asks for***
    - e.g. `fact.temporal.date`
  - **Context**: ***What the question is about***
    - e.g. *begin of trip*
  - **Form**: ***How the question is asked***
    - Politeness and formality levels

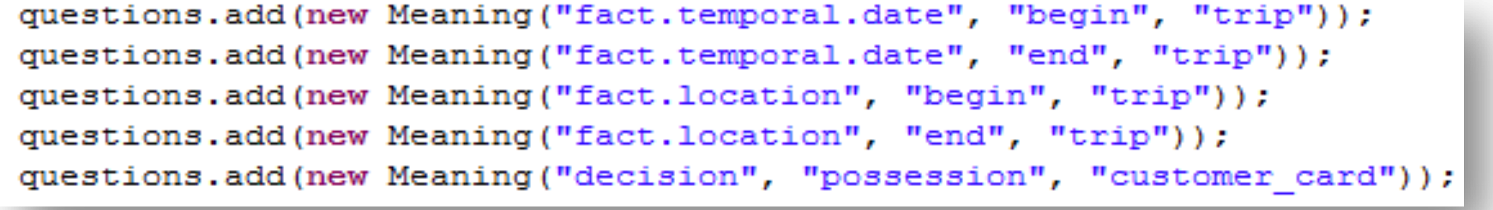
„Where do you want to go?“

„Tell me your destination!“

- Lexicon: AQD annotations (Ontology)
  - „where“ can be used in a spatial dimension in any context
  - „go“ can be used in a spatial dimension in the context of the end of a trip or in a temporal dimension in the context of the beginning of a trip
- Grammar: OpenCCG
  - Different question styles (structure /syntax)
    - Select words according to meaning specified by the AQD

Selected Aspect: Question Generation

- The following code...



```
questions.add(new Meaning("fact.temporal.date", "begin", "trip"));
questions.add(new Meaning("fact.temporal.date", "end", "trip"));
questions.add(new Meaning("fact.location", "begin", "trip"));
questions.add(new Meaning("fact.location", "end", "trip"));
questions.add(new Meaning("decision", "possession", "customer_card"));
```

- ... leads to different surface forms on runtime
  - Politeness and formality is set on a global dialogue level

S: Departure date ***please****!*
U: ...
S: ***Now tell me*** *your return date!*
U: ...
S: ***Tell me*** *your departure city!*
U: ...
S: And the destination ***please****!*
U: ...
S: Do you have a customer card?
U: ...

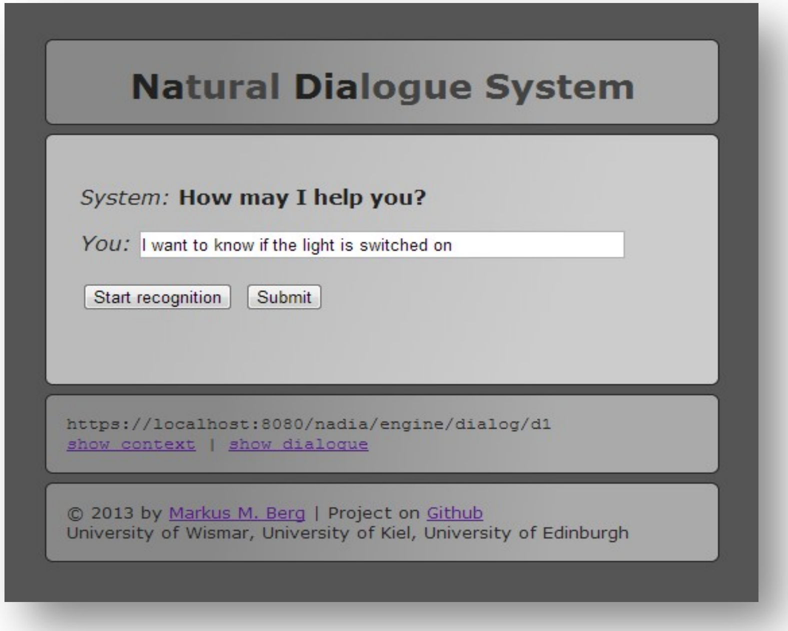
Formality =2, politeness=1

S: ***When do you*** *want to travel?*
U: ...
S: ***Can you now tell me*** *when you want to return?*
U: ...
S: ***Please tell me*** *your departure city!*
U: ...
S: And ***where do you*** *want to go?*
U: ...
S: Do you have a customer card?
U: ...

Formality =2, politeness=4

NADIA

- The *Natural Dialogue System* processes the dialogue description
- *NADIA*:
  - REST-based
  - Java
  - embedded Jetty-Server
  - Uses OpenCCG
- Several user interfaces
  - Web UI: Google TTS/ASR
  - Skype
  - Console
  - …



Natural Dialogue System

System: **How may I help you?**

You: I want to know if the light is switched on

https://localhost:8080/nadia/engine/dialog/d1
show context | show dialogue

© 2013 by Markus M. Berg | Project on Github
University of Wismar, University of Kiel, University of Edinburgh

Example: Booking Dialogue

- System: **How may I help you?**
- *User: I'd like to get price information about a flight.*
- **Where do you want to depart?**
- *In Hamburg.*
- **Can you please tell me where you want to go?**
  - *Can you tell me something about Edinburgh?*
  - **Edinburgh is the capital city of Scotland, situated on the southern shore of the Firth of Forth. With a population of 482,640 […].**
- **Where do you want to go?**
  - *And how is the weather in Edinburgh?*
  - **16°C.**
- **Can you tell me where you want to go?**
- *Okay, I want to go to Edinburgh on 26/04/2015.*
- …

Evaluation

Facts

- Experts were presented the software and a manual
- Novelty: 4.3 / 5
- Demand: 4.4 / 5
- Contribution to more natural dialogues: 4.3 / 5

Opinions

The model allows “*to specify the structure of dialogues more fully than in existing freely-available systems* [which] ***makes the process of designing dialogues [...] much faster and simpler***“

“Nadia deals very effectively with user-driven shifts in dialogue context. In this respect ***it makes a very valuable contribution to development of dialogue systems***“

Benefits

- Teaching & Rapid Prototyping
  - Show effects of different features:
    - Dialogue acts
    - Sub dialogues
    - Open-ended questions
    - Over-informative answers
    - Natural Language Generation
  - Declarative approach: Change the settings without changing the code
  - No need to programme the whole engine
  - Easily develop own dialogues and connect to new back ends
  - Web-based (no installation required)

Thank you for your attention!

Anything **unclear**?
Feel free to ask!

... or drop me a line later
mail@mmberg.net

Last words...

- This presentation refers to the following paper:
  - Berg, Markus M.
    NADIA: A Simplified Approach Towards the Development of Natural Dialogue Systems.
    Natural Language Processing and Information Systems, 20th International Conference on Applications of Natural Language to Information Systems, 2015.
  -

- Bibliography

  - **[Jokinen and McTear]** Kristiina Jokinen and Michael F. McTear. Spoken Dialogue Systems. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers, 2009.
  - **[Bringert]** Björn Bringert. Programming Language Techniques for Natural Language Applications. PhD thesis, University of Gothenburg, 2008.
  - **[Pfleger]** Norbert Pfleger. Context-based multimodal interpretation: an integrated approach to multimodal fusion and discourse processing. PhD thesis, Universität des Saarlandes, 2008.
  - **[Glass and Weinstein]** James R. Glass and Eugene Weinstein. Speechbuilder: Facilitating Spoken Dialogue Sys- tem Development. In Proc. of the 7th European Conference on Speech Communications and Technology, INTERSPEECH, pages 1335–1338, 2001.
  - **[OpenCCG]** http://openccg.sourceforge.net/