Rapid Dialog Prototyping

Martin Rajman, Miroslav Melichar
{martin.rajman,miroslav.melichar}@epfl.ch
Artificial Intelligence laboratory, Swiss Federal Institute of Technology
Lausanne, Switzerland

Introduction

- **Rapid dialog prototyping** is a significant part in the development process of interactive systems, especially for the ones with a **vocal interface**.

- However, due to the **complexity** of spoken languages and their strong **dependences with the context**, there does not exist yet a really generic approach for dialog design; each application requires the development of a **specific model**.

Designing a dialog model
a 3 steps approach

A possible approach for the realization of a dialog model consists in 3 steps:

- An initial **Wizard of Oz experiment** to determine the ***behavior of the users*** when faced with a simulation of the targeted dialog system.

- The use of a **rapid dialog prototyping tool** to build ***a first prototype for the dialog model***.

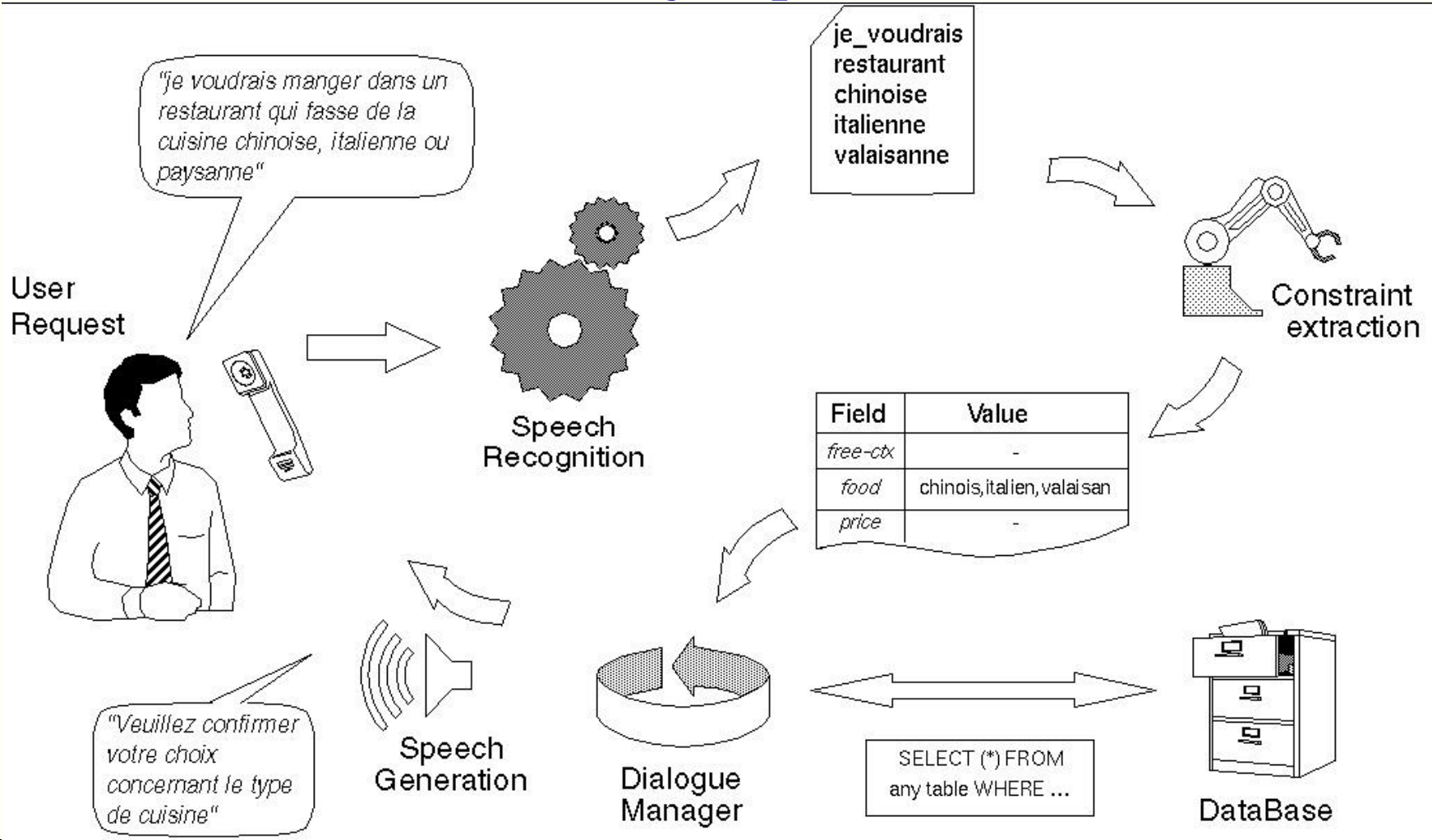
- The **implementation and validation** of an ***operational run-time version*** of the dialog model.

Designing a dialog model
an example: the Rest'Info system

- **The Rest'Info system** is a **vocal phone-based information server** providing information about the restaurants of the city of Martigny (Switzerland).

- The goal of the dialog is to **gather enough information** about the requirements of the user (food type, price range, location,…) in order to provide a **list of relevant restaurants.**

the Rest'Info synoptic architecture



Designing a dialog model
step 1: Wizard of Oz experiment (1)

- **Problem:** Real life conditions need to be taken into account to produce and validate a good dialog model for the targeted application.

- **Solution:** A “Wizard of Oz” experiment, consisting in observing the behaviour of potential users when faced with a simulation of the system.

Wizard of Oz experiment (2)
the main steps

- A **human operator** (the wizard) is trained to ***behave in a way*** that is representative of the ***current abilities*** of the targeted system.
  - **Human-to-human interactions** in the framework of the selected task are ***observed and analyzed*** (initial input).
  - The task is analyzed and represented in the form of a **frame** to be filled in.
  - The results of the analysis are converted into a **set of predefined (prerecorded) questions** (one for each of the slot in the frame) and an **initial generic structure for the dialog nodes**.
- The wizard is used to ***simulate the targeted system*** and some interactions with users are **recorded** and **analyzed**.
- A **first dialog model** is produced.

Wizard of Oz experiment (3)
the Rest'Info frame

type of kitchen	Italian, Chinese, French, traditional,…
location	near to the center, near to the train station,…
price range	cheap, medium, expensive
time	lunch, dinner,…
date	Monday, Tuesday,…, weekend,…, today

Wizard of Oz experiment (4)
Generic dialog node: categorization of the answers

- **OK –** the user answers the question in a usable way.
- **Request for repetition –** the user asks for the repetition of the question.
- **Request for assistance –** the user does not know how to answer the question.
- **Out of context –** the user answers something, which the system can interpret (even partially) as an answer to a different question.
- **Unusable –** nothing can be extracted from the answer.
- **Incoherence –** the answer provided by the user is incoherent with a previously provided one.

Wizard of Oz experiment (5)
Generic dialog node: operator view

Message d'accueil.

bonjour bonsoir

Analyse de reponse:

>indication des champs >avertissement

>indication_valeur >reponse incoherente >reponse OK >reponse hors-contexte >demande de repetition

Reponse incoherente.

repas repas2

cuisine cuisine2

resto resto2

jour jour2

prix prix2

localisation localisation2

Analyse de reponse:

>arret du dialogue >resultats [0 1 2-3 >3] >avertissement

>indication_valeur >reponse OK >reponse hors-contexte >demande de repetition

http://liawww.epfl.ch/~trutnev/InfoVox

Wizard of Oz experiment (6): Example of interaction

- S: Hello! Welcome to the Rest'Info information server on the restaurants of the city of Martigny.

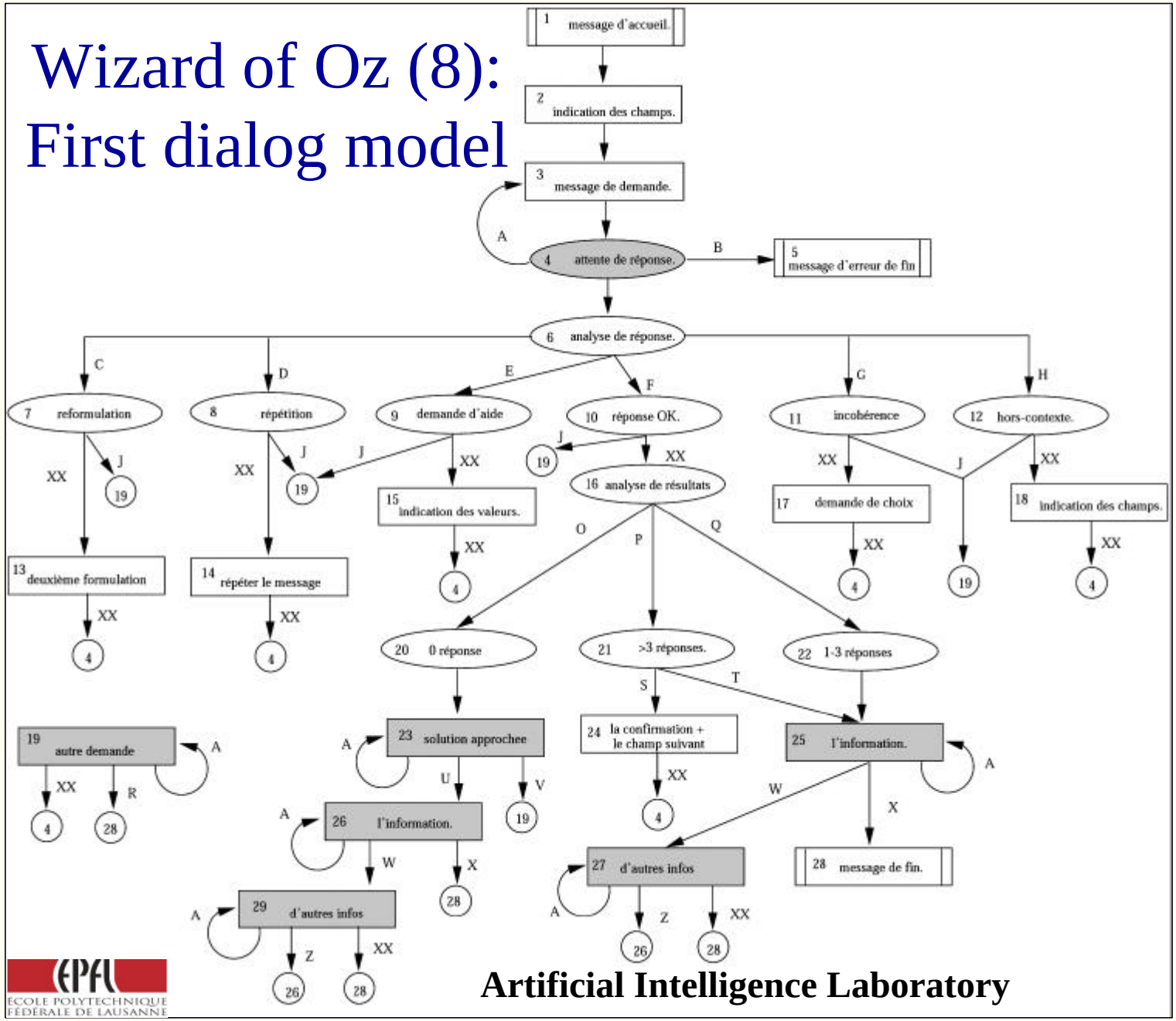
  You can make a search on the type of kitchen, the price, the approximate location of the restaurant or its opening days and hours.

  Please define your request...
- U: I would like to know where there is a pizzeria.
- S: The possible locations are: near to the center, near to the train station, near to the hospital, other locations or no preference.
- U: near to the center.
- S: Are you looking a restaurant for lunch or for dinner?
- U: For the evening.
- S: On what day do you plan to eat?
- U: Next Saturday.
- S: How much do you plan to spend per person?
- U: I don't know, I think we will be 10 persons... The price is not so important.
- S: Here is the answer to your request: Italian restaurant “La Nonna” ...

Wizard of Oz experiment (7)
Some facts

- **Duration: 38 days** (since 2/11/1999 until 17/01/2000)
- **Dialogs: 255** (1 to 6 dialog per person, mean 2.5 d/p)
- **Participants: 99**
  - 9 project members (3 LIA, 4IDIAP, 1 SwissCom, 1 Omedia)
    - 35 dialogs
  - 27 project member collaborators (16 LIA, 9 IDIAP, 1 SwissCom, 1 Omedia)
    - 66 dialogs
  - 48 project members relatives (friend, family, …)
    - 129 dialogs
  - 15 relatives of project member relatives
    - 25 dialogs

Wizard of Oz (8): First dialog model



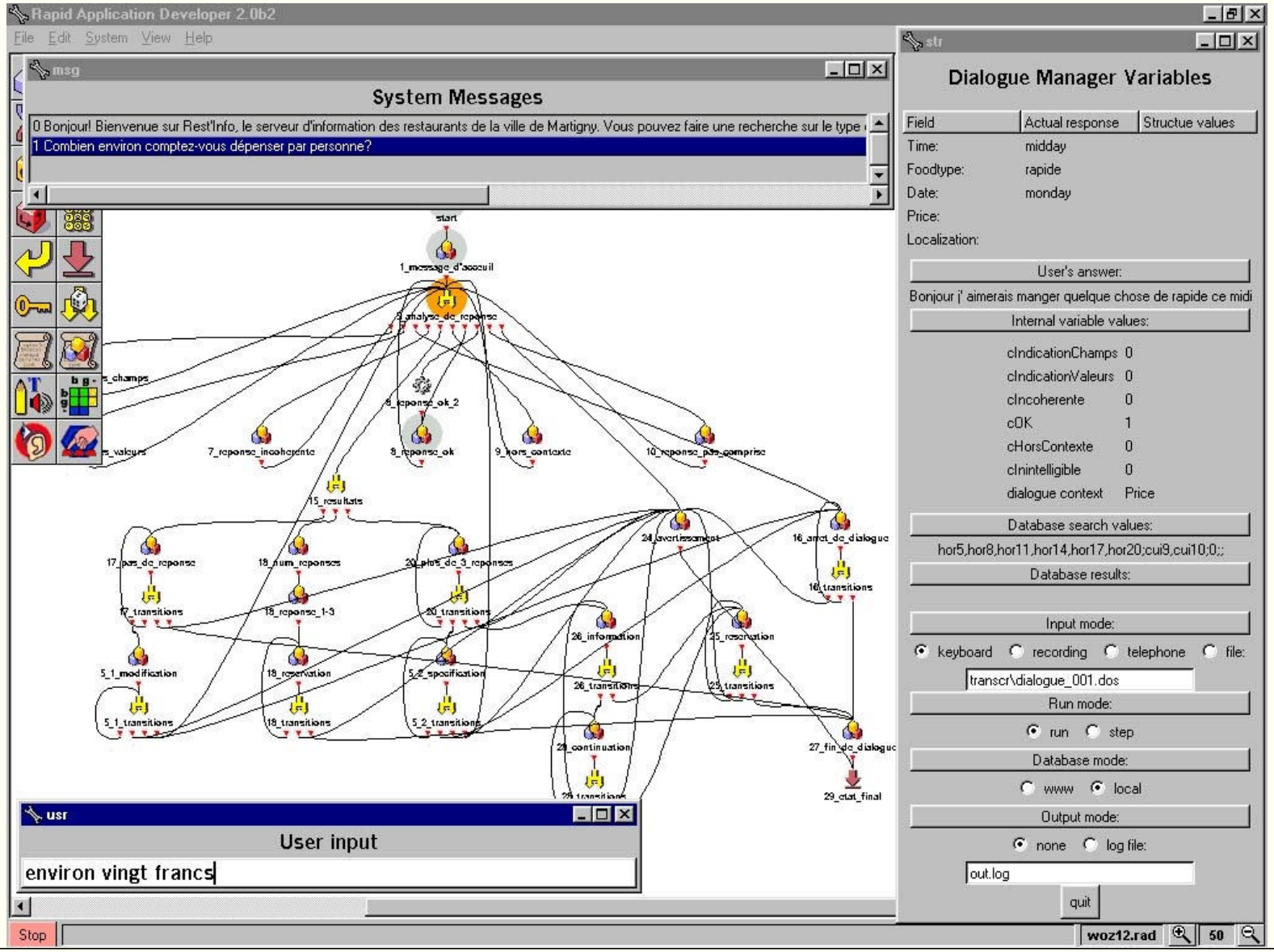
Designing the dialog model
step 2: Rapid dialog prototyping (1)

- The first step leads to an **initial dialog model** description with **nodes**, **connections** between nodes and (informal) **transition rules** defined.
- A rapid dialog prototyping tool is then used to produce a **first implementation** of the initial dialog model.
- The resulting prototype is then **incrementally refined** (node modification, transition rules made progressively formal and implemented).
- Within this framework, a **RAD tool is of invaluable help**.

Rapid dialog prototyping (2)
Requirements

- Implementation which can **easily** (and quickly) be **modified**, in order to experiment with different dialog strategies.
- Short implementation **time**.
- **Graphical display** showing the active machine state, state conditions, internal system variables, system messages, etc.
- **Audio** output capabilities.
- Interface to an external **speech recognizer** using data file exchange.
- Built-in or SQL based interface to a database.

Rapid dialog prototyping (3): the CSLU toolkit



Designing the dialog model
step 3: Building the run-time version

- Once the dialog model is **stabilized**, the **run-time version** of the system can be build. In the case of Rest'Info a C++ program.
- **The specificity** of the run-time version is its full **integration** with the rest of the information system (connection with phone access, connection with real size database,…), its **robustness** and the quality of its **design**.
- A final important step is the **evaluation of the run-time version** of the system that needs to be done with randomly chosen external users (external field test).

Tutorial

- The goal of this tutorial is to use the CSLU's RAD toolkit, to illustrate in a concrete way a **small experiment** in the field of **rapid prototyping** of dialog models.
- We will put the focus on the management of a very simple interaction with a human interlocutor. The objective is to produce a **dialog model** usable to ask the very simple following question:

  "**What is your favorite color?**".
- It is of significant interest to point out that even within the framework of such an elementary application, the **difficulties** arising in dialog management are **quite the same** ones as with a more **complex application**.

Examples of answers
“What color do you like at most?”

- Yellow!
- Well, I like violet blue purple.
- ahh hem [break] me I like I like rather colors hem hot colors.
- What! I haven’t understood, what did you say?
- Sorry! Why you want to know that?
- Yeah, ok, what are the choice?
- Oh! That! But me, I wanted to go to the movies.
- ahhh [break] I never eat oranges!
- humm.

Rapid Application Developer (RAD)

- The CSLU **Toolkit** [Center for Spoken Language Understanding] is a comprehensive suite of tools to enable **exploration**, **learning**, and **research** into speech and human-computer interaction. It was created to provide the basic framework and tools for people to build, investigate and use interactive language systems.
- These tools (available for US and UK English, and also Spanish) incorporate:
  - leading-edge **speech recognition**
  - natural **language understanding**
  - **speech synthesis**
  - **facial animation** technologies
- The main module of the toolkit is the Rapid Application Developer (RAD), a tool for creating structured (basically state-based, system initiative) dialogs (dialog manager) between users and computer. Since the kernel of RAD is a TCL program, it can easily be extended (**frame-based dialog**, **mixed initiative**,…)