

Metodologia di composizione delle regole di adattamento

Survey

Nel presente documento si riassumono i contenuti del deliverable "**D2.3_2 Metodologia di composizione delle regole di adattamento**" relativo all'attività *RI 2.3 Definizione di modelli per la descrizione di regole di adattamento di contenuti multimediali e servizi interattivi*, nell'ambito del secondo Obiettivo Realizzativo (OR 2) "Studio di servizi e contenuti intelligenti".

In SAPI verrà effettuato l'adattamento delle *Entity* a una data *situazione* (*user-context*) attraverso la definizione ed esecuzione di **un appropriato piano di composizione delle regole di adattamento** selezionate tra quelle proprie e quelle relative alle *Entity* nel caso quest'ultime siano *Entity* intelligenti. Ricordiamo che per *Entity* intelligente in SAPI si intende qualsiasi entità annotata semanticamente dotata di regole di adattamento proprie. Il processo di composizione delle regole consiste nel definire un workflow (**adaptation path**) di regole atomiche o composte allo scopo di raggiungere un obiettivo di adattamento più o meno complesso (**adaptation goal**). Quindi, data una *Entità* ed una *Situazione* o in particolare un *cambiamento di una situazione*, l'*adattore* individua un *adaptation goal* da raggiungere attraverso una *composizione di regole* atomiche e/o composte rappresentante il percorso di adattamento (*adaptation path*) *ottimo*.

Nel Deliverable D.2.3_1 si è mostrato come una regola di adattamento SAPI possa essere descritta alla stessa stregua di un servizio (di adattamento) nell'accezione ontologica di OWL-S. Infatti, una regola SAPI posto:

$$\text{Event} \wedge \text{Condition} = \text{Precondition}$$

può essere informalmente scritta come segue:

If

Precondition

Then

Perform (Adaptation Process)

Se, inoltre, si introducono nella descrizione semantica della regola "as a whole" dei *quality code*, si ottiene come risultato che una regola *r* può essere formalmente rappresentata dalla seguente 6-pla:

$$r = (P, I_p, O_p, C_R, E_R, Q_R)$$

dove:

- P è un processo (metodo) di adattamento atomico o composto (Horn *rule head*);
- I_p sono i parametri in input del processo di adattamento;
- O_p sono i parametri in output del processo di adattamento;
- C_R sono le precondizioni del processo (Horn *rule body*);
- E_R sono gli effetti (nell'accezione OWL-S) del processo di adattamento;
- Q_R sono i parametri (attributi) di qualità della regola (ad es., il tempo di esecuzione, il costo, etc.).

Nella head di ciascuna regola appariranno quindi le condizioni relative alle caratteristiche relative alla data situazione (user-context):

$$\mathbf{If} \left(((C_D = (d_1, d_2, \dots, d_k)) \right.$$

$$\mathbf{AND} (C_N = (n_1, n_2, \dots, n_l)) \mathbf{AND} (C_U = (u_1, u_2, \dots, u_n)) \mathbf{AND} (C_E = (e_1, e_2, \dots, e_l)) \left. \right)$$

$$\mathbf{Then} (Entity\ Adaptation\ Plan = Pa)$$

dove:

- C_D sono le caratteristiche del dispositivo d'utente;
- C_N sono le caratteristiche del canale di comunicazione (Rete);
- C_U sono le caratteristiche dell' *utente*;
- C_E sono le caratteristiche dell' *environment*;
- Pa rappresenta il piano (processo) di adattamento della Entity.

Definizione 1 : Stato di una Entity

Per stato \mathbf{S} di una Entity, $S(E)$, si intende un' istanza della n-pla di metadati che la descrive. Per esempio, nel caso di una Immagine \mathbf{I} descritta dalla 4-pla (*formato, profondità di colore, altezza, larghezza*),

$$\mathbf{S}(\mathbf{I}) = (jpeg, 24\ bpp, 200\ pixels, 400\ pixels)$$

Definizione 2 : Task di trasformazione

Per Task di trasformazione \mathbf{t} di una Entity intendiamo uno step atomico del meccanismo astratto di trasformazione dell' Entity collocato all'interno di un adaptation plan. Un esempio di pipeline (workflow) di task di trasformazione è rappresentata in Figura 1.

Un task di trasformazione è descritto informalmente e qualitativamente in termini di Input e Output e della funzione di trasformazione che esegue.

Alcuni esempi di funzioni di trasformazione sono le seguenti:

- compressione;
- decompressione;
- scaling;
- riduzione della profondità di colore;
- conversione di formato;
- traduzione da una modalità di interazione ad un'altra;
- summarization;
- etc.

In SAPI, la *sequenza di trasformazione* corretta ($\{t_1 \rightarrow t_2 \rightarrow \dots \rightarrow t_n\}$) sarà univocamente determinata.

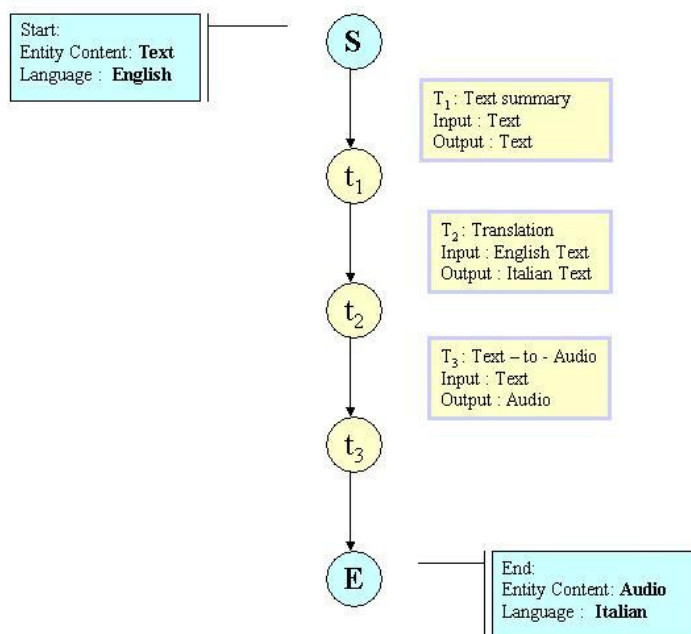


Figura 1 - Esempio di pipeline di Task di Trasformazione di una Entity.

Definizione 3 : Task-Rule Mapping function

Siano T e R rispettivamente l'insieme dei task di trasformazione e l'insieme delle regole di adattamento. Verrà denominata *Task-Rule Mapping function* la seguente funzione:

$$\rho : T \rightarrow \mathbf{n}(R)$$

dove $\mathbf{n}(R)$ è un power set di R .

Per ciascun task t_i , applicando la funzione ρ si individuano le regole di adattamento che contengono nella *head* i metodi di adattamento in grado di effettuare il task t_i .

Definizione 4 : Grafo di adattamento

A partire da una data pipeline di trasformazione $T = \{t_1 \rightarrow t_2 \rightarrow \dots \rightarrow t_n\}$, viene definito Grafo di adattamento il Grafo Diretto Aciclico così ottenuto:

1. Per ciascun task t_i , applicando la funzione \mathbf{p} si individuano (*Rule Discovery*) le regole di adattamento candidate che contengono nella *head* i metodi di adattamento che possono eseguire t_i ; si otterrà così un insieme di regole di adattamento r_{ij} per ciascun task t_i .

$$G = \{ \langle \mathbf{t}_1, r_{11}, r_{12}, \dots \rangle, \langle \mathbf{t}_2, r_{21}, r_{22}, \dots \rangle, \dots, \langle \mathbf{t}_n, r_{n1}, r_{n2}, \dots \rangle \}$$

2. Per connettere due regole r_{ki} r_{k+1j} devono essere soddisfatte le seguenti condizioni di compatibilità:
 - $r_{k+1j.Pre} \subseteq (r_{ki.Pre} \cup r_{ki.C_Eff})$, dove $r_{k+1j.Pre}$ sono le precondizioni di $r_{k+1j.Pre}$, $r_{ki.Pre}$ sono le precondizioni di r_{ki} e $r_{ki.C_Eff}$ sono le condizioni che devono eventualmente essere soddisfatte per gli effetti di r_{ki} ;
 - parametri QoS compatibili con quelli richiesti.

Un esempio del risultato che si otterrà è rappresentato dal grafo di Figura 2.

3. dal grafo così ottenuto bisogna rimuovere i nodi non interconnessi;

Nel grafo di Figura 2 sono presenti tre possibili *adaptation path* :

- $P_1 = \{(t_1, r_{11}), (t_2, r_{22}), (t_3, r_{31})\}$;
- $P_2 = \{(t_1, r_{12}), (t_2, r_{22}), (t_3, r_{31})\}$;
- $P_3 = \{(t_1, r_{12}), (t_2, r_{23}), (t_3, r_{32})\}$.

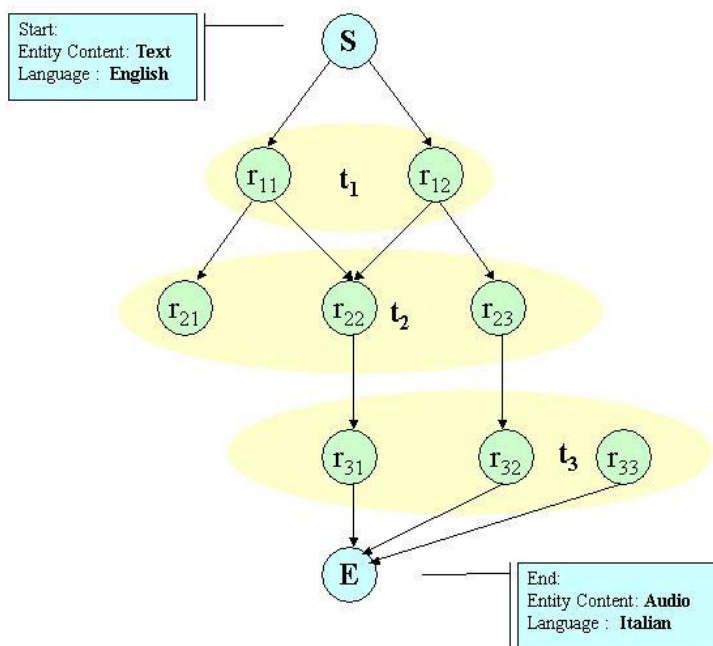


Figura 2 - Esempio di Grafo di Adattamento di una Entity.

Trovare il path di adattamento più conveniente tra quelli candidati è un problema di ottimizzazione che verrà trattato più avanti.

Determinazione della sequenza dei task di trasformazione

Il problema della determinazione della sequenza dei task (astratti) di trasformazione di una Entity può essere enunciato nel modo seguente:

"Data una Entity di un certo tipo in un determinato stato (iniziale), qual' è la sequenza di task di trasformazione da effettuare per portare l' Entity in uno stato (finale) tale che rappresenti, tra quelli possibili, il più adatto ad una determinata (istanza di) Situazione."

Un modo immediato e concettualmente semplice per risolvere tale problema è quello rule-based, vale a dire: una volta definito lo spazio degli stati di una Entity E_s , lo spazio delle Situazioni S e quello degli Adaptation Goal G si potrebbe definire un insieme di regole R del tipo

$$\begin{array}{l}
 \text{If } (E_{si} \wedge S_j) \\
 \text{Then} \\
 G_{ij}
 \end{array} \quad (1)$$

Introdurre poi un letterale che rappresenti semanticamente un obiettivo di adattamento (ad es. portare la Entity nello stato finale *AudioItalianLanguage*) o una sequenza di task di trasformazione (*EnglishToItalianTranslation* → *TextToSpeech*) è sostanzialmente la stessa cosa.

Ovviamente una tale soluzione ha lo svantaggio che la cardinalità dell'insieme R dipende strettamente dalla complessità delle Situazioni e della tipologia delle Entity ed in certi casi può crescere a dismisura. Ma, come fare altrimenti a conoscere l'adaptation goal? In ogni problema di AI planning il Goal è conosciuto a priori e fa parte integrante dell'enunciato del problema!

La soluzione proposta in SAPI, illustrata in figura 3, si basa su questo assunto e ha come scopo quello di sintetizzare l'obiettivo di adattamento finale a partire da obiettivi parziali (corrispondenti ai task di trasformazione) dedotti attraverso l'applicazione di opportune Regole di Trasformazione alle singole caratteristiche o a gruppi di caratteristiche di una situazione. In altre parole, ciò equivale a decomporre una data situazione in situazioni elementari, in particolare costituite da una sola caratteristica. Applicando le Regole di trasformazione alle singole situazioni elementari, piuttosto che all'intera situazione, le Regole di trasformazione saranno certamente in numero minore ed enormemente semplificate. La necessità di prevedere situazioni elementari con più di una caratteristica nasce dal fatto che alcune caratteristiche per loro natura vanno considerate contemporaneamente "as whole" dalle Regole di trasformazione. È questo il caso per esempio delle abilità dell'utente e delle capability del dispositivo d'utente : per un utente non vedente che utilizzi un dispositivo che è dotato della modalità vocale (con altoparlante e microfono funzionanti) una Entity di tipo *Testo* può essere trasformata in una Entity di tipo *Audio*. Un altro caso è il seguente: se un utente si mette in movimento, è dotato di palmare ed è notte allora è opportuno ingrandire la dimensione dei caratteri di una Entity di tipo *Testo* ed evidenziarli (highlighting). Per contro, nel caso si abbia una Entity di tipo *Testo* scritta in Inglese è sufficiente conoscere la lingua o le lingue specificate nelle preferenze dell'utente per decidere se va tradotta oppure no.

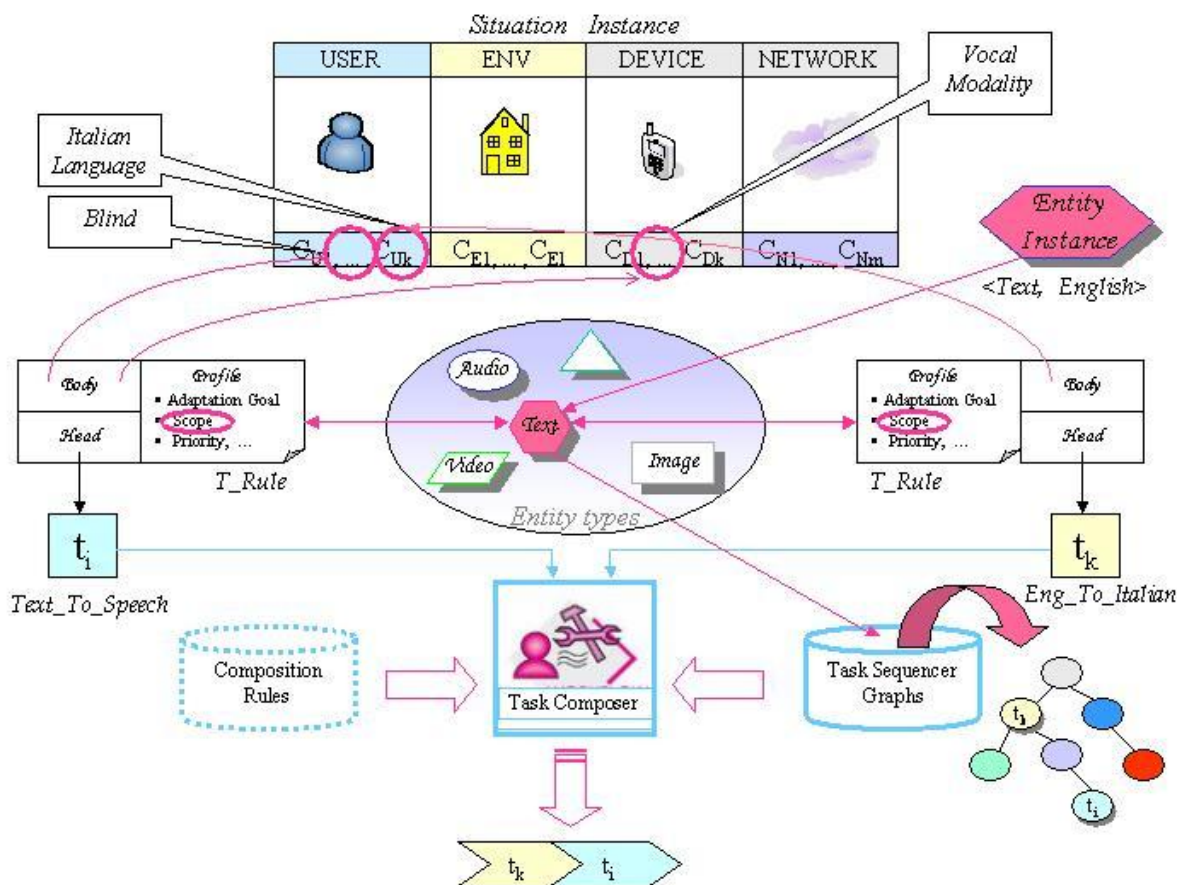


Figura 3 - Determinazione della sequenza dei task di trasformazione di una Entity.

La fattibilità di tale approccio è subordinata dal sussistere delle seguenti condizioni al contorno:

- Per ciascun tipo di Entity (*Testo, Audio, Video, Immagine*) le caratteristiche della Situazione saranno raggruppate in modo da costituire un insieme di situazioni elementari;
- Una regola di trasformazione applicata ad una Entity, in un certo stato e ad una data istanza di situazione elementare, può non essere attivata (Antecedente = False);
- Una regola di trasformazione attivata (Antecedente = True) individua almeno un task di trasformazione;
- Se due Regole di trasformazione sono conflittuali, vale a dire individuano due o più task tra loro incompatibili, verranno presi in considerazione solo i task individuati dalla regola avente *priorità* più alta;

- È possibile definire in fase di design dei grafi diretti aciclici (Task Sequencer Graphs in figura 3) in grado di indicare la sequenza temporale di composizione dei singoli task di trasformazione individuati dalle regole di trasformazione; ciò equivale ad individuare l'obiettivo di adattamento finale. In figura 4 è riportato un esempio parziale di grafo sequencer per le Entity di tipo *Testo*. Tali Grafi possono avere anche nodi di tipo decisionale intervallati a quelli di tipo "task di trasformazione astratto" anche se, in questa fase di design di SAPI non se ne intravede ancora la necessità.

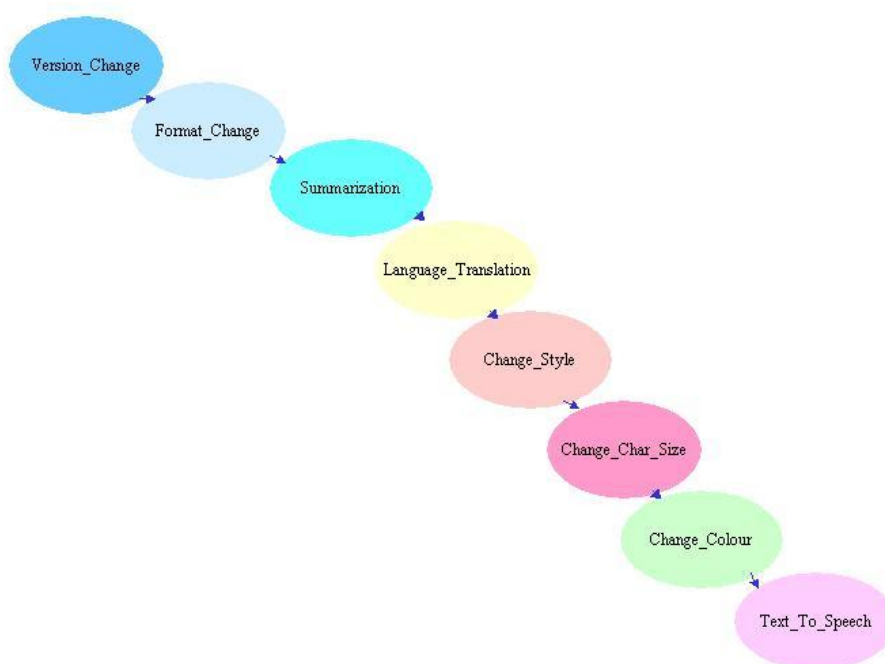


Figura 4 - Esempio di Task Sequencer Graph per le Entity di tipo Testo.

Come si può notare in figura 3, si è considerata l'opportunità, nell'eventualità nascesse l'esigenza in una fase più avanzata del progetto per casi non previsti in questa fase di design, di utilizzare delle Regole di composizione dei task di trasformazione. Tali regole sarebbero in ogni caso delle regole di Horn del tipo Condition-Action.

Ricerca del path di adattamento ottimale

In generale, data una *Entity* e una particolare *Situazione*, possono esistere più *Entity adaptation path* che raggiungono lo stesso risultato (adaptation goal) in termini di adattamento della Entity alla Situazione. È molto importante quindi mettere a punto una strategia di selezione del percorso di adattamento ottimale. Tale strategia deve necessariamente basarsi su un set di criteri di qualità, vale a dire su un insieme di proprietà non funzionali. In SAPI verranno considerati due parametri di qualità: il *costo* e il *tempo di esecuzione* di una regola di adattamento. Altri parametri possono essere aggiunti senza alterare fundamentalmente la strategia di selezione che verrà proposta di seguito. Tale

strategia si basa sulla tecnica *multi-criteria* SAW (Simple Additive Weighting) che ha come scopo quello di ottenere uno *score* a partire da un insieme di dimensioni opportunamente pesate aventi differenti unità di misura. Grazie alla sua semplicità computazionale e alla sua scalabilità in termini di numero di parametri qualitativi utilizzabili il metodo SAW è largamente applicato in vari ambiti applicativi.

A questo punto è possibile definire in SAPI la strategia di selezione del percorso di adattamento come, ad esempio, uno dei due seguenti problemi di ottimizzazione:

1. Trovare l'adaptation path avente il minimo costo e con un tempo di esecuzione non eccedente un tempo prestabilito T_{max} ; oppure
2. Trovare l'adaptation path avente il minimo tempo di esecuzione e con un costo non eccedente C_{max} .

Nell'ipotesi in cui per SAPI fosse sufficiente considerare il solo parametro *tempo di esecuzione*, un'alternativa molto valida al metodo SAW è rappresentata dall'algoritmo di Dijkstra modificato per i Grafi Diretti Aciclici. In quest'ultimo caso, utilizzando la notazione Big-O, l'algoritmo modificato avrebbe un tempo di esecuzione pari a $O(E+V)$ con **E** uguale al numero di archi (**E**dges) e **V** uguale al numero di nodi (**V**ertices).

Terminazione, confluenza e consistenza del set di regole

Garantire la *Terminazione* e la *Confluenza* di un insieme di Regole è un problema tipico dei *Database attivi*. La Terminazione in particolare, a meno di restrizioni, è un problema indecidibile.

In SAPI, per la metodologia di composizione delle regole adottata tali problemi sono automaticamente risolti. Per quanto riguarda la **Terminazione**, nella metodologia proposta è infatti implicito il vincolo che una regola attiva non possa autonomamente attivare direttamente o indirettamente altre regole. Se così non fosse, non sarebbe garantita l'aciclicità del grafo di adattamento che per scelta è un DAG (Directed Acyclic Graph). Che le regole siano poi a due a due commutabili (**Confluenza**), vale a dire che ordini di esecuzione diversi generino il medesimo stato finale dell'Entity da adattare, non è un requisito di SAPI, in quanto la *sequenza di trasformazione* corretta ($\{t_1 \rightarrow t_2 \rightarrow \dots \rightarrow t_n\}$) è determinata per ogni coppia <Entità, Situazione>, in modo da imporre alle regole coinvolte un ordine di esecuzione prestabilito (*adaptation path*)

Un insieme di regole è consistente se e solo se non vi sono regole sovrapposte o conflittuali.

La *sovrapposizione* delle regole ha luogo quando più di una *condizione* (body) conduce alla stessa *azione* (head). Ciò non implica necessariamente che sia sbagliato, ma è buona pratica, onde evitare comportamenti anomali o non desiderati, rilevare tutte le sovrapposizioni presenti nel rule set e verificare che esse siano tutte e sole quelle volute in fase progettuale.

Regole di Horn conflittuali occorrono quando la stessa *condizione* può essere vera per due o più regole, attivate dallo stesso *evento*, che conducono a differenti *azioni* (head).

Al fine di semplificare il check di consistenza dell'insieme di regole di SAPI, le regole in SAPI verranno raggruppate secondo i seguenti criteri di **classificazione**:

1. in base agli eventi che le innescano;
2. in base alle condizioni (*body*) che devono essere soddisfatte per la loro occorrenza;
3. in base alle azioni (*head*) che esse intraprendono.