Frame-Based Dialogue Systems, VoiceXML

Pavel Cenek

Laboratory of Speech and Dialogue
Faculty of Informatics
Masaryk University
Brno

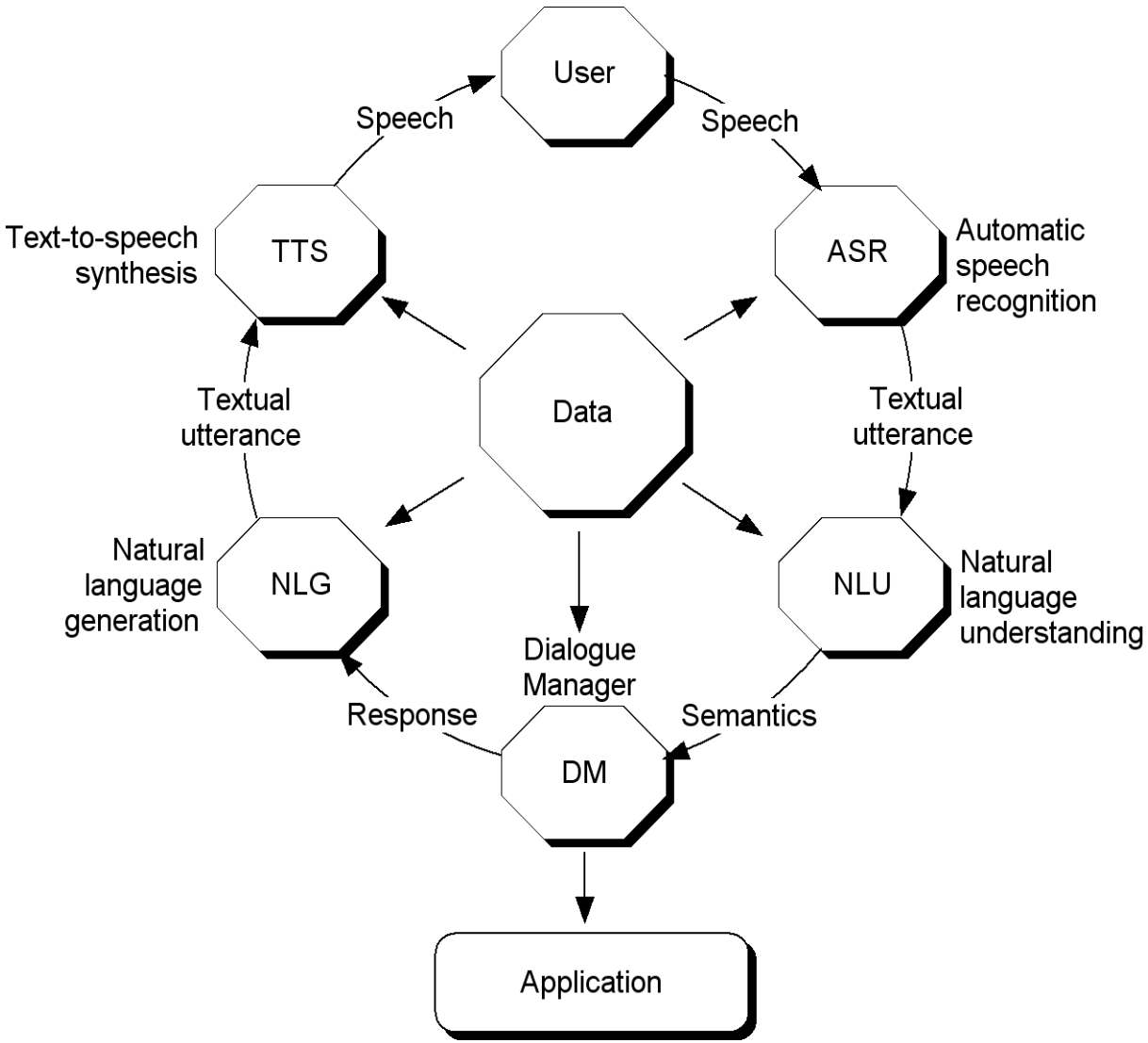
Agenda

- **Dialogue system (DS)**
- Frame-based DS
- VoiceXML

Dialogue System (DS)

- Computer based system
- Communicates with the user by means of natural language
  - Spoken or written form
- Interactive system

Dialogue System Structure



DS Components
Automatic Speech Recognition

- Transforms speech to text
- Two basic types
  - Grammar-based ASR
    - The set of accepted phrases defined by regular/context-free grammars (i.e. language model in the form of a grammar)
    - Usually speaker independent
  - Dictation machine
    - Recognizes “any utterance”
    - N-gram language model
    - Often speaker dependent

DS Components
Natural Language Understanding

- Analyzes textual utterance and returns its formal **semantic representation**
  - Logical formula
  - Name/value pairs
  - …
- When using a grammar-based ASR, semantics is usually encoded in the grammar (referred to as **semantic-based grammar**)

DS Components
Dialogue Manager

- Coordinates activity of all components
- Maintains representation of the current state of the dialogue
- Communicates with external applications
- **Decides about the next dialogue step**

DS Components
Natural Language Generation

- Produces a textual utterance (so called **surface realization**) from an internal (formal) representation of the answer
- The surface realization can include formatting information
  - Speaking style, prosody, pauses
  - Earcons
  - Background sounds
  - ...

DS Components
Text-to-Speech

- Renders an acoustic representation of the surface realization

DS Taxonomy According to the Dialogue Management Approach

- Finite-state dialogue systems
  - Dialogue expressed as a network of states connected by edges
- Frame-based dialogue systems
- Agent-based dialogue systems
  - Communication modelled as an interaction of two intelligent agents that are able to reason
  - Agents have some mental attitudes, e.g. beliefs, desires, intentions and goals

Agenda

- Dialogue system (DS)
- **Frame-based DS**
- VoiceXML

Frame-Based Dialogue Systems

- Based on the slot-filling concept
  - Slots represent “containers” for information that must be elicited from the user
  - Semantics is expressed as name/value pairs
- Slots are stored in a structure called **frame**
- Manageable with the current level of technology

Frame-based Dialogue Systems (2)

Prompt: Where and when do you want to travel?
Grammar: <departure and arrival city, date and time specification>
Help: Please specify the departure and arrival city, date and time

FROM
Prompt: From which city are you leaving?
Grammar: <city specification>
Help: Tell me the name of the city you want to leave from

TO
Prompt: To which city do you want to travel?
Grammar: <city specification>
Help: Tell me the name of the city you want to travel to

WHEN
Prompt: When do you want to travel?
Grammar: <date and time specification>
Help: Please specify date and time of your journey

Filled: SELECT * FROM connections WHERE departure like 'FROM' AND destination like 'TO' *AND* time like *'WHEN'*

Dialogue Strategies

- The task of the dialogue strategy is to decide what the next step of the dialogue will be
- The decision is made based on
  - Semantics of the last user's utterance
  - History of the conversation
  - Knowledge of the domain
  - Knowledge of the user (user model)

Dialogue Strategies (2)

- Local dialogue strategies
  - Control subdialogues with the aim of eliciting values of one or several slots or a special command from the user
- Global dialogue strategies
  - Process the newly filled slots and plan the continuation of the dialogue on the global level

Local Dialogue Strategies

- Request for help
- No input from the user
- Rejection of utterance by the speech recognizer (no match)
- Information about current context
- Repetition of last system prompt (reprompt)
- Pause/Resume
- Restart
- Transfer to operator

Global Dialogue Strategies

- Confirmation strategy
  - No confirmation
  - Implicit confirmation
  - Explicit confirmation

  - Selection based on the degree of certainty that the recognized information is correct
  - Problem: How to recognize that the user corrects a recognition error instead of giving new information

Global Dialogue Strategies (2)

- Integration of the newly acquired information
  - Filled only empty slots
  - Entered identical values for previously filled slots
  - Entered new values for previously filled slots
    - Problem: which value is valid
    - The new value refine the previously entered value (e.g. less than 15000 + more than 10000)

Global Dialogue Strategies (3)

- Relaxing of overconstrained requests
  - No solution satisfying the criteria specified by the user (no suitable object, no item in the database etc.)
  - The value of a slot must be erased/changed
  - Problem: how to select an appropriate candidate

Global Dialogue Strategies (4)

- Dialogue initiative control
  - As long as the dialogue proceeds well, the dialogue system leaves initiative to the user
  - The user can use a large scale of utterances and the conversation can cover a larger part of the domain (i.e. several slots in one dialogue step)
  - When problems arise, the dialogue system must control the conversation and ask more focused questions
  - Problem: How to recognize that problems arose

Global Dialogue Strategies (5)

- Conversational focus selection
  - Problem: which slot should be selected as the topic of conversation (if the conversation is controlled by the system)
  - Important question – the order in which slots are discussed can significantly reduce the length of the dialogue

Global Dialogue Strategies – Summary

- Confirmation strategy
- Integration of the newly acquired information
- Relaxing of overconstrained requests
- Dialogue initiative control
- Conversational focus selection

Domain Representation in Frame-based DS

- Domain (task) represented by
  - Structure of the frame
  - Conditions for filling slot values
  - Relations among slot values
  - Slot priority
- Other possibilities
  - Control table
  - Path constraints

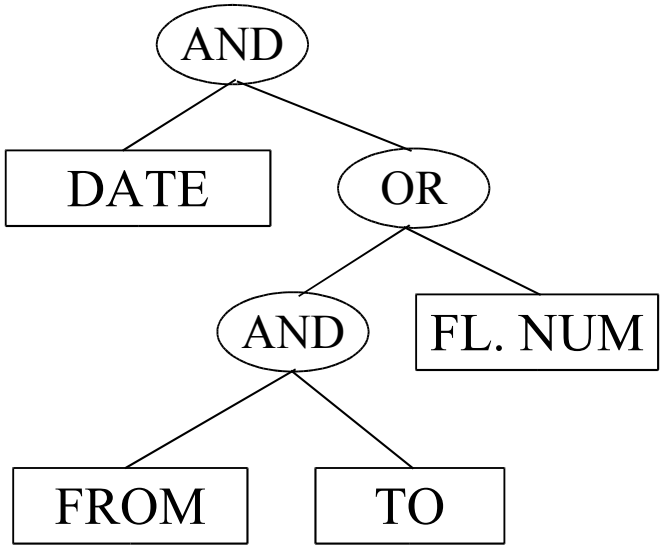
Frame as a Form

FROM
TO
WHEN

- Flat
- Suitable for simple domains
  - The flat structure can be compensated by richer relations among slot values and conditions for filling slots

Hierarchical frame

- Reflects internal structure of the task
- Example: Airplane ticket reservation domain



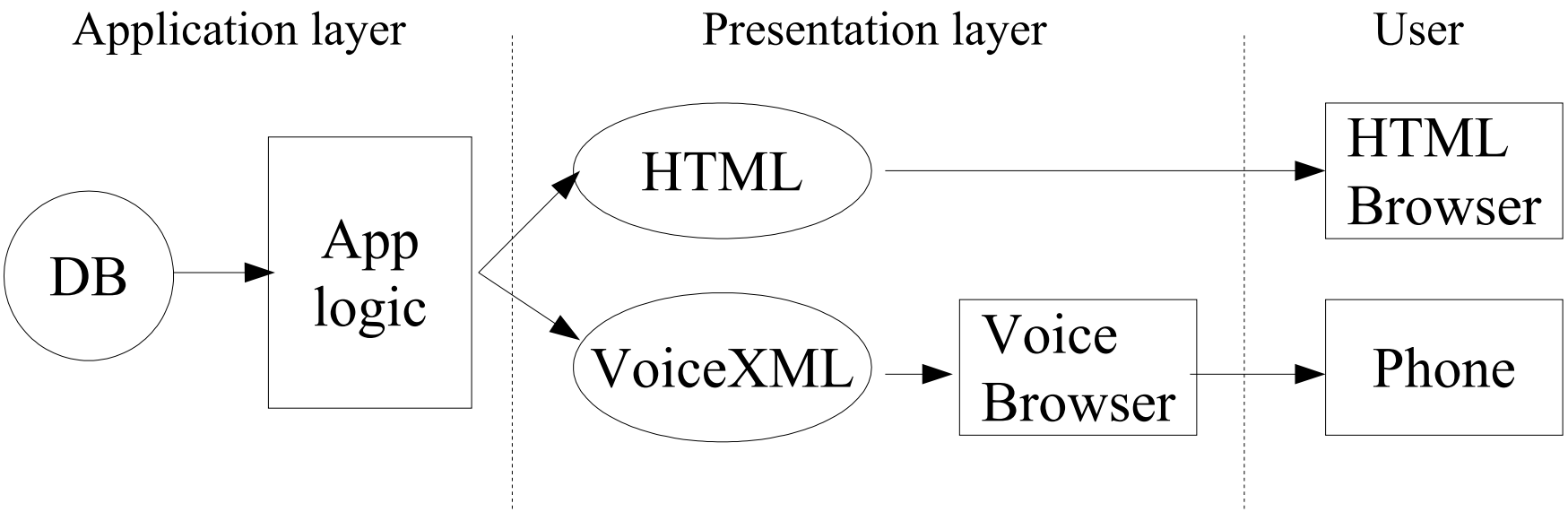
Agenda

- Dialogue system (DS)
- Frame-based DS
- **VoiceXML**

VoiceXML

- Markup language with XML syntax designed for creating audio dialogues featuring
  - Speech recognition and DTMF input
  - Recording of spoken input
  - Speech synthesis and digitized audio playback
  - Mixed initiative conversation
- Declarative with procedural parts
- W3C **Voice Browser** Activity
  - W3C Voice Browser Working Group
  - *Applying Web technology to enable users to access services from their telephone via a combination of speech and DTMF*

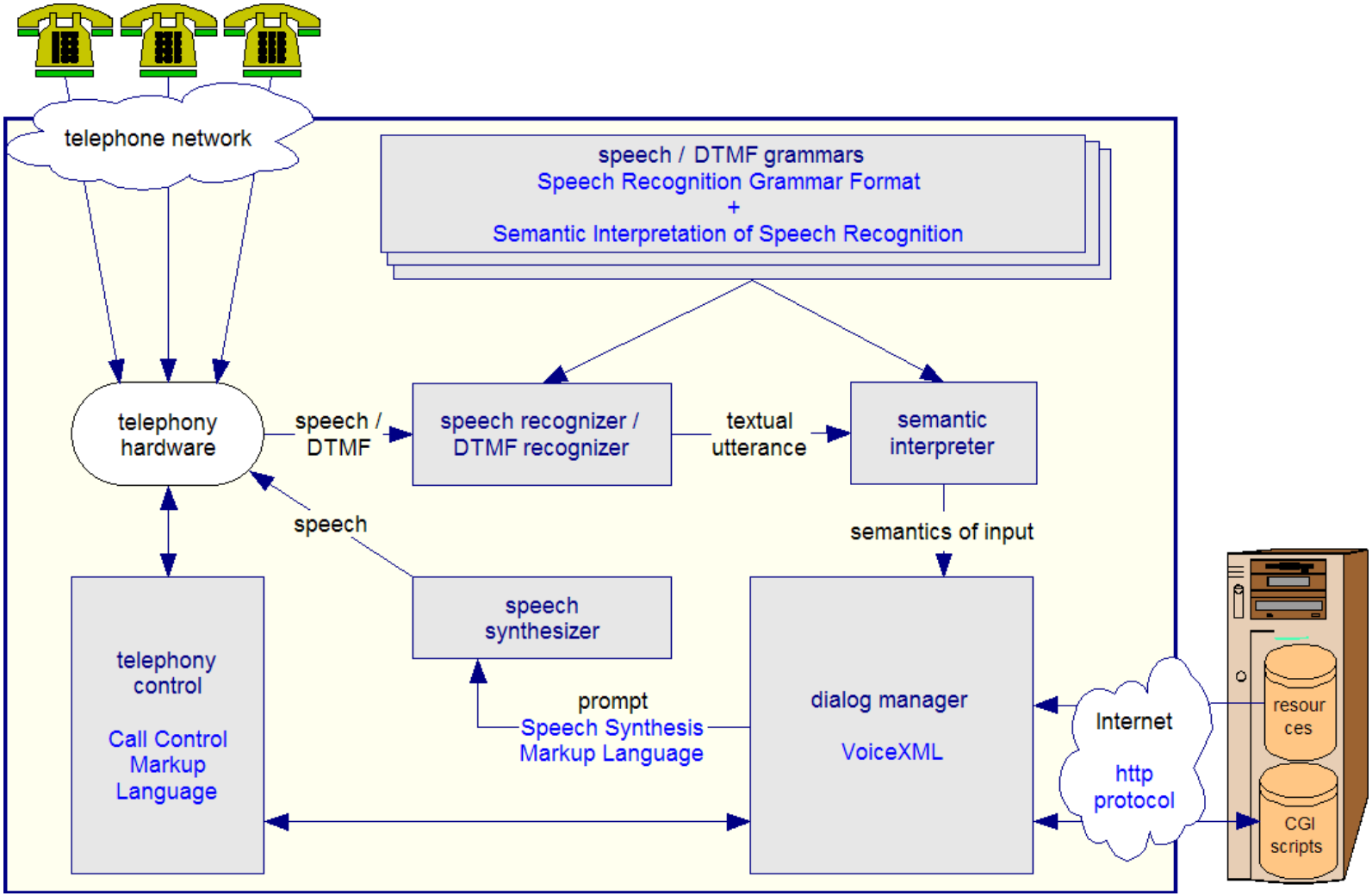
Application Architecture



VoiceXML (2)

- **W3C Speech Interface Framework**
  - **VoiceXML 2.0 (VXML)** – dialogue management
  - **Speech Recognition Grammar Specification (SRGS)** – grammar syntax
  - **Semantic Interpretation for SR (SISR)** – process of semantic interpretation of utterances
  - **Speech Synthesis Markup Language (SSML)** – description of the utterance surface realization
  - **Call Control XML (CCXML)** – telephony support
  - http://www.w3.org/Voice/

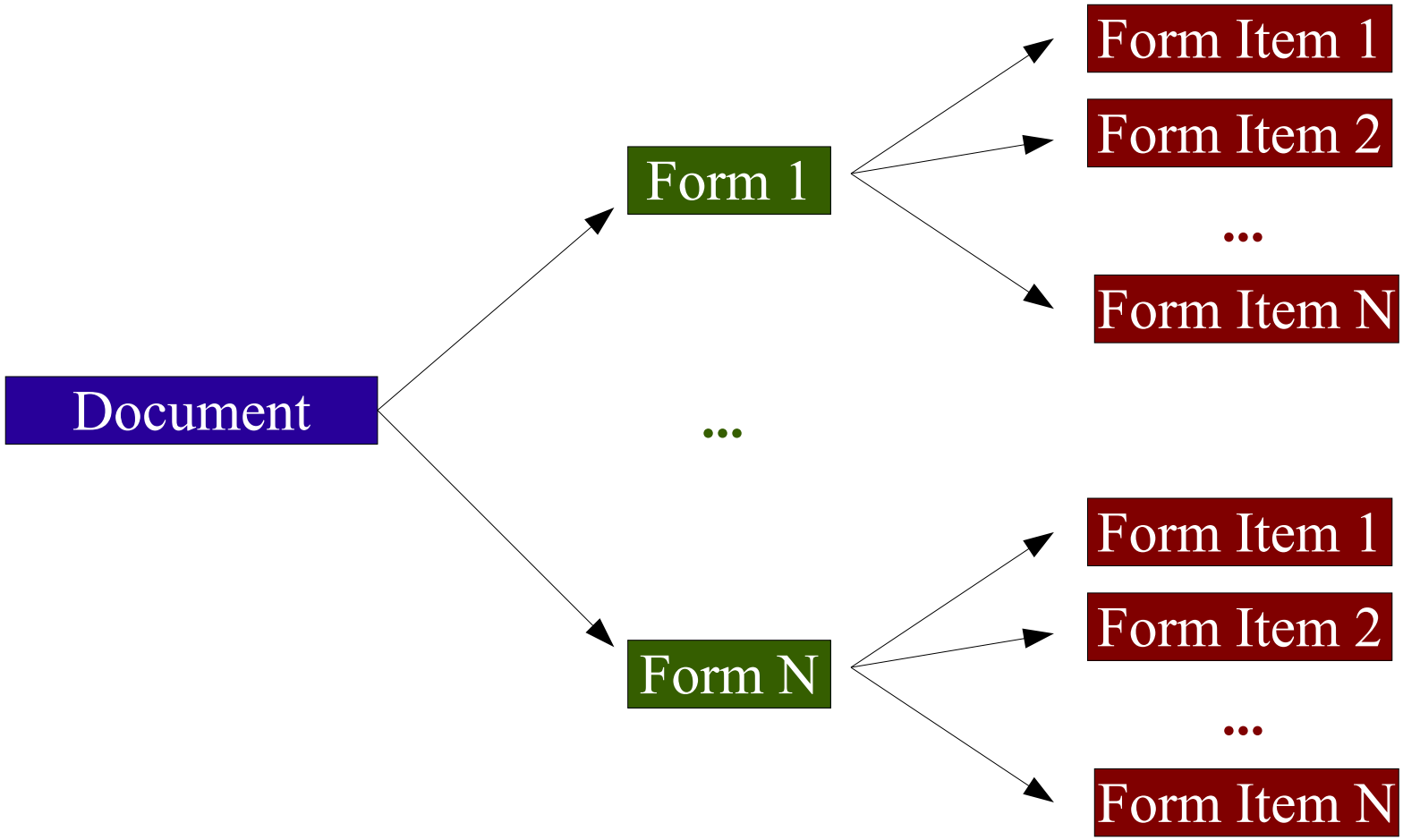
Voice Browser



VoiceXML Platform OptimTalk

- OptimSys http://www.optimsys.cz/
- Free for private and educational use and for non-profit research
- Extremely modular and extensible
- Can work on a desktop computer with microphone and speakers
- Special features for research

VoiceXML Document Structure



Form Items

- Form items
  - **<block>** contains executable code
  - **<field>** gathers input form the user
  - **<initial>** defines first step of a mixed-initiative conversation
  - **<subdialog>** calling “subrutine” (mechanism for reusing common dialogs)
  - **<record>** records user's input
  - **<object>** calls platform specific extensions
  - **<transfer>** transfers to another phone number

Variables and Scripts

- VoiceXML exploits ECMAScript
- Each variable in VoiceXML is an ECMAScript variable (types: undefined, null, bool, integer, double, string, object)
- Each expression in VoiceXML is an ECMAScript expression
- Variable is declared using the **<var>** tag (e.g. <var name="age" expr="20">)
- Variable can be later assigned value using the **<assign>** tag (e.g. <assign name="age" expr="age+1">)

Variables and Scripts (2)

- Each variable **must be declared** before it is used
- The **<script>** tag can contain or refer to some ECMAScript code
- Declaration of a variable in a script is equivalent to the declaration using the <var> tag

Form Interpretation Algorithm (FIA)

- **Applies to one form**, no implicit transition between forms is performed by the interpreter
- Each form item has an attribute **name** specifying name of a **variable that is declared by the interpreter and associated with the form item** (if the attribute is missing, a name is generated internally)
- At the beginning, the variable value is undefined (so called **unvisited form item**)
- FIA finds the first unvisited form item in document order and interprets it
- Interpretation of each form item leads under normal cirstumcances to filling in a value into its associated variable
- FIA ends when there are no more unvisited form items or if the interpretation is explicitly transited to another <form>

Form Item <block>

- Block of executable code
- Before <block> is interpreted, its associated variable is set to true to prevent it from being visited again
- Interpreting <block> means to execute code contained in the <block>, in particular
  - <goto> transition among form items/forms/documents
  - <if><elseif><else> conditional execution
  - <prompt> prompt said to the user
  - <assign>
  - <script>

Form Item <field>

- Gathers input form the user
- Contains specification of
  - Prompts that should be spoken to the users – tag **<prompt>**
  - Grammars that define set of acceptable user's utterances and their semantics – tag **<grammar>**
  - Actions that should be performed when the form item variable is filled (tag **<filled>**)

<prompt>

- Content of the tag is the SSML language
  - **<audio>** plays an audio file
  - **<emphasis>** emphasises its content
  - **<break>** places a break into the speech
  - **<voice>** sets voice parameters
  - **<prosody>** influences prosody of the speech

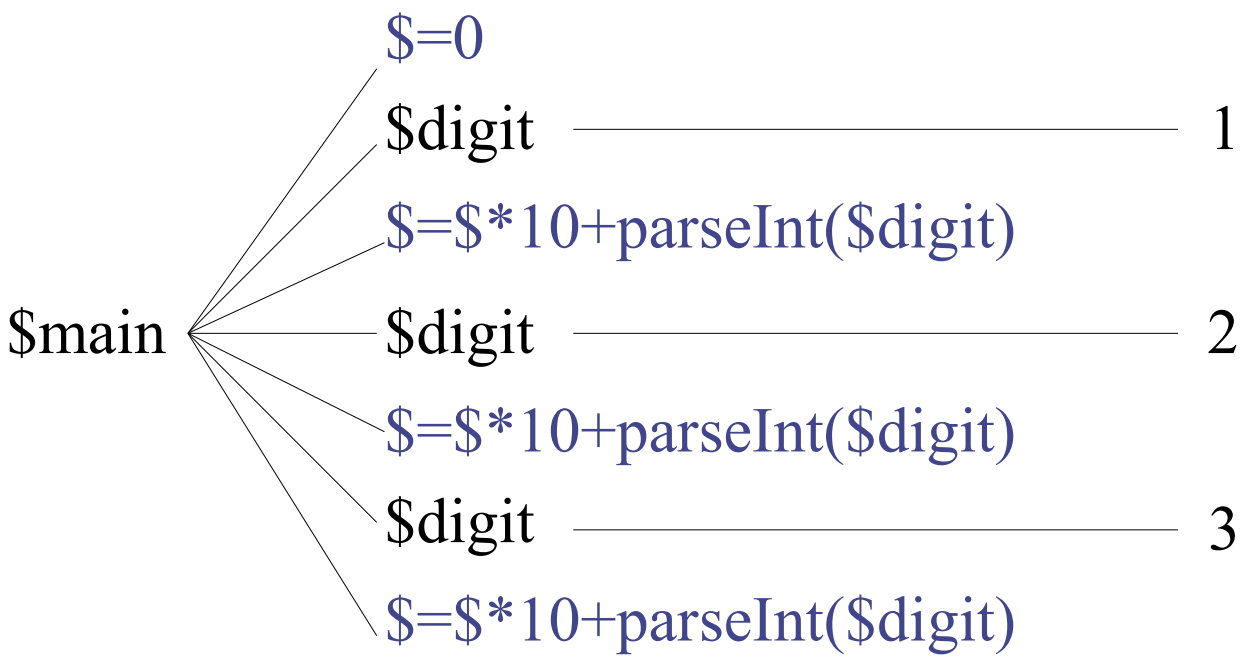
<grammar>

- Content of the tag is the SRGS language, content of the semantic tags is the SISR language
- SRGS has two equivalent forms – XML form and ABNF form
- If user's utterance matches the grammar, the corresponding parse tree is used for semantic interpretation

<grammar> (2)

- ```
  #ABNF 1.0 UTF-8;
  language en;  mode dtmf;  root $main;

  public $main = {$=0;} ($digit {$=$*10+parseInt($digit)})<1-10>;
         $digit = 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0;
  ```
- Utterance 123


  ```

<filled>

- Descendant of <form> or a form item
- Content of the <filled> tag is executable code
- After interpreting a form item, the interpreter iterates through all <filled> in document order and executes that ones which fire.
- There is no priority of form descendants and form item descendatns!
- <filled> can fire if a specified combination of form items has been filled (and at least one in the last dialogue step) or any of specified form items has been filled

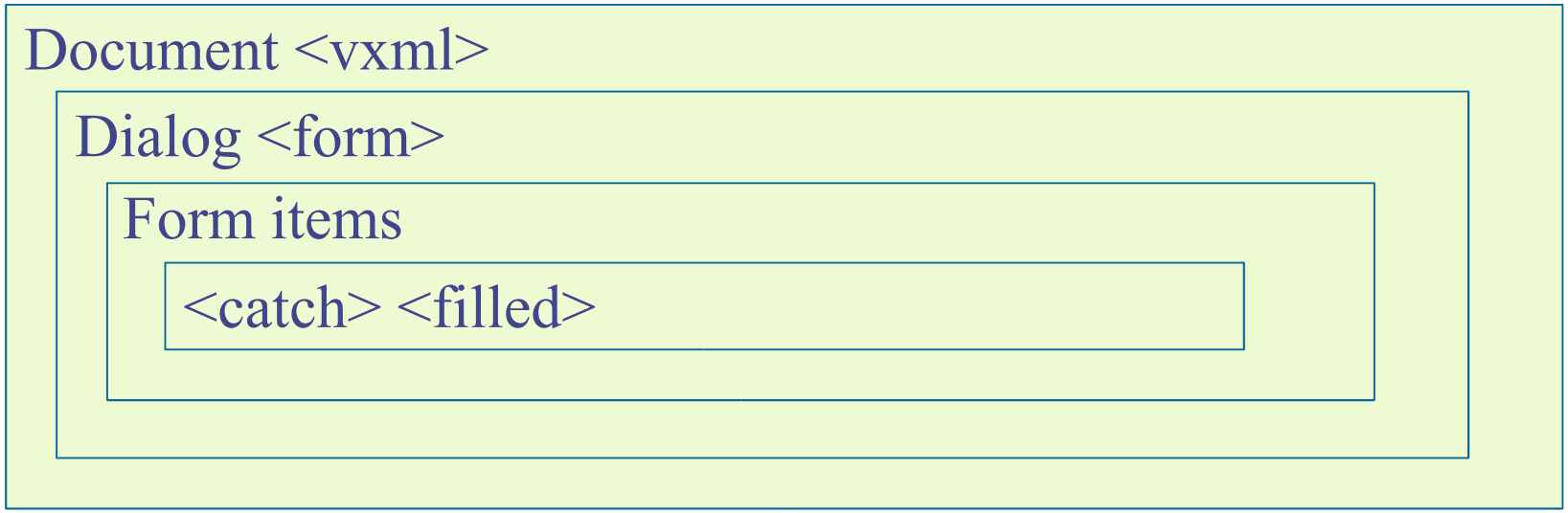
Events

- Events are named objects that are generated as reaction to the occurrence of a particular situation or condition
  - User does not respond (noinput)
  - User's response is not intelligible (nomatch)
  - An error occurred (error.*)
  - Explicitely thrown using the **<throw>** tag
- Events are caught by event handlers (<catch event="name">, <nomatch>, <noinput>, <error>, <help>)
- Event handlers contain executable code

Links

- The voice analogy to HTML hyperlinks
- <link> contains <grammar>s
- When a grammar contained in the link is matched, the action defined by the link is performed
- The action can be
  - Transition to a place in a document
  - Throwing an event with specified name

Scopes



- Scopes can contain in general
  - Grammars
  - Event handlers
  - Variables and scripts
- Not all the items are allowed in each of the scopes
- Grammars are matched and event handlers and variables searched from the most inner scope

Counters

- Each form item and each event has associated a counter
- The counters are reset when the form is entered and increased each time the prompt in the form item is spoken / the event is thrown
- This allows for tapered prompting and different reactions on repeatedly occuring events

Mixed Initiative Dialogues (Form Item <initial>)

- <initial> defines prompts and event handlers used for opening the dialogue
- When interpreted
  - Computer speaks the prompts (an open ended question)
  - User answers with an utterance. User has a higher freedom of utterance formulation and can fill multiple form items in a single step
  - Form-level (and higher level) grammars are active to match the utterance
  - Form-level <filled> sections triggered by a combination of filled form items are executed
- Values of unfilled form items are elicited in a directed dialogue in next iterations of the FIA

Thank you for your attention

Questions?