Interakce člověk–počítač v přirozeném jazyce (ICP)

LS 2013 — Dialogové systémy

Tino Haderlein

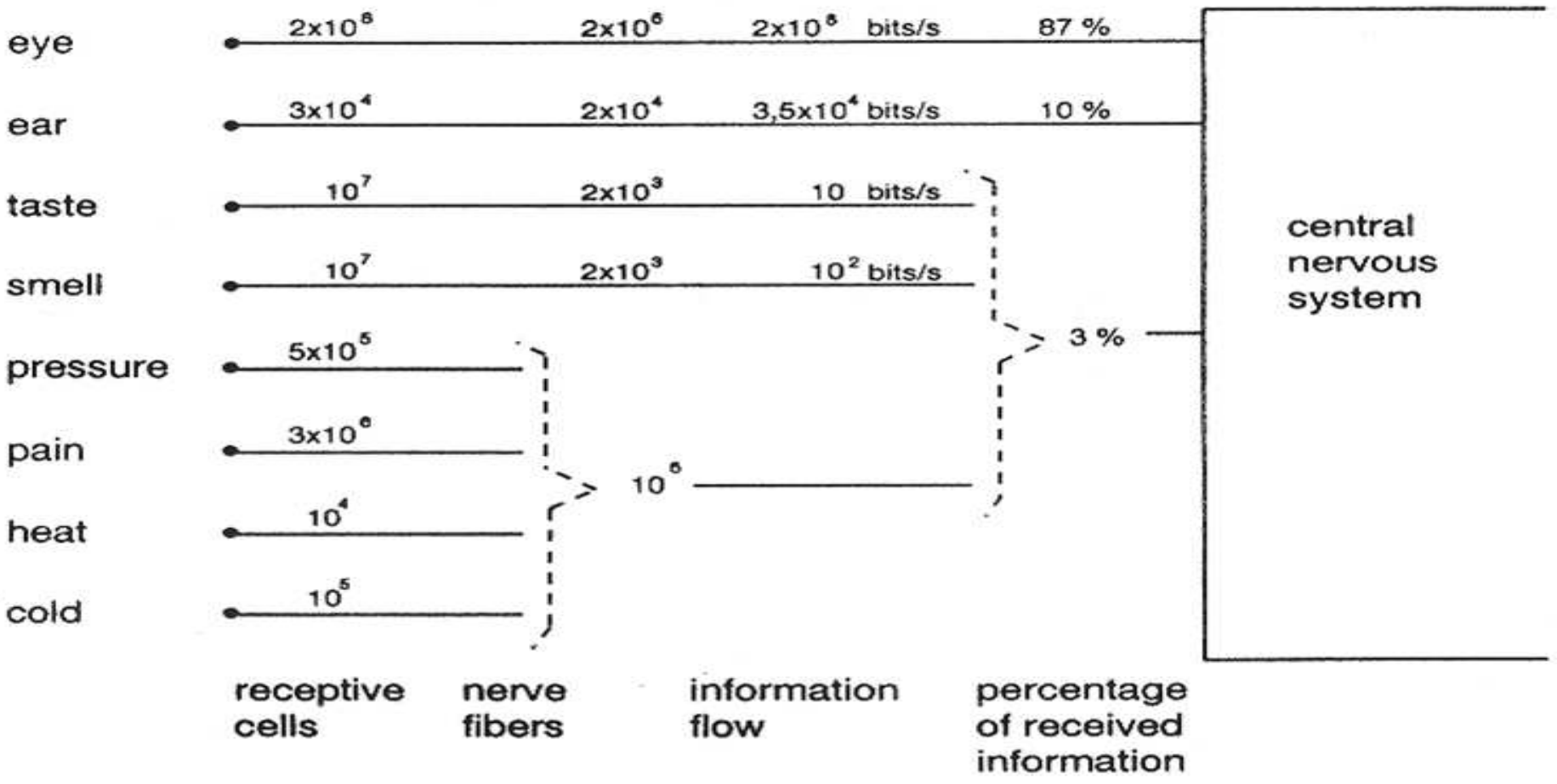
Katedra informatiky a výpočetní techniky (KIV)
Západočeská univerzita v Plzni

Dialogové systémy

systémy komunikující s uživatelem pomocí dialogu

- přirozenou řečí, i přes telefon (voice over IP, VoIP):
  - izolované
  - kontinuální
  - spontánní
- pomocí DTMF (dual-tone multi-frequency, tónová volba)
- psaným textem (přes klávesnici nebo rukopisně), snímaným textem (scanner)
- znakovou řečí
- mimikou
- nebo kombinací těchto možností (multimodální)

Přenos a užití informací různými smysly



Druhy přirozeně mluvících systémů

- systémy otázka – odpověď
- jednoduché technické konzultační systémy
- informační dialogové systémy (information retrieval dialogue systems)
- systémy s bází znalostí a expertní systémy
- jazykové portály

Příklady pro aplikace:

- dotaz předpovědi počasí přes telefon
- dialog s navigačním systémem v autě
- objednávka zboží přes telefon
- ...

Dialogové systémy

výhody:

- přirozená komunikace
- při multimodální analýze přístupné i nezkušeným nebo postiženým uživatelům
- zpracování více informací, např. i aktuálního stavu uživatele podle mimiky, řečové prosodie atd.
- přístupnost přes různé informační kanály

nevýhody:

- případné přetížení uživatele spoustou informací
- případná nedůvěra uživatele vůči stroji

Struktura informačního dialogového systému

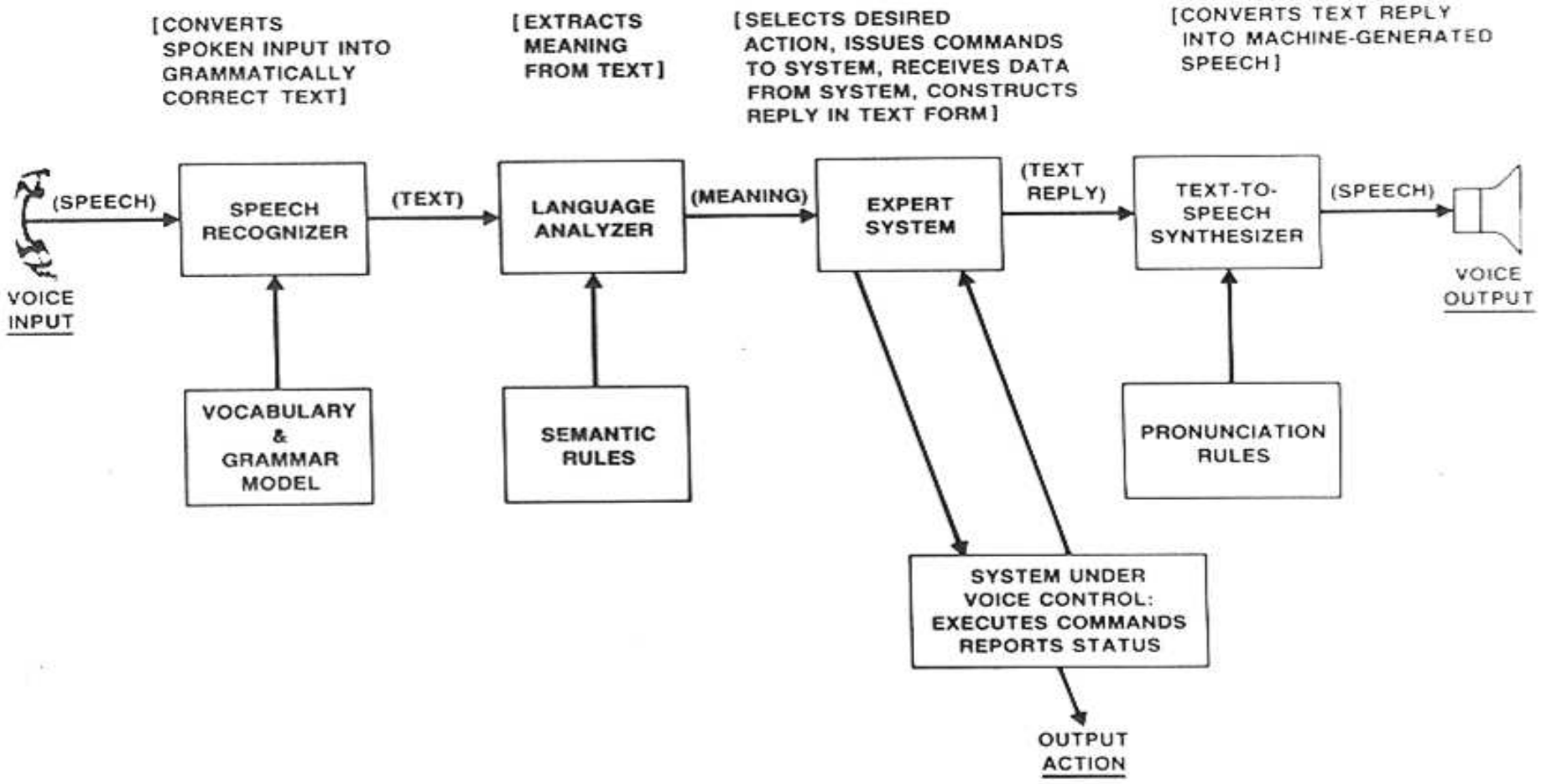
information retrieval dialogue system:

1. vstupní řetěz programových modulů
   - koncové zařízení (umožňuje uživateli komunikaci se systémem)
   - telefon (VoIP, DTMF), mikrofon atd.

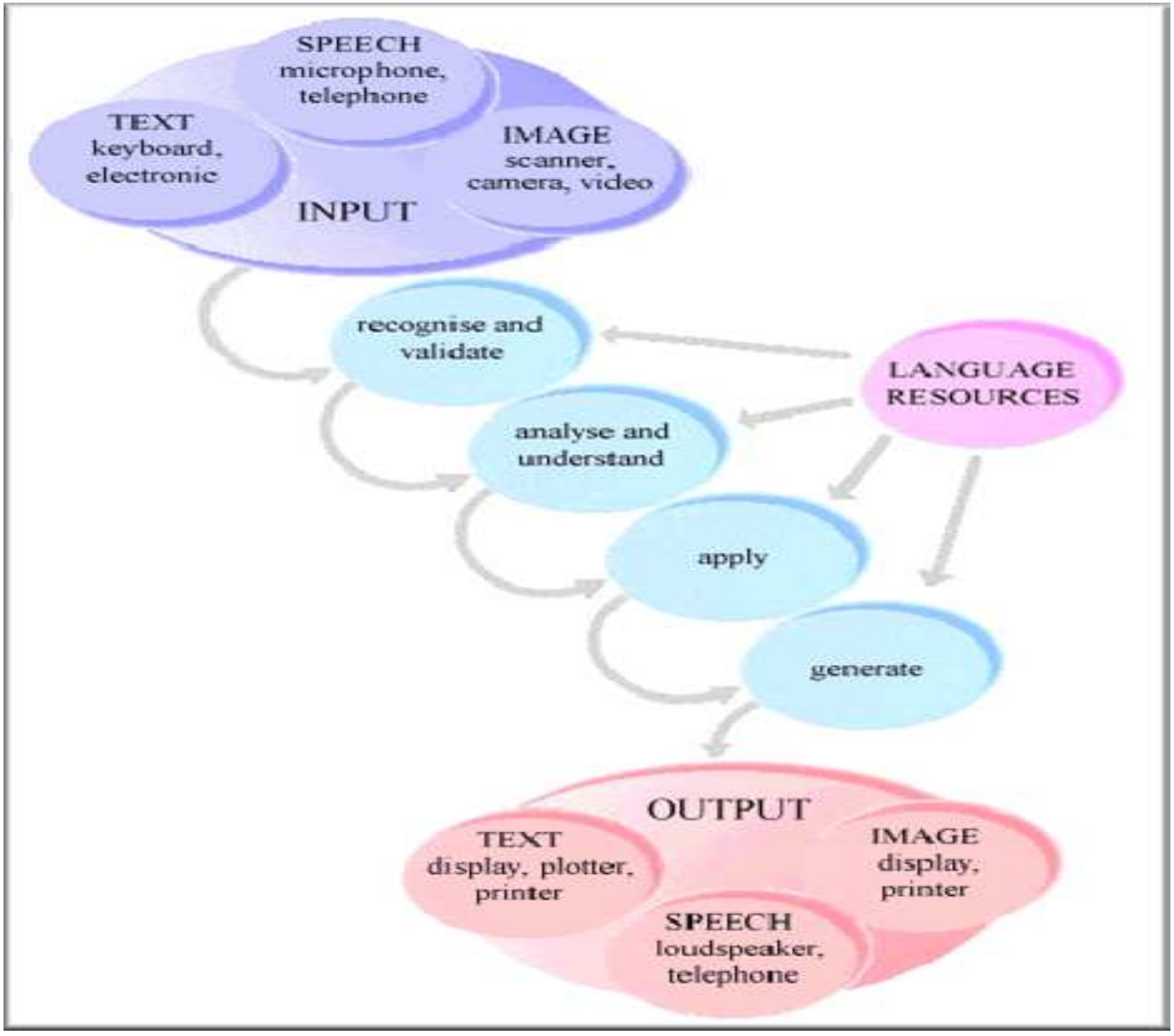
   rozpoznávač slov:
   - modul předzpracování signálu
   - modul akusticko-fonetické analýzy
   - modul rozpoznávání slov

   parser: modul lingvistické, sémantické analýzy
2. dialogový manažer
   - modul řízení dialogu a dotazů databáze
   - modul vytvoření odpovědí (prompts)
3. výstupní řetěz programových modulů
   - řečový syntetizér (**text-to-speech system**)
   - komunikační modul

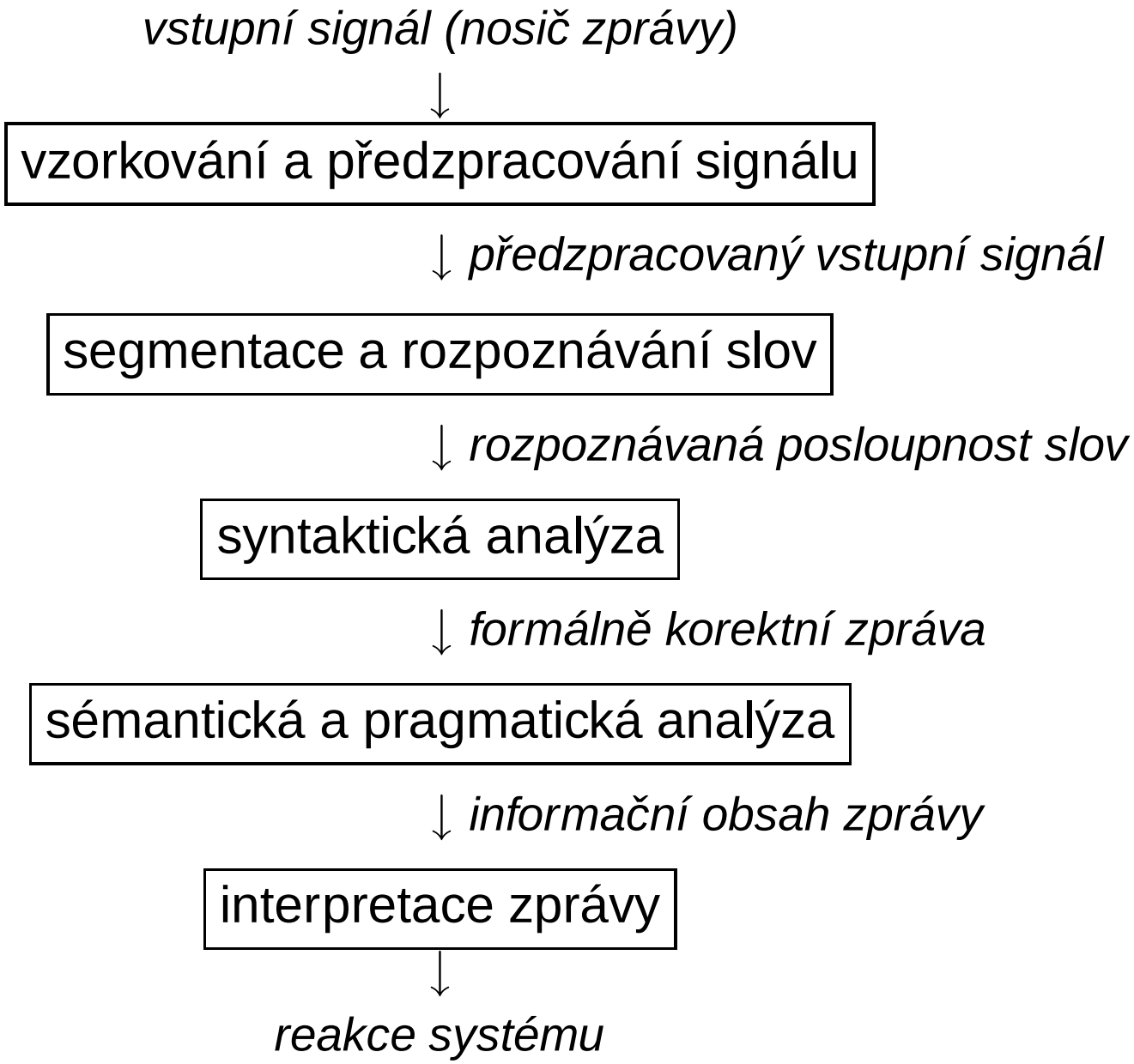
Struktura dialogového systému



Struktura dialogového systému



Zpracování vstupu v přirozeném jazyce



Definice

- diskurz: všeobecný rozhovor, běžně mezi vícero osobami; víckrát se překrývající promluvy, cocktail-party effect
- dialog: rozhovor dvou (nebo vícero) účastníků
- anglický pravopis:
  „dialogue“: užíván ve většině zemí, v USA jen v netechnickém kontextu
  „dialog“: užíván hlavně v USA v technickém kontextu
- promluva: souvislé sdělení, které učiní jeden účastník dialogu k druhému
- obrat: reakce (promluva) druhého účastníka
- dialogová strategie: postup, který k dané promluvě vytvoří následující promluvu

Počítačová lingvistika: jazykové procesy

explanandum	**explanans**
hlásky (tokens)	fonetika
skupiny hlásek/ekvivalenční třídy	
foném: nejmenší jednotka, která změní význam	
skupiny fonémů	fonologie
morfém: nejmenší jednotka, která nese význam	
alomorfy: morfémy s ekvivalentním významem	morfologie
skupiny morfémů: tvary slova	
(skloňované tvary jednoho slova)	
slova: ekvivalenční třídy tvarů slov	
fráze: skupiny slov	syntax
věty (úplné, gramatické)	
promluvy (logicky správné anebo ne)	sémantika/
řečové akty (měnící stav dialogu)	pragmatika

Lingvistické roviny při rozčlenění věty

- do posloupnosti písmen:
  `d-n-e-s-k-a-j-e-k-r-á-s-n-é-j-a-r-n-í-p-o-č-a-s-í`
- do posloupnosti slabik:
  `dnes - ka - je - krás - né - jar - ní - po - ča - sí`
- do posloupnosti morfémů:
  `dnes - ka - je - krás - né - jar - ní - po - čas - í`
- do posloupnosti tvarů slov:
  `dneska - je - krásné - jarní - počasí`
- do posloupnosti částí věty:
  `dneska - je - krásné jarní počasí`

Kooperativní dialogy

- dialog jako kooperativní akce: Rozhodující pro úspěch dialogu nejsou individuální intence jednotlivých dialogových partnerů, ale jejich společné intence.
- teorie her: Dialog je jako hra dvou účastníků.
- hodnotící funkce $E(L)$: přiřazuje každému dialogu L reálné číslo
- dialogová komunikace $M = (S_1, S_2, E_1, E_2)$ s dialogovými strategiemi S_i a hodnotícími funkcemi účastníka i
- $E_1 = E_2$: účastníci maji stejný cíl (kooperativní)
- $E_1 \neq E_2$: cíle účastníků se odlišují
- $E_1 = -E_2$: protichůdné cíle

Iniciativa v dialogu

- iniciativa uživatele *[příklad: informace o vlakových spojení]*:

  Uživatel: Pojede zítra ráno kolem osmi vlak z Plzně do Prahy?
  Systém: V 8:08 hodin pojede rychlík R 753 [...]

- iniciativa systému *[příklad: pro informace o pojištění auta]*:

  Systém: Zadejte své jméno a příjmení.
  Uživatelka: Káťa Jandová
  Systém: Zadejte SPZ svého auta.
  Uživatelka: ACD 09-57

- smíšená iniciativa *[příklad: výsledky fotbalových her]*:

  Uživatel: Máš výsledky posledních her?
  Systém: Chcete výsledky první anebo druhé ligy?
  Uživatel: První.
  Systém: V neděli 24.2.2013 hráli [...]

Zdroje znalostí dialogového systému

- stav dialogu (krok dialogu, předchozí vstupy a výstupy)
- lingvistické znalosti (gramatika, výskyt řečových segmentů atd.)
- model úlohy (reprezentace informací, které během dialogu hledáme)
- model uživatele:
  - statický: jazyk, věk, ...
  - dynamický: hlas, emoce, intence, ...
- model domény (oblasti) systému
- model znalostí o světě (např.: Únor je po lednu.)
- modelace znalostí: pro různé typy znalostí různě složité

Grounding

- Komunikace předpokládá, že účastníci mají množinu stejných znalostí.
- V dialogu vznikají nové společné znalosti $\rightarrow$ grounding.

Metody:

- positivní evidence: potvrzení („ano“, kývnutí), pozornost, iniciace příští promluvy, ...
- negativní evidence: otázky týkající se porozumění („Prosím?“, „Kdo?“), nevhodná parafráze (zopakování zprávy vlastními slovy)

Modelace dialogu

- sítě přechodů, např. Petriho sítě, konečné automaty
- regulární výrazy
- syntaktické diagramy
- gramatiky
- atributové gramatiky
- modely řízené událostmi
- HIT (hierarchic interaction graph templates)
- SDL (specification and description symbolic language)

Formální definice sítě přechodů

Dialogový model $D = (Q, T, intf, P, A, \delta)$, kde

- Q: konečná množina stavů
- T: konečná množina tzv. tokens (které uživatel může systému poslat pomocí interakčních objektů)
- $intf$: objekt propojovací jednotky aplikace (interface)
- P: konečná množina predikátů $P|intf \rightarrow \{true, false\}$
- A: konečná množina činností
- δ: přechodová funkce $Q \times T \rightarrow (P \times \{true, false\} \rightarrow A \times Q)$

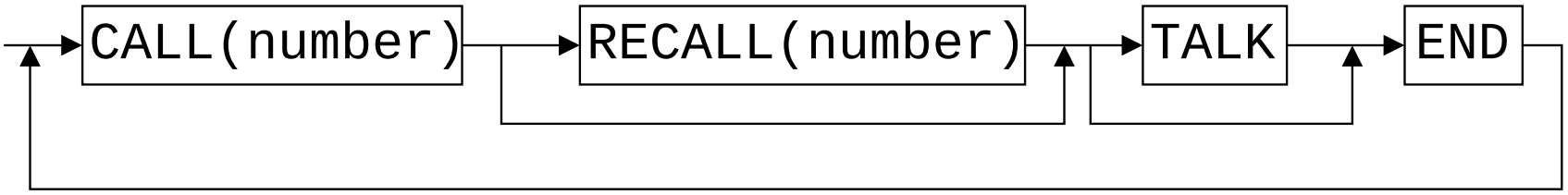
Regulární výrazy pro modelaci dialogů

příklad: telefonický hovor

```
(CALL(number) [RECALL(number)] [TALK] END)
```

Syntaktické diagramy

příklad: telefonický hovor



Gramatiky pro modelaci dialogů

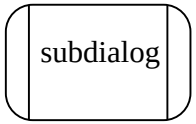
příklad: telefonický hovor

```
dialog      ::= phone-call dialog
phone-call  ::= call end
                | call talk end
                | call call end
                | call call talk end
call        ::= CALL(number)
talk        ::= TALK
end         ::= END
```

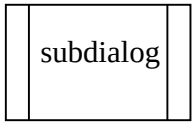
Modely řízené událostmi

- angl.: event-response systems
- nemají hierarchickou strukturu
- nemají modulový koncept
- dialog je reprezentován posloupností párů *(podmínka, činnost)*
- kdykoli je podmínka splněna $\rightarrow$ činnost
- podmínky mohou být splněny událostmi

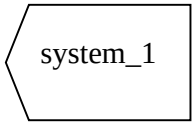
Notace SDL



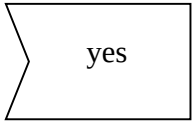
počáteční symbol



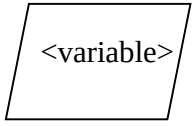
volání subdialogu



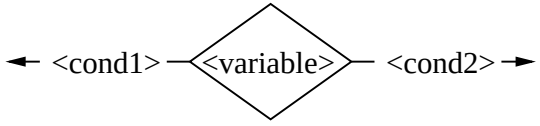
výstup systému; alternativy se číslují



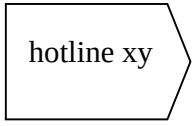
vstup uživatelem



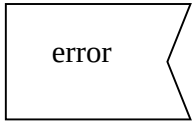
uložení proměnné



rozvětvení; tady závislé na hodnotě proměnné

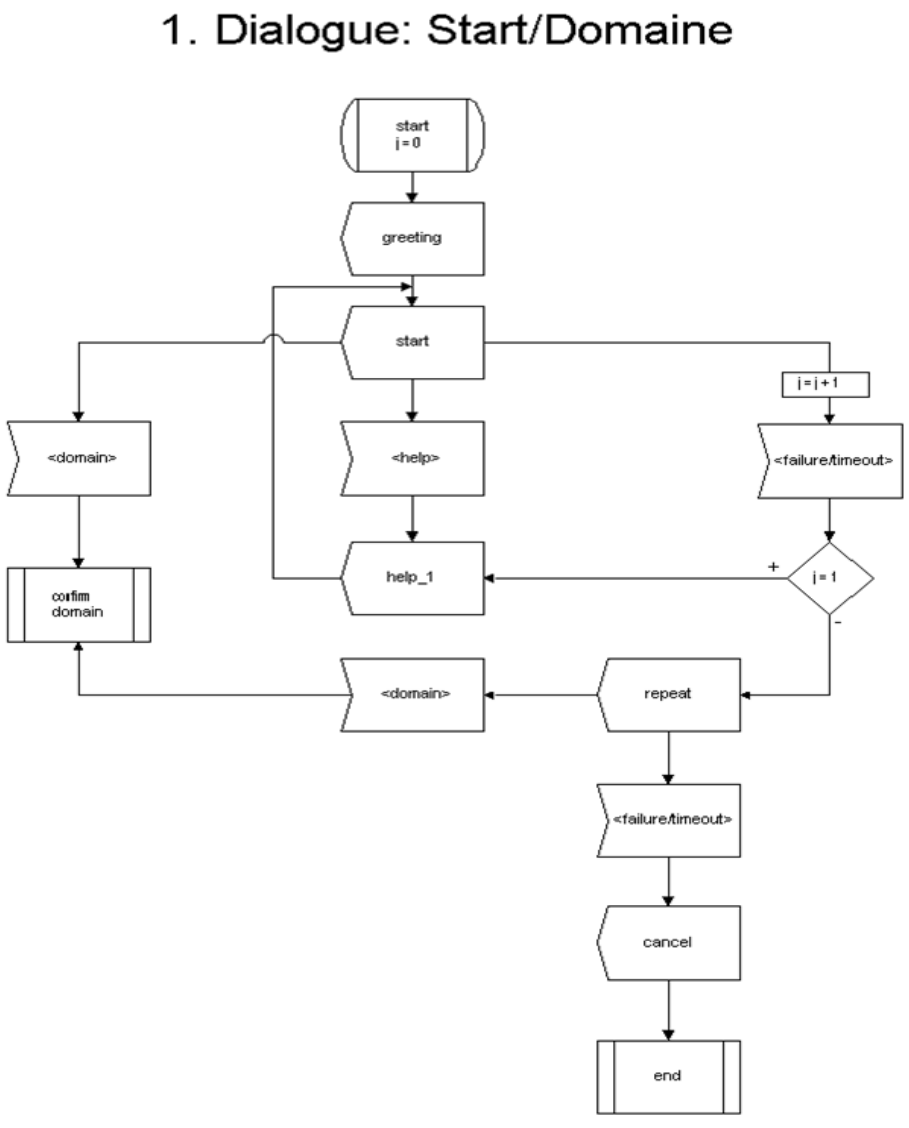


zpráva pro kontrolu systému, která přiřazuje dodané „hotline xy“ telefonní číslo

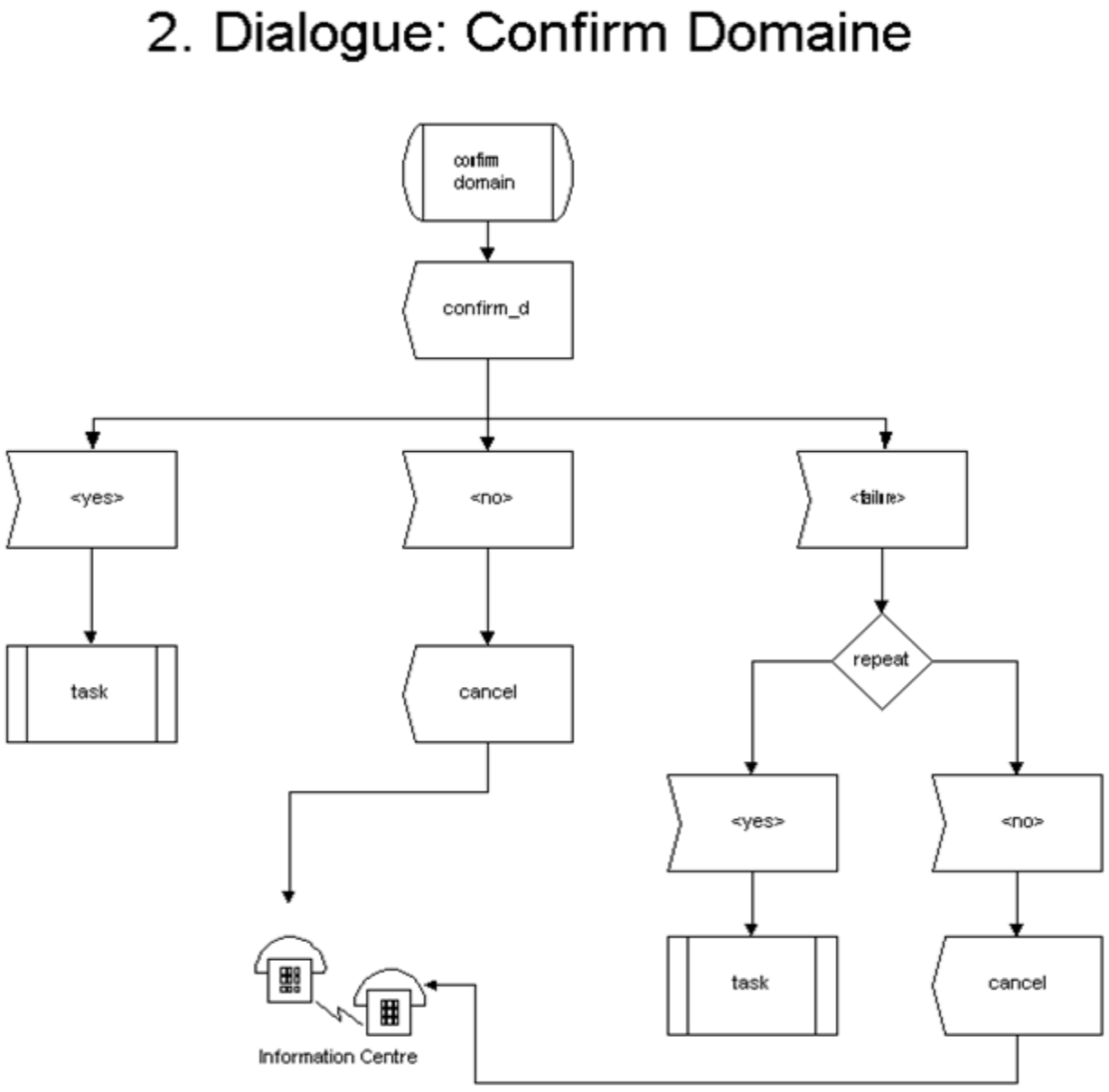


zpráva od systémové kontroly (když žádný operátor není k dispozici)

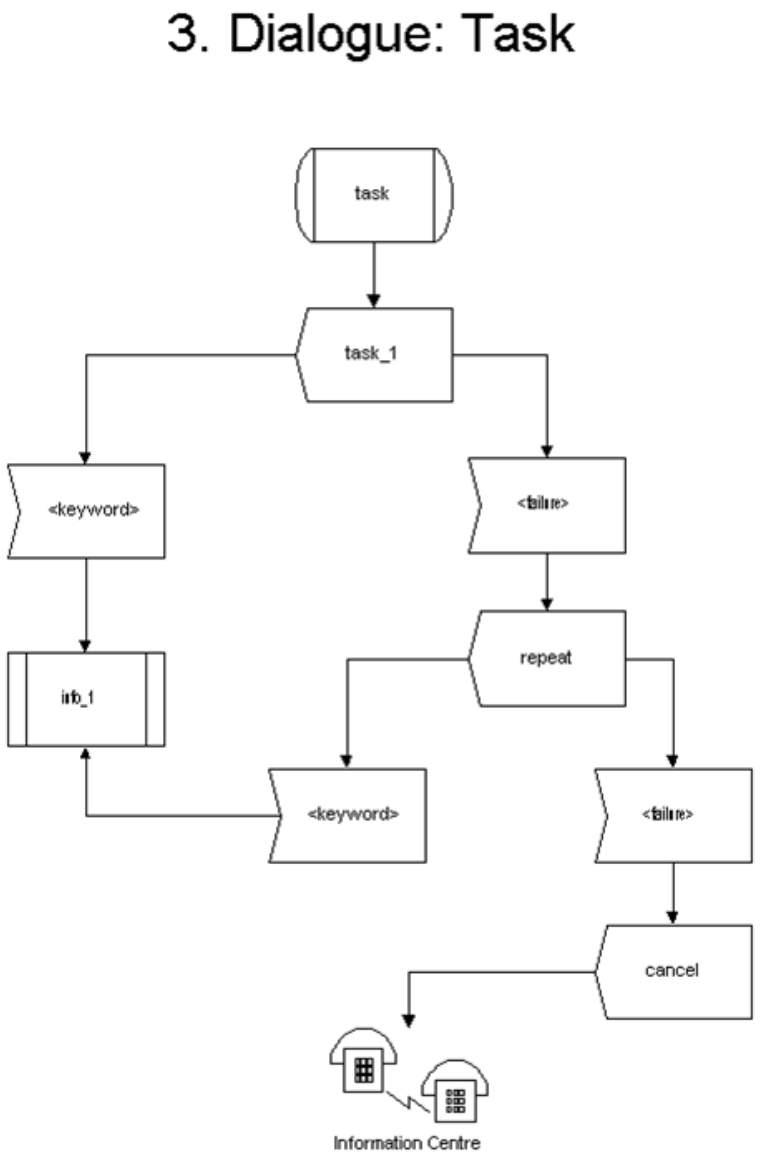
Příklad popisu dialogu v SDL



Příklad popisu dialogu v SDL



Příklad popisu dialogu v SDL



Otázky pro vytvoření dialogového modelu

- Co je účel dialogu?
- Kdo bude volat?
- Má řeč být formální a konzervativní anebo neformální?
- Má hlas pro výstupní promluvy znít mužsky, žensky, staře, mladě, vážně?
- Máme užívat znělky nebo hudbu během dialogu (zdůraznění důležitých dialogových kroků, čas na rozmyšlenou pro uživatele)?
- Zvuky a znělky mohou i vyvolat asociace, aby si uživatel lépe zapamatoval informace.

Finite state model dialogu

- Konečný automat (finite state machine, specializace sítě přechodů): výpočetní model s konečným počtem stavů; dialog přechází z jednoho stavu na další, až se dostane do výsledného stavu.
- Vstupy uživatele jsou zpracovávané krok za krokem pomocí pořadí, které je definované v modelu.
- Technická realizace je jednoduchá, proto je metoda často užívaná.
- různé typy, např. Mealeyho automat
- Nevýhody:
  - Odchylky od definovaného postupu dialogu nejsou možné.
  - Iniciativa uživatelem není plánovaná.
  - Odpovědi uživatele se často nehodí k modelu.

Dialog jako Mealeyho automat

- konečný automat s výstupem
- $M = (S, \Sigma, \Lambda, T, G, s_0)$, kde
  S: neprázdná, konečná množina stavů
  Σ: konečná vstupní abeceda
  Λ: konečná výstupní abeceda
  T: přechodová funkce ($T : S \times \Sigma \rightarrow S$)
  G: výstupní funkce ($G : S \times \Sigma \rightarrow \Lambda$)
  s_0: počáteční stav

Frame-based model dialogu

- Dialogový manažer zkusí vyplnit mezery (slots) v předloze (frame, template) vstupy uživatele.
- Pořadí je flexibilní.
- Příští krok dialogu je zvolen vzhledem k informacím, které ještě chybí.
- Vedení dialogu je přirozenější (flexibilita, doplnění vícero proměnných v jednom kroku).
- Kvůli flexibilitě je metoda často užívaná.
- Iniciativa uživatele je přesto omezená.

Pawlakův informační systém

- Zdzisław Pawlak (1926-2006), Politechnika Warszawska, Polská akademie věd
- informační systém: $S = < U, A >$, kde
  U: univerzum prvků (objektů)
  A: množina atributů
- V: množina hodnot atributů
- $g : U \times A \rightarrow V$
- popis vztahů mezi objekty a jejich atributy
- pro dialogové systémy užíváno pro hledání minimální množiny hodnot z A, které určují objekty z U
- aplikace: vyhledávání informací v relačních databázích

Dialog(ue) Control Tasks

MAIN DIALOGUE CONTROL TASKS

- initial greeting
- generation of a first question
- output of the question to the text-to-speech component
- passing control to speech recognition and speech understanding
- modifying these components according to the current system status
- processing of the results obtained from speech understanding
- combining this information with the current system status
- selection of the next question based on the new system status
- generation of the exact form of this question
- passing control to speech recognition and speech understanding
- processing of the results obtained from speech understanding
- verification of previously obtained results
- access to the database
- output of query results, possibly after converting them into a suitable form
- dealing with special situations during the dialogue, like errors from the database, repetition of a question, repetition of results
- monitoring of the dialogue progress
- closing ceremony

Logický průběh dialogu

- Důležité: Každá promluva uživatele musí dostat odpovídající reakci systému (prompt).
- To znamená, že systém musí být připraven na všechno, co by člověk mohl říct.
- Uživatel může např. říct „Servis.“ anebo „Někoho ze servisu, prosím.“ anebo „Spojte mě prosím s někým ze servisu.“
- Důležitý pro uživatele je přesný návod, **co** může být řečeno **kdy**.
- Struktura jazykového portálu musí být přehledná.
- Systém musí vést uživatele přes dialog.
- Dialog s dialogovým systémem má být co nejpodobnější přirozenému rozhovoru.
- Systém nesmí dát uživateli příliš hodně informací najednou.
- V každém kroku dialogu má systém maximálně nabídnout čtyři alternativy.
- Systém musí zohledňovat věk a úroveň vzdělání uživatele.

Nejdůležitější aspekt řízení dialogu

Potvrzení správnosti vstupních dat
různé dialogové strategie:

- no-confirm strategy (systém nepotvrzuje nic)
- separate-confirmation strategy (každý údaj je potvrzen jednotlivě)
- confirm-alone strategy (všechny údaje jsou potvrzeny dohromady)
- confirm-plus-initiative strategy (potvrzení údajú systémem a iniciativa na zisk dalších dat)

Každou strategii musí řídit dialogový manažer.

Přirozenost dialogu

- Když je dialog příliš přirozený, je možné, že si uživatel už není vědom, že mluví se strojem.
- Pak se může stát, že uživatel může mluvit nespisovně a že systém je přetížený.
- Měla by být vždy možnost kontaktovat pracovníka/pracovnici, aby uživatel mohl mluvit přímo s někým z firmy.
- Reakce systému (prompts) mají být krátké:
  - „Promiňte, já jsem vašemu vstupu nerozuměl. Mohl(a) byste prosím váš dotaz zopakovat?“ *(příliš dlouhé)*
  - „Promiňte, nerozuměl jsem. Prosím zopakujte.“ *(kratší a přirozenější)*
  - „Cože?“ *(ještě kratší, ale suboptimální z jiných důvodů)*
- Uživatel by měl mít možnost dostat pomoc (návod k obsluze).
- Před uvedením do provozu by výrobce měl provést různé testy, např. s pracovníky, kteří se systémem ještě nemají zkušenost.

Přirozenost dialogu – ELIZA

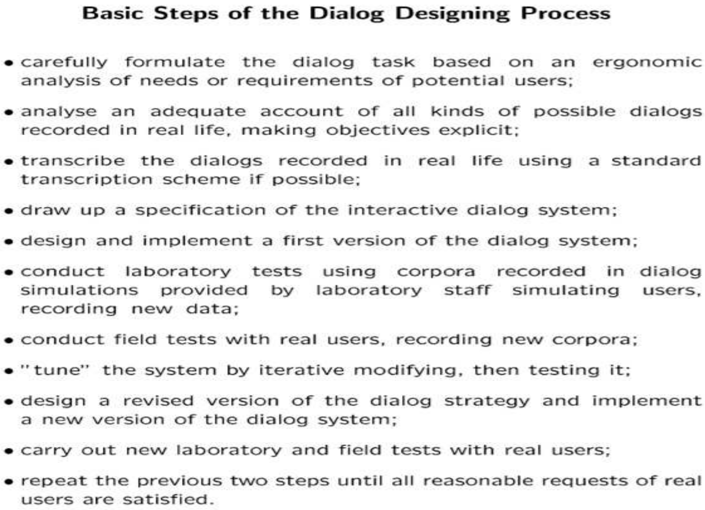
- Joseph Weizenbaum (1923-2008), MIT
- ELIZA: důkaz, že program, který vede dialog, nepotřebuje inteligenci
- mj. simulace „psychoterapeuta“
- skládá odpověď z klíčových slov a kopírovaných úseků vstupu; příklad:

Uživatel: Jsem pořád smutný.
ELIZA: Proč říkáte, že jste pořád smutný?

anebo

Uživatel: Mám problém s otcem. *[otec: klíčové slovo]*
ELIZA: Povídejte mi víc o své rodině.

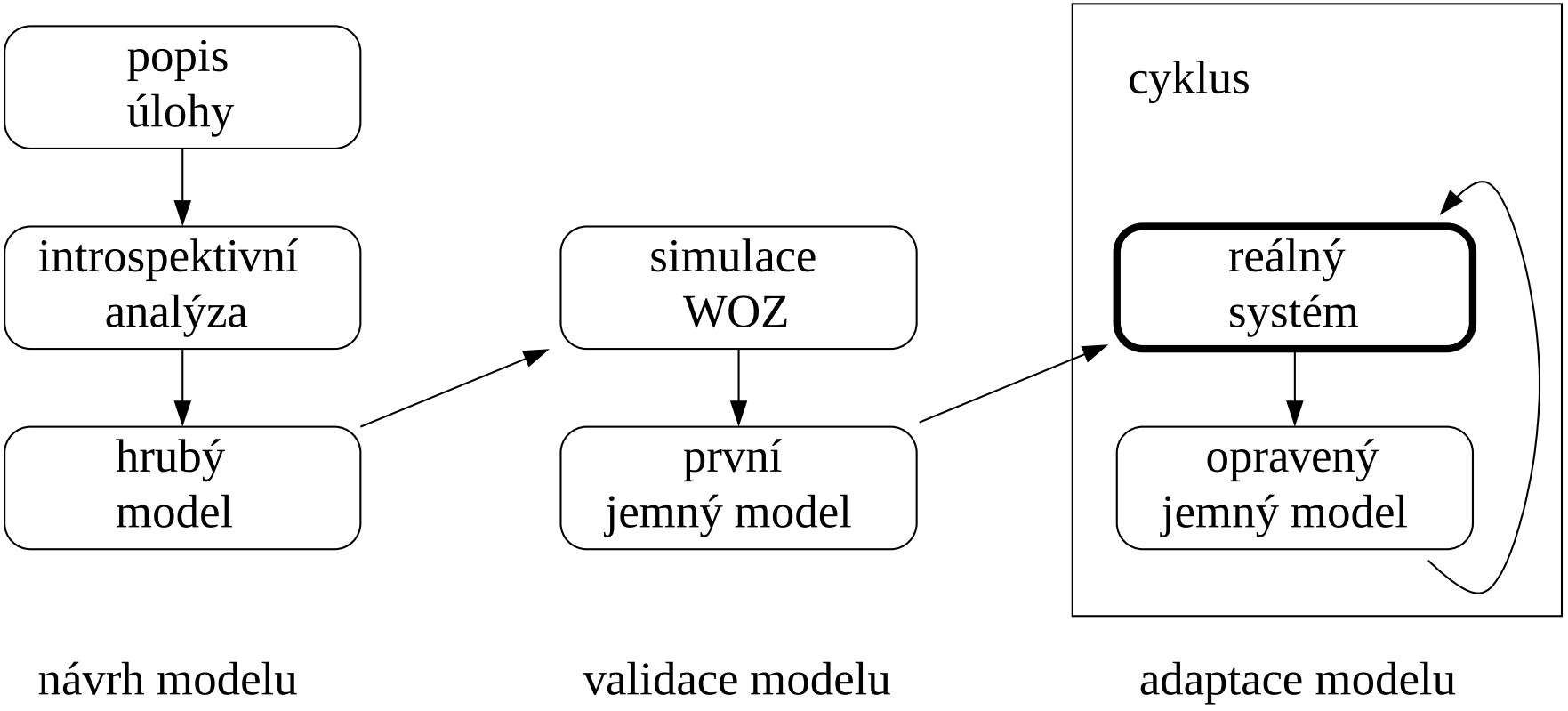
Dialog Design



Basic Steps of the Dialog Designing Process

- carefully formulate the dialog task based on an ergonomic analysis of needs or requirements of potential users;
- analyse an adequate account of all kinds of possible dialogs recorded in real life, making objectives explicit;
- transcribe the dialogs recorded in real life using a standard transcription scheme if possible;
- draw up a specification of the interactive dialog system;
- design and implement a first version of the dialog system;
- conduct laboratory tests using corpora recorded in dialog simulations provided by laboratory staff simulating users, recording new data;
- conduct field tests with real users, recording new corpora;
- "tune" the system by iterative modifying, then testing it;
- design a revised version of the dialog strategy and implement a new version of the dialog system;
- carry out new laboratory and field tests with real users;
- repeat the previous two steps until all reasonable requests of real users are satisfied.

Návrh dialogů a modelu



Simulace typu Wizard-of-Oz (WOZ)

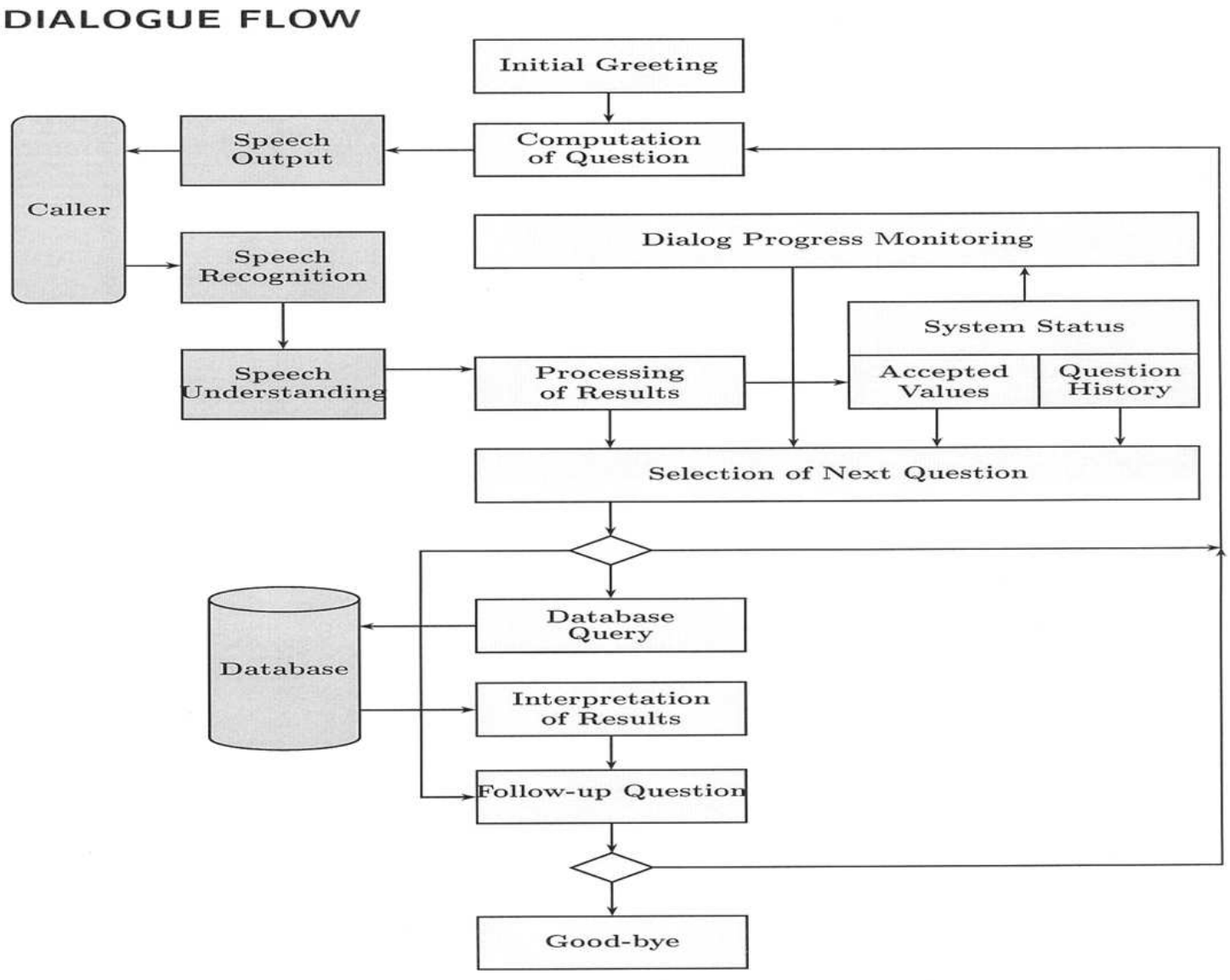
pro získání realistických dat pro modelaci dialogů:
simulace chování interaktivního automatu člověkem

- mluvením s uživatelem syntetickým anebo změněným hlasem
- výběrem a puštěním předem definovaných odpovědí
- ruční modifikací některých parametrů simulačního systému
- užíváním osoby pro simulaci integrace existujících komponent

Dialogové strategie

- Pro kontrolu správnosti vstupních dat by je systém měl zopakovat a umožnit uživateli korektury.
- Uživatel by měl mít možnost přerušit systém
  - pro zrychlení přechodu na příští dialogový krok,
  - pro žádost o opakování promluv,
  - pro korekturu nesprávně zpracovaných údajů,
  - pro opuštění dialogového kroku, dialogu nebo programu
  - pro žádost o pomoc (návod k obsluze).
- Dialogová strategie by se měla přizpůsobit uživateli.

Dialogový tok



Struktura dialogového manažeru

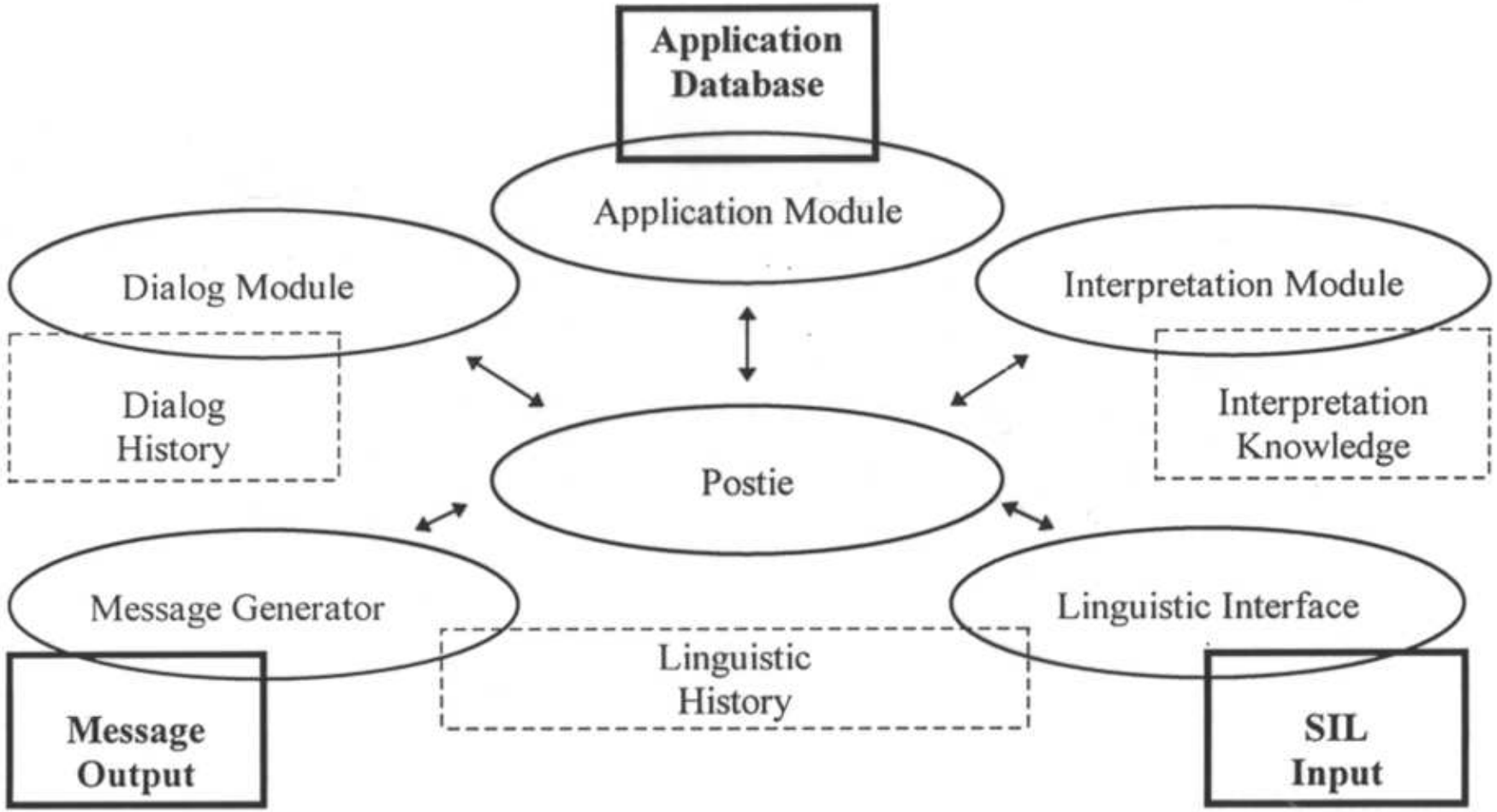
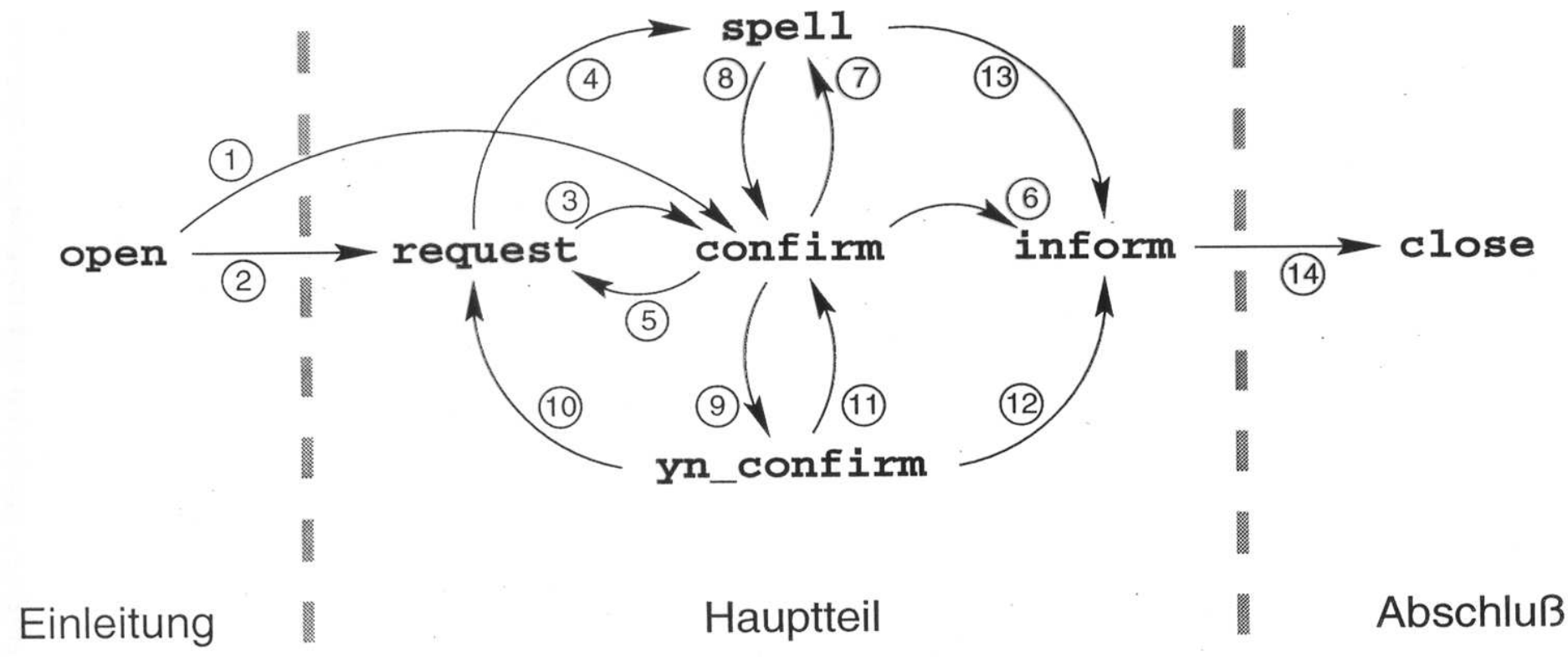


Figure 1: Modular structure of the Dialog Manager

SIL = Semantic Interface Language

Automat dialogových strategií



úvod

hlavní část

závěr

Modelace dialogové strategie v praxi

- procedurální/objektově orientované programovací jazyky a příslušné knihovny, např. C, C++, Java
- logické programovací jazyky, např. Prolog
- VoiceXML (otevřený standard)
- CallXML (otevřený standard)
- ...

Možné zdroje chyb systému

- periferie: telefon, přenosový protokol
- rozpoznávání řeči
- sémantická analýza
- dialogová strategie
- generátor promluv systému
- řečový syntetizér

Typické chyby uživatele

- Uživatel přidá víc informace než je požadováno:
  systém: V kolik hodin chcete jít do kina?
  uživatel: Zítra večer v šest hodin do kina Cinestar na film Já, robot.
- Uživatel odpoví na jinou otázku:
  systém: V kolik hodin chcete jít do kina?
  uživatel: Do kina Cinestar na film Já, robot.
- Uživatel přidá příliš málo informace:
  systém: V kolik hodin chcete jít do kina?
  uživatel: Zítra.
- Uživatel ignoruje informace:
  systém: Tenhle film zítra nedávají.
  uživatel: Chci ho ale přesto vidět.
- Uživatel opustí doménu dialogu:
  systém: V kolik hodin chcete jít do kina?
  uživatel: Přítelkyně má narozeniny, proto chci koupit pár vstupenek.

Vedení dialogu s (dobrým) dialogovým systémem

Systém: Dobrý den, tady je Fränki firmy Sympalog, informační systém o kinech ve středních Frankách... ***[barge in]***

Uživatel: Chci vidět film Já, robot.

Systém: Chcete vidět film Já, robot. Kde?

Uživatel: **Ne, myslel jsem Noční hlídka**, film Noční hlídka, **dneska večer**. *Korektury jsou vždycky možné. Systém umí rozlišovat nepřímé údaje. Systém umí zpracovat dodatečné údaje.*

Systém: Chcete vidět film Noční hlídka dneska mezi 18 a 22 hodin. Kde?

Uživatel: V Erlangenu.

Systém: Film Noční hlídka dávají v kině Manhattan v 20 hodin.

Uživatel: Dávají **ho** i později? *Systém si pamatuje hodnoty parametrů.*

Systém: **Film Noční hlídka** nedávají v kině Manhattan v Erlangenu po 20 hodin.

Kroky syntézy řeči (TTS)

- analýza struktury
  - detekce odstavců, vět, ...
  - příklad: tečka v „Ústí n.L.“ není konec věty
- předzpracování textu
  - zpracování zkratek, zkratkových slov (acronyms), údajů dat, časů, čísel atd.
  - příklad: „Zákon č. 309/1999 Sb.“, „my_addr@provider.com“ $\longrightarrow$ text, ktery má být mluven
- převod text$\rightarrow$foném
- analýza prosodie
- vytvoření řečového zvuku

Metody syntézy řeči (TTS)

- nutné: fonetický přepis textu
- syntéza ve frekvenční oblasti:
  - simuluje chování artikulačních orgánů
  - volba parametrů: základní frekvence (znělé hlásky) popř. generátor šumu (neznělé hlásky), harmonické frekvence, intenzita, ...
- syntéza v časové oblasti:
  - spojování řečových segmentů
  - často užívané segmenty: difony, trifony, ale i alofony, slabičné segmenty, ...
- postprocessing: doplění intonace, přízvuku, ...

Syntéza slova „ahoj“ v časové oblasti

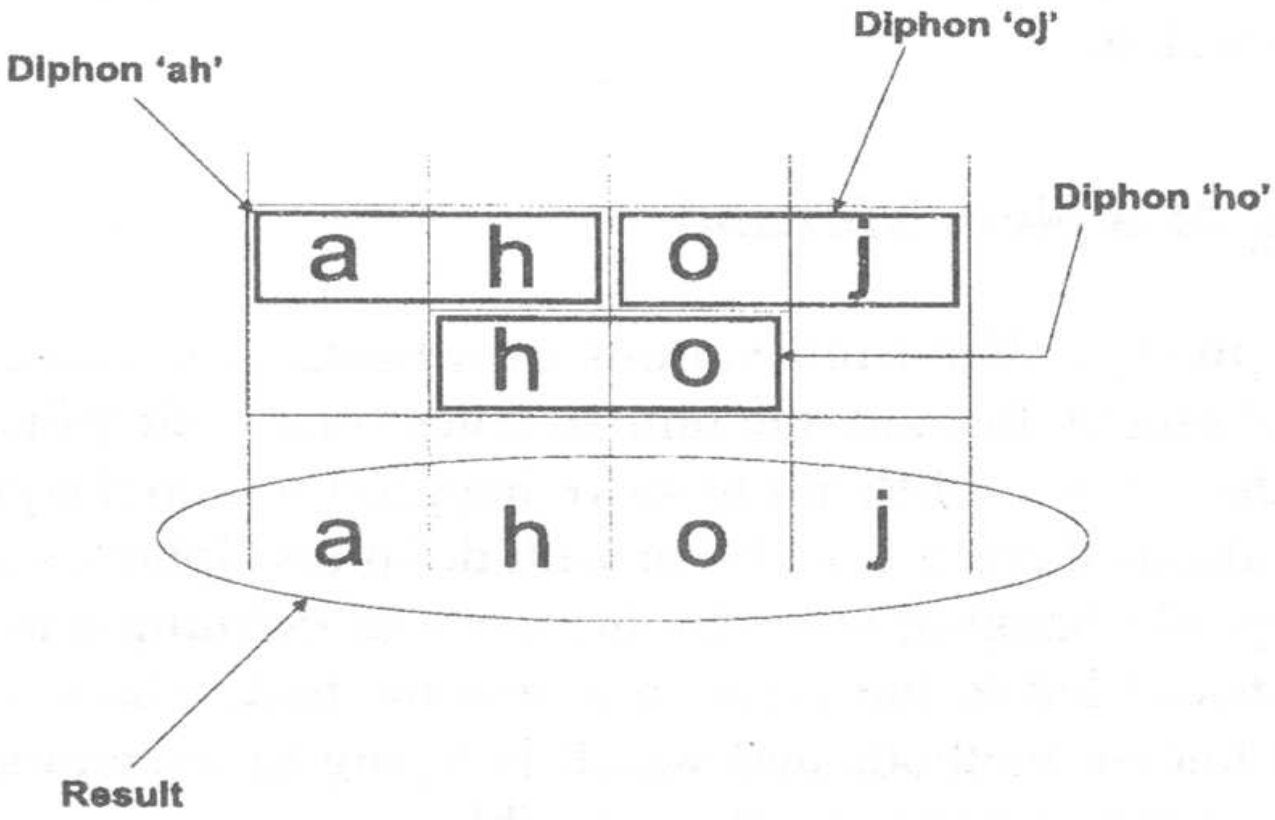


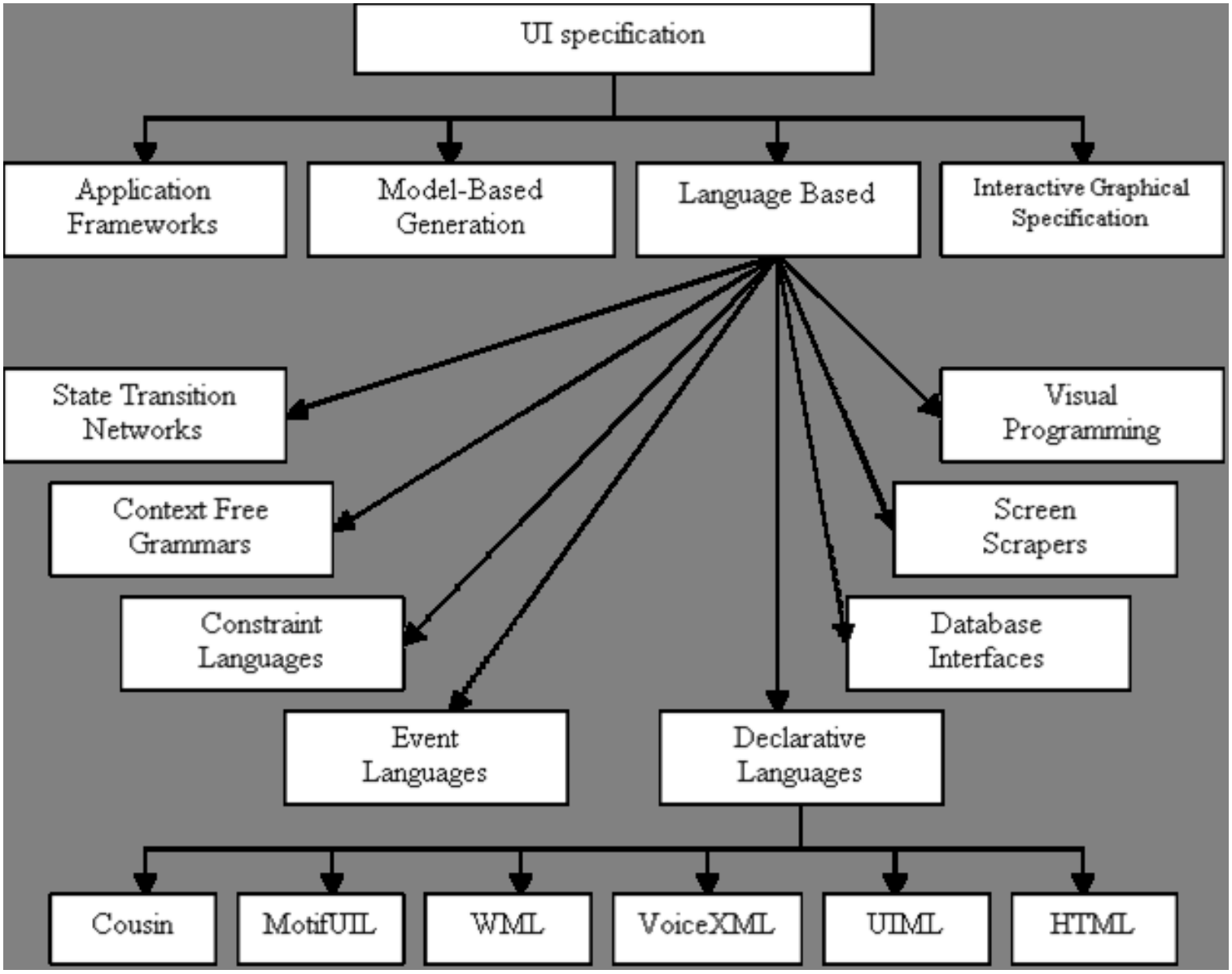
Fig. 1. Synthesis of word "ahoj".

Poznámka: Podle definice této přednášky difon obsahuje jen půlky dvou sousedících fónů; bifon obsahuje celé fóny.

VoiceXML

- standardní XML formát pro specifikaci interakčních dialogů mezi člověkem a počítačem
- HTML: interpretován vizuálním webovým prohlížečem
  VoiceXML: interpretován prohlížečem hlasu
- základní prvky: „tags“ pro příkazy o syntéze řeči, rozpoznávání řeči, dialogovém managementu a pouštění nahrávek
- aktuální verze: 2.1 (2007)

Programovací jazyky



WWW Consortium: Speech Interface Framework

defining markup languages for applications supporting speech input and output

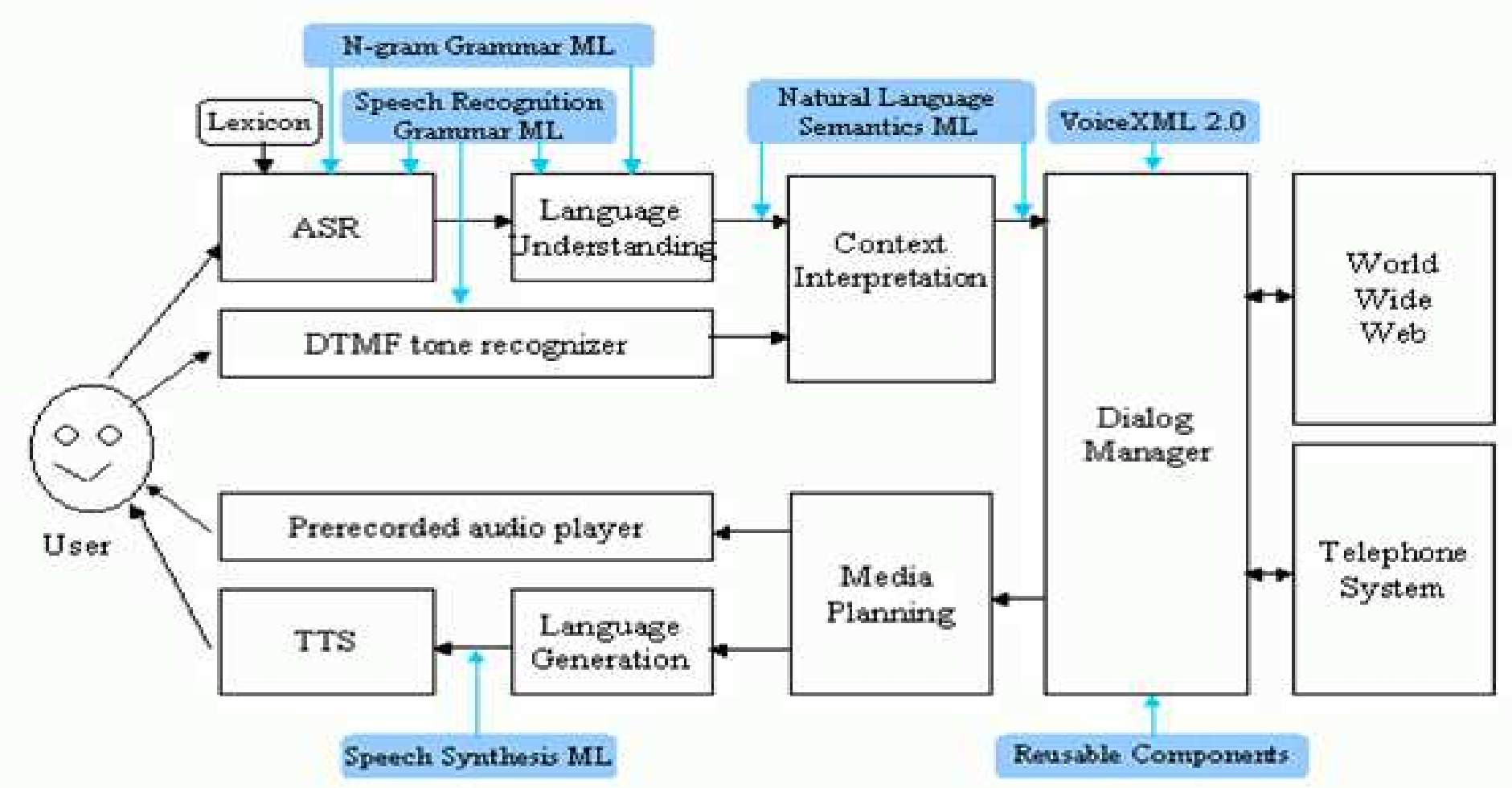


Figure 1. W3C Speech Interface Framework

Nejjednodušší VoiceXML program

```
<?xml version="1.0"?>
<vxml version="2.0">
      xmlns="http://www.w3.org/2001/vxml"
      xml:lang="en-US">
<!-- hello world example -->

<form>
  <block>
    Hello World!
  </block>
</form>
</vxml>
```

Opravdový VoiceXML program

```
<vxml version="2.0"
      xmlns="http://www.w3.org/2001/vxml"
      xml:lang="en-US">
<!-- hello world example -->

<var name="hi" expr="'Hello World!'"/>

<form id="say_hi">
  <block>
    <prompt><value expr="hi"/></prompt>
    <goto next="#say_goodbye"/>
  </block>
</form>
<form id="say_goodbye">
  <block>
    Goodbye!
  </block>
</form>
</vxml>
```

VoiceXML – libovolné tags

```
<?xml version=”1.0” encoding=”UTF-8”?>
<vxml version=”2.0”>
      xmlns=”http://www.w3.org/2001/vxml”
      xml:lang=”cs-cz”>
<firma>
  <jmeno_firmy>EuroSoftware ČR s.r.o.</jmeno_firmy>
  <oddeleni>
    <jmeno_oddeleni> vedení </jmeno_oddeleni>
      <osoba>
        <prijmeni> Mlynář </prijmeni>
        <jmeno> Roland </jmeno>
        <pozice> vedoucí </pozice>
        <ulice> Schumannova 12 </ulice>
        <obec> 330 49 Deštný Hradec </obec>
      </osoba>
[...]
</firma>
```

Další jednoduchý VoiceXML program

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="1.0">
<!-- drink example -->
  <form>
    <field name="drink">
     <prompt>Would you like coffee, tea, milk, or
             nothing ?</prompt>
     <grammar src="drink.gram" type="application/x-jsgf"/>
    </field>
    <block>
     <submit next="http://www.drink.example/drink2.asp"/>
    </block>
  </form>
</vxml>
```

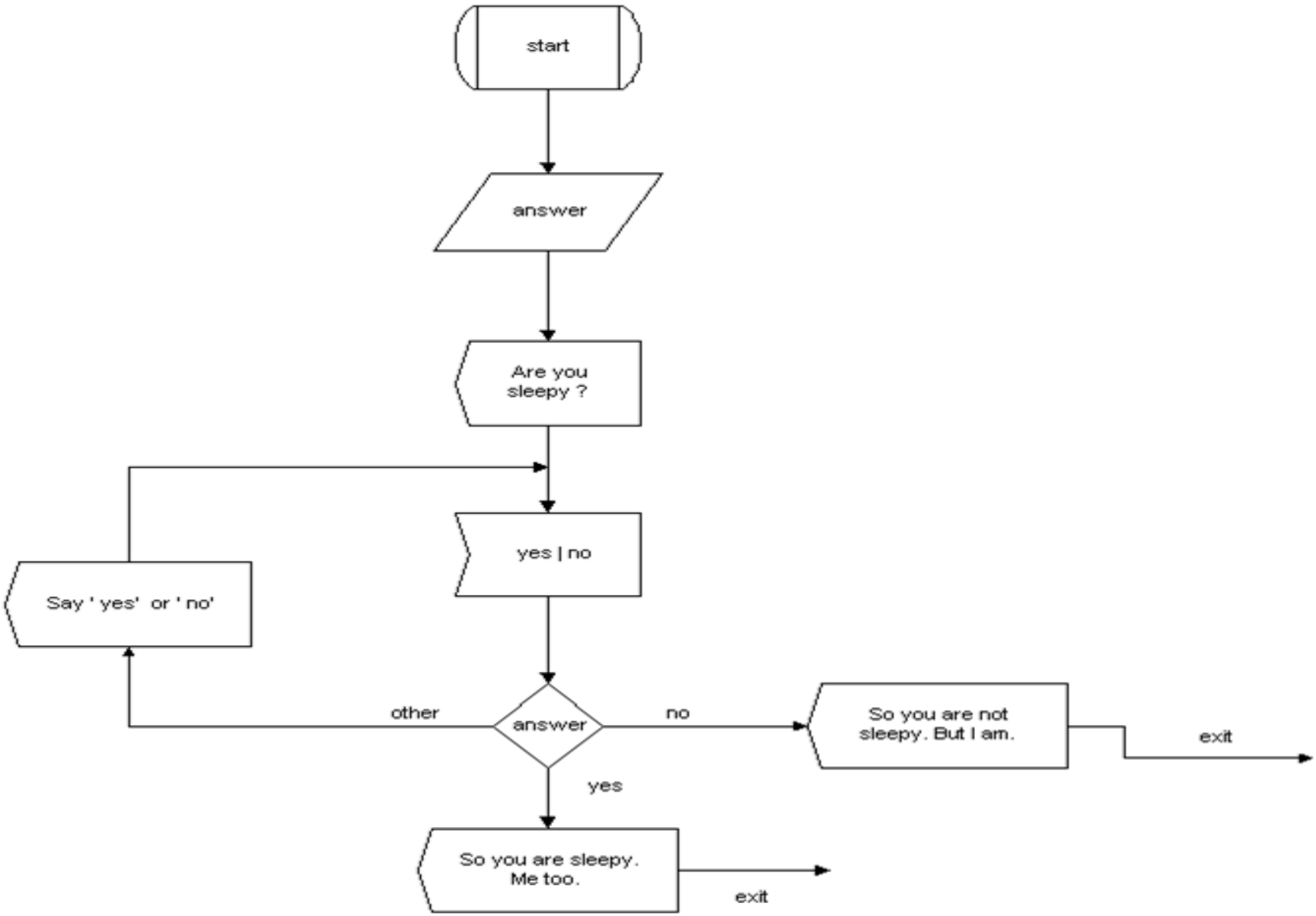
VoiceXML: užití proměnných

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="2.0">
<!-- Variables Example -->

<var name="a" expr="1"/>
<form id="start">
  <var name="b" expr="'hello'"/>
  <block>
    <prompt> Variable a has value: <value expr="a"/> </prompt>
    <prompt> Let's try to evaluate an easy expression: a+1 =
                                                  <value expr="a+1"/>
      </prompt>
    <prompt> Variable b has value: <value expr="b"/> </prompt>
    <prompt> Let's try string concatenation: b+' world' =
                                              <value expr="b+' world'"/>
      </prompt>
  </block>
</form>
</vxml>
```

VoiceXML: užití proměnných

```
Variable a has value: 1
Let’s try to evaluate an easy expression: a+1 = 2
Variable b has value: hello
Let’s try string concatenation: b+’ world’ = hello world
```

VoiceXML: grammar and event handling example



VoiceXML: grammar and event handling example

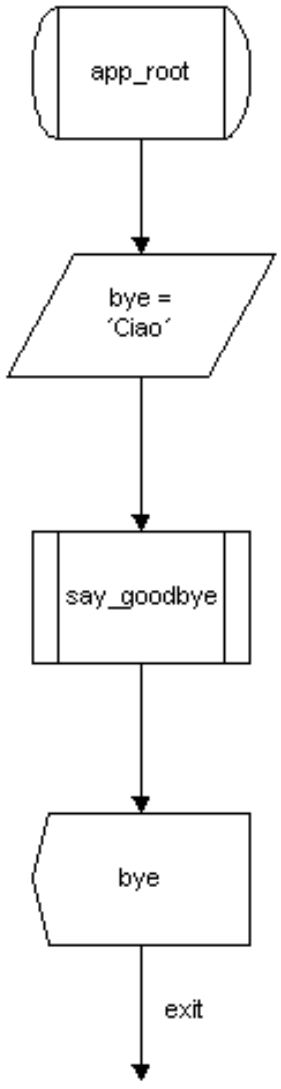
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="1.0">

<form id="start">
  <field name="answer">
    <catch event="noinput">
      <prompt> Hey, don’t sleep! </prompt>
    </catch>
    <catch event="nomatch">
      <prompt> say ’yes’ or ’no’ </prompt>
    </catch>
    <grammar> yes{yes} | y{yes} | no{no} | n{no} </grammar>
    <prompt> Are you sleepy? </prompt>
    <filled>
      <if cond="answer==’yes’">
        <prompt> So you are sleepy. Me too. </prompt>
      <else/>
        <prompt> So you are not sleepy. But I am. </prompt>
      </if>
    </filled>
  </field>
</form>
</vxml>
```

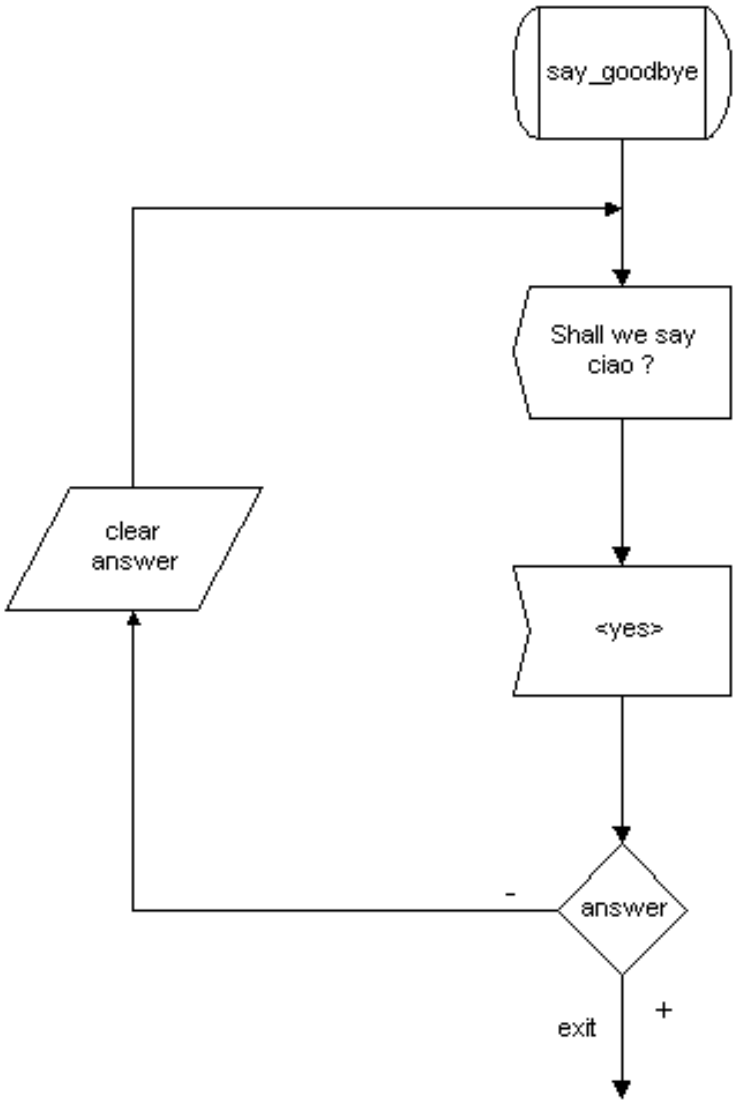
VoiceXML: grammar and event handling example

Možné vedení dialogu:

```
1)  Are you sleepy?
    yes
    So you are sleepy. Me too.

2)  Are you sleepy?
    no
    So you are not sleepy. But I am.

3)  Are you sleepy?
    y
    So you are sleepy. Me too.

4)  Are you sleepy?
    si
    say ’yes’ or ’no’
    ciao
    say ’yes’ or ’no’
    no
    So you are not sleepy. But I am.
```

VoiceXML: multi-document application

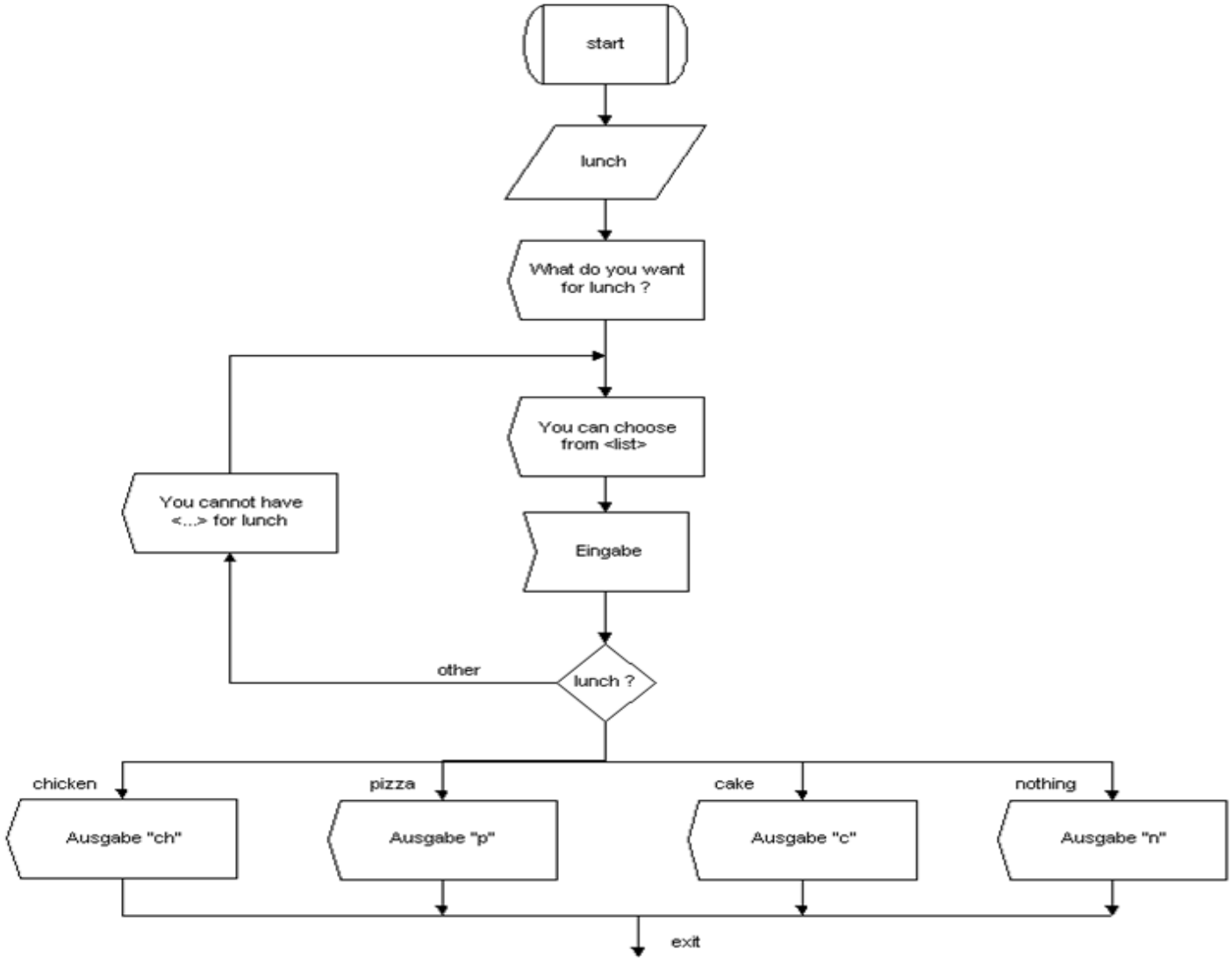




VoiceXML: multi-document application

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="2.0">
<!-- Application root document -->
  <var name="bye" expr="Ciao"/>
    <grammar> <rule id="root" scope="public">
                yes{yes} | y{yes} | no{no} | n{no} </rule> </grammar>
    <goto next="#say_goodbye"/>
</vxml>

<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="2.0">
<!-- Application leaf document -->
  <form id="say_goodbye">
   <field name="answer" type="boolean">
    <prompt> Shall we say <value expr="application.bye"/> ? </prompt>
    <filled>
      <if cond="answer">
        <exit/>
      </if>
      <clear namelist="answer"/>
    </filled>
   </field>
  </form>
</vxml>
```

VoiceXML: výběr z vícero alternativ



VoiceXML: výběr z vícero alternativ

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="1.0" >
<!-- Options, link and shadow variables example -->

<catch event="noinput">
  <prompt> Hey, don’t sleep! </prompt>
</catch>
<catch event="exit">
  <prompt> Exit! </prompt>
  <exit/>
</catch>
<link event="help">
  <grammar> [please] help [me] [please] </grammar>
</link>
<link event="exit">
  <grammar> exit | quit | q </grammar>
</link>
```

VoiceXML: výběr z vícero alternativ

```
<form id="start">
  <field name="lunch">
    <catch event="nomatch">
      <prompt> You cannot have <value expr="lunch$.utterance+' '"/>
                                                          for lunch.
 You can choose from <enumerate/> </prompt>
      <!-- shadow variable used here to get user's utterance -->
    </catch>
    <catch event="help">
      <prompt> Just say what you want for lunch. </prompt>  </catch>
    <option dtmf="1" value="ch"> chicken </option>
    <option dtmf="2" value="p"> pizza </option>
    <option dtmf="3" value="c"> cake </option>
    <option dtmf="4" value="n"> nothing </option>
    <prompt> What do you want for lunch? You can choose from
                                          <enumerate/> </prompt>
```

VoiceXML: výběr z vícero alternativ

```
    <filled>
      <if cond="lunch==’ch’">
        <prompt> You will have chicken for lunch </prompt>
        <prompt> I like chicken too </prompt>
      <elseif cond="lunch==’p’"/>
        <prompt> You will have pizza for lunch </prompt>
        <prompt> Italia fan?  </prompt>
      <elseif cond="lunch==’c’"/>
        <prompt> You will have cake for lunch </prompt>
        <prompt> You will be fat! </prompt>
      <else cond="lunch==’n’"/>
        <prompt> You will have nothing for lunch </prompt>
        <prompt> Diet? </prompt>
      </if>
    </filled>
  </field>
</form>
</vxml>
```

VoiceXML: výběr z vícero alternativ

Možné vedení dialogu:

```
1) What do you want for lunch? You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   nic

   You cannot have  nic  for lunch. You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   chicken

   You will have chicken for lunch
   I like chicken too
```

VoiceXML: výběr z vícero alternativ

```
2) What do you want for lunch? You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   ano
   You cannot have  ano  for lunch. You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   co?
   You cannot have  co?  for lunch. You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   pizza
   You will have pizza for lunch
   Italia fan?
```

VoiceXML: výběr z vícero alternativ

```
3) What do you want for lunch? You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   nic

   You cannot have  nic  for lunch. You can choose from
   1.  chicken
   2.  pizza
   3.  cake
   4.  nothing

   nothing

   You will have nothing for lunch
   Diet?
```

VoiceXML: soubor gramatiky pro dotazy o vlacích

```
grammar IC_Auskunft;                              // name

public <main> = <quest1> | <quest2> | <bye>;
<quest1> = <please>[<day>][<time>]<prep> <city_gn>;
<quest2> = <prep> <city_gn>;
<please> = chtěl bych;
<day> = dneska | zítra;
<time> = ráno | odpoledne | dopoledne | večer;
<city_gn> = Plzně | Prahy | Brna;
<prep> = z | do;
<bye> = [děkuju] na slyšenou;
```

VoiceXML: příklad pro různé koncepty

```
<?xml version="1.0">
<vxml version="2.0">
<var name="xyz" expr="16"/>  <!-- definice promennych -->
<form id="form_name">
  <block>                    <!-- vystup bez podminky -->
    <prompt> text anebo <value expr="hi"/> </prompt>
    <goto next="#next_module"/>
  </block>
  <field name="answer">    <!-- vstup reci -->
    <grammar> yes{yes} | no{no} </grammar>
    <catch event="noinput">       <!-- udalosti -->
      <prompt> Hey, don’t sleep! </prompt>
    </catch>
    <catch event="nomatch">
      <prompt> say ’yes’ or ’no’ </prompt>
    </catch>
```

VoiceXML: příklad pro různé koncepty

```
    <filled>      <!-- az podminka bude splnena -->
      <if cond="answer==’yes’">
        <prompt> Answer 1 </prompt>
      <else/>
        <prompt> Answer 2 </prompt>
      </if>
    </filled>
  </field>
</form>
</vxml>
```

VoiceXML: jednoduché vedení dialogu

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<vxml version="2.0">
<!-- Weather example -->

<catch event="noinput">
  <prompt> Hey, don’t sleep! </prompt>
</catch>
<catch event="exit">
  <prompt> Exit! </prompt>
  <exit/>
</catch>
<link event="exit">
  <grammar> exit | quit | q </grammar>
</link>
```

VoiceXML: jednoduché vedení dialogu

```
<form id="weather_info">
 <block>Welcome to the weather information service.</block>
 <field name="state">
  <prompt>What state?</prompt>
   <grammar src="state.gram"  type="application/x-jsgf"/>
  <catch event="help">
     <grammar> [please] help [me] [please] </grammar>
     Please speak the state for which you want the weather.
  </catch>
 </field>
 <field name="city">
  <prompt>What city?</prompt>
  <grammar src="city.gram" type="application/x-jsgf"/>
  <catch event="help">
     <grammar> [please] help [me] [please] </grammar>
     Please speak the city for which you want the weather.
  </catch>
 </field>
```

VoiceXML: jednoduché vedení dialogu

```
 <block>
  <prompt> Transmission of program control to the weather
           information system </prompt>
  <prompt> to provide the information about the weather
                    in <value expr="city$.utterance"/>,
  <value expr="state$.utterance"/> </prompt>
  <submit next="/servlet/weather" namelist="city state"/>
 </block>
</form>
</vxml>
```

VoiceXML: jednoduché vedení dialogu

Možné vedení dialogu:

```
Welcome to the weather information service.
What state?
Georgia
What city?
Tbilisi
Unrecognized input
What city?
Macon
Transmission of program control to the weather
information system to provide the information about
the weather in Macon, Georgia
```

Zdroje

- V. Matoušek, Natürlichsprachliche Mensch-Computer Interaktion mit VoiceXML. Západočeská univerzita v Plzni, 2004. http:\\www.kiv.zcu.cz/studies/predmety/icp
- L. Bártek. Dialogové systémy. Masarykova univerzita, Brno, 2011. http:\\is.muni.cz/el/1433/jaro2012/PA156/um/prednasky/prednasky.pdf
- S. Schaden. Dialogmodelle und Dialogmodellierung. Universität Duisburg-Essen, 2007. http:\\www.shameacademy.de/chapters/mmi/ppt/ \ vortrag_schaden_dialog.pdf