

Laurea Specialistica in Informatica
a.a. 2007-2008

Interazione Uomo-Macchina II:

Interfacce Intelligenti

Fiorella de Rosis

Laboratori di
Irene Mazzotta e Nicole Novielli

1

Introduzione

Unità 1: Ragionamento logico:

- Formalizzazione
- Risoluzione

Unità 2: Generazione di linguaggio naturale

- Teorie
- Metodi

Unità 3: Ragionamento incerto

- Reti Causali Probabilistiche
- Reti dinamiche

Unità 4: Modelli di Utente

- Modelli logici
- Modelli con incertezza

Unità 5: Comprensione del linguaggio naturale

Unità 6: Simulazione di dialoghi

- **Modelli basati su ATN**
- **Modelli basati su IS**

Unità 7: Affective Computing

Programma
del Corso

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 <i>modello dell'utente</i> è funzione del contesto: memorizza gli scambi avvenuti e i 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 lo realizza per passi, adattando 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 goals e beliefs e dei piani da realizzare 3

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).

Ripartiamo dai nostri esempi.

Un dialogo molto semplice: Information-giving, system initiative, formal attitude

S: Welcome 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: When would you like to leave?

U: On May 7th

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

U: Regular

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

.....

Qui, il piano di S è ottenere da U le informazioni di cui ha bisogno per prenotargli il volo e vendergli il biglietto (luogo di partenza e di arrivo, data, tipo di tariffa,...). 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 nel corso del dialogo, ma al massimo piccoli aggiustamenti.

Una prima variante: Information-giving, user moves to interpret

S: Welcome 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* and *have not much money to spend*.

S: OK. When would you like to leave?

.....

Qui, S lascia U più libero nell'interazione. E U fornisce diverse informazioni allo stesso tempo (luogo di partenza e di arrivo). Può farlo anche in ordine diverso da quello previsto nel piano di S ("*I would like to go to London on May 7th*"). Fornisce inoltre informazioni che richiedono un ragionamento, da parte di S, per essere interpretate e tradotte in informazioni utili per la scelta del volo.

Una seconda variante:

Information-giving, mixed initiative, friendly attitude

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

U: *Hi Irene, nice to meet you! My name is Carlo.* I would like to go to London starting from Bari, and have not much money to spend. *Can you help me?*

S: *I'll do my best for you Carlo!* There are several low-cost and very convenient companies going there. Do you mind going from Fiumicino to Ciampino, when you are in Rome?

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

S: Good. Just tell me when exactly you would like to leave then.

.....

Qui, S ha un doppio goal: dare informazioni e 'stabilire un clima di cooperazione' con U, avendo riconosciuto in lui un atteggiamento amichevole. Il suo piano, quindi, si complica. Il linguaggio d'interazione di U è meno rigido, e le mosse che può fare sono definite in modo più flessibile. Anche lo stile adottato da S è più amichevole.

7

Metodi per la simulazione di dialoghi

Vedremo due dei metodi principali per la simulazione di dialoghi:

- a. Simulatori a stati finiti e
- b. Simulatori basati sull'Information state.

Ne confronteremo vantaggi e svantaggi.

Cominciamo con il primo.

Simulatori a Stati Finiti, eventualmente 'aumentati'

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.

Cominciamo dalla formulazione più semplice...

9

Simulatori a Stati Finiti

Siano S il sistema e U l'utente;
supponiamo che S e U si alternino nelle mosse di dialogo.

Chiamiamo:

SM: $\{sm_1, sm_2, \dots, sm_n\}$ l'insieme delle *mosse che il sistema può compiere*;

UM: $\{um_1, um_2, \dots, um_p\}$ l'insieme delle *mosse che l'utente può compiere*.

S: $\{s_0, s_1, s_2, \dots, s_q\}$ l'insieme degli *stati* del dialogo; questo insieme potrà essere distinto in due sottoinsiemi:

ASM: $\{s_i, s_j, \dots\}$ insieme degli stati raggiunti dopo una mossa di S
(che rappresentiamo con cerchietti *gialli*)

AUM: $\{s_h, s_k, \dots\}$ insieme degli stati raggiunti dopo una mossa di U
(che rappresentiamo con cerchietti *verdi*)

Inoltre, chiameremo:

IS: $\{s_0\}$: lo stato iniziale del dialogo

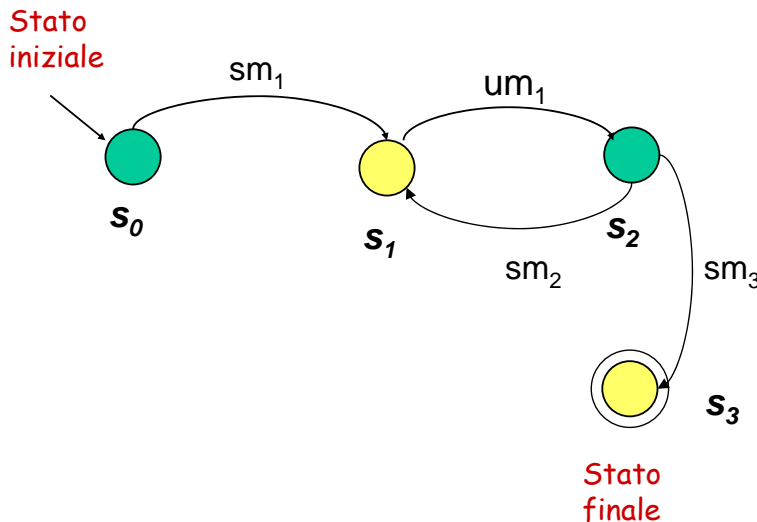
ES: $\{s_m, s_p, \dots\}$: l'insieme degli stati finali del dialogo (che rappresentiamo con un doppio cerchietto, verde o giallo).

... *vediamo subito un esempio...* 10

Associamo agli archi le mosse di S e di U

$(s_0, sm_1, s_1); (s_1, um_1, s_2); (s_2, sm_2, s_1); (s_2, sm_3, s_3);$

IS : s_0 ; ES = s_3



S fa la prima mossa.

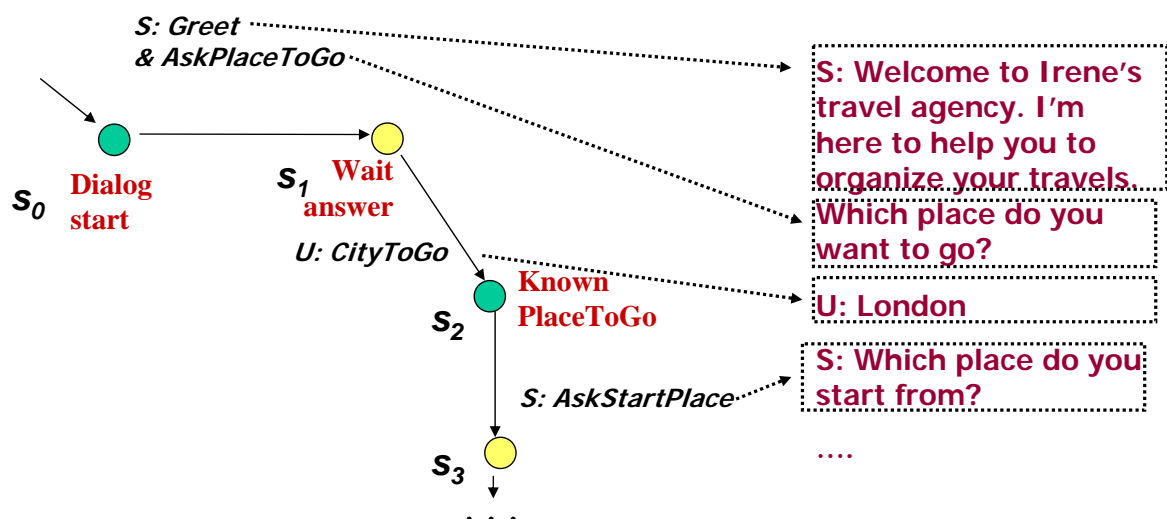
s_0, s_2 rappresentano stati in cui ha mosso S;

s_1, s_3 stati in cui ha mosso U.

In questo caso c'è un unico stato finale, ma potrebbero essercene diversi: ad esempio, S o U potrebbero concludere il dialogo.

11

Esempio: il dialogo 'di base' della Travel Agency

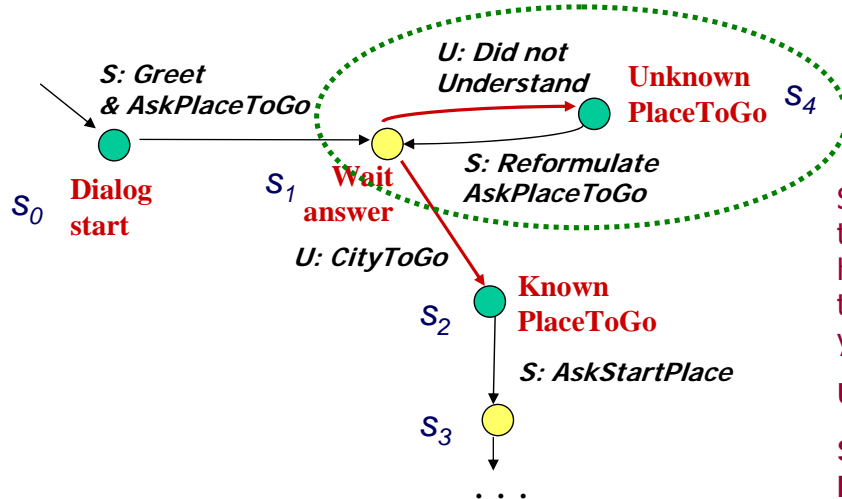


Nota: abbiamo semplificato la descrizione degli atti comunicativi, rispetto a quella introdotta nell'Unità 3.

*Abbiamo aggiunto agli stati una **descrizione della situazione del sistema**: ' s_0 : Dialog start', ' s_1 : Wait answer', ' s_2 : Known PlaceToGo',...*

12

Esempio: lo stesso dialogo, con un caso di incomprensione



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

U: Eeeeh???

S: Sorry: I need to know the name of the place in which you want to go

U: OK, London

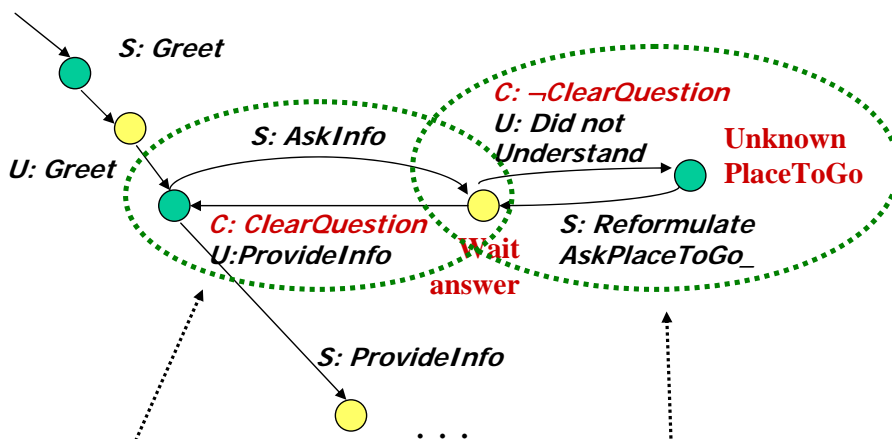
S: Which place do you start from?

....

13

Qui, abbiamo inserito un nuovo stato ('S₄: Unknown PlaceToGo') e creato una 'digressione' rispetto al piano principale, per gestire la situazione. Ora, da uno stato possono partire diversi archi!

Generalizziamo, e aggiungiamo agli archi una condizione



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

U: Hi.

S: Which place do you want to go?

U: Eh???

S: I need to know the name of the place in which you want to go

U: OK, London

S: Which place do you start from?

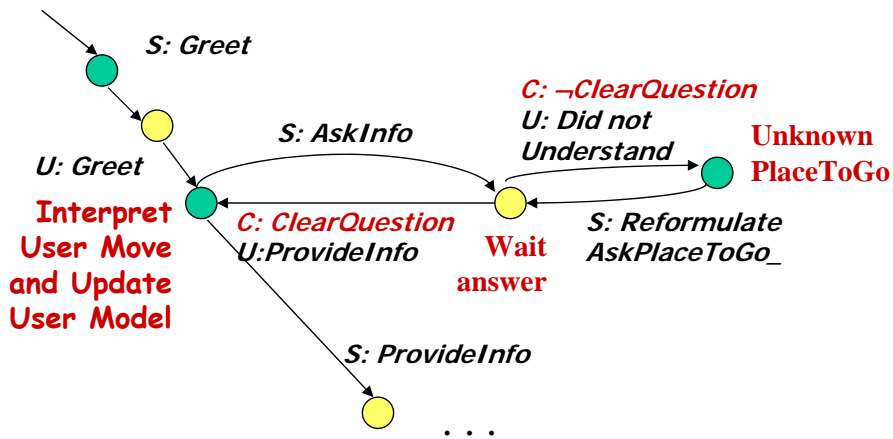
....

Fase di question-answering con domanda di S compresa da U

Fase di question-answering con domanda di S **non** compresa da U

14

Ora, integriamo l'informazione sugli stati



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

U: Hi.

S: Which place do you want to go?

U: Eh???

S: I need to know the name of the place in which you want to go

U: OK, London

S: Which place do you start from?

....

Qui, abbiamo aggiunto agli stati 'verdi' una descrizione delle **azioni svolte dal sistema** dopo una user move: sua interpretazione e aggiornamento del modello di utente.

Questo significa che il sistema deve includere **operazioni di lettura / scrittura da registri o db**, o di chiamata a software esterni di gestione dello UM: ad esempio, interpretazione di una mossa in linguaggio naturale e aggiornamento di un BN)

15

Quali azioni può svolgere il sistema?

A seconda della modalità d'interazione scelta, il sistema può **preparare l'acquisizione della prossima mossa dell'utente** mostrando semplicemente un elenco di risposte possibili oppure un campo di testo libero da riempire o preparandosi a registrare un comando vocale.

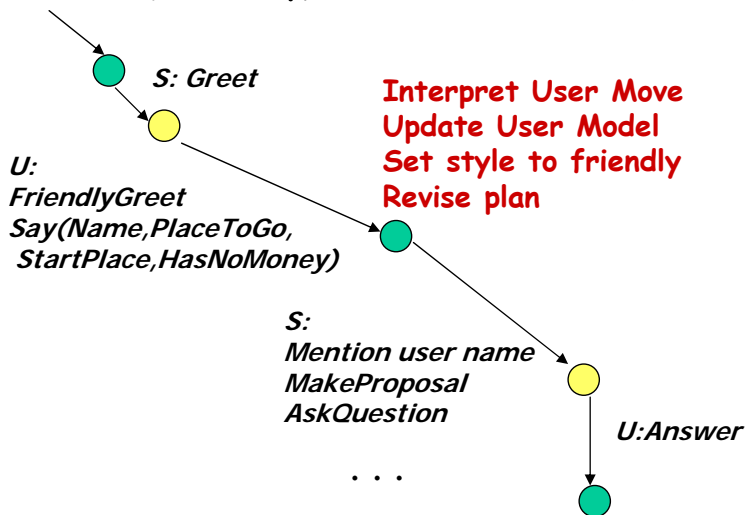
Le azioni di **interpretazione della mossa** possono consistere nella memorizzazione dell'opzione selezionata nell'elenco, o nel riconoscimento di una stringa in linguaggio naturale, con i metodi che abbiamo visto nell'Unità precedente.

L'aggiornamento del modello di utente può consistere in semplici operazioni di lettura/aggiornamento di un DB o può richiedere quelle forme di ragionamento logico o incerto, con propagazione di evidenza in una RCP, che abbiamo visto nelle scorse unità.

La **preparazione della mossa successiva** dipende dalla mossa precedente dell'utente (e, più in generale, dalla 'storia del dialogo')

16

Vediamo ora la seconda variante del dialogo



*L'andamento del dialogo è molto meno prevedibile che nei casi precedenti.
La ATN che permette di rappresentare tutte le situazioni possibili di un dialogo 'libero' diventerebbe molto più complessa.
Dovremo pensare ad una soluzione alternativa.*

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

U: Hi Irene, nice to meet you! My name is Carlo. I would like to go from Bari to London, and haven't much money to spend. Can you help me?

S: I'll do my best for you Carlo! There are several low-cost and convenient companies going there. Do you mind going from Fiumicino to Ciampino, when you arrive to Rome?

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

....

17

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

Studiamo la struttura del dialogo dell'American Express

18

Codifichiamo le mosse di S e di U

S: A at American Express may I help you?
 U: yeah this is B. I need to make some travel arrangements

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: ok thank you very much
 S: thanks a lot

Greet
 SayName & RequestTA

AskInfo
 ProvideInfo
 AskInfo
 DoubtfullInfo
 Specify
 Confirm

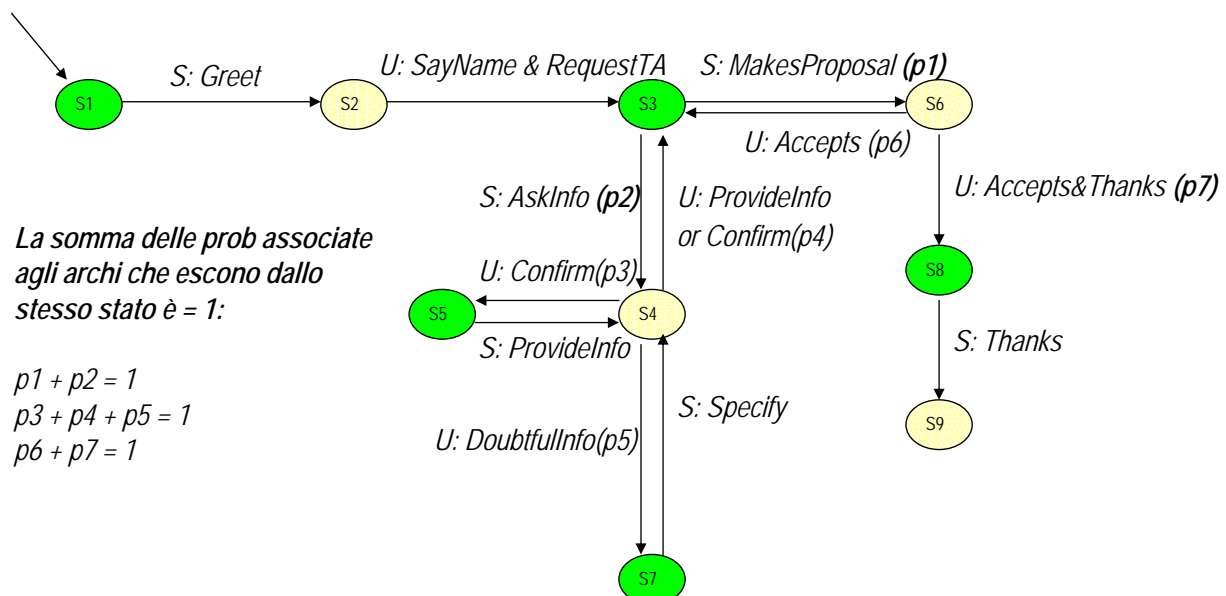
ProvideInfo
 Confirm
 AskInfo
 ProvideInfo
 AskInfo
 DoubtfullInfo

Specify
 Confirm
 MakeProposal
 Confirm

MakesProposal
 Accepts
 MakesProposal
 Accepts&Thanks
 Thanks

19

Rappresentiamo il dialogo come diagramma a SF associando agli archi dei valori di probabilità

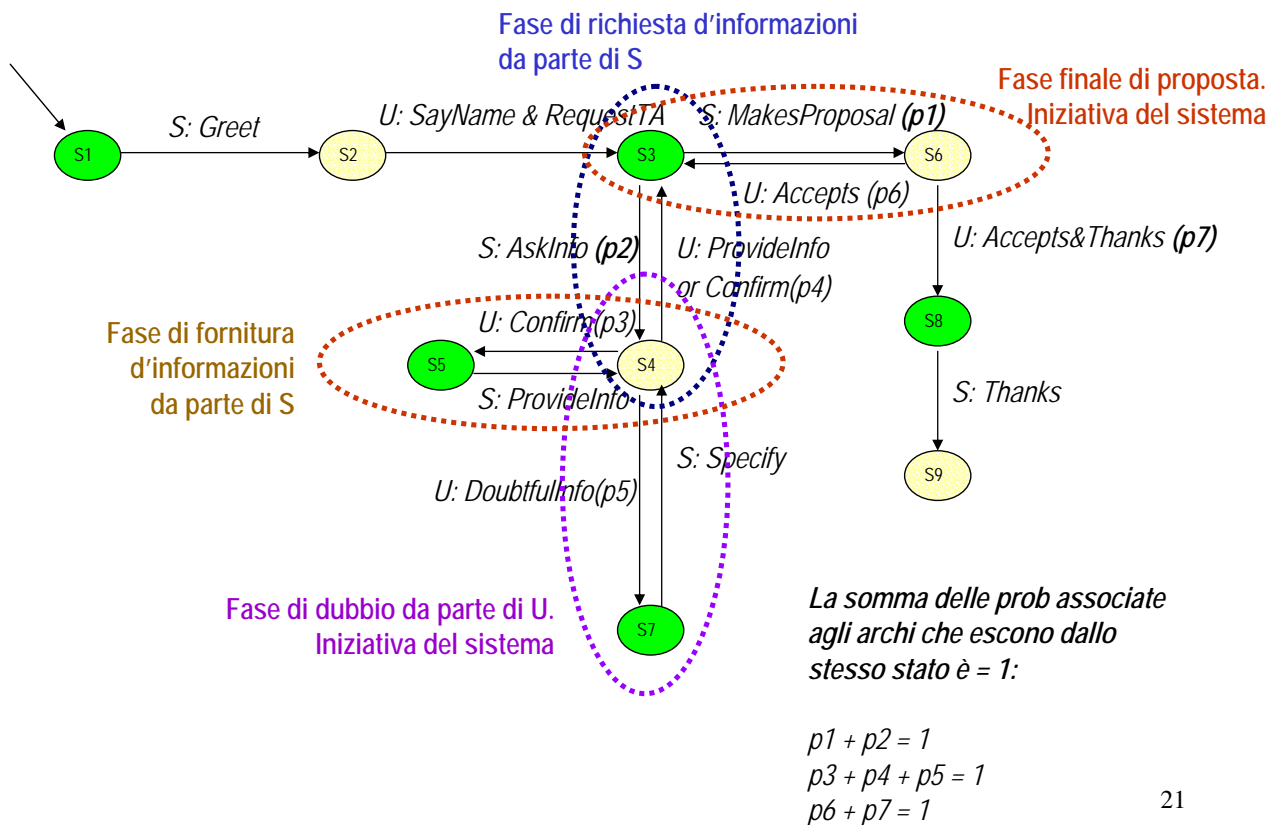


Il dialogo precedente passa fra gli stati:

S1,s2,s3,s4,s3,s4,s7,s4,s5,s4,s3,s4,s7,s4,s3,s6,s3,s6,s3,s6,s8,s9

20

Possiamo identificare alcune 'fasi' del dialogo



21

Dialoghi come Augmented Transition Networks (ATN)

Una ATN è un diagramma di transizione fra gli stati, ai cui archi possono essere associati una *label* e una *condizione*.

Le **labels** possono denotare mosse (di S o di U) oppure chiamate ad **altre ATN**: si tratta quindi di diagrammi ricorsivi.

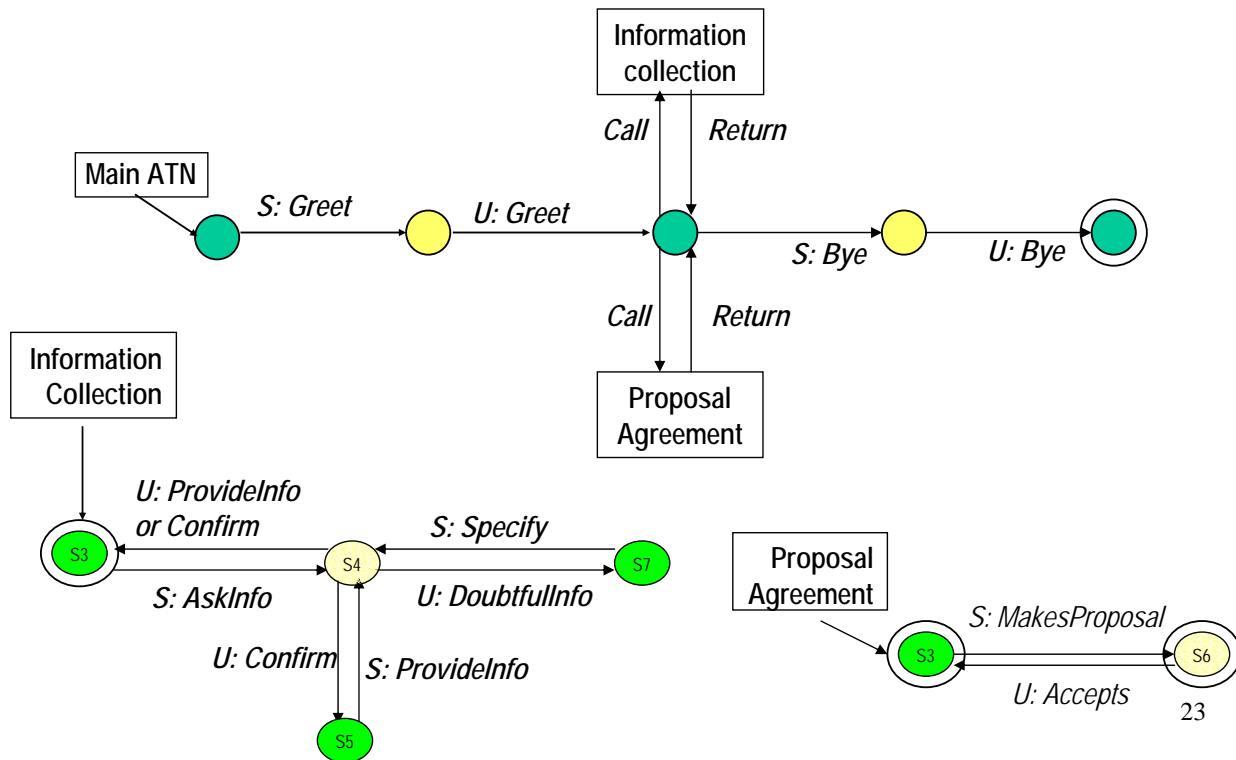
Le **condizioni** devono essere soddisfatte perché l'arco possa essere attraversato.

Un insieme di **registri** permette di memorizzare risultati intermedi oppure lo stato dell'esplorazione dell'ATN.

22

Introduciamo l'innestamento del diagramma dell'AmExpr

S'introduce uno stack che permette di saltare dall'ATN principale ad una secondaria.



Come rappresentare l'ATN che descrive l'andamento di un dialogo? Uno sguardo agli scripting language

Cosa abbiamo bisogno di rappresentare?

Siamo partiti da una struttura del tipo:

$(s_0, sm_1, s_1); (s_1, um_1, s_2); (s_2, sm_2, s_1); (s_2, sm_3, s_3);$

IS : s_0 ; ES = s_3

L'abbiamo via via arricchita, inserendo:

label associate agli archi;

condizioni associate agli archi;

azioni associate agli stati.

Abbiamo quindi bisogno di rappresentare una struttura del tipo:

$(s_i, cond_h), sm_i, s_j);$

Abbiamo quindi bisogno di estendere il linguaggio definito all'inizio:

ATN_NAME: $name_w$ una stringa che indica il nome dell'ATN

SM: $\{sm_1, sm_2, \dots sm_n\}$ l'insieme delle *mosse che il sistema può compiere*;

UM: $\{um_1, um_2, \dots um_p\}$ l'insieme delle *mosse che l'utente può compiere*.

S: $\{s_0, s_1, s_2, \dots s_q\}$ l'insieme degli *stati* del dialogo; questo insieme potrà essere distinto in due sottoinsiemi:

ASM: $\{s_i, s_j, \dots\}$ insieme degli stati raggiunti dopo una mossa di S
(che rappresentiamo con cerchietti *gialli*)

AUM: $\{s_h, s_k, \dots\}$ insieme degli stati raggiunti dopo una mossa di U
(che rappresentiamo con cerchietti *verdi*)

IS: $\{s_0\}$: lo stato iniziale del dialogo

ES: $\{s_m, s_p, \dots\}$: l'insieme degli stati finali del dialogo (che rappresentiamo con un doppio cerchietto, verde o giallo).

COND: $\{c_1, c_2, \dots c_r\}$: l'insieme delle condizioni associate agli archi

SYS_ACT: $\{act_1, act_2, \dots act_r\}$: l'insieme delle azioni, associate agli stati appartenenti a AUM, che il sistema può compiere.

La descrizione dell'ATN sarà del tipo:

$(s_h, (um_j, p_j), s_k);$

Da $s_h \in ASM$ (*stato giallo*),
con la user move um_j e livello di probabilità p_j
si passa a $s_k \in AUM$



$(s_k, ((sm_h \vee name_w), c_i, p_j), s_r);$

Da $s_k \in AUM$ (*stato verde*), in cui S compie l'azione act_q ,
con la system move sm_h o una call all' ATN $name_w$,
se la condizione c_i è verificata, e livello di probabilità p_j
si passa a $s_r \in ASM$



Quale rappresentazione posso usare?

Esistono diverse rappresentazioni possibili.

Ad esempio,

una struttura xml

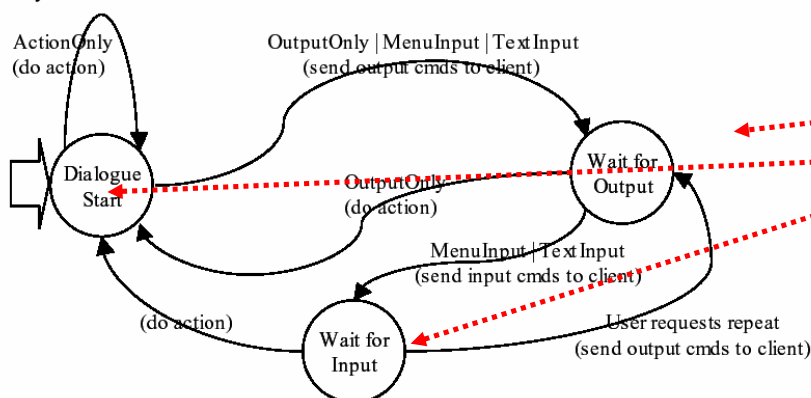
uno scripting language di tipo diverso.

Vediamone uno piuttosto noto, alla base di un simulatore di dialogo sviluppato (... arieccolo) all'MIT.

27

Il Sistema FitTrack dell'MIT

obiettivo: dare suggerimenti sullo sport, seguendo i soggetti nel tempo



Gli stati descrivono la situazione del sistema

Figure 8-7. Fragment of ATN Executive State Machine

28

Il Sistema FitTrack dell'MIT

obiettivo: dare suggerimenti sullo sport, seguendo i soggetti nel tempo

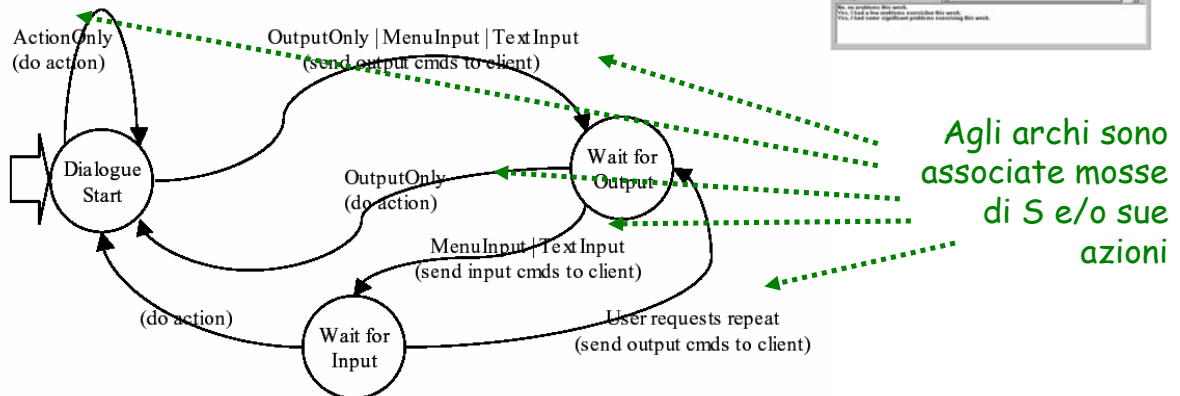
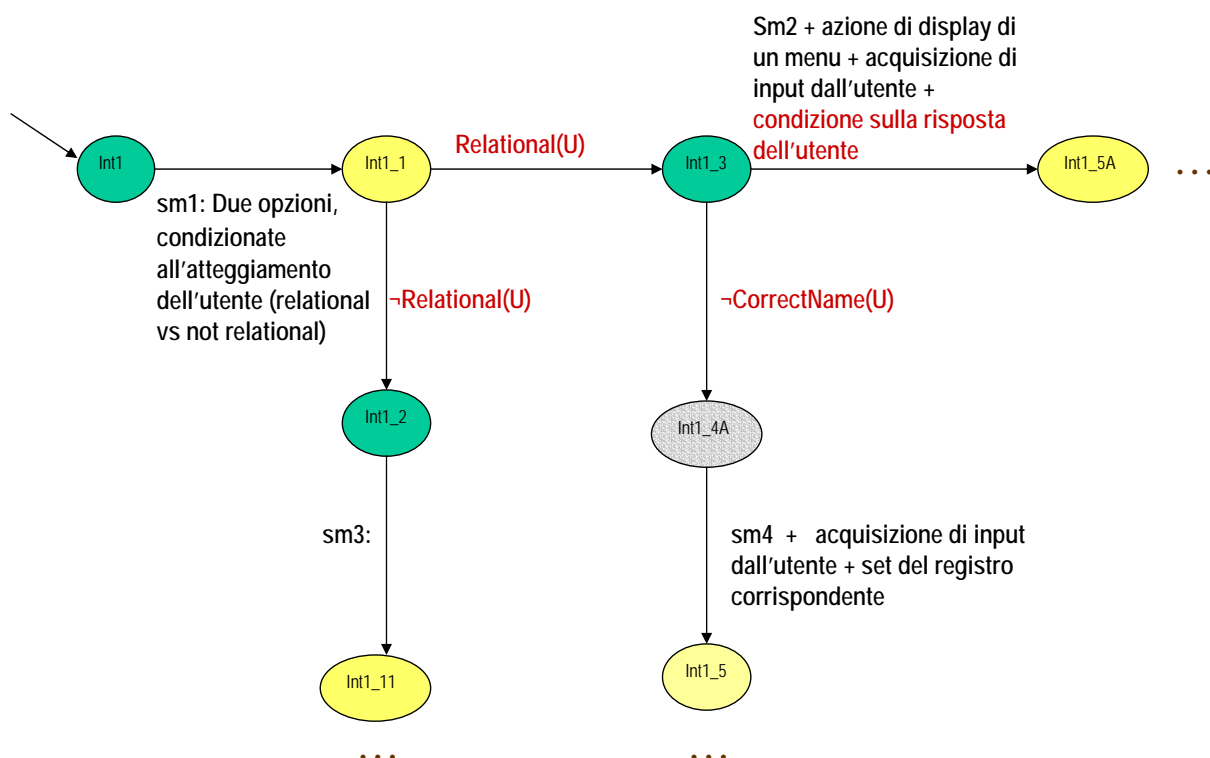


Figure 8-7. Fragment of ATN Executive State Machine

Una macchina a stati finiti + uno *stack* e una serie di registri.
 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*).
 Operazioni di lettura e scrittura da registri consentono di realizzare l'adattamento.

Rappresentiamo un frammento di dialogo



Quale scripting language?

Una volta definita la struttura del dialogo, occorre definire:

- la modalità d'interazione: menu-based vs testo libero vs speech per U; testuale o speech o agente animato per S;
- i 'registri' che S deve gestire;
- le 'azioni' (funzioni) che S deve essere in grado di compiere;
- il linguaggio in cui rappresentare l'ATN;

In FitTrack, le funzioni sono:

- gestione di registri
 - sulle feature di U
 - sull'esercizio fisico utile per U
 - generali
- input (get text from um)
- output (display sm)
- salto al nuovo stato

31

Funzioni in FitTrack

Script Primitives

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.


32

... Continuano le funzioni di FitTrack

ATN Change Functions	State	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).

33

Lo Scripting Language di FitTrack

STATE: Si  Tag per l'animazione del character

AGENT: \$<BEAT>Are you going to work out tomorrow?</BEAT>\$

USERMENU: I can't. => \$ Go("Sj"); \$

 Yep. => \$ Go("Sk"); \$

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

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

34

Questo frammento di script corrisponde al lucido che descriveva l'ATN

"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"); $
```

35

FitTrack: Un esempio di dialogo

Fase iniziale di raccolta d'informazioni

-
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

36

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)

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).

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 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?

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.³⁹

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, accettare, ...
*coinvolgono il passaggio d'informazioni su 'atteggiamenti di gentilezza' di S*⁴⁰

Elaborazione degli atti comunicativi di U

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

Siano

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)

41

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...

42

Ragionamento sull'atto comunicativo di U

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

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

KnowAbout(S, WeFo°(Finistere)) ? No!

S conclude che non le conosce

¬KnowAbout(S, WeFo°(Finistere))

Ma S il Finistere è nel nord della Francia

Is-in(Finistere, NorthernFrance)

e, in particolare, la Manica è nel nord della Francia

Is-in(Channel, NorthernFrance)

e S conosce le previsioni in questa zona

KnowAbout(S, WeFo°(NorthernFrance))

S deduce allora che U voglia conoscere le previsioni nel nord della Francia

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

e, in particolare, nella Manica

(Bel S WantToKnow(U, WeFo°(Channel)))

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

Bel S [WantToKnow (U,x) \wedge ¬KnowAbout(S,x) \wedge (Similar(x,y) \vee IsIn(x,y)) \wedge KnowAbout(S,y) \rightarrow WantToKnow(U,y)]

43

Traduzione del ragionamento nella move successiva

Piano che combina:

Answer (S ¬KnowAbout(S, WeFo°(Finistere)))

e

Answer(S KnowAbout(S WeFo°(Channel)))

Answer(S KnowAbout(S WeFo°(NorthernFrance)))

... 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 delle Answer e della Offer)

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.

45

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'):

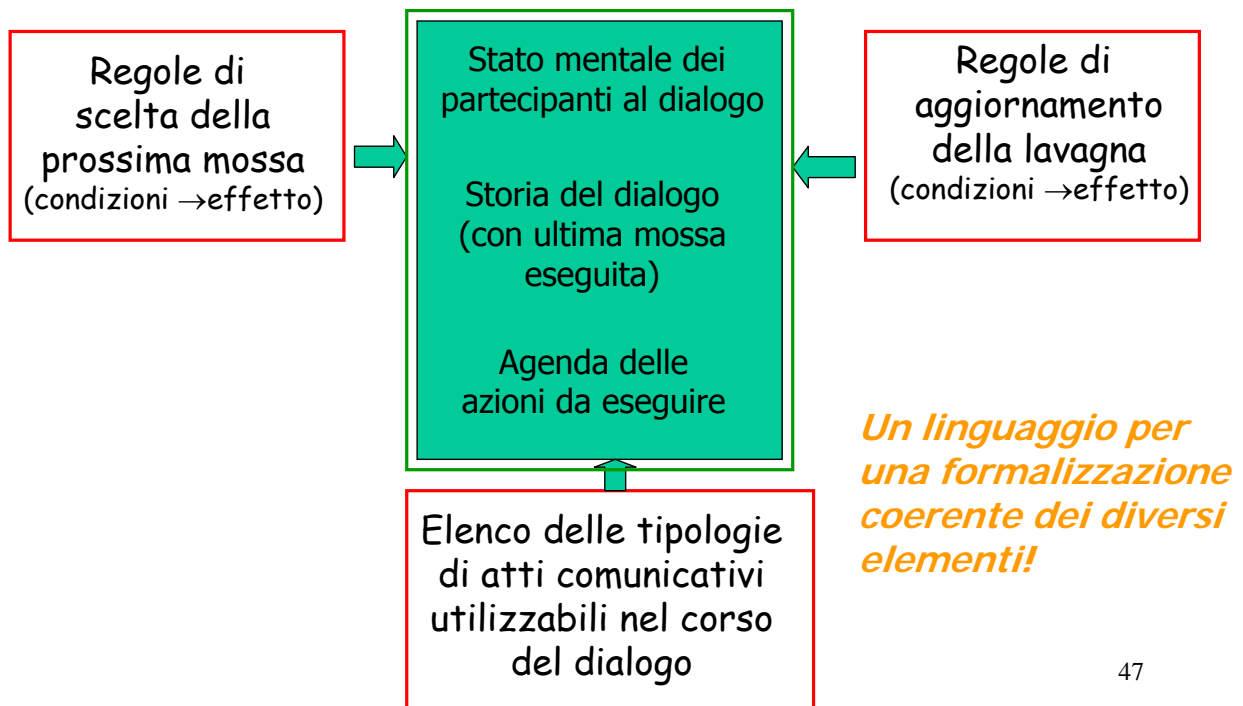
Su U:	WantToKnow(U, WeFo°(Finistere))	shared
	WantToKnow(U, WeFo°(NorthernFrance))	private
	WantToKnow(U, WeFo°(Channel))	private

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)))
Answer (S KnowAbout(S, WeFo°(NorthernFrance)))
OfferInfo (S, WeFo°(Channel))
```

46

Una Lavagna 'attiva' (Information State)



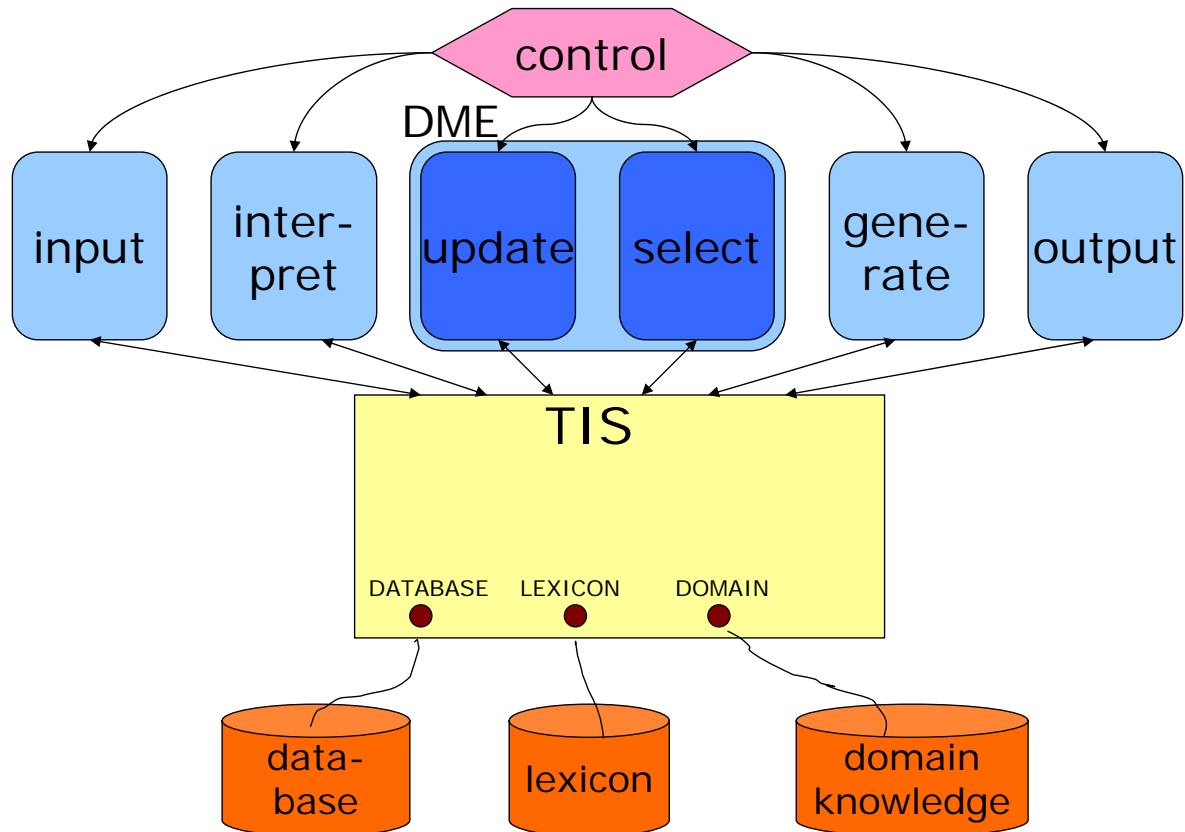
47

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**

48

Architettura di Trindikit



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

49

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.

50

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:

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

51

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

52

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
```

54

Generatore

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

55

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 vent'anni devo già pensare alla salute?

S5: Ma correre tiene anche in forma!

56

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.

57

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 APML 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.⁵⁸

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'.

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!

60

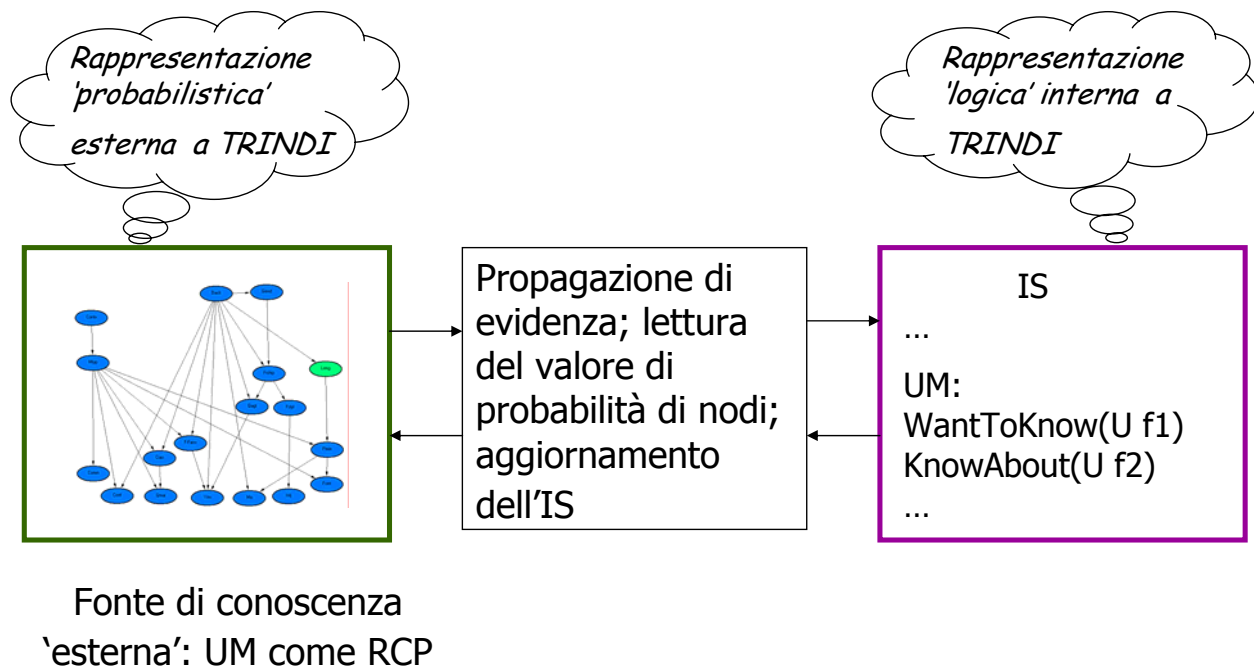
Collegamento con moduli esterni

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
- ...

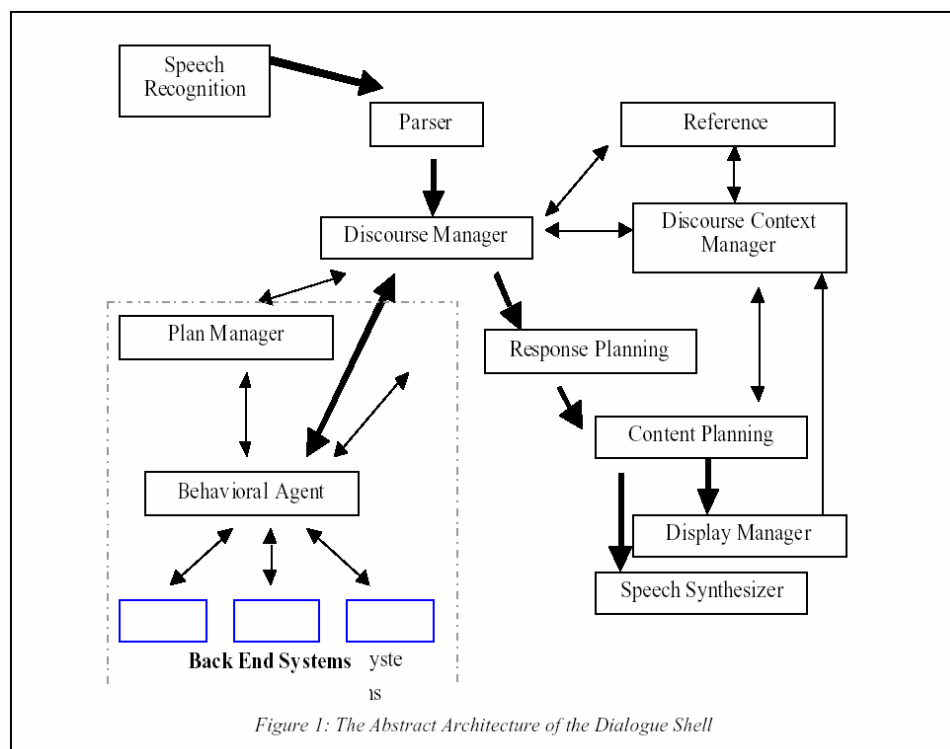
61

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



62

Architettura (generale) di un simulatore di dialoghi



63

Riferimenti

- D R Traum e S Larsson: The information state approach to dialogue management
<http://www.di.uniba.it/intint/people/ArticoliDid/Dialogo.zip>