Progetto Sogei-ISTI/CNR

Rapporto

Titolo:	Metodi per Sistemi Workflow Interattivi
Autori:	Fabio Paternò, Carmen Santoro
Organizzazione:	ISTI-CNR
Data:	12 Dicembre 2003
Commenti:	

Metodi per Sistemi Workflow Interattivi

F.Paternò, C.Santoro

ISTI-CNR

Laboratorio HIIS

{fabio.paterno,carmen.santoro}@isti.cnr.it

12/12/03

Introduzione

I sistemi di workflow hanno come obiettivo di fornire supporto automatico ai processi di un'organizzazione. Essi incorporano delle regole che definiscono come possono essere eseguite le attività delle persone, come possono comunicare tra loro, e come possono essere manipolati gli oggetti di interesse. In alcuni casi queste regole possono essere rigide ed entrare in conflitto con pattern di attività più informali e meno strutturati che contribuiscono ad un più snello svolgimento dei processi.

Nella task analysis l'obiettivo è di identificare i compiti che gli utenti desiderano svolgere allo scopo di raggiungere i loro obiettivi. Oltre ad identificare i compiti si tende a capire i principali attributi (come frequenza, precondizioni, oggetti da manipolare…). Dai risultati di una task analysis si possono creare dei modelli di task che descrivono le relazioni semantiche e temporali che caratterizzano l'esecuzione dei vari compiti.

Quando metodi basati sui task considerano applicazioni multi utente si trovano spesso ad affrontare problematiche simili a quelle dei sistemi di workflow: identificare i ruoli coinvolti nel sistema, gli oggetti manipolati, i vincoli tra le attività svolte da persone con diversi ruoli e così via. Ecco la necessità di definire metodi capaci di usare le tecniche sviluppate in questi due contesti in modo integrato e cercando di sfruttare anche possibili sinergie: ad esempio nella task analysis si presta maggiore attenzione ai fattori umani e vi sono metodi che consentono di usare l'informazione che contiene un modello dei task allo scopo di progettare interfacce utenti efficaci ed effettive o per valutarne l'usabilità.

All'origine di questi due approcci possiamo identificare due principali differenze: la granularità delle attività considerate e l'area scientifica da dove sono nate. Nei sistemi di workflow si tende a considerare attività di durata maggiore, interi processi aziendali, mentre nella task analysis si considerano anche attività molto brevi che si svolgono tramite interazioni elementari con calcolatori o altri sistemi. I sistemi di workflow sono nati prevalentemente nella comunità delle basi di dati dove il problema principale considerato era consentire un accesso a dati consistente da parte di vari utenti con diversi diritti. I sistemi di analisi e modellazione dei compiti sono nati nella comunità che studia l'interazione uomo-macchina dove si presta una particolare attenzione a che i sistemi, anche quelli cooperativi, forniscano interfacce usabili che consentono lo svolgimento di attività in modo efficace, efficiente e con soddisfazione personale.

Lo scopo di questo studio è di identificare approcci importanti in questa ottica e analizzarli per individuare metodi capaci di usare in modo integrato tecniche di workflow e di task modelling. Lo

scopo di tali metodi e di analizzare meglio come le attività si svolgono in un'organizzazione, come progettare le interazioni con i sistemi di supporto e come valutare la bontà delle soluzioni adottate. Inoltre, si analizzerà quale supporto automatico si può fornire per questo metodo, analizzando anche strumenti esistenti. In particolare sono stati identificati quattro approcci significativi che sono stati sviluppati da gruppi diversi: WIDE che è stato sviluppato nell'ambito di un progetto Europeo, Action Workflow, che fu originariamente sviluppato da Winograd [MWFF92], UML che è uno standard dell'ingegneria del software e ConcurTaskTrees (CTT), una notazione per modelli di task, anche cooperativi, ed il relativo ambiente che sono stati sviluppati all'ISTI-CNR di Pisa.

Il rapporto richiama alcuni concetti di base dei sistemi di workflow, in particolare per coloro che non sono familiari con essi, poi fornisce una descrizione sintetica dei quattro approcci considerati, discutendo relativi vantaggi e svantaggi.

1. Concetti di Base

Negli ultimi anni i sistemi di workflow sono diventati un argomento importante a livello di ricerca, ma soprattutto nelle aziende. Il workflow ha una natura multi-disciplinare coinvolgendo molti aspetti dell'informatica: basi di dati, sistemi distribuiti client/server, gestione delle transazioni, ri-ingegneria dei processi aziendali, integrazione di sistemi legacy e nuove applicazioni, ambienti software ed hardware eterogenei. E' anche in corso un processo di standardizzazione nell'ambito del Workflow Management Coalition (http://www.wfmc.org/). In generale i sistemi di workflow sono capaci di produrre soluzioni efficaci a processi che vanno dalla gestione di documenti, allo sviluppo di modelli complessi di workflow, alla cooperazione di gruppi di individui. Essi devono poter cooperare con le basi di dati, le reti, i sistemi operativi e le strutture hardware esistenti per poter garantire appieno le funzionalità offerte, così come devono garantire la cooperazione tra i sistemi di produttività individuale del sistema informatico in uso. Pur essendoci diversi sistemi di workflow, esistono delle problematiche comuni che vanno dalla conoscenza di quali attività siano supportate dalle applicazioni esistenti a quali siano effettivamente integrabili e a che livello, fino alla consapevolezza dello sforzo necessario per raggiungere l'integrazione.

Varie definizioni di workflow (letteralmente "flusso di lavoro") sono state proposte. Una è l'attività che progetta e mette in pratica l'**automazione dei processi aziendali**, cioè l'automazione dei processi di lavoro grazie ai quali **documenti** ed **informazioni** sono passati da una persona (o una macchina) a un'altra secondo precise **regole procedurali** per ottenere un **risultato**. Il funzionamento di una organizzazione può quindi essere rappresentato come una "serie di processi", cioè gruppi di attività collegate fra loro in modo logico per ottenere lo "scopo" desiderato. In sintesi possiamo considerarlo come l'insieme delle attività svolte collettivamente e volte al raggiungimento di un obiettivo aziendale comune. Tale obiettivo spesso si traduce in un incremento della produttività e dell'efficienza.

L'integrazione di un sistema di workflow in una realtà di produzione, per farne uno strumento valido di miglioramento dell'organizzazione e svolgimento dei vari processi, comporta una serie di passi necessari per una corretta cooperazione con il sistema informatico preesistente che possono essere schematizzati in poche fasi chiavi che sono:

- *la fase di modellazione dei processi*, che consiste in un'analisi critica accurata dei processi già in uso al fine di poter identificare chiaramente le esigenze aziendali effettive e determinare quali siano quelle procedure che possono essere rielaborate dal prodotto di modellazione. A seguito verrà posta la stesura di un modello descrittivo dei processi sviluppando la definizione del workflow che specifica quelli che sono i tratti chiave come

quali attività li costituiscono, la ripartizione dei compiti tra i vari individui, l'ordine di esecuzione di tali attività, le priorità, le scadenze, ecc.;
- *la fase di riprogettazione*, che consiste nell'ottimizzazione dei processi di business approfondendo i tratti salienti di ciascuno individuandone le debolezze in modo da poter apportare tutte le modifiche necessarie;
- *una fase di implementazione e automazione del workflow*, che consiste nella messa in atto concreta delle fasi di studio precedentemente svolte integrando al sistema informatico presente un'attività di controllo.

Più in generale il lavoro di Howard "The Workflow Management Market" ha proposto la definizione di un modello ideale di workflow. Tra l'altro, il modello esprime quelli che sono i cinque ambienti procedurali che dovrebbero essere soddisfatti dal prodotto ideale:

l'ambiente di analisi, che comprende funzionalità relative all'analisi dei processi, alla loro modellazione e simulazione, all'analisi delle risorse e del rapporto costo/benefici;

l'ambiente di sviluppo, che comprende funzionalità relative alle applicazioni di workflow management come la definizione del percorso delle informazioni, delle scadenze e delle priorità di esecuzione dei compiti, ecc..

l'ambiente di lavoro dell'utente finale, che comprende funzionalità relative alla definizione del piano di sicurezza dei dati, alla definizione delle viste dei compiti e dei meccanismi di notifica delle scadenze, ecc..

l'ambiente di amministrazione, che comprende funzionalità relative allo sospensione di un flusso di lavoro, al bilanciamento dei lavori, al controllo delle scadenze, ecc..

il motore di workflow, che comprende funzionalità relative alla gestione del traffico interno al sistema, alla integrazione con altri sistemi di workflow, al supporto di automazione del riassegnamento dei lavori, di avvio di azioni di contromisura ad eventi in generale come ad un avviso di scadenza non rispettata, etc..

2. Vantaggi del Work-Flow

Più i processi sono chiari e definiti, più il lavoro si svolge in modo vantaggioso: in breve, meno sforzo e più risultati: si eliminano le attività non necessarie che costano e rallentano il raggiungimento dell'obbiettivo, oppure si modificano per farle divenire più efficienti (anche attraverso l'uso di tecnologie avanzate).

I vantaggi economici che si possono ottenere nell'organizzare una attività secondo la logica del "Work-Flow" sono enormi, molto superiori a quelli raggiungibili con una buona organizzazione basata su mansionari o perfino sulle Procedure dei Sistemi Qualità (come ad esempio ISO9000/Vision2000), di cui possono essere viste come naturale evoluzione. Vi è chi ha calcolato che una organizzazione media può risparmiare centinaia di milioni all'anno con un investimento tutto sommato modesto rispetto ai ritorni ottenuti.

L'applicazione di questa tecnologia consente di raggiungere:

- Maggior controllo dei processi in atto
- Maggiore flessibilità
- Migliore servizio (poiché i processi sono più veloci e precisi)

- Facilità di implementazione dell'offerta, grazie al risparmio di tempo e risorse che può essere impiegato nell'attività di miglioramento.

3.Action Workflow

Molti approcci per la gestione dei workflow sono strutturati attorno al dominio dei processi informativi, ossia essi partono con una classe di oggetti di informazione quali form per poi definire come "workflow" una sequenza di azioni che deve essere eseguita su tali oggetti. La struttura organizzativa primaria è dunque il "passaggio" di questi oggetti di informazione attraverso gli utenti insieme alla specifica di azioni automatiche che devono essere intraprese all'interno di tale "passaggio". In un certo senso tutto ciò è molto simile al trattamento dei materiali, in cui le varie parti vengono passate da una "stazione" ad un'altra in una fabbrica/stabilimento per il loro trattamento e alcune delle attività che compongono tali processi sono intraprese da macchinari automatizzati. La Figura 1 mostra la sequenza base delle azioni in un action workflow loop. C'è sempre un cliente identificato ed un performer (esecutore), e il ciclo gestisce una particolare azione che l'esecutore accetta di completare per la soddisfazione del cliente.

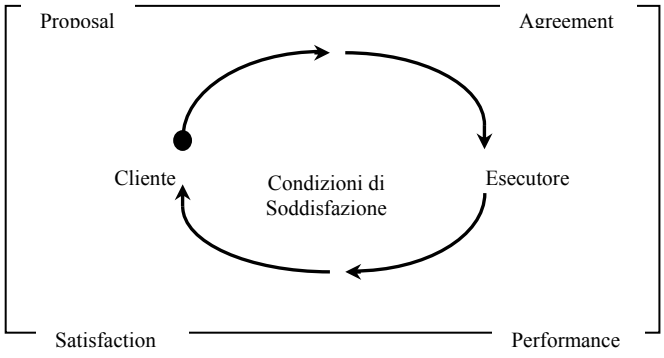

Fig. 1: Un Action Workflow Loop

Il loop procede in 4 fasi:

1) **Proposal** (proposta)
Il cliente richiede (oppure l'esecutore offre) il completamento di una particolare azione in base ad alcune stabilite condizioni di soddisfazione;

2) **Agreement** (accordo)
Le due parti arrivano ad un comune accordo sulle condizioni di soddisfazione, le quali condizioni includono anche le tempistiche entro le quali ulteriori passi saranno intrapresi.

3) **Performance** (esecuzione)

L'esecutore dichiara al cliente che l'azione è stata completata;

4) **Satisfaction** (soddisfazione/accettazione)

Il cliente dichiara all'esecutore che il completamento dell'azione è stato soddisfacente;

Un processo è dunque una rete di transazioni tra persone (clienti ed esecutori) che produce azioni coordinate ai fini della soddisfazione del cliente. All'interno delle suddette fasi ci possono essere azioni addizionali, come ad esempio chiarimenti e/o ulteriori negoziazioni sulle condizioni di soddisfazione. La struttura di tali azioni è definita da "speech acts" attraverso cui le persone si coordinano, piuttosto che dalle specifiche azioni eseguite dagli individui per raggiungere le condizioni di soddisfazione. La caratteristica fondamentale di Action Workflow (nonché differenza chiave rispetto ad altri approcci di modellazione di workflow), sta infatti proprio in questo shift di attenzione dalla struttura del task in sé alla struttura di coordinamento.

In un approccio di workflow più tradizionale le azioni di coordinamento sono viste come un tipo di task oppure come un flusso di informazioni tra task. Nella prospettiva seguita da Action Workflow, i task sono definiti dalle richieste e dagli impegni (committment) espressi nei loop. Questo shift in qualche misura può essere paragonato allo spostarsi da una visione di una rete come collezione di nodi (che hanno link tra loro) al vedere tale rete come una collezione di link (con nodi condivisi). Nonostante gli elementi siano gli stessi, il diverso punto di partenza porta a differenti potenzialità per rappresentare e supportare le varie attività.

La semplice struttura a loop del workflow è allo stesso tempo generale e universale. E' generale in quanto essa occorre ogni volta che esiste un coordinamento tra le persone, qualsiasi cosa esse stiano facendo. Le parole "cliente " ed "esecutore" si applicano a persone all'interno di una singola organizzazione, così come attraverso confini. La struttura del loop è universale in quanto è indipendente da qualsiasi cultura, linguaggio o mezzo di comunicazione attraverso cui è condotta. Ci sono infinite variazioni nello specifico di come i vari passi possono venire intrapresi, quali altri loop vengono attivati e come le persone rispondono a interruzioni ("breakdown") all'interno di esse, ma la struttura base è la stessa. L'Action Workflow Loop può essere paragonato ad un elemento atomico nella chimica delle interazioni. Ad ogni fase ulteriori cicli di negoziazione possono essere iniziati e svolti, ottenendo così una scomposizione gerarchica delle attività in ulteriori livelli di dettaglio e descrivendo così i complessi fenomeni delle organizzazioni. A differenza del sequential tracking delle form che si può trovare in altri approcci di modellazione e supporto di workflow, l'approccio seguito da Action Workflow progetta (e supporta la ri-progettazione) di un processo aziendale come collezione di loop intercorrelati, ciascuno con i propri completamenti e possibilità di interruzione.

L'obiettivo di Action Workflow sta nel rivelare/evidenziare gli elementi chiave dei workflow e le loro relazioni rispetto a completezze e incompletezze che possono essere riconosciute vitali per l'organizzazione. Ulteriori opportunità per migliorare la performance vengono dall'abilità di identificare, osservare e anticipare potenziali "breakdown" o fallimenti nel raggiungimento di un soddisfacente completamento. Dalle mappe e dalle discussioni associate è possibile identificare posti dove breakdown possono verificarsi su una base ricorrente e vedere quali passi addizionali o workflow possono essere piazzati per anticipare o affrontare eventuali problemi. L'esplicita articolazione della struttura in clienti, esecutori e condizioni di soddisfazione porta a identificare nuovi tipi di offerte o richieste che possono essere fatte. Sulla base di queste ultime, nuove strutture di workflow possono essere istituite.

3.1 Un esempio

Consideriamo ora un esempio di applicazione per la gestione delle review di job candidate per mostrare in che modo viene utilizzato Action Workflow.

Il processo si compone di quattro loop centrali, così come è mostrato in Figura 2. Le linee che connettono i loop mostrano dipendenze tra loro, con ognuna connessa all'appropriato quadrante del loop, in accordo a quali aspetti della struttura del workflow essi completeranno. Ogni loop sta per un workflow ricorrente, con il cliente identificato sulla sinistra e l'esecutore sulla destra. Le linee che connettono i loop di workflow indicano relazioni di triggering e di dipendenza tra loro. I cerchi numerati indicano form e altre rappresentazioni esterne che giocano un ruolo nel processo.

Il processo di review di un candidato inizia quando il direttore del personale fa una richiesta al manager del personale di gestire la review di un particolare candidato. Il manager inizia il processo con il riempire una form online con l'informazione come gli intervistatori, posizioni ricercate per il candidato, skill richiesti (Figura 2). Il workflow "Schedule interviews" corrisponde alla seconda fase (agreement) del workflow principale: il manager accetta di fare il lavoro come richiesto dal direttore, una volta che le interviste sono state schedulate. Nella fase di agreement uno specifico tempo di completamento può essere promesso. Una volta che il review process raggiunge l'agreement, la fase di "Performance" inizia e i workflow "Submit evaluation form" sono automaticamente iniziati, uno per ognuno dei selezionati interviewers. Ancora, forms sono definite per ognuno dei partecipanti e usate nel fare azioni nel workflow. Una volta che una interview è stata schedulata per una data particolare, tutti i workflow per sottomettere i report di valutazione sono iniziati e diretti ai selezionati interviewers per essere completati.

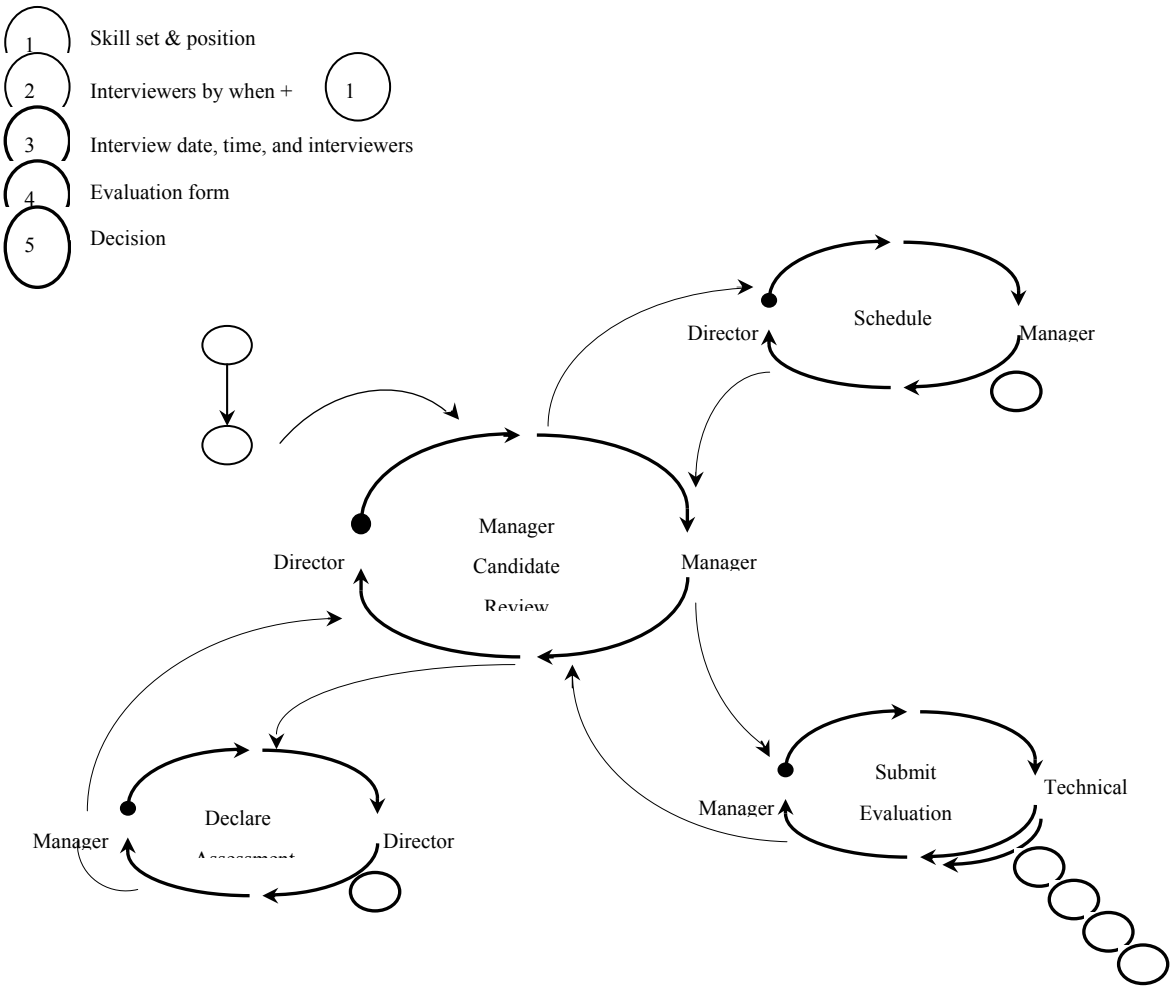

Fig. 2: Un esempio di specifica di workflow utilizzando Action Workflow

La teoria dell'Action Workflow fornisce un punto di partenza che è molto differente da approcci tradizionali di workflow. Quando un analista chiede alle persone di un'organizzazione "Come è strutturato il lavoro qui?", la risposta naturale è quella di iniziare a guardare le form e le procedure. Gli autori di Action Workflow respingono esplicitamente tutto questo, ignorando le form e chiedendo "Cosa state facendo effettivamente?". Senza la struttura dell'AW questa domanda potrebbe sembrare senza significato, ma con l'AW c'è una specifica direzione verso cui muoversi: "Chi sono i clienti e gli esecutori? Quali sono le condizioni di soddisfazione in ogni loop? Come viene eseguita ciascuna delle quattro fasi? Come sono correlati l'uno all'altro i vari loop?

Queste domande portano all'identificazione di quei posti dove confusioni possono portare a workflow incompleti, incomprensioni di risultati e non effettivo flusso di informazioni. Tutto ciò può dunque portare a nuove form e procedure, piuttosto che semplicemente automatizzare le vecchie. Così come gli autori di Action Workflow rivendicano nel loro lavoro, i tradizionali metodi di workflow sono da sempre stati *production-centered,* e focalizzati sull'efficienza e controllo, mentre il loro approccio è essenzialmente satisfaction-centered, con un focus centrale sugli impegni, condizioni di soddisfazione e completamento a tempo debito.

WIDE

1. Il modello WIDE per i workflow

Il modello per la rappresentazione dei processi proposto nel progetto EU-ESPRIT WIDE (Workflow on Intelligent Distributed database Environment, http://www.wide.ad.jp/) si colloca nel filone dei modelli che consentono di descrivere processi come insiemi di attività tra loro collegate da vincoli di precedenza e punti di sincronizzazione. Caratteristiche particolare del modello WIDE sono la possibilità di descrivere processi in modo flessibile, in particolare per quanto riguarda il trattamento delle eccezioni.

Il modello dei processi di WIDE è associato ad altri due modelli che completano la descrizione dei processi con la descrizione delle informazioni a essi associate e degli agenti che svolgono attività nei processi. Pertanto il modello WIDE completo è strutturato in tre differenti modelli:

- il *Modello dell'Organizzazione*, che consente di descrivere la struttura dell'organizzazione e gli agenti che ne fanno parte, indipendentemente dalla descrizione dei singoli processi.
- il *Modello delle Informazioni*, che consente di descrivere i dati e i documenti necessari all'esecuzione di un processo, in particolare in vista di un supporto informatizzato tramite workflow;
- il *Modello dei Processi*, che definisce le attività che fanno parte del processo e l'ordine in cui tali attività devono essere eseguite;

Nel seguito descriveremo brevemente i tre modelli. La descrizione completa del modello è presentata in [WIDE97].

1.1 Il Modello dell'Organizzazione

Lo scopo del modello dell'organizzazione è di descrivere la parte dell'organizzazione coinvolta nella gestione e nell'esecuzione dei processi (e non di modellare l'organizzazione in sé), e di come

questa è collegata ai task e ai workflow, al fine di consentire la specifica degli agenti (o dei gruppi di agenti) che hanno il diritto di eseguire un certo task. E' importante sottolineare che parte di questo modello è fortemente correlata alla struttura dell'organizzazione, ed è per questo motivo che è estremamente difficile generare un modello dell'organizzazione che vada bene per ogni tipo di business. Tuttavia, lo scopo qui è quello di creare una comprensione comune dei vari concetti chiave in modo da essere preparati per un loro adattamento ai vari tipi di organizzazioni.

Una caratteristica importante di WIDE è la netta separazione tra la descrizione dell'organizzazione e quella dei processi. Infatti, molti workflow sono in genere definiti nell'ambito della stessa organizzazione, e viceversa la specifica di uno stesso processo di workflow può essere eseguito nel contesto di diverse organizzazioni.

Il modello dell'organizzazione può essere definito in tempi differenti durante la vita della specifica di un workflow. Innanzitutto, durante la modellazione dei processi, il workflow viene specificato insieme ai *ruoli* che sono autorizzati ad eseguire ciascun task. Il passo successivo è popolare l'organizzazione in termini delle entità definite di seguito. Il terzo passo è quello di stabilire il link tra i due modelli. Per quello ogni ruolo nella specifica workflow viene mappato in una o più entità nel modello dell'organizzazione; questa associazione è, ovviamente, dipendente dall'organizzazione. Infine, la specifica workflow è pronta per essere effettivamente messa in atto tramite la creazione di istanze ("cases").

Questo approccio consente al progettista di workflow di modificare la popolazione dell'organizzazione, perfino le associazioni tra la specifica workflow e le entità dell'organizzazione senza modificare la specifica dei workflow. Questa caratteristica è molto importante in particolar modo per le grandi aziende, che spesso definiscono i loro processi in modo centralizzato, li distribuiscono, e poi l'associazione è eseguita ad ogni ramo. Questo meccanismo permette la condivisione e il riuso di modelli attraverso l'intera organizzazione.

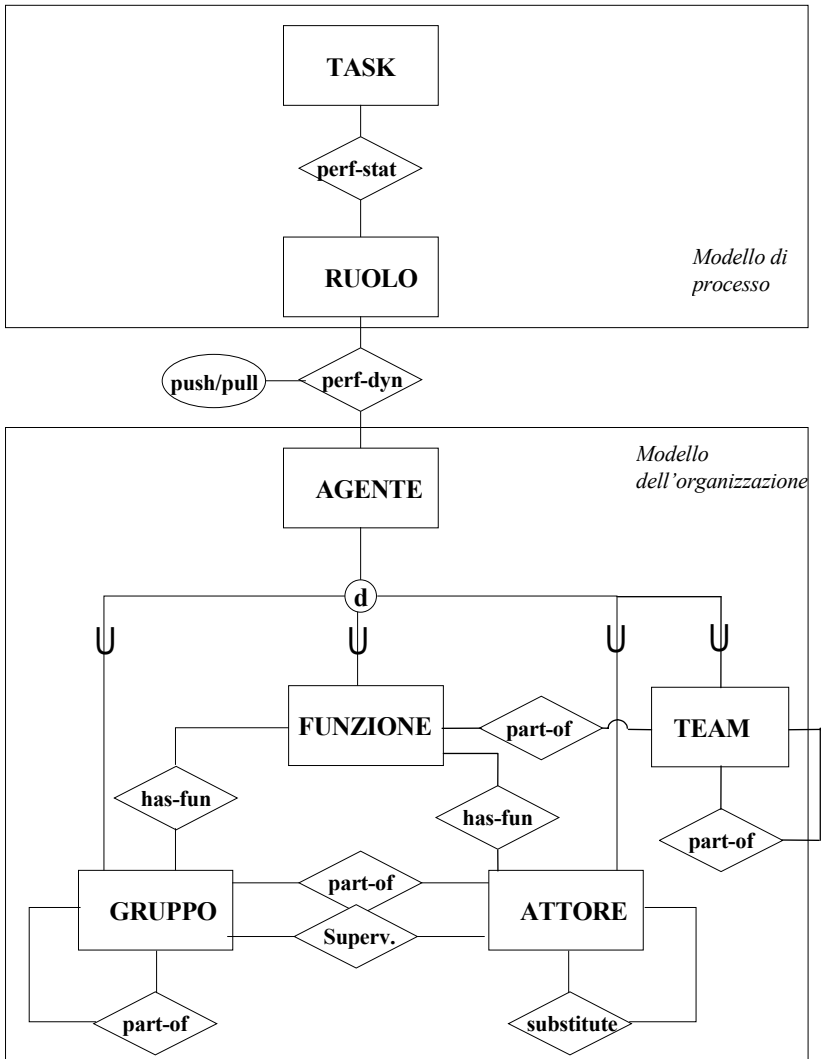

Figura 3 - Modello dell'Organizzazione di WIDE e sua relazione con il modello dei processi

1.1.1 Il modello

Le differenti entità e relazioni che compongono questo modello sono descritte nelle seguenti sottosezioni. Una rappresentazione parziale del modello proposto, omettendo gli attributi, è mostrato in Figura 3. Come si vede da questa figura, il modello dei processi e il modello dell'organizzazione possono essere individuate chiaramente e sono correlate dinamicamente attraverso la relazione "*performs-dinamically*" (perf-dyn).

1.1.1.1 Entità

Le seguenti entità sono identificate nel modello dei dati per l'assegnamento dei task (fig. 3):

Task. Un *task* è una attività atomica di lavoro nel modello dei processi del workflow, eseguita da una singola entità.

Ruolo. Un *ruolo* rappresenta una descrizione delle processing entitities che possono e hanno il permesso di eseguire un task specifico

Agente. Un agente, il concetto più generico nel modello dell'organizzazione, è una processing entità che può eseguire (direttamente o indirettamente) task nell'esecuzione di un processo di workflow. Un agente può a sua volta essere una funzione, un attore, un gruppo, o un team:

Funzione. Una *funzione* è una specifica di un insieme di gruppi, attori e team che hanno le stesse capacità;

Attore. Un *attore* è un umano o un componente meccanico o elettronico in grado di eseguire attività, e quindi di interfacciarsi con il sistema di gestione dei workflow;

Gruppo. Un *gruppo* rappresenta un insieme di attori che hanno caratteristiche comuni, ad esempio sono assegnati allo stesso progetto o sono nella stessa sede;

Team. Un *team* è una lista di funzioni necessarie a svolgere un certo compito; ad esempio, un team può essere definito come composto da un senior manager, un junior manager, e una segretaria;

Non tutti i concetti sono allo stesso livello di astrazione. Gli *attori* ed i *gruppi* sono concetti concreti, mentre le *funzioni* e i *team* sono invece concetti astratti che devono essere istanziati in concetti concreti.

1.1.1.2 Relazioni

La figura 3 mostra anche le relazioni tra le entità. Le relazioni sono fondamentalmente di due tipi:

- quelle che sono usate per collegare il modello dell'Organizzazione e di Processo e
- quelle che sono interne all'Organizzazione e cercano di soddisfare differenti strategie di assegnamento.

Il primo gruppo comprende le relazioni di assegnamento statico (*performs-statically*) e assegnamento dinamico (*performs-dynamically*).

Performs-statically. La relazione di assegnamento statico (*perf-stat*) collega un task ad uno o più ruoli, al fine di descrivere le capacità richieste per eseguire il task;

Performs-dinamically. La relazione di assegnamento dinamico (*perf-dyn*) collega i ruoli con gli agenti che sono in grado di ricoprire quel ruolo; la cardinalità della relazione è molti a molti, dato che più agenti possono ricoprire lo stesso ruolo, e un agente può ricoprire più ruoli.

Le relazioni nel secondo gruppo sono più dipendenti dall'organizzazione e possono essere implementate in modi differenti all'interno dell'organizzazione e tipicamente definite in fase di specifica dell'organizzazione. Questo gruppo contiene le seguenti relazioni:

Has-function. (*has-fun*) Questa relazione associa un attore o un gruppo ad una funzione. La cardinalità di tale relazione è n:m;

Part-of. Collega un attore a un gruppo di cui fa parte, un sottogruppo al gruppo di cui fa parte il sottogruppo, un sotto-team al team di cui fa parte oppure una funzione ad un team di cui fa parte, etc.

Substitute. Questa relazione collega un attore ad un altro attore che lo può sostituire nel caso che il primo attore non sia disponibile. La cardinalità di questa relazione è n:1;

Supervises. Collega un attore ad un gruppo. L'attore svolge mansioni di supervisione sul gruppo ed i suoi membri. La cardinalità di questa relazione è n:1.

1.1.2 Modalità di assegnamento di un task

Quando un task deve essere assegnato ad una data entità, ci possono essere diversi modi di procedere. Al fine di supportare questa operazione, ogni entità dell'organizzazione capace di ricevere lavoro (gruppo o attore) è rappresentata nel sistema da un *task desk* che contiene tutti i task da eseguire da quella entità.

Per l'assegnamento di un task ad una entità possiamo avere o una strategia del tipo *pull* o una di tipo *push.* Con il push è il sistema o un utente chiave che "spinge" (push) il task nel task-desk dell'utente selezionato, e quindi l'utente riceve dal sistema il task da eseguire. Seguendo il modello di tipo pull, il sistema mette solamente il task all'interno di un task-desk condiviso, e quindi sta al singolo utente che ha accesso a quel particolare task-desk condiviso di "tirare" (pull) il task nel suo proprio task desk. In questo caso l'utente seleziona il lavoro da fare. Una sostanziale differenza tra i due metodi sta nel comportamento attivo o passivo dell'utente rispetto al sistema: nel sistema push l'utente semplicemente aspetta il lavoro, nella metodologia di tipo pull è l'utente che deve prendere i lavori da uno spazio condiviso.

1.2 Il modello delle Informazioni

Il modello delle informazioni descrive le informazioni coinvolte nella definizione di un workflow. Il modello WIDE consente la specifica di quattro tipi di informazioni: variabili e elementi di documentazione (form, documenti e cartelle).

- **Variabili.** Le variabili sono elementi basici di informazione, disponibili a tutti i task nel modello dei processi. Possono essere usate semplicemente per memorizzare informazione importante nel modello oppure, quando applicate ad un modello di processo per controllare l'esecuzione di un flusso. Una variabile è caratterizzata dal suo tipo (intero, stringa, durata, data, enumerazione), il suo nome, il suo valore iniziale (se esiste), la sua descrizione e la sua categoria. Il concetto di categoria permette ai progettisti di applicare alcuni criteri di classificazione all'intero insieme di variabili. Le variabili, per default, non sono condivise tra diversi cases (istanze di specifiche di workflow)

- **Form**. E' un insieme di campi di dati il cui contenuto può essere controllato dal WorfFlow Management System (WFMS). Una form contiene dati a cui l'utente può accedere. Una specifica form può essere utilizzata in differenti modi in diversi task, per questo motivo si dice che i task gestiscono *views* delle form. Una form viene definita in due passi: innanzitutto viene definita la sua semantica, in termini delle variabili che la compongono, e poi viene definito come visualizzare la form.

- **Documenti.** sono insiemi di informazioni il cui contenuto non è prodotto direttamente dal WFMS. I documenti sono creati e manipolati per mezzo di applicazioni esterne. Esempi di documenti sono immagini, file di testo o in formati proprietari (come Microsoft Word).

- **Cartelle.** sono combinazioni di form, documenti o anche di altre cartelle. Il contenuto di una cartella può variare durante il tempo in funzione di come evolve il flusso del processo. Inoltre, una cartella può essere definita come vuota o avere un contenuto iniziale. Le cartelle sono utili quando c'è la necessità di raggruppare logicamente un insieme di elementi che devono essere trasferiti tra i vari task.

Documenti e form sono i "mattoni" che costituiscono la documentazione, mentre le cartelle consentono di strutturare l'informazione secondo una struttura ad albero. A tutti i vari elementi di documentazione può essere assegnata una condizione. In questo caso il task usa tale elemento se la condizione è verificata nel momento in cui il task è pronto per essere eseguito.

1.3 Il modello dei Processi

Il modello dei Processi descrive il comportamento di una specifica di workflow, cioè come il processo evolve dal suo stato iniziale ad uno dei suoi stati finali. Gli elementi fondamentali da descrivere sono:

- *Task*, ossia le unità elementari di lavoro all'interno di una specifica workflow
- *Connettori*, che specificano come i vari task vengono connessi l'uno all'altro
- Altre unità di esecuzione come ad esempio sotto-processi, etc. che permettono una progettazione chiara e facilitano il riuso e la distribuzione.

1.3.1 Task

I task sono le unità di lavoro elementari che complessivamente portano al raggiungimento dell'obiettivo del workflow. Il motore del workflow si preoccupa di determinare quando un certo task deve iniziare la sua esecuzione e di assegnarlo ad uno specifico agente, seguendo differenti strategie di schedulazione per l'assegnamento dei task. Un task è caratterizzato da:

- un *nome*, unico all'interno del processo;
- una *descrizione*, in linguaggio naturale, del lavoro da eseguire per completare il task
- un riferimento all'esecutore del task, il ruolo, che sarà usato per stabilire l'associazione con il modello dell'Organizzazione.
- un insieme di dati associati al task, necessari per la sua esecuzione (tipicamente: form, documenti e cartelle)
- un insieme di diritti per le possibili azioni da eseguire sul task
- le sue caratteristiche transazionali
- il suo comportamento eccezionale

1.3.2 Connettori

Le connessioni modellano l'interazione fra i task e definiscono la *"colla"* tra i vari task che permette al case di passare dal task iniziale al task finale. Un task può avere una sola connessione in uscita ed una in ingresso. Due task A e B possono essere connessi direttamente (tramite una freccia, nel linguaggio grafico), con la semantica intuitiva che non appena A termina, B viene mandato in esecuzione. In tutti gli altri casi, le connessioni fra task sono mediate da *connettori*. Un connettore, per esempio, può essere un connettore di tipo fork, per iniziare esecuzioni concorrenti di task, oppure un connettore *join* per sincronizzare task dopo una esecuzione concorrente.

Connettori fork

I connettori di tipo fork sono preceduti da un task, chiamato *predecessore*, e seguiti da un numero di task, chiamati *successori*. I connettori di tipo fork sono classificati come segue:

- *totale*: al temine del predecessore tutti i successori sono pronti per essere eseguiti.
- *condizionale*: ad ogni successore è associata una condizione: al termine del predecessore, le condizioni vengono valutate istantaneamente e solo i task successori la cui condizione è vera saranno pronti per essere eseguiti.
- *condizionale con mutua esclusione:* come il condizionale ma con la differenza che solo una condizione può essere vera.

Connettori join

I connettori di tipo join sono preceduti da molti task, chiamati *predecessori*, e seguiti da un task, chiamato *successore*, oppure da un altro connettore I connettori di tipo join sono classificati in:

- *totale*: il successore viene attivato solo al termine di tutti i predecessori;
- *parziale*: al connettore join è associato un valore *k*: il successore viene attivato non appena k predecessori con lo stesso numero di attivazione sono terminati. La terminazione di ulteriori predecessori con lo stesso numero di attivazione non ha nessun effetto.
- *ciclico*: un'istanza del successore viene attivata tutte le volte che un predecessore termina, fin quando una condizione viene verificata.

La rappresentazione grafica di task e connettori è mostrata in figura 4.

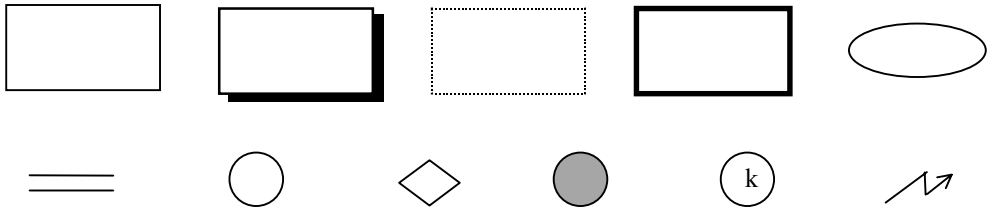

Figura 4 - Simboli del modello WIDE. Da sinistra a destra, dall'alto al basso:

1)task, 2) sottoprocesso,3) supertask, 4)business transaction, 5)wait task;

1)simbolo di inizio-fine, 2)fork/join totale, 3)fork condizionale, 4)ciclo, 5)join parziale, 6)trigger

1.3.3 Simboli di *inizio-fine* processo

I simboli di inizio-fine segnalano i momenti della creazione e completamento di istanze di workflow (cases). Ogni workflow ha un simbolo di inizio e uno o più simboli di fine processo. Il simbolo di inizio ha uno o più successori (se ha più di un successore, il simbolo deve essere preceduto da un connettore fork), e analogamente il simbolo di fine ha uno o più task predecessori.

1.3.4 Wait task

Il *wait task* è un particolare tipo di task che non compie azioni e che non deve essere assegnato ed eseguito da un agente. La condizione associata alla definizione del wait task può essere definita come un predicato sui dati dell'applicazione e/o sui documenti, sul tempo, o sull'occorrenza di eventi esterni. Non appena la condizione è verificata, il task è completato.

1.3.5 Multitask

Un *multitask* definisce un insieme di task che eseguono esattamente lo stesso lavoro in parallelo, con alcuni parametri che possono variare tra le differenti istanze. L'attivazione del multitask corrisponde all'attivazione contemporanea di più istanze dello stesso task; tutte le istanze hanno lo stesso numero di attivazione, che coincide con quello del multitask. Il multitask ha un duplice scopo: consente di specificare in modo compatto un insieme di task che compiono la stessa funzione e consente di definire a tempo di esecuzione il numero delle istanze che devono essere

attivate, che può dipendere dal valore di una variabile del workflow. Ogni multitask ha un parametro di input che indica il numero di istanze di task che diventano pronte quando il predecessore del multitask termina. E' anche possibile specificare quando un multitask deve essere considerato finito, associando ad esso un valore soglia o una espressione chiamata *quorum*. Quando il numero di istanze di task che sono terminate raggiunge il quorum il multitask termina e il suo successore diviene pronto per essere eseguito.

1.3.6 Sottoprocessi, supertask e business transactions

Sottoprocessi, supertask e business transaction sono costrutti che raggruppano insiemi di task, consentendo di modularizzare la specifica di un workflow. Sono instanziati quando vengono raggiunti dal flusso di controllo.

- Un *sottoprocesso* è un modo di modularizzare la specifica di un workflow in termini di pezzi più piccoli, che hanno un insieme di dati in ingresso e in uscita passati al sottoprocesso e ritornati al processo padre al termine del sottoprocesso.
- Come i sottoprocessi, anche i *supertask* sono composti da un insieme di task, collegati tramite connettori. Il supertask non ha però parametri di ingresso, e vede le stesse variabili del processo nel quale è definito.
- La *business transaction* è l'elemento di base del modello transazionale di WIDE. Il concetto di business transaction ha lo scopo di consentire la specifica, a livello di workflow, di proprietà transazionali caratteristiche delle operazioni su basi di dati. Una *business transaction* raggruppa task che formano un'unità transazionale, ovvero che devono essere eseguiti in modo atomico e isolato rispetto agli altri task. In WIDE, ogni task deve fare parte di una business transaction, pertanto, un workflow può essere visto come un insieme di business transaction.

Un esempio di applicazione del linguaggio WIDE è mostrato in figura 5, in cui si specifica il processo di approvazione delle spese di trasferta di un dipendente, in funzione dei costi previsti per l'albergo e per il vettore.

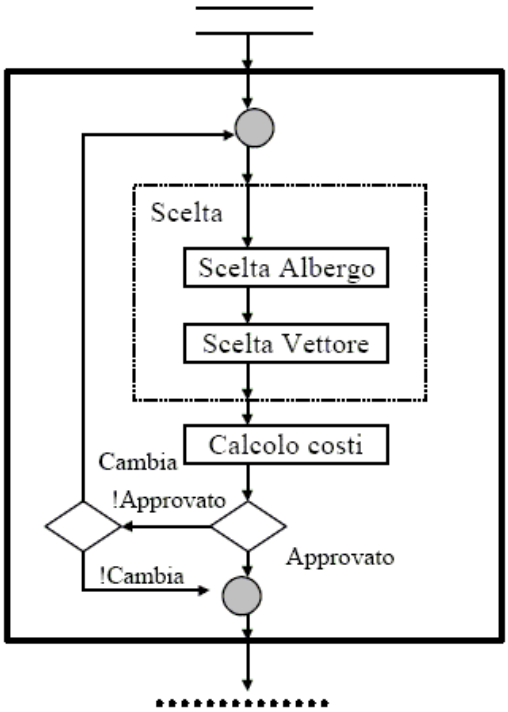

Figura 5 – Un semplice workflow specificato in WIDE

1.3.7 Eccezioni

Inoltre, la specifica di un workflow può includere anche *eccezioni*, che consentono di descrivere in modo compatto alcune situazioni di tipo anomalo che si possono verificare durante l'esecuzione del processo e che richiedono un particolare trattamento, quale l'esecuzione di specifiche attività o l'aggiornamento di alcuni dati del processo o l'alterazione del normale flusso di esecuzione del processo. Una delle caratteristiche più significative ed interessanti di WIDE è proprio la possibilità di definire e gestire situazioni eccezionali, ovvero situazioni che fanno parte della semantica di un processo, pur non facendo parte della sua esecuzione normale.

Le eccezioni hanno caratteristiche diverse rispetto al flusso normale del processo; infatti, le eccezioni sono spesso asincrone, ovvero possono capitare in qualunque momento durante l'esecuzione del processo, e non solo in corrispondenza dell'esecuzione di uno specifico task. Ad esempio, la cancellazione di un viaggio in un processo di prenotazione viaggi o l'incidente di un auto in un processo di noleggio auto possono capitare in vari momenti durante l'esecuzione di questi processi. Inoltre spesso l'azione necessaria per gestire un'eccezione non comporta necessariamente l'attivazione di un task: al contrario, l'azione opportuna può essere l'invio di un messaggio ad un agente per segnalare la situazione eccezionale, o la cancellazione del processo.

Dato che un'eccezione non è direttamente causata dall'esecuzione di un task, e non viene gestita attivando un task, non è possibile rappresentarla tramite diramazioni nel grafo che descrive il processo. Per questo, nel progetto WIDE sono stati studiati altri formalismi per la specifica dei comportamenti eccezionali, ed il formalismo che è stato scelto è quello di regole *Evento-Condizione-Azione* (ECA), nelle quali l'evento indica l'occorrenza di una situazione potenzialmente eccezionale, la condizione verifica che l'evento corrisponde effettivamente ad una situazione eccezionale che deve essere gestita, mentre l'azione, eseguita solo se la condizione vera, reagisce all'evento, al fine di gestire la situazione eccezionale.

UML

UML (Unified Modelling Language) [BRJ99] è un ambiente di modellazione che è diventato uno standard de facto nel campo dell'ingegneria del software. Esso fornisce un insieme di rappresentazioni per specificare, costruire, rappresentare in modo visuale e documentare sistemi software. L'approcio risultante è basato su concetti sviluppati precedentemente, come il lavoro precedente di Booch, l'Object Modelling Technique (OMT) di Rumbaugh's e la OOSE (Object-Oriented Software Engineering) di Jacobson.

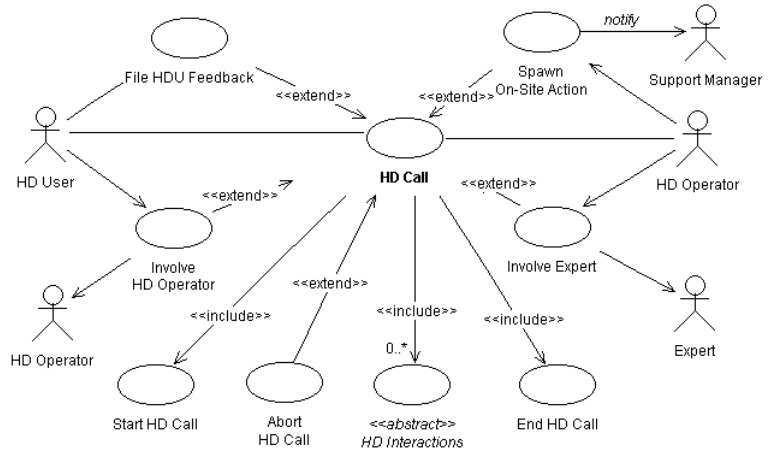

Figura 6: Esempio di Use Case Diagram.

Una delle motivazioni principali dietro lo sviluppo di UML fu l'integrazione delle pratiche nello sviluppo software, tenendo presente numerosi punti di vista basati su livelli di astrazione, domini di applicazioni, architetture, stadi nel ciclo di vita, tecniche di implementazione, ecc.. Le seguenti nove rappresentazioni sono supportate da UML:

- *Class diagrams*: tali diagrammi sono usati per descrivere la struttura logica del sistema: la struttura delle entità che appartengono al sistema, e la natura delle loro relazioni. UML non impone una stretta distinzione tra diagrammi di classi e di oggetti. Un certo diagramma può contenere un misto di classi ed oggetti.
- *Use case diagrams*: questi diagrammi motrano le relazioni tra attori e use cases dentro un sistema. Uno use case diagram è un grafo di attori, un insieme di use cases nell'ambito di un sistema, l'indicazione delle associazioni tra attori e use cases, e generalizzazioni tra use cases. Figura 5 fornisce un esempio di Use case diagram.
- *Interaction diagrams*: un pattern di interazione tra oggetti è mostrato tramite un interaction diagram. Questi diagrammi possono essere rappresentati in due modalità basate sulla stessa informazione ma che evidenziano aspetti diversi: *sequence diagrams* e *collaboration diagrams*. I Sequence diagrams mostrano un'interazione secondo l'evoluzione temporale. In particolare, essi mostrano una tavola con gli oggetti che partecipano all'interazione su un asse ed I messaggi che essi si scambiano durante l'evoluzione temporale lungo l'altro asse. I Collaboration diagrams mostrano le relazioni tra oggetti che sono rappresentate tramite archi e sono utili per comprendere gli effetti su un certo oggetto e per la progettazione delle procedure.

- *State diagrams*: questi diagrammi mostrano le sequenze di stati attraverso cui un oggetto od un'interazione vanno durante la loro vita in risposta agli stimuli ricevuti, insieme con le risposte e le azioni. La semantica e la notazione sono essenzialmenre quelle degli statecharts di David Harel [H87] con qualche modifica minore. Il suo lavoro fu un importante progresso rispetto i tradizionali diagrammi a stati finiti.
- *Object diagrams*: essi danno una vista statica di un insieme di oggetti e le loro relazioni che possono essere trovati nei class diagrams.
- *Activity diagrams* rappresentano la componente dinamica di un sistema che mostra il flusso di controllo tra un insieme di attività. Essi possono essere usati per rappresentare diversi ti di attività: flussi di lavoro, operazioni di un oggetto, ecc..
- *Component diagrams,* mirano a rappresentare la struttura delle componenti che sono composte di classi, interface, o collaborazioni.
- *Deployment diagrams* rappresentano un'architettura composta di nodi di elaborazione a run-time e le relative componenti.

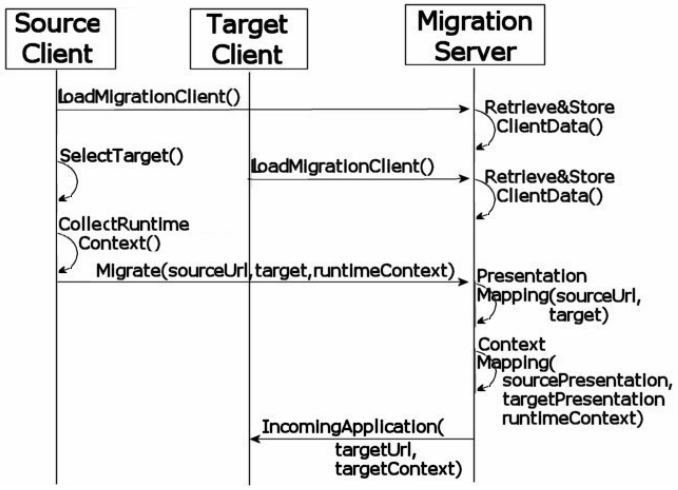

Figura 7: Esempio di Sequence Diagram.

UML è stato principalmente usato per progettare sistemi software prestando poca attenzione agli aspetti di interface utenti. E' più orientato a supportare una progettazione di un sistema funzionalmente corretto ed ingegnerizzato. Due domande si pongono quando si confrontano approcci basati su task e UML:

- Come possono essere posizionati gli approcci basati su task nell'insieme di rappresentazioni fornite da UML ?
- Come è possibile integrare approcci basati su task con UML?

Riguardo la prima questione, in generale possiamo notare che i modelli di task forniscono una vista più orientate all'utente del comportamento di un'applicazione mentre UML è più orientato a fornire rappresentazioni del comportamento interno di un sistema. Riguardo le specifiche tecniche usate

dentro UML possiamo notare che gli use cases sono abbastanza simili al risultato di un'analisi di task perchè entrambi sono orientate a supportare la fase di analisi dei requisiti. Gli Activity diagrams hanno scopi simili alle notazioni per i modelli di task, tuttavia non sembrano particolarmente flessibili, per esempio per mostrare diversi livelli di astrazione e le loro relazioni, mentre gli sequence diagrams indicano le attività che sono eseguite includendo gli attori ed i messaggi che sono scambiati, fornendo così descrizioni che considerano aspetti simili a quelli considerati nei modelli di task ma più limitati all'implementazione software ed a specifiche interazioni.

Riguardo al seconda questione varie soluzioni sono possibili. In [LPL98] è proposto un metodo che supporta la trasformazione da un sequence diagram in un modello di task rimuovendo l'informazione interna orientata al sistema che non è necessaria in un modello di task. Il motivo per tale approccio è che UML è largamente usato e quindi in questo modo ci si può avvantaggiare da sorgenti di conoscenze pre-esistenti riusando l'informazione che essi contengono allo scopo di ridurre lo sforzo richiesto per sviluppare modelli di task che possono essere usati per supportare meglio la documentazione una volta che sono sviluppati. Una soluzione alternativa è di sviluppare modelli di task usando le informazioni contenute negli use cases e negli activity diagrams, allora l'informazione nei modelli di task possono essere usati per sviluppare la progettazione del sistema in termini di interaction e object diagrams catturando meglio gli aspetti orientati agli utenti.

ConcurTaskTrees

L'analisi dei task consiste nel capire quali sono le attività che si deve compiere e come queste devono essere eseguite per raggiungere determinati obiettivi. In questo modo, essa può fornire un aiuto per migliorare la modellazione nonché la progettazione e l'implementazione del sistema informatico, focalizzando il ruolo dell'attività umana nell'ambito di applicazioni interattive.

L'analisi dei task nasce con due impieghi principali :

- per identificare concetti e procedure di base che possano servire per la redazione di tutorial, per insegnare ad un eventuale utente come utilizzare il computer, per eseguire una determinata applicazione
- per redigere manuali o routine di aiuto che siano di supporto per l'utente nell'individuare gli obiettivi che egli vuole realizzare nonché spiegare quali procedure effettuare per raggiungere tali finalità.

Ruolo molto importante, oramai, dell'analisi dei task, è quello di poter essere di supporto per la progettazione software, a causa della complessità sempre crescente delle applicazioni, per cui si moltiplicano anche le difficoltà nella organizzazione e nella gestione del processo di modellazione di tali applicazioni.

Un task, in sostanza, descrive un'attività tesa a raggiungere un obiettivo più generico. La combinazione di un insieme di tasks in base a relazioni di dipendenza logica e temporale permette di definire il compito dei task che l'applicazione supporta.

Diamo una definizione di task che risulterà molto utile per capire le successive considerazioni che verranno fatte. Un task definisce come l'utente può raggiungere un obiettivo in uno specifico dominio di applicazione. L'obiettivo è una modifica desiderata dello stato di un sistema o una domanda verso di lui.

Infine, introduco una tra le varie forme che può assumere il risultato dell'analisi dei task è l'analisi dei task di tipo gerarchico (HTA) che per la prima volta venne sviluppata da Annett e Duncan [AD67]. I vantaggi della disposizione gerarchica dei task:

- La naturale decomposizione gerarchica dei task può assistere gli analisti nello strutturare la loro raccolta di dati e l'attività di documentazione. È possibile focalizzare l'attenzione su sotto parti costituenti i task, senza perdere la visione globale dell'attività e, la natura top-down del metodo, assicura la completezza della descrizione della realtà analizzata.
- Il diagramma gerarchico è relativamente semplice da capire.
- Il diagramma rappresenta uno strumento importante per la raccolta iniziale di dati e per la seguente loro approvazione.

La chiarezza e struttura di questo tipo di rappresentazione assiste nella fornitura di informazioni per i progettisti del sistema. Facendo un esempio, supponiamo di dovere risolvere il problema relativo alla compilazione di una domanda di partecipazione ad un concorso di bellezza e della sua spedizione. In base al concetto di analisi dei task noi possiamo suddividere l'attività principale in una serie di sotto attività e poi metterle in relazione tra loro. L'obiettivo primario, che è quello di preparare il modulo di domanda, può essere ad esempio decomposto in due sotto obiettivi: la redazione del modulo e la possibilità di farne una copia, e conseguentemente la sua compilazione. Inoltre il task relativo alla compilazione può essere ulteriormente scomposto in modo tale da poter inserire semplice testo oppure anche delle immagini.

Riassumendo graficamente il tutto si ottiene:

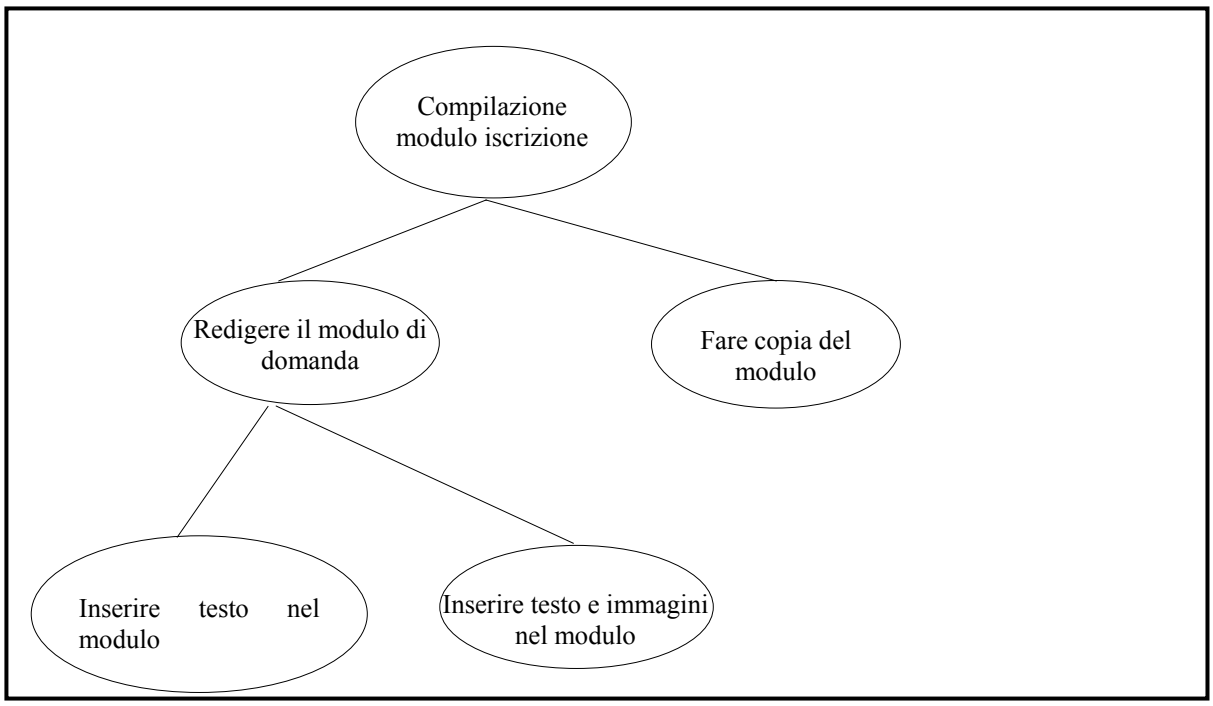

Figura 8: Esempio di Sequence Diagram.

Come si può vedere dal semplice esempio un generico problema viene analizzato e decomposto in una serie di sotto problemi. È importante, però, a questo punto, capire il legame esistente tra questi ultimi in quanto una loro scorretta risoluzione può compromettere quella dell'intera applicazione. Si

pensi al fatto di eseguire la copia della domanda prima che questa sia stata compilata. Si capisce che, senza un opportuno ordine di esecuzione, si possono avere delle situazioni inconsistenti.

È per questo che dall'analisi dei task si passa poi attraverso alla modellazione di un ordinamento logico tra le varie azioni da compiere.

Sempre nell'esempio, la copia della domanda deve essere fatta solo quando si è finito di scriverla e non prima, per cui si capisce come sia conveniente per i progettisti, gli sviluppatori, gli psicologi e gli esperti del dominio di applicazione studiare i task che un determinato problema da risolvere dovrebbe supportare. Per questo fine è molto importante avere una notazione per progettare la specifica del modello dei task in modo tale che:

- essi siano facili da comprendere e da usare, migliorando così la cooperazione tra le persone coinvolte nella soluzione di un preciso problema
- essi siano capaci di strutturare un'ampia gamma di specifiche;
    - la loro semantica sia definita in maniera tale da evitare situazioni di ambiguità nella comunicazione

La Notazione

La notazione che consideriamo per la rappresentazione dei modelli, permette di strutturare gerarchicamente le diverse attività che compongono l' applicazione nomadica da progettare, mentre le diverse relazioni temporali che legano i task possono essere formalizzate attraverso un ricco insieme di operatori. Per ogni task, inoltre, possono essere specificate ulteriori informazioni come il tipo, gli oggetti manipolabili e diversi altri attributi come la frequenza di esecuzione e la sua eventuale compatibilità con i diversi tipi di dispositivi considerati.

Le caratteristiche principali di questa notazione sono [P99]:

- *Focus sulle attività.* Come già enunciato precedentemente, permette di concentrarsi sugli aspetti interessanti prescindendo da quelli implementativi.
- *Struttura gerarchica*. L' utilizzo di una struttura gerarchica fornisce un elevato grado di intuitività alla modellazione. In questo caso il suo utilizzo presenta due vantaggi principali: fornisce un grado elevato di granularità, consentendo il riuso di strutture di task sia di grosse che di piccole dimensioni, inoltre, consente di definire le strutture da riutilizzare sia ad alto che a basso livello semantico.
- *Sintassi grafica*. In questo caso l' utilizzo di tecniche visuali per l' editing dei modelli risulta di più facile comprensione. Il modello riflette la struttura logica delle attività e, per questo motivo, ha una forma ad albero.
- *Notazione concorrente*. Per definire le relazioni tra i task, vengono messi a disposizione un numero elevato di operatori temporali. Altri approcci prevedono che la specifica delle relazioni temporali avvenga implicitamente, tuttavia, nel nostro caso è necessario che queste dipendenze siano esplicitate, di modo che in un momento successivo sia possibile, analizzando il modello, stabilire gli insiemi dei task disponibili per essere eseguiti in un determinato momento. La possibilità di ricavare questi insiemi, come vedremo nel prossimo capitolo, è fondamentale al per la determinazione dell' interfaccia utente relativa al modello considerato.

- *Allocazione dei task*. Possono essere indicate quattro categorie differenti di task: task astratti, ovvero task composti da un certo numero di subtask di categorie differenti, task utente, task di applicazione e task interattivi.
- *Oggetti manipolati*. Una volta identificato il task, è necessario poter stabilire gli oggetti che si devono manipolare affinché ne sia consentito il completamento. Possono essere specificati sia elementi del dominio dell' applicazione da modellare, che oggetti dell' interfaccia utente.

In Figura 7 viene mostrato un semplice esempio di modello dei task rappresentato nella notazione scelta.

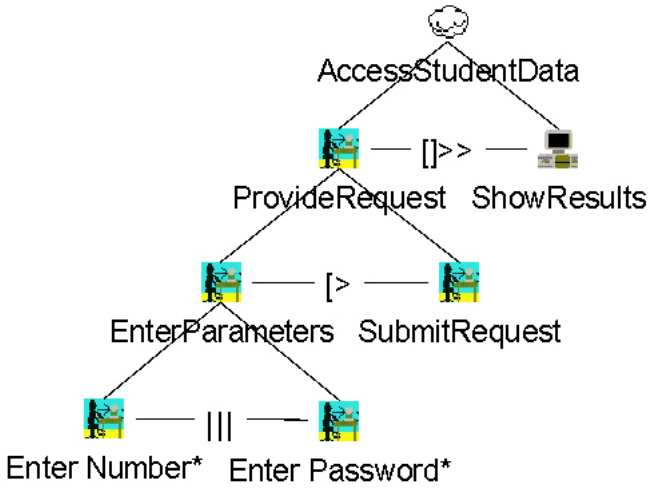

Figura 9 : Semplice esempio di modello dei task in questa notazione

L' esempio è decisamente semplice, illustra le attività eseguite da uno studente nell' accedere al proprio profilo accademico. In principio, è necessario che venga inserito il proprio numero di matricola con la relativa password; a questo punto è possibile sottomettere la richiesta, completando l' autenticazione personale e accedendo ai propri dati personali.

Operatori disponibili

Di seguito viene riportata la lista degli operatori temporali che sono messi a disposizione dalla notazione.

- T1 ||| T2, *Concorrenza*. Task eseguibili in qualsiasi ordine, senza alcuna limitazione.

- T1 |[]| T2, *Concorrenza con scambio di informazioni*. Come prima, ma è necessaria una sincronizzazione che consenta la trasmissione delle informazioni da un task all' altro.

- T1 [] T2, *Scelta*. E' possibile selezionare quale task eseguire, tra i diversi disponibili. Una volta effettuata questa scelta gli altri task diventano ineseguibili e rimangono tali almeno fino al completamento del task stabilito.

- T1 |=| T2, *Ordine Indipendente*. I task possono essere eseguiti in qualsiasi ordine, iniziare l' esecuzione di uno, però, inibisce, fino al suo completamento, l' eseguibilità degli altri.

- T1 [> T2, *Disattivazione*. Il primo task viene disattivato in modo definitivo contestualmente all' esecuzione della prima azione del secondo task.

- T1 >> T2, *Attivazione*. La terminazione del primo task rende eseguibile il secondo.

- T1 []>> T2, *Attivazione con scambio di informazioni.* Come il caso precedente, con in più una trasmissione di informazioni dal primo al secondo task.

- T1 |> T2, *Sospensione-Ripristino*. Il secondo task può arbitrariamente sospendere l' esecuzione del primo, che riprenderà, nello stato raggiunto prima della sospensione, una volta terminata l' esecuzione del secondo.

- T*, *Iterazione.* Un task iterativo, una volta completato, riprende la sua esecuzione dall' inizio, a meno che non venga disabilitato da altri operatori.

- T(n), *Iterazione finita.* Viene utilizzato questo operatore quando è noto a priori quante volte verrà ripetuta l' esecuzione del task.

- [T], *Task opzionale.* La performance del task è opzionale.

- *Ricorsione.* Nel sottoalbero del task ricorsivo, compare un' altra istanza del medesimo task. La ricorsione viene ripetuta fino a quando qualche task ne inibisce l' esecuzione.

Supporto Automatico

Verranno descritte, adesso, le funzionalità principali messe a disposizione dal tool CTTE che supporta la modellazione dei task [MPS02]. Nella seguente Figura 10 viene mostrata uno screenshot del tool in funzione.

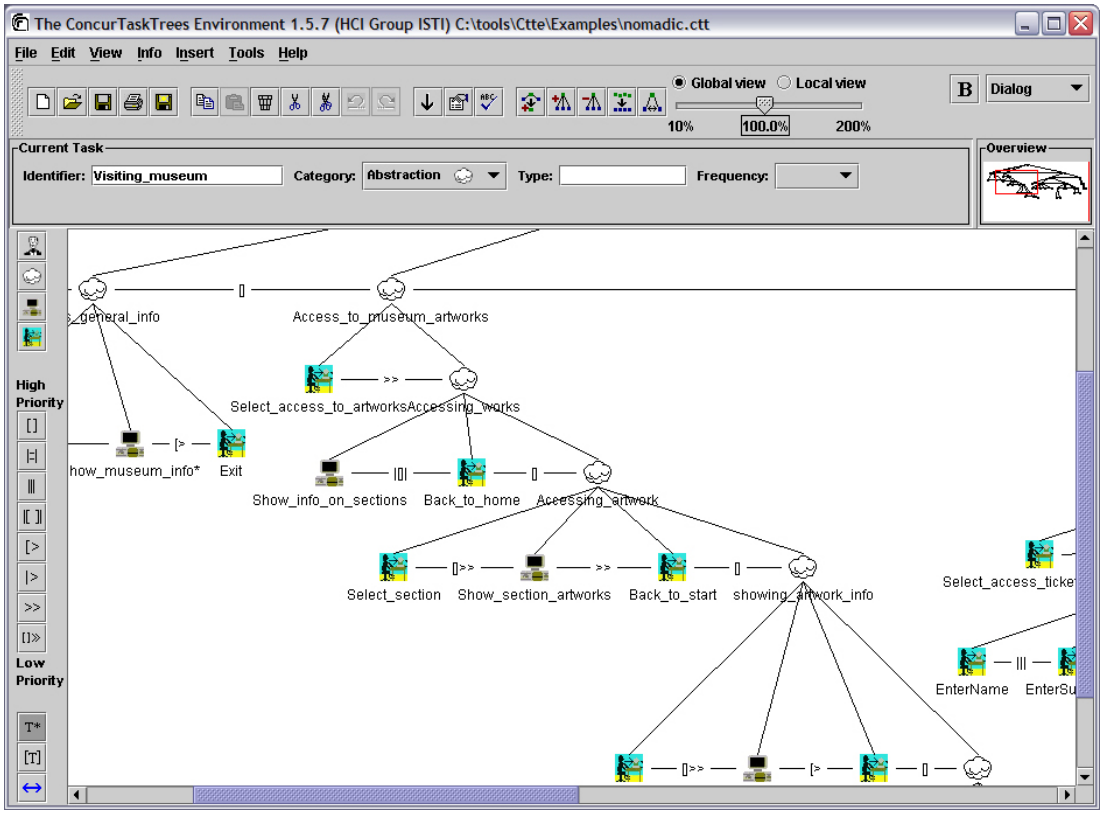

Figura 10 : Cattura di una tipica sessione di utilizzo del CTTE

- *Editing.* E' possibile inserire figli o fratelli di un altro task, selezionandone la tipologia desiderata. Sono disponibili, inoltre, le classiche tecniche di cut&paste che facilitano la copia di interi sottoalberi. La modifica delle relazioni temporali è estremamente semplice e permette di connettere e disconnettere nodi e sottoalberi molto velocemente.
- *Viste multiple.* Nella modellazione di applicazioni reali il modello dei task può raggiungere dimensioni estremamente grosse. In questi casi, l' utilizzo di scrollbar è utile ma non completamente sufficiente. Per meglio supportare l' analisi delle attività in questi casi, viene fornita una piccola overview che ne rappresenta la struttura logica senza riportarne i dettagli.
- *Supporto per applicazioni cooperative.* E' possibile rappresentare applicazioni cooperative definendo i diversi ruoli coinvolti e, per ciascuno di essi, le attività che vengono portate a termine contestualmente alla cooperazione. Sia il modello collaborativo generale, che quello di ciascun ruolo, vengono definiti mediante un albero delle attività per ognuno.
- *Descrizioni informali.* Viene messa a disposizione la possibilità di importare una descrizione testuale informale di un tipico scenario di utilizzo e di utilizzarla per la raccolta delle informazioni relative al modello da specificare.
- *Controllo completezza.* CTTE è in grado di verificare se la specifica del modello è significativa in termini della notazione utilizzata.
- *Salvataggio in diversi formati.* In qualsiasi momento, è possibile salvare o importare parti del modello utilizzando diversi formati. A questo proposito è stato definito un linguaggio XML per la dichiarazione di specifiche secondo la notazione scelta. E' inoltre possibile esportare l' albero modellato sotto forma di immagine JPG.

- *Confronto tra modelli.* Nel caso in cui sia necessario confrontare due diverse specifiche tra di loro, CTTE mette a disposizione una funzionalità in grado di evidenziare le differenze esistenti tra due modelli in esame. Affinché il confronto abbia significato, è necessario che i due modelli presentino diverse analogie, ad esempio, che si riferiscano ai medesimi ruoli etc.
- *Specializzazione del modello per una piattaforma.* Di ogni task è possibile, in accordo con la notazione utilizzata, specificare un numero elevato di proprietà; si può, ad esempio, indicare il tempo di esecuzione di una attività, descriverne gli output e gli input ed elencare le caratteristiche degli oggetti la cui manipolazione è necessaria al completamento dell' azione. Inoltre, viene consentito selezionare le piattaforme con le quali è compatibile l' esecuzione del task; la scelta può essere effettuata tra alcuni valori predefiniti messi a disposizione dal CTTE, oppure, nel caso questi non siano sufficienti, definendone all' occorrenza opportuni altri. Il pannello per la definizione di queste caratteristiche viene mostrato in Figura 11.

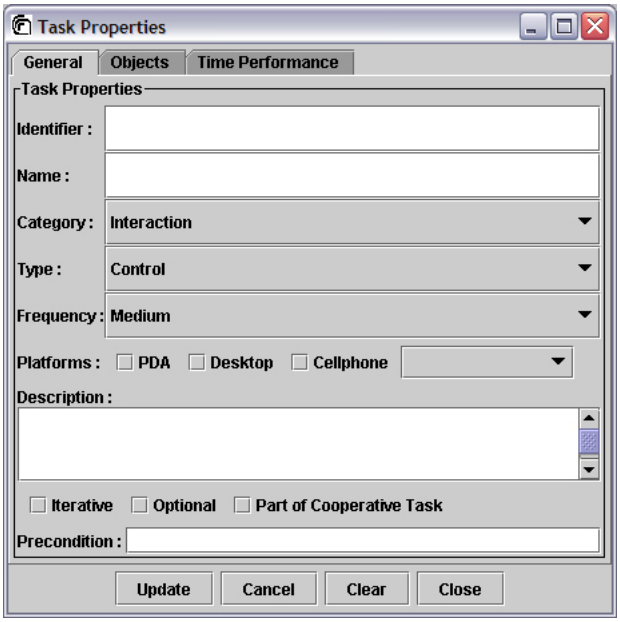

Figura 11 : Informazioni generali che è possibile specificare per ogni task

- *Simulatore.* CTTE consente di verificare la correttezza del modello specificato, anche attraverso la possibilità di simulare il comportamento del sistema, in corrispondenza delle diverse interazioni possibili che l' utente può compiere. Ad ogni istante, viene mostrata la lista dei task attualmente disponibili per essere eseguiti, l' interazione con il progettista avviene mediante la selezione del task del quale se ne intende simulare l' esecuzione. In questo modo, il sistema risponde alle interazioni dell' utente spostandosi nello stato risultante. Questa funzione è molto importante, permette all' analista di verificare se il comportamento è veramente quello che si intendeva modellare.

Vantaggi e svantaggi delle diverse tecniche

Le notazioni CTT, WIDE e GTA messe a confronto

Nella Figura 10 vengono mostrati i modelli di task, realizzati con le notazioni CTT, WIDE e GTA (un'altra notazione per task modelling sviluppata all'Università di Amsterdam), in grado di descrivere lo stesso caso. Due agenti, l'*Agente 1* e l'*Agente 2*, eseguono ciascuno due task, sequenzialmente: l'*Agente 1* esegue prima "Task 1.1" e poi "Task 1.2", mentre l'*Agente 2* esegue prima "Task 2.1" e poi "Task 2.2". `E importante anche l'ordine con il quale vengono eseguiti i task dei due diversi agenti. In particolare il primo task dell'*Agente 2* (il"Task 2.1") deve essere eseguito successivamente al secondo task dell'*Agente 1* (il "Task 1.2").

Analizzando i modelli realizzati si può notare come non tutte le notazioni permettano di ottenere modelli di task dotati dello stesso livello di espressività.

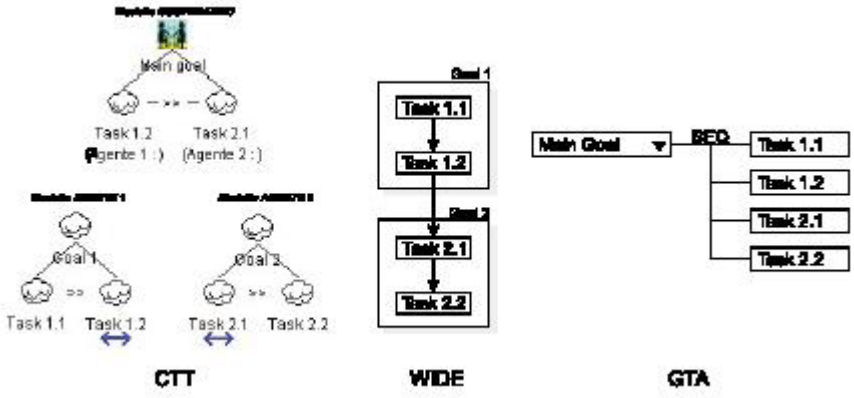

Figura 12: Confronto CTT, WIDE e GTA

Il modello CTT `e il più completo, dal momento che fornisce informazioni su quali siano i task da completare, sul loro ordine di esecuzione e sugli agenti responsabili. Per ottenere questo risultato è necessario realizzare tre distinte rappresentazioni (due relative agli agenti coinvolti ed una per il modello cooperativo).

WIDE e GTA permettono di specificare tutti i task della procedura nello stesso modello, ma in questo modo non integrano esplicitamente le informazioni relative alla delega dei task ai ruoli e agli agenti. Queste informazioni vengono definite a parte e a livello di singolo task.

CTT e GTA, rappresentando il modello con una forma ad albero, focalizzano l'attenzione sulla struttura dei task. La forma adottata da WIDE, invece, permette di evidenziare meglio il flusso di esecuzione del modello. Con WIDE è possibile rappresentare solo un sottoinsieme delle relazioni esprimibili in GTA o CTT (anche se questo non emerge dalla Figura 10, dal momento che l'unica relazione definita tra i task è la sequenza).

GTA enfatizza la decomposizione gerarchica (analogamente a CTT), l'uso di relazioni tra task (analogamente a CTT) e la definizione degli oggetti manipolati dai task (in modo analogo a CTT) e include inoltre aspetti tipici di un sistema di workflow come la descrizione della struttura organizzativa.

Riferimenti

[BRJ99] Booch, G., Rumbaugh, J., Jacobson, I., *Unified Modeling Language Reference Manual,* Addison Wesley, 1999.

[H87] Harel, D., "Statecharts: A Visual Formalism for Complex Systems", *Science of Computer programming*, 8, 1987.

[LPL98] Lu, S., Paris, C., Linden, K., "Towards the Automatic Construction of Task Models from Object-Oriented Diagrams", *Proceedings EHCI'98*, Kluwer, 1998.

[MWFF92] Medina-Mora, R., Winograd, T., Flores, R., Flores, F., The action workflow approach to workflow management technology, Proceedings of the 1992 ACM conference on Computer-supported cooperative work, CSCW 92, Toronto, Ontario, Canada.

[MPS02] Mori G., Paternò F., Santoro C., "CTTE: Support for Developing and Analysing Task Models for Interactive System Design", IEEE Transactions on Software Engineering, IEEE Press, Agosto 2002, Vol. 28, No. 8, pp. 797-813.

[P99] Paternò, F., Model-Based Design and Evaluation of Interactive Application. Springer Verlag, ISBN 1-85233-155-0, 1999.

[WIDE97] The WIDE Workflow Model and Language. WIDE Esprit Project 20280, Deliverable 5.1.2

Altri riferimenti:

Workflow Bibliography: http://www.comp.lancs.ac.uk/computing/research/cseg/projects/evaluation/MSc_biblio.html

Action Workflow: http://hci.stanford.edu/winograd/action.html

WfMC published documents: http://www.wfmc.org/standards/docs.htm

Workflow handbook 2003: http://www.wfmc.org/information/handbook2003.htm

Prodotti WfM: http://is.twi.tudelft.nl/~hommes/toolsub.html - 20