

Laurea Specialistica in Informatica
a.a. 2005-2006

Interazione Uomo-Macchina II:

Interfacce Intelligenti

Fiorella de Rosis

1

Introduzione

Prima parte: Formalizzazione e Ragionamento

- 1.1. Ragionamento logico:
 - Formalizzazione
 - Risoluzione
- 1.2. Ragionamento incerto
 - Reti Causali Probabilistiche
 - Reti dinamiche
 - Apprendimento di Reti

Seconda parte: Modelli di Utente

- 2.1. Modelli logici
- 2.2. Modelli con incertezza

Terza parte: Interazione in linguaggio naturale

- 3.1. Generazione di messaggi
 - Introduzione
 - Teorie
 - Metodi
- 3.2. Comprensione di messaggi

Quarta parte: Simulazione di dialoghi

Partiamo dall' esempio
del trascritto di un dialogo telefonico
(speech-based)
sulla prenotazione di voli,
che avevamo introdotto all'inizio del Corso.

Ragioniamo su come questo tipo di dialoghi possa
essere simulato

3

S: A at American Express may I help you?

U: yeah this is B. I need to make some travel arrangements
[recording notification, charge number, travel advance, phone]

S: ok and what do you need to do?

U: ok on June sixth from San Jose to Denver, United

S: leaving at what time?

U: I believe there's one leaving at eleven o'clock in the morning

S: yeah that's United flight four seventy

U: ok

S: leaves at eleven a.m. and arrives Denver at two twenty p.m. out of San Jose

U: that's the one

S: and a return?

U: return June seventh

S: leaving at what time?

U: uh I believe there's one at five sixteen

S: yeah that's United flight six fifty nine it arrives back into San Jose at six forty two

U: ok

S: let me just see what sort of fare I can get you on this

U: all right

S: ok the best fare I can get you on this would be five sixty eight round trip. That does
have a twenty five percent change cancellation penalty on it

U: ok

S: And we'd have to ticket it by Tuesday

U: right, that's fine

[hotel, car, seating, mileage plus number]

U: ok thank you very much

S: thanks a lot

(Trascritto di un dialogo
telefonico American
Express)

Simulatori di Dialoghi vs Generatori di Monologhi

Generazione di Monologhi	Simulazione di Dialoghi
Il <i>modello dell'utente</i> viene definito inizialmente e non varia durante il processo di generazione	Il modello è funzione del contesto: scambi avvenuti e ragionamenti effettuati da S durante il dialogo
Gli <i>obiettivi comunicativi</i> vengono definiti inizialmente e quindi tradotti in un piano.	S ha degli obiettivi iniziali, ma può rivederli sulla base di nuove esigenze emerse nel corso del dialogo
Il <i>piano del discorso</i> viene definito inizialmente e quindi tradotto in un messaggio.	S definisce un 'piano globale' ma adatta al contesto i piani realizzati in ogni passo del dialogo.
Le <i>fonti d'informazione</i> sono: <ul style="list-style-type: none"> • i piani o gli operatori di piano, • gli schemi per la generazione superficiale • il modello di U. 	Alle fonti necessarie per generare monologhi si aggiunge una 'storia del dialogo'.
Le <i>componenti principali</i> sono: un pianificatore, un (eventuale) revisore del piano e un generatore superficiale.	Alle componenti necessarie per generare un monologo si aggiunge un revisore del modello di utente, dei propri goal e belief e dei piani da realizzare ⁵

Complessità dei Dialoghi

I dialoghi possono essere più o meno semplici, a seconda dell'iniziativa lasciata all'utente (far domande o semplicemente rispondere alle domande del sistema) e della complessità del ragionamento che il sistema deve eseguire per rispondere a queste domande.

Gli elementi che contribuiscono a definire la complessità del simulatore sono l'*obiettivo* del dialogo e l'*iniziativa* (del sistema o dell'utente).

Facciamo qualche esempio.

Un dialogo molto semplice:

Information-giving, system initiative, formal attitude

S: Wellcome to Irene's travel agency. I'm here to help you to buy your travel tickets. Which place do you want to go?

U: London

S: Which place do you start from?

U: Bari

S: Do you prefer travelling with a regular or a low-cost company?

U: Regular

S: Which tarif do you prefer? Economic or business?

.....

Qui, S guida il dialogo, e le risposte di U appartengono a categorie predefinite. Non è necessario un parser complesso né una interpretazione della mossa di U. Non sono necessarie revisioni di goal e di piani, ma al massimo piccoli aggiustamenti.

Un dialogo più complesso:

Information-giving, mixed initiative, friendly attitude

S: Wellcome to Irene's travel agency. I'm here to help you to organize your travels.

U: I would like to go to London starting from Bari in the month of may, and have not much money to spend.

S: Oh, London! I would love going there! There are several low-cost companies going there. Let's see what we can do. Do you mind to transfer from Fiumicino to Ciampino, when you are in Rome?

U: No problem: I'm young and used to travel.

S: Good, no problem then. Just tell me when exactly you would like to leave.

.....

Qui, S ha un doppio goal: dare informazioni e 'stabilire un clima di cooperazione' con U. Il linguaggio d'interazione di U è abbastanza rigido, e le mosse che può fare sono predefinite.

Un dialogo ancora più complesso: information-giving, mixed initiative

S: Dovresti andare a correre, Giuseppe! *Prima mossa persuasiva*

U: Perché?

S: Perché sei giovane, ci tieni alla tua salute, e correre fa bene alla salute. *Linea di ragionamento seguita*

U: Ma ti pare che, a vent'anni, già devo pensare alla salute?

S: Secondo me sì. Ma comunque, correre aiuta anche a tenersi in forma. *Correzione della strategia persuasiva*

U: E chi te l'ha detto?

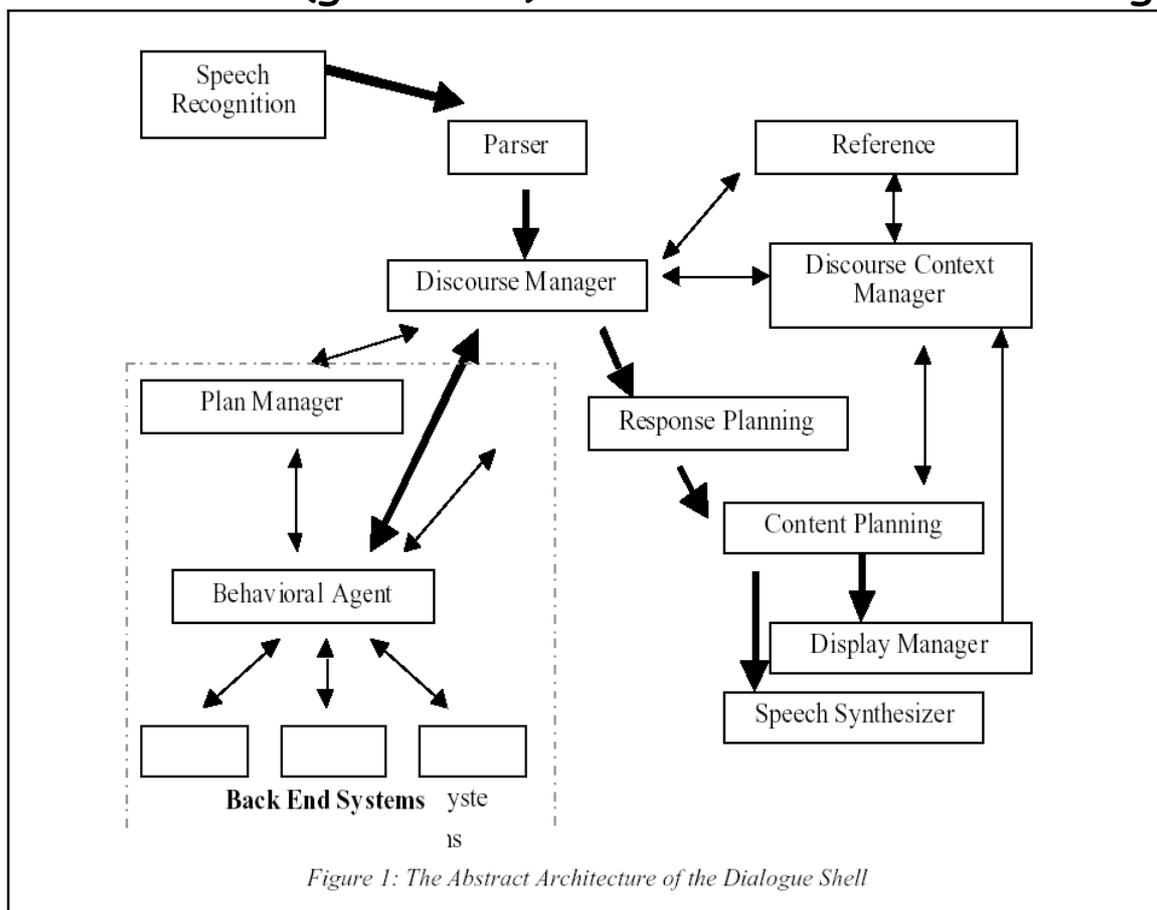
S: Lo dicono studi epidemiologici svolti in diversi paesi, da istituti di ricerca qualificati. *Giustificazione di un passo di ragionamento*

U: Ma io detesto correre. *ragionamento*

...

Qui, ad ogni passo U può fare mosse di tipo diverso: chiedere più informazioni, verificare come S sta ragionando, obiettare, ...

Architettura (generale) di un simulatore di dialoghi



Module	Function	<i>Ruolo delle diverse componenti</i>
Speech Recognition (SR)	Transforming speech input into a word stream or word lattice	
Parser	Transforming the SR output into interpretations, each a set of conventional speech acts, using full and robust parsing techniques	
Reference Manager (REF)	Identifying the most salient referents for referring expressions such as noun phrases	
Discourse Context Manager	Maintaining the global (topic flow) and local (salience with a topic) discourse context	
Discourse Manager (DM)	Identifying the intended speech act, current task, current step in the current task, and system obligations arising from the dialogue	
Behavioral Agent (BA)	Determines system actions (e.g., answer a question, notify of a problem, request clarification); Manages the interface to the back-end systems.	
Plan Manager	Constructing, modifying, evaluating, and executing plans (whether they are the subject of the conversation or the task being executed)	
World KB	Maintains a description of the current state of the world under differing assumptions (e.g., based on different plans or hypotheses)	
Response Planner	Determining the best communicative act(s) (and their content) to accomplish the system's current goals and discourse obligations	
Content Planner	Determining how to realize the planned speech acts	
Display Manager	Managing the visual presentations given the available hardware.	

Table 2: The Key Modules in the Dialogue Shell in the Abstract Architecture

Relazione fra Metodi e Task

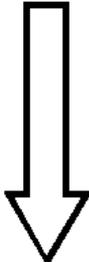
Technique Used	Example Task	Task Complexity	Dialogue Phenomena handled
Finite-state Script	Long-distance dialing	least complex	User answers questions
Frame-based	Getting train arrival and departure information		User asks questions, simple clarifications by system
Sets of Contexts	Travel booking agent		Shifts between predetermined topics
Plan-based Models	Kitchen design consultant		Dynamically generated topic structures, collaborative negotiation subdialogues
Agent-based Models	Disaster relief management		Different modalities (e.g., planned world and actual world)
			most complex

Figure 1: Dialogue and Task Complexity

Vedremo ora due dei metodi principali per la simulazione di dialoghi:

- a. Simulatori a stati finiti e
- b. Simulatori basati sull'Information state, confrontandone vantaggi e svantaggi.

13

Simulatori a Stati Finiti, eventualmente 'augmentati' (ATN = Augmented Transition Networks)

Sono utilizzabili quando l'iniziativa è del sistema, che segue un piano predefinito per raccogliere informazioni dall'utente.

Gli stati rappresentano il risultato dell'esecuzione di una mossa di dialogo nello stato precedente.

Per ogni stato, viene definita una lista di possibili mosse successive.

Ad ogni stato sono associate delle 'azioni' che il sistema esegue.

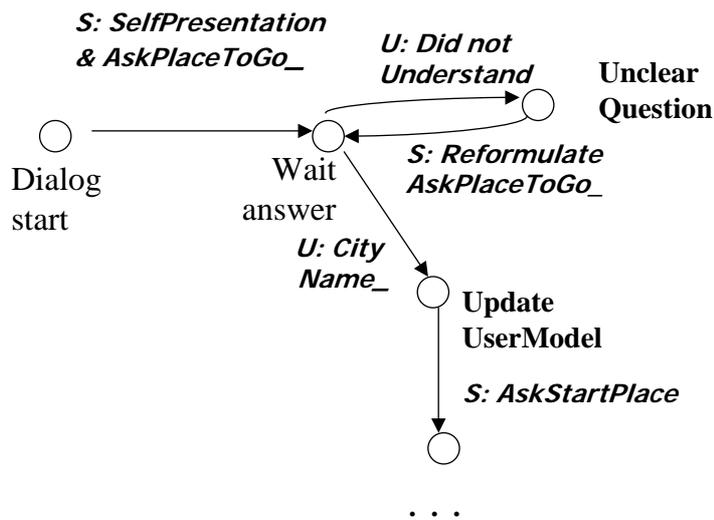
Il passaggio da uno stato all'altro è reso possibile dal verificarsi di una serie di condizioni, legate alla mossa eseguita dall'utente e al valore di alcuni registri.

... vediamo qualche esempio ...

14

Una ATN molto semplice

per il dialogo della Travel Agency di Irene



S: Wellcome to Irene's travel agency. I'm here to help you to organize your travels. Which place do you want to go?

U: London

S: Which place do you start from?

U: Bari

S: Do you prefer travelling with a regular or a low-cost company?

U: Regular

S: Which tarif do you prefer? Economic or business?

Vediamo ora un esempio di sistema basato su ATN.
 Obiettivo: dare suggerimenti sullo sport utilizzando un Agente Animato Conversazionale

Esercizio 1

Prova a simulare come varia, ad ogni passo, il modello dell'utente costruito da S, nel dialogo precedente.

Il Sistema FitTrack dell' MIT

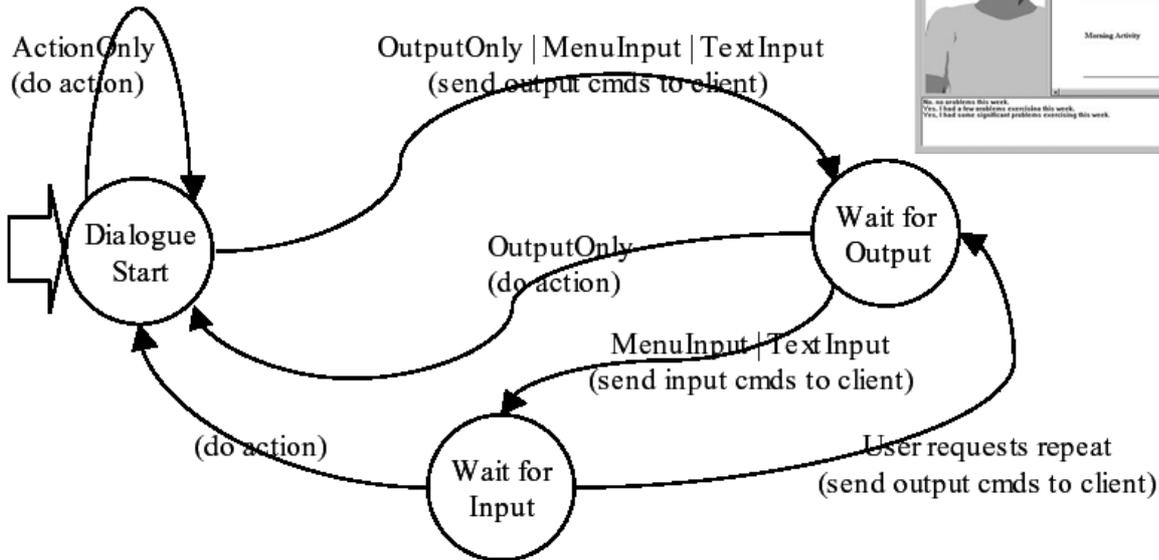


Figure 8-7. Fragment of ATN Executive State Machine

Una macchina a stati finiti + uno *stack*.

L'esecuzione può passare da un nodo all'altro, oppure al nodo iniziale di un'altra ATN, inserendo il nome del nodo attuale nello stack (*innestamento*).

Lo Scripting Language di FitTrack

```

STATE: Si
AGENT: $<BEAT>Are you going to work out tomorrow?</BEAT>$
USERMENU: I can't. => $ Go("Sj"); $
           Yep. => $ Go("Sk"); $
    
```

← Tag per l'animazione del character

Oltre ad azioni di Go, è possibile inserire altre funzioni, di azione sullo stack (PUSH e POP), per realizzare l'innestamento fra ATN.

... un compilatore produce l'eseguibile...

FitTrack: Esempi per diverse 'personalità' dell'agente

"Int1" script

```
STATE: Int1
AGENT_REL: $ <BEAT> <HAPPY/> Hi there. To talk to me, <TEXTBOX> just click on one
of the options in the menu below </TEXTBOX>. <NEUTRAL/> </BEAT> $
AGENT_NREL: $ <BEAT> <NEUTRAL/> Hi there. To talk to me, <TEXTBOX> just click on
one of the options in the menu below </TEXTBOX>. </BEAT> $
REPEAT: $ <BEAT>Just click the mouse below where it says OK.</BEAT> $
USERMENU: OK, I understand that this is where I click. => $ GO("INT1_1"); $

STATE: INT1_1
ACTION: $ if(ISREL()) GO("INT1_3"); else GO("INT1_2"); $

STATE: INT1_2
AGENT: $ <BEAT> Good. I'm going to be your exercise advisor. </BEAT> $
ACTION: $ GO("INT1_11"); $

STATE: INT1_3
AGENT: $ <BEAT> Good. My name is Laura. <YOU> You are =| GET("NAME") |= , is that right?
</YOU> </BEAT> $
USERMENU: Yep, that's right. => $ GO("INT1_5A"); $
No, actually that's not right. => $ GO("INT1_4"); $

STATE: INT1_4
AGENT: $ <BEAT> Oh, I'm sorry. What is your name? </BEAT> $
USERTEXT: My name is: => $ String entry=GETTEXT();
ALERT("WRONG NAME: not "+GET("NAME")+ " corrected to "+entry);
entry=FILTERNAME(entry);
if(entry.equals("NOTCHECKED")) entry="";
SET("NAME",entry);
GO("INT1_5"); $
```

Script Primitives

Funzioni in FitTrack

Type	Function	Purpose
Property Access Functions	GET(<property>)	Looks up single-valued property value in database, returned as a String.
	GETINT(<property>)	Returns value as an integer.
	UNDEFINED(<value>)	Returns 'true' if the property value is undefined.
	SET(<property>,<value>)	Sets a single-valued property in the database.
	GET_SESSION(<property>,<index>)	Looks up historical property value. If (integer) index is positive, this returns the value for the specified session. If index is negative, this uses relative indexing (e.g., "-1" returns the value from the previous session). If zero, this returns the value from the current session.
	GET_SESSION(<property>)	Looks up historical property value for the current session.
	SET_SESSION(<property>,<value>)	Sets a historical property value, indexed to the current session.
	EXISTS_SESSION(<property>,<value>)	Returns most recent session index in which <property> had <value>.
	COUNT_SESSION(<property>,<value>)	Returns the number of sessions in which <property> had value <value>.
	GET_EQ(<property>,<value>)	Returns 'true' if single-valued <property> had value <value>.
	GET_SESSION_EQ(<property>,<value>)	Returns 'true' if historical <property> has value <value> in current session.
	GET_SESSION_EQ(<property>,<index>,<value>)	Returns 'true' if historical <property> has value <value> in the specified session.

... Continuano le funzioni di FitTrack

ATN Change Functions	GO(<state>)	Branches to the named state.
	PUSH(<nextState>, <returnState>)	Pushes <returnState> on the runtime stack and branches to <nextState>.
	POP()	Pops the runtime stack and branches to the returned state.
User Interface Functions	GETTEXT()	Returns the text entered by the user.
	CONTENT(<n>)	Displays educational content page for session <n> in the main browser window.
Database lookup	ISREL()	Returns 'true' if user is in RELATIONAL condition of the study, else 'false' (if in NON-RELATIONAL).
	NAME()	Returns subject's given name.

Script Primitives, continued

Type	Function	Purpose
Miscellaneous	PICK(<n>)	Returns a random number between 1 and <n>.
	REFER_SESSION(<n>)	Returns an English expression referring to the time of the specified session (e.g., "YESTERDAY", "LAST TUESDAY", etc.)
	GOALTIME()	Returns the number of minutes the user should be exercising today.
	GOALSTEPS()	Returns the number of steps the user should be walking today.
	ALERT(<string>)	Enters message into the Alert table in the database (reviewed daily by operator for exceptions).

21

FitTrack: Un esempio di dialogo

1. A: Hi!
2. U: Hello.
3. A: So, now I'd like to ask you a little about your exercise behavior. Do you currently do any kind of sport?
4. U: No, I don't
5. A: Do you currently do any other kind of aerobic exercise regularly?
6. U: I go to the gym
7. A: And, do you think you would like to exercise by yourself, or with a buddy?
8. U: By myself
9. A: Great. So, did you manage to go for a walk since we last talked?
10. U: No, didn't get the chance.
11. A: OK
12. U: Before we chat again, do you think you could get out and go for another walk? It doesn't matter how long or how far.
13. A: Sure.
14. A: Okay. Here is some information about how to make a walk a workout. Casual strolls don't quite count.
15. A: Remember, when we end our sessions, you can spend as much time as you like browsing this library. Just close the Fit Track application when you are finished.
16. A: Have a great morning, and the rest of your day too. Goodbye.

Figure 8-8. Sample NON-RELATIONAL Interaction (A=Agent; U=User)

Unico caso in cui U prende l'iniziativa, nel dialogo

L'agente suggerisce 'come' praticare lo sport

22

FitTrack: Un dialogo con 'smalltalk'

1. A: Hi Chrissy.
2. U: Hi Laura, great to see you.
3. A: How are you doing today?
4. U: I'm not feeling so great.
5. A: I'm sorry to hear that.
6. U: Tell me more about how you are feeling
7. U: I'm sick
8. A: I'm sorry you're sick. What do you think is wrong?
9. U: I have the flu
10. A: You must be feeling miserable. Have you been to see a doctor?
11. U: Yes.
12. A: That's good. It can make you feel better just seeing a health professional about a problem. Do you think this illness might affect your ability to exercise?
13. U: Definitely.
14. A: I'm sorry to hear that. As soon as you feel well enough, you might try to get out for a short walk. The fresh air will do you good, and it will help you to stay in the habit.
15. A: So, are you a student?
16. U: Yes
17. A: Do you go to MIT?
18. U: No
19. A: Oh, what school do you go to?
20. U: Cambridge Continuing Education, right now. I am working toward a degree at Tufts.
21. A: Really? Are you a grad student, or an undergraduate?
22. U: Senior
23. A: Cool. What's your major?
24. U: English
25. A: So, now I'd like to ask you a little about your exercise behavior. Do you currently do any kind of sport?
26. U: No, I don't
27. A : Do you currently do any other kind of aerobic exercise regularly?
28. U: I do something else

← **Smalltalk**

← **Come nel Dialogo Precedente (move 3)**

Esercizio 2

Prova a simulare con un simulatore a stati finiti una parte del dialogo dell'American Express.

Vantaggi e limiti dei metodi script-based

Il numero di stati attraverso i quali il sistema può passare è finito e predeterminato.

La logica di sviluppo del dialogo è espressa in modo molto chiaro

L'iniziativa è fondamentalmente del sistema

Gli utenti devono rispondere con il livello di dettaglio e l'ordine di presentazione delle informazioni previsti dal sistema

E' difficile realizzare un sistema che adatta il suo comportamento alle caratteristiche dell'utente (utilizzando un modello del suo stato mentale)

25

Un Dialogo in ARTIMIS (France Telecom)

S: Welcome to the ARTIMIS server. How can I help you?

U: Hello, I would like to know the sea weather forecast for the Finistère area.

S: There is no sea weather forecast server for the Finistère area.

But there are some for the Channel and for the Atlantic.
Are you interested in one of these?

U: Yes, mmm the Channel

S: There are two servers for coastal weather forecast for the Channel.
Their phone numbers are.... Would you like more information?

U: Does it have specific weather forecast for sailing?

S: No it doesn't.

There is no weather forecast specialized for sailing for the Channel.
But there are some for France. Is that OK?

U: Yes.

.....
S capisce qual è l'obiettivo di U e non si limita a rispondere alla sua richiesta, ma cerca di rispondere a questi obiettivi.

26

Per formalizzare l'esempio precedente,
ricordiamo la Classificazione di Searle
degli atti comunicativi, in funzione degli 'obiettivi comunicativi'

Constativi:

- informare, avvertire
esprimono il desiderio che H creda in un determinato fatto
- affermare, annunciare, rispondere, sostenere, confermare, negare,
...
esprimono il desiderio che H creda che S crede in un determinato fatto
- chiedere, domandare:
esprimono il desiderio di conoscere il pensiero di H su un determinato fatto

Direttivi: suggerire, ordinare, istruire, permettere,...

esprimono il desiderio che H compia una determinata azione

Commissivi: offrire, promettere, garantire, ...

esprimono l'intenzione, da parte di S, di compiere una determinata azione

Riconoscitivi: scusarsi, congratularsi, ringraziare,...

coinvolgono il passaggio d'informazioni su 'atteggiamenti di gentilezza' di S

Elaborazione degli atti comunicativi di U

Analizziamo le 'mosse' di U nel dialogo precedente, formalizzandole in termini dei seguenti atti comunicativi:

x variabile che denota un information item; a variabile che denota un'azione; φ formula; ag variabile che denota S/U

Constativi

Inform / AskInfo (ag x)

Answer / Announce / Confirm / AskIf (ag φ)

Direttivi

Request / Order (ag a)

Commissivi

Offer / Promise (ag a)

Riconoscitivi

Apologize / Congratulate / Thank / Acknowledge (ag φ)

Elaborazione degli atti comunicativi di U

U: Hello, I would like to know the sea weather forecast for the Finistère area.

a) *Riconoscimento*: traduzione della frase in LN in forma simbolica:

U ha chiesto informazioni sulle previsioni del tempo nel Finistère

AskInfo(U, WeFo°(Finistere))

b) *Interpretazione*: ragionamento sullo stato mentale di U che ha prodotto quell'atto comunicativo:

U vuole conoscere le previsioni del tempo nell'area

(Bel S WantToKnow(U, WeFo°(Finistere))) *

c) Ragionamento su di sé (come rispondo alla sua mossa?)

* ... ricordiamo le ipotesi sull'atto comunicativo di AskInfo...

29

Ragionamento sull'atto comunicativo di U

c) Ragionamento su di sé (come rispondo alla sua mossa?)

KnowAbout(S, WeFo°(Finistere)) ? No!

S non le conosce

Is-in(Finistere, NorthernFrance)

KnowAbout(S, WeFo°(NorthernFrance))

Il Finistere è nel nord della Francia

e S conosce le previsioni in questa zona

(Bel S WantToKnow(U, WeFo°(NorthernFrance)))

S crede che U voglia conoscere le previsioni nel nord della Francia

IsIn(Channel, NorthernFrance)

(Bel S WantToKnow(U, WeFo°(Channel)))

In particolare, la Manica è nel nord della Francia

S crede che U voglia conoscere le previsioni nella Manica

Regola applicata da S (ragionamento su U, su S e sul dominio)

$$\text{Bel } S (\text{WantToKnow}(U,x) \wedge \neg \text{KnowAbout}(S,x) \wedge (\text{Similar}(x,y) \vee \text{IsIn}(x,y)) \wedge \text{KnowAbout}(S,y) \rightarrow \text{WantToKnow}(U,y))$$

Traduzione del ragionamento nella move successiva

Piano che combina:

Answer (S \neg KnowAbout(S, WeFo°(Finistere)))

e

Answer(S KnowAbout(S WeFo°(Channel)))

... con una relazione retorica di 'Contrast'...

e

OfferInfo (S WeFo°(Channel))

Con una 'Joint'

Per giustificare perché
non risponde
direttamente alla
domanda di U

Per verificare se la
sua ipotesi è
corretta

S: There is no sea wheather forecast server for the
Finistère area. But there are some for the Channel and
for the Atlantic.

Are you interested in one of these?

(traduzione in LN della Answer e della Offer)

31

Cosa occorre per gestire questo tipo di dialogo?

Fonti d'informazione:

Un *modello dell'utente*, in cui memorizzare le informazioni
acquisite (o inferite) durante il dialogo.

Una base di *conoscenza sul dominio*.

Una *agenda* delle cose da fare.

Una *'storia' del dialogo*, e in particolare dell'ultima *'mossa'* (chi
ha parlato e cosa ha detto, quale piano ha applicato il
sistema, ecc).

Capacità di ragionamento:

Interpretare la mossa dell'utente.

Aggiornare il suo modello garantendone la consistenza.

Scegliere il piano successivo ed eseguirlo.

Tradurre il piano in linguaggio naturale o in forma multimediale.

32

Una soluzione possibile: una 'Lavagna'

E' necessario uno spazio di memoria (una 'blackboard')
in cui memorizzare i risultati del ragionamento,
con una differenziazione fra fatti acquisiti ('shared') e fatti inferiti
(private):

Su U: WantToKnow(U, WeFo°(Finistere))	shared
WantToKnow(U, WeFo°(NorthernFrance))	private
WantToKnow(U, WeFo°(Channel))	private

Nella stessa blackboard si può registrare la 'storia del dialogo':

AskInfo(U, WeFo°(Finistere))
Answer (S, ¬ KnowAbout(S, WeFo°(Finistere)))
Answer (S KnowAbout(S, WeFo°(Channel)))
OfferInfo (S, WeFo°(Channel))

33

Una Lavagna 'attiva' (Information State)



34

Approccio al Dialog Management basato sull' Information State

- *Information State (IS)*
 - an abstract data structure (record, DRS, set, stack etc.)
 - accessed by dialogue system modules using conditions and operations
- *Dialogue Moves*
 - utterance function (ask, answer, request etc.)
- *Update rules*
 - Modify IS based on on observed moves
 - Select moves to be performed
- IS Approach implemented in TrindiKit

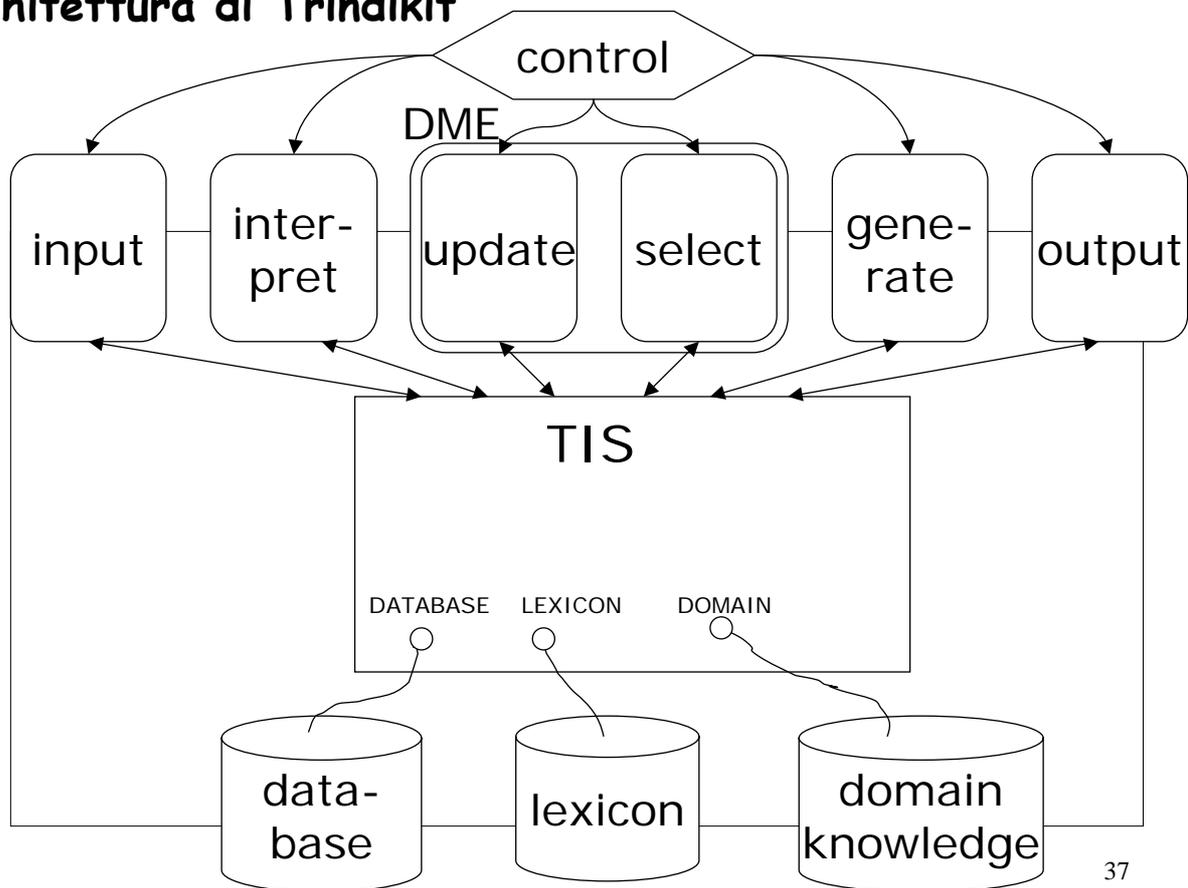
35

Trindikit

Un simulatore di dialoghi in Prolog basato sul metodo dell'information state,
sviluppato nell'ambito di un Progetto Europeo.

36

Architettura di Trindikit



La struttura dell'IS

PRIVATE	:	BEL	: Set(Prop)
		AGENDA	: Stack(Action)
SHARED	:	BEL	: Set(Prop)
		QUD	: Stack(Question)
		LM	: Move

BEL: conoscenza acquisita durante il dialogo

AGENDA : stack delle azioni da eseguire

QUD : questions under discussion

LM: latest dialogue move performed

PRIVATE : conoscenza inferita dal sistema

SHARED : conoscenza esplicitamente dichiarata durante il dialogo

L'IS rappresenta quindi, in particolare, lo User Model, la storia del dialogo e un'agenda delle cose da fare, mentre la conoscenza sul dominio è rappresentata in una struttura separata.

Update rule

Permettono di aggiornare l'IS dopo una 'mossa' di S o di U.

Modello di Utente:

- dopo un atto comunicativo CA di U: aggiornamento dell'UM con gli scopi di CA(U); *(perché U ha detto CA(U)?)*
- dopo un atto comunicativo di S: aggiornamento dell'UM con gli effetti di CA(S); *(quali effetti presumibilmente ha prodotto CA(S)?)*

Esempio:

$(\text{AskInfo } U, x) \rightarrow \text{WantToKnow}(U, x)$

$(\text{Inform } U, \varphi) \rightarrow \text{Bel } U \varphi$

Ma l'aggiornamento può riguardare anche le implicazioni di quanto U o S hanno detto.

Ad esempio, in un lucido precedente avevamo:

$(\text{WantToKnow}(U, x) \wedge (\text{Similar}(x, y) \vee \text{IsIn}(x, y))) \rightarrow \text{WantToKnow}(U, y)$

39

Select rule

Permettono ad S di selezionare la prossima 'mossa' da compiere (singolo atto comunicativo o piano), sulla base del contenuto dell'IS (UM, stato del dialogo, obiettivi che S intende raggiungere).

Esempio: Answer directly

$(\text{WantToKnow}(U, x) \wedge \text{KnowAbout}(S, x)) \rightarrow \text{InformAbout}(S, x)$

Esempio: Answer indirectly

$(\text{WantToKnow}(U, x) \wedge \neg \text{KnowAbout}(S, x) \wedge (\text{Similar}(x, y) \vee \text{IsIn}(x, y))) \wedge \text{KnowAbout}(S, y) \rightarrow \text{InformAbout}(S, y)$

Esempio: OpenSmallTalk *

$(\text{LM}(U, \text{Greet}) \wedge (\text{Bel } S \neg \text{Socialized}(U)) \wedge (\text{Goal } S \text{Socialized}(U))) \rightarrow \text{NextPlan}(S, \text{SmallTalk})$

* Apre un dialogo di smalltalk all'inizio dell'interazione, come nell'esempio di FitTrack

Struttura di controllo

```
control_algorithm( [ ! reset,
    * [ * [ ( ! select ,
        ( ( or( or( or( fstRec(private^agenda, ifem(,_,_ ) ), fstRec(
private^agenda, if(,_,_ ) ) ),fstRec( private^agenda, ifthen(,_,_ ) ) ),fstRec(
private^agenda, end) ) , ! [ ! adequate, ! update, !goal_check, ! print_state ] );
        ! [ ! generate,
            ! output,
            ! update,
                ! goal_check,
            ! print_state ] ) ),
        ( or( emptyRec( private^agenda ),
            fstRec( private^agenda, raise(_^(task=_)) ) ) ),

        program_state:val(run),
        ! input,
            ! interpret,
            ! update,
        ! print_state
    ],
    program_state:val(run)
]
    ] ).
```

41

Struttura di controllo

```
control_algorithm
While system_goals = empty:
    Select next move from agenda
        if nil, then
            select next goal; read nextplan
            add (nextplan, agenda)
        Generate(car(nextplan), system_move)
        Output system_move
        Read user_input
        Interpret (user_input, user_move)
        Update IS: add(user_move, dialog_history, )
        Update IS: integrate(user_move, user_model)
        Update IS: revise(system_goals)
end
```

42

Generatore

Traduce le singole mosse che il sistema ha stabilito di compiere
(rappresentate in forma simbolica)
in frasi in linguaggio naturale
oppure frasi annotate in un linguaggio di markup,
applicando uno dei metodi
che abbiamo esaminato, in breve, nelle unità precedenti.

43

Vediamo un esempio di dialogo con un Agente animato simile a FitTrack

S1: Dovresti andare a correre, Giuseppe!

U1: Perché?

S2: Perché ci tieni alla tua salute e correre fa bene alla salute.

U2: *Ma tu ci vai, a correre?*

S3: *Io sono un agente artificiale, non posso correre!*

U3: *Oh peccato, mi sarebbe piaciuto fare una corsetta con te!*

S4: *Grazie, sei gentile. Chissà, in un mondo virtuale! Per ora, puoi andare a correre con un'amica, se vuoi restare in buona salute...*

U4: Ma ti pare che a ventanni devo già pensare alla salute?

S5: Ma correre tiene anche in forma!

44

Dialogue games

L'esempio precedente fa riferimento alla categoria dei *dialogue games*, in cui (in una certa fase del dialogo) *i due interlocutori hanno goal diversi*:

S vuole persuadere U a fare sport:

U vuole scherzare con S.

S potrebbe rifiutare il goal di U. Ma invece adotta un atteggiamento 'aperto'. Lo accetta temporaneamente (in S3 e S4), per poi riprendere il suo (nella stessa S4).

Vediamo la dinamica del contenuto dell'IS in questo dialogo.

Ne descriveremo una possibile simulazione introducendo nell'IS una agenda e senza formalizzare né la rappresentazione della conoscenza né il suo trattamento.

45

Piano iniziale di S: *Persuadere U motivando*:

```
<DPL><plan name="RatPers">
  <goal name="Persuade" term="S-Do(Mary Run)" RR="Motivation"> ← Subgoal1
  < communicative_act name="Suggest" term="S-Do(Giuseppe Run)" type="N">
  </communicative_act>
    <goal name="DescribeWhy" term="S-Do(Giuseppe Run)" RR="Joint"> Subgoal2
      <communicative_act name="Remind" term="V-Goal(Giuseppe
GoodHealth)">
      </communicative_act>
      <communicative_act name="Inform" term="Implies(Run
GoodHealth)">
      </communicative_act>
    </goal>
  </goal>
</plan></DPL>
```

S mette il piano in una 'agenda' nell'IS, gestita come stack:

```
(communicative_act name=Suggest, term=S-Do(Giuseppe Run))
((communicative_act name=Remind, term=V-Goal(Giuseppe GoodHealth));
(communicative_act name=Inform, term=Implies(Run GoodHealth)))
```

Inizia ad eseguire il suo piano, considerando il primo elemento dello stack:

S1: Dovresti andare a correre, Giuseppe!

Cede il controllo al generatore superficiale, che inserisce i tag APLM nella 'mossa di dialogo' e invia l'input al player del character animato.

Passa il controllo al modulo di *input interpretation* e osserva la reazione dell'utente.⁴⁶

U1: Perché?

Input Interpretation interpreta la User move come richiesta di giustificazione del suggerimento.

Questo attiva il Subgoal2 di S, che determina l'esecuzione delle altre due componenti dell'agenda:

**((communicative_act name=Remind, term=V-Goal(Giuseppe GoodHealth));
(communicative_act name=Inform, term=Implies(Run GoodHealth)))**

S2: Perché ci tieni alla tua salute e correre fa bene alla salute.

L'agenda di S, ora, è vuota. U risponde:

U2: Ma tu ci vai a correre?

Input Interpretation interpreta la User move come AskWhether; interroga la sua base di conoscenza sull'Agente; scopre di non poter correre perché è un agente artificiale.

Applicando una select-rule, pone la sua risposta in cima all'agenda:

**((communicative_act name=Answer, term=notCan(S,run));
(communicative_act name=Inform, term=ArtificialAgent(S)))**

S3: Io non posso correre, sono un agente artificiale!

Ora U risponde:

U3: Oh peccato, mi sarebbe piaciuto fare una corsetta con te!

Input Interpretation interpreta la User move come 'SocializationAttempt'.

47

Applicando una select-rule, decide di rispondere ricambiando la gentilezza e riprendendo il suo suggerimento iniziale, adattato al contesto del dialogo.

Pone in cima alla sua agenda una risposta:

**(communicative_act name=Acknowledge, term=Kind(U));
(communicative_act name=Suggest, term=RunWith(U,friend));
(communicative_act name=Remind, term=V-Goal(Giuseppe GoodHealth));**

S4: Grazie, sei gentile. Chissà, in un mondo virtuale! Per ora, puoi andare a correre con un'amica, se vuoi restare in buona salute...

U4: Ma ti pare che a vent'anni devo già pensare alla salute?

Input Interpretation interpreta la User move come rifiuto della motivazione.

S capisce che la sua strategia persuasiva è fallita e cerca di riparare cercandone una alternativa. Trova che un'altra ragione per correre è l'essere in forma. Aggiunge all'agenda un nuovo piano di persuasione:

(communicative_act name=Inform, term=Implies(Run InShape))

S5: Ma correre tiene anche in forma!

48

Esercizio 3

Sviluppa l'esempio precedente piu' in dettaglio,
e cioè definendo:
update-rule, select-rule, rappresentazione dello user
model, storia del dialogo ecc.

49

Esercizio 4

Sviluppa nei dettagli questo esempio:

S1: Mangi la frutta Maria?

U1: Si, ne vado matta!

S2: Hai ragione, soprattutto in estate la frutta è fantastica!

E i vegetali, li mangi?

...

Nota che il piano iniziale di S è, in questo caso, raccogliere informazioni sulle abitudini alimentari di U (se mangia frutta e vegetali).

Il piano viene messo momentaneamente da parte da S per rispondere con gentilezza a quello che interpreta una manifestazione di 'social attitude' da parte di U

50

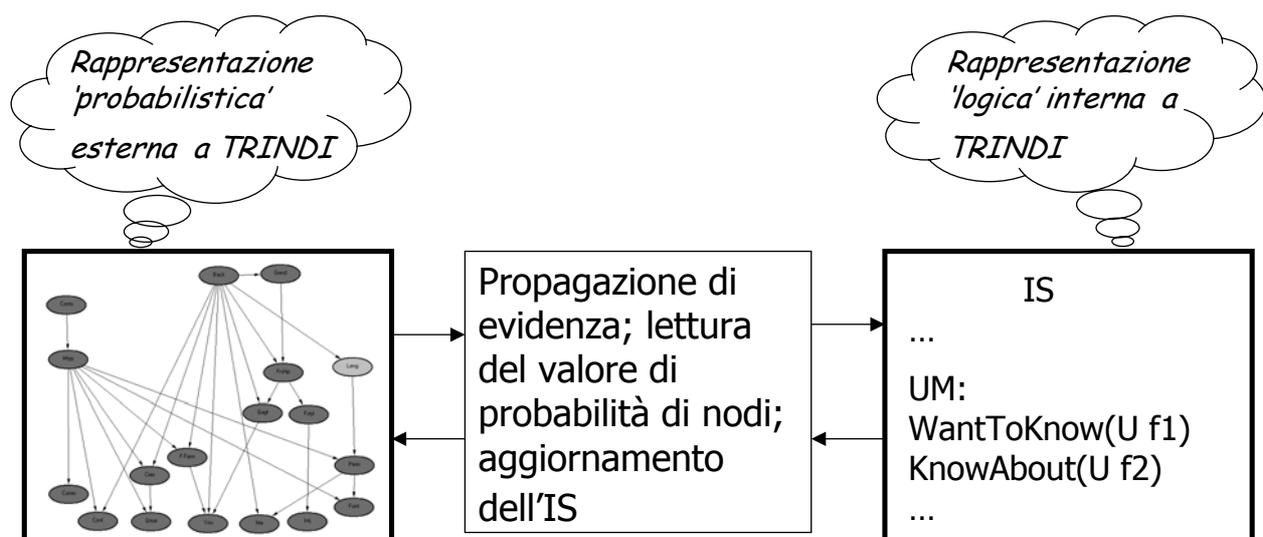
Collegamento con moduli esterni

Possono essere diversi, a seconda del tipo di dialogo che si vuole simulare:

- un generatore di linguaggio naturale più sofisticato
- un player di agenti animati conversazionali
- un modulo di user modeling basato su RCP
- un modulo di selezione di strategie persuasive
- un parser delle move dell'utente
- ...

51

Un esempio: come realizzare la consistenza fra modalità diverse per la rappresentazione della conoscenza relativa all'utente



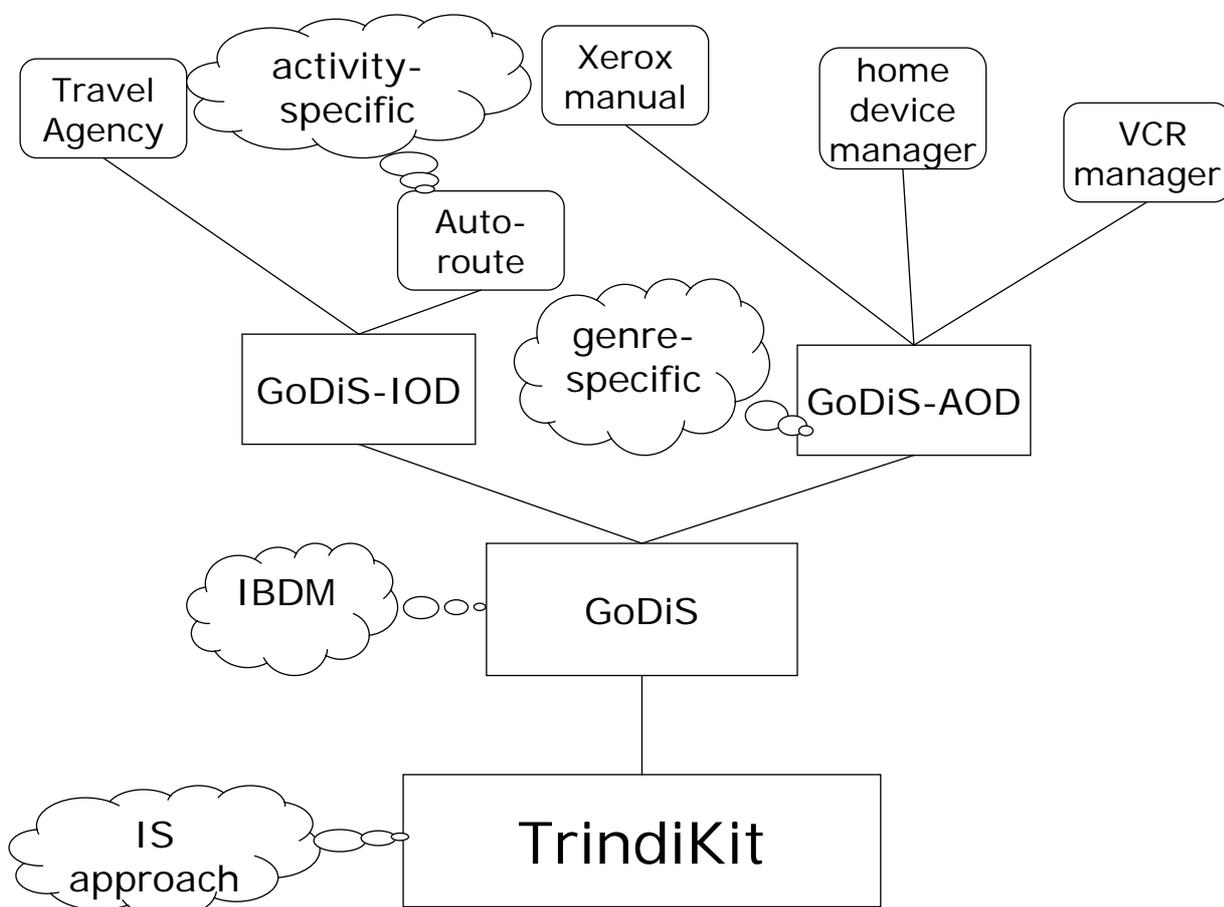
Fonte di conoscenza 'esterna': UM come RCP

52

GoDiS: an issue-based dialogue system

- Built using TrindiKit
 - Toolkit for implementing and experimenting with dialogue systems based on the information state approach
- Explores and implements Issue-based Dialogue Management (IBDM)
- Extends theory to more flexible dialogue
 - Multiple tasks, information sharing between tasks
 - Interactive Communication Management (ICM), including feedback, and grounding
 - Question accommodation
 - Negotiation of alternatives
 - Menu based action oriented dialogue

53



54

Un esempio di dialoghi in Godis

S: Wellcome to the travel agency!

U: Flights to Paris

S: Okay, you want to know about price. A flight. To Paris. Let's see.
What city do you want to go from?

U: London

S: Okay, from London. When do you want to travel?

U: In April, as cheap as possible.

S: Okay, in April, economy class. What day do you want to travel?

U: The fifth

S: Okay. The price is 300\$.

Nota:

un sistema basato sull'IS può simulare, nel caso più semplice, gli stessi dialoghi simulabili con un sistema basato su ATN.

55

Riferimenti

- D R Traum e S Larsson: The information state approach to dialogue management

<http://www.di.uniba.it/intint/people/ArticoliDid/Dialogo.zip>

e, per chi ha un interesse particolare sull'argomento, la Tesi di PhD di Timothy Bickmore (all'MIT-Boston), scaricabile dalla rete.

56