UN SISTEMA DI SUPPORTO DECISIONALE ATTIVO PER LA GESTIONE DELLA EMERGENZA: APPROCCI DI SVILUPPO BOTTOM-UP E TOP-DOWN

Claudio Balducelli, Adam M. Gadomski,
ENEA, CR Casaccia, 00060 Roma
balducelli_c@casaccia.enea.it, gadomski_a@casaccia.enea.it

SOMMARIO

Il lavoro descrive un Sistema di Supporto Decisionale Attivo, volto ad aiutare gli operatori ed i managers della emergenza ad identificare possibili errori, omissioni od incompletezze all'interno delle Procedure di Gestione delle Emergenze, a suggerire le risorse e gli interventi necessari. Questo stesso tipo di strumento può anche essere usato come sistema di addestramento per gli operatori specialmente per migliorare l'utilizzo delle risorse durante le emergenze. Il sistema fa uso di un interfaccia di *progetto ed esecuzione di procedure* di emergenza connessa ad un *Tool-Kit operativo* di supporto all'operatore durante la esecuzione delle procedure stesse. La problematica più generale di sviluppo di sistemi di supporto alla gestione dell'emergenza viene affrontata sia con un approccio *top-down*, ossia partendo dei requisiti funzionali che dovrebbero guidare lo sviluppo strutturato di sistemi IDSS (Sistemi di Supporto Decisionale Attivo/Intelligente), che *bottom-up*, ossia partendo dalla esigenza di realizzare rapidamente, tramite la prototipazione incrementale, ambienti non del tutto definiti, ma funzionanti, da migliorare in corso d'opera.

1. INTRODUZIONE

Il presente lavoro si inquadra all'interno della attività di ricerca svolta in ENEA in relazione allo sviluppo di un sistema di gestione della emergenza di tipo Intelligent Decision Support System (IDSS).

Il progetto è stato realizzato come parte del programma quadro MINDES(Managerial Intelligent Node for Decisional Emergency Support). Questo programma quadro rappresenta un contributo di ENEA alla iniziativa GEMINI (Global Emergency Management Information Network) del Comitato G7 [1]. L'obbiettivo principale di MINDES [2] è rappresentato dal supporto intelligente alla gestione delle Emergenze Industriali. La tecnologia usata si basa sullo sviluppo di sistemi orientati agli agenti, che siano in grado di fornire non solo dati e/o informazioni, ma anche di suggerire azioni e/o piani di intervento. Sistemi di supporto decisionale locali (basati sulla architettura MINDES) dovrebbero essere in futuro connessi con altri simili centri di gestione della emergenza nell'ambito di un sistema distribuito per la gestione delle emergenze, allo scopo di ridurre la probabilità di errori umani durante le decisioni ad alto rischio.

Un framework concettuale iniziale per tale tipo di sistemi è stato teoricamente definito in [3]. Ulteriori passi, supportati, in parallelo, dallo sviluppo di prototipi, sono stati fatti nelle tre seguenti direzioni strategiche:

1. **Identificazione e specificazione bottom-up** delle funzioni svolte da un gestore dell'emergenza, avente un ruolo definito, come la direzione di impianto petrolchimico.
2. **Modellazione top-down** delle generiche attività dei gestori della emergenza, delle modalità di collaborazione reciproca, indipendentemente dai rispettivi ruoli specifici.
3. **Sviluppo di agenti software** che supportino autonomamente le funzioni cognitive dei decisori.

Le esperienze nei progetti ISEM[4], MUSTER[5], CIPRODS [6], GEO[7], hanno evidenziato che la prima difficoltà nello sviluppo di sistemi di tipo IDSS per i gestori della emergenza, non si riferisce alle tecnologie del sofware bensì alla vaghezza ed incompletezza dei requisiti utente. Gli utenti potenziali di un IDSS sono persone estremamente pratiche ed hanno seri problemi a strutturare e descrivere mentalmente la loro attività, con vincoli di tipo logico-temporale, sotto forma di categorie generali, che rappresentino completamente il dominio di emergenza di loro competenza. Fino ad oggi, questo problema, pur essendo stato largamente affrontato, non è stato risolto completamente dagli ingegneri della conoscenza e dai modellisti di sistemi.

La complessità delle decisioni che i gestori delle emergenze devono prendere, è accentuata dal fatto che esse si possono riferire a diversi domini di intervento [8] così come evidenziato in fig. 1.

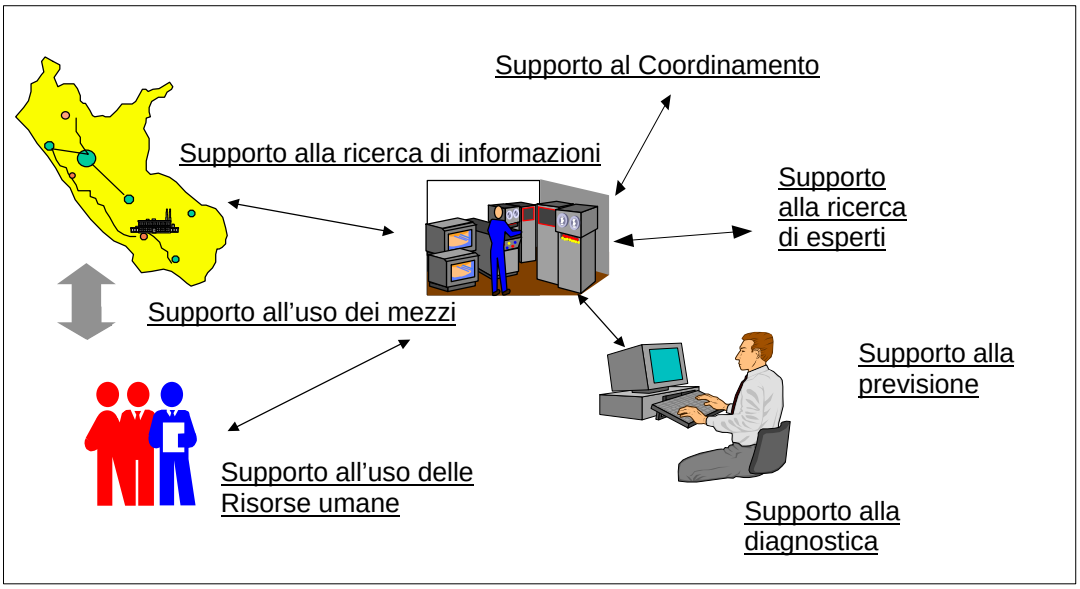

Fig. 1 – Possibili domini di intervento di un IDSS.

Nel nostro contesto, l'IDSS è pensato principalmente essere di supporto alla scelta di decisioni orientate alla esecuzione di azioni o di piani alternativi. Tali tipi di decisioni consistono nella ricerca di informazioni e di dati per la scelta degli interventi più appropriati e nella loro esecuzione con l'aiuto degli strumenti disponibili.

Partendo dalla analisi di differenti modelli di scenari più o meno generici viene confermata la presenza del ciclo ripetitivo delle attività di decision-making. Detto ciclo è diviso nelle seguenti generiche fasi[3] :

P1. Valutazione della situazione corrente (**situation assessment**),
P2. Ricerca delle cause,
P3. Valutazione/Predizione delle conseguenze,
P4. Pianificazione degli interventi(**task planning**),
P5. Esecuzione della azione corrente(**action planning**).

Nelle situazioni concrete, alcune di queste fasi sono omesse o realizzate in modo diverso tramite una loro decomposizione all'interno del processo cognitivo dei decisori. In pratica, all'interno di domini di emergenza molto strutturati, come impianti ad alto rischio, le fasi precedenti possono essere supportate dal seguente ciclo di funzioni computerizzate:

- rilevazione ed analisi di un evento anormale – questa è l'attività iniziale di **situation assessment**,
- scelta delle procedure adeguate – questa è una attività di **task planning** ma solo ridotta alla scelta di un piano di intervento all'interno di un dispositivo emergenza disponibile per gli operatori di impianto,
- scelta di un azione all'interno della procedura – questa è una attività di **action planning** ma solo ridotta alla scelta di azioni alternative sotto condizioni predefinite,
- messa a disposizione dell'utente di un set di strumenti software di supporto alla esecuzione della azione quali strumenti di ricerca di numeri telefonici, localizzazione delle risorse, modelli di simulazione etc.

2. PROGETTO BOTTOM-UP DI UN IDSS

Il progetto di tipo bottom-up è un approccio incrementale applicabile per lo sviluppo di sistemi qualitativamente innovativi, per i quali la fascia di applicazione e la complessità delle funzioni svolte difficilmente possono essere completamente definite sulla base dei requisiti utente. Questo approccio ha un carattere fortemente esplorativo ed è orientato alla verifica dell'utilità e della applicabilità di nuovi metodi software e tecnologie di tipo IDSS.

Nel caso di DSS attivi a supporto della gestione della emergenza, l'approccio bottom-up viene applicato nel seguente modo:

1. ***Analisi degli eventi*** previsti nel dominio applicativo considerato.
2. ***Analisi di funzioni modellabili*** che fanno parte della attività di un manager/operatore dell'emergenza.
3. ***Loro implementazione*** come funzioni di un DSS.

Il sistema DSS, così costruito, è del tipo **menù-driven**, poiché il sistema non contiene un modello del comportamento dell'utente basato sugli obbiettivi di gestione della emergenza.

La attivazione delle funzioni messe a disposizione dal sistema sono usualmente guidate dagli eventi esterni in accordo allo schema:

scenario relativo un evento -> piani necessari ad affrontare l'evento

dove la necessità è arbitrariamente riconosciuta dall'utente umano in base alla sua maggiore o minore esperienza del problema da affrontare.

Il sintomo iniziale può essere, ad esempio, la segnalazione di un incendio, l'informazione circa una esplosione

Qui la tendenza del modellista software è quella di catturate il maggior numero di dettagli correlati all'evento iniziatore della emergenza, o comunque all'evento più importante, al fine di poterlo caratterizzare nel modo migliore possibile.

Sfortunatamente, nella situazione reale, le caratteristiche particolari possono essere presenti in numero così elevato, che le loro combinazioni non possono in nessun caso essere tutte considerate e previste. Se si tenta di inserire nel sistema tutte le combinazioni possibili si arriva al paradosso di costruire sì un sistema che dà risultati più precisi nei casi considerati, ma che ha poca capacità di apprendere ed è poco adattabile a situazioni che si discostano anche in maniera minima da quelle considerate.

Nella fig 2 è riportato, sotto forma di albero, lo schema di una implementazione software che può essere vista in modo bottom-up o top-down. Come si vede si parte dallo sviluppo di una certa funzione (giudicata più importante) e dei relativi metodi di supporto per un certo evento, senza tener conto del fatto che altri eventi possono essere presenti (allo stesso livello di evento iniziatore dello scenario) o a livello più basso (eventi intermedi nello scenario). Tutti questi ulteriori eventi possono avere caratteristiche parzialmente simili, e potrebbero essere affrontati con funzioni e metodi anche essi parzialmente simili. Se la implementazione è completamente bottom-up le funzioni implementate inizialmente nel ciclo di vita del software, sono molto efficienti nel trattare gli eventi cui si riferiscono ma poco utilizzabili per gli altri. Quindi si ritiene opportuno migliorare il progetto di tipo bottom-up con alcuni vincoli di tipo to-down.

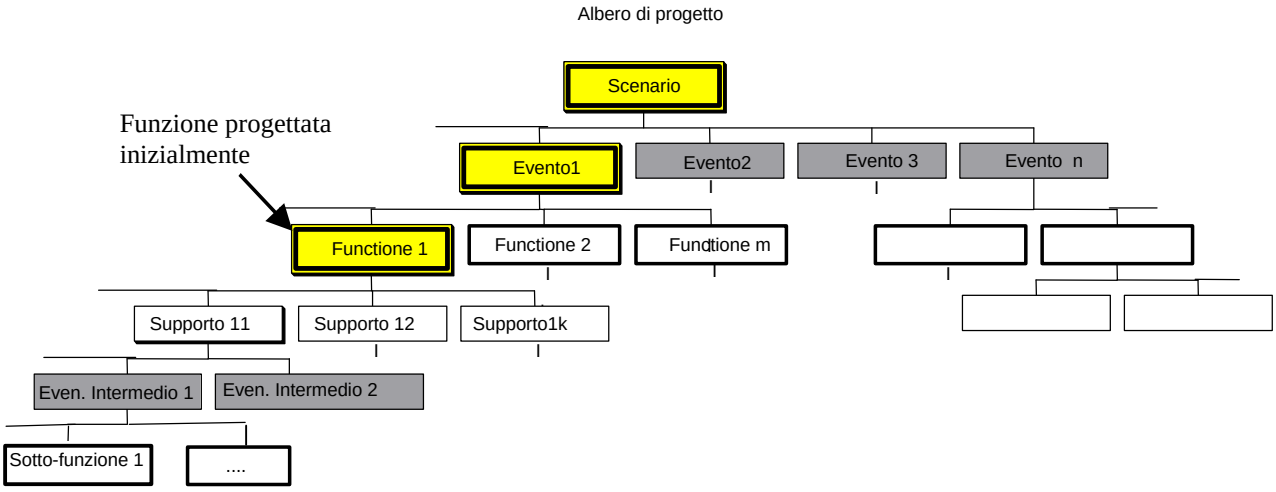

Fig. 2. Sviluppo bottom-up e top-down di un DSS attivo

3. LA PRIMA IMPLEMENTAZIONE BOTTOM-UP DEL SISTEMA GEO

Il sistema GEO (un DSS attivo per la Gestione delle Emergenze su reti petrolifere e depositi) [7], possiede una interfaccia basata su un Tool-Kit di supporto alla emergenza. Il Tool-Kit è un insieme di strumenti operazionali indirizzabili tramite conoscenza procedurale relativa alla gestione ed al controllo in tempo reale di incidenti severi su un deposito petrolifero.

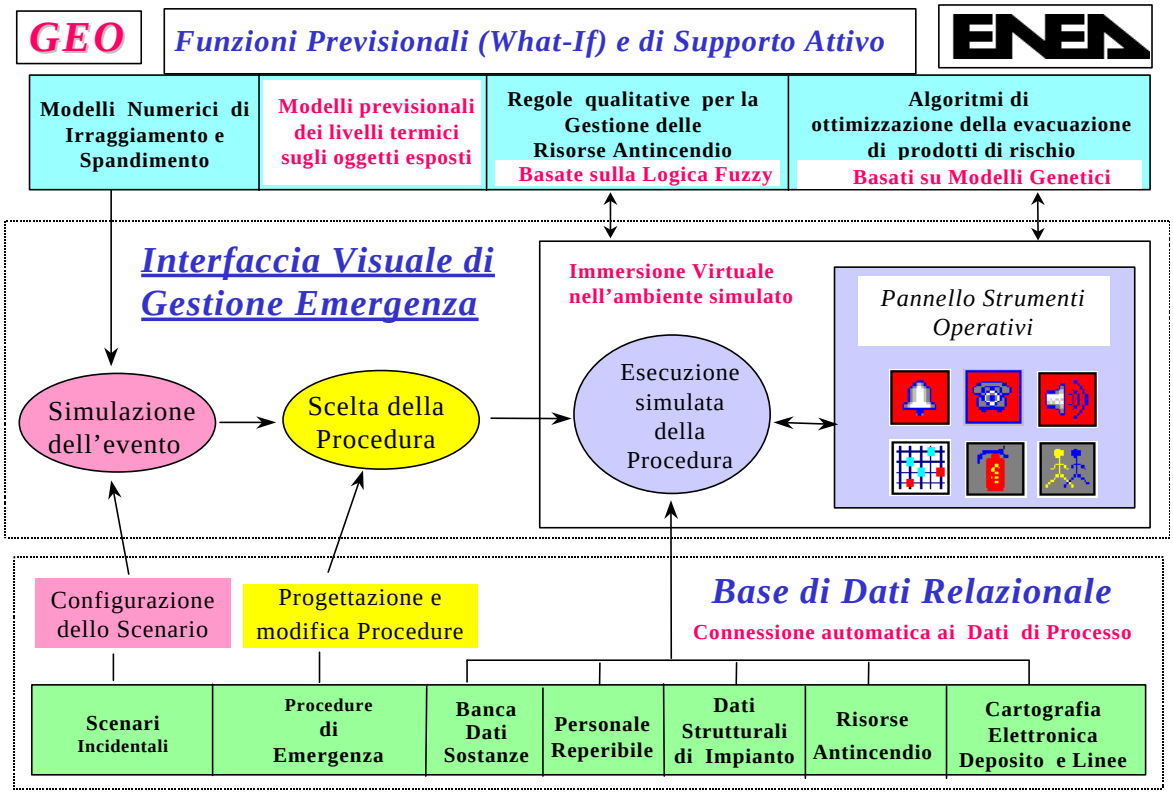

Fig. 3 – Architettura del sistema GEO

La interfaccia è stata sviluppata usando una implementazione incrementale di tipo bottom-up: partendo cioè da un dominio di emergenza ben preciso, con eventi particolari, e ruoli specifici degli operatori preposti alla gestione della emergenza.

Il sistema usa anche un insieme di modelli di simulazione numerici e discreti dei più importanti eventi nel dominio considerato.

L'architettura sofware è stata in seguito generalizzata per produrre uno strumento adattabile ad una larga classe di domini di intervento. L'interfaccia è stata progettata e sviluppata utilizzando la programmazione visuale e facendo uso dell'approccio funzionale per visualizzare e gestire le informazioni.

Lo stesso sistema può anche essere usato nel training operativo, specialmente al fine di migliorare la capacità degli operatori nell'utilizzo delle risorse durante le principali emergenze previste.

Come si vede dalla figura, il sistema è strutturato come una shell connessa con un Data Base relazionale da un lato e con gli output prodotti da modelli di simulazione dall'altro.

3.1 Configurazione di scenari di emergenza

GEO permette all'utente di configurare ed eseguire Scenari di Emergenza tramite funzioni di supporto al set-up di *dati primari* e *secondari* relativi al dominio fisico.

- i dati *primari* sono dati che direttamente caratterizzano lo scenario della emergenza come ad esempio il tipo di evento, la data e l'ora dell'evento, i sistemi e gli oggetti coinvolti etc.

- i dati *secondari* caratterizzano invece le condizioni sotto cui l'evento stesso avviene come ad esempio le condizioni meteo, e lo stato di funzionamento dei componenti di impianto.

3.2 Progettazione delle Procedure di Emergenza

GEO cointiene un Editor di Procedure the, durante la sessione di set-up del sistema, permette all'utente di progettare ed implementare le procedure di emergenza, e di connettere le procedure stesse ai diversi tipi di eventi possibili. La struttura delle procedure è suddivisa in *fasi* (parti del piano associate alla realizzazione di obbiettivi operativi più generali), dove ogni fase è suddivisa in diversi *compiti* (parti del piano associate alla realizzazione di più obbiettivi operativi elementari).

Testi esplicativi e vincoli di precedenza possono essere introdotti fra i compiti e le fasi in cui la procedura è suddivisa. L'Editor di procedura permette all'utente di collegare gli eventi (dati di tipo primario nello scenario configurato) alle procedure, ove più tipi di evento possono essere collegati alla stessa procedura.

3.3. Barra di menù degli Strumenti Operativi (Tool-Kit)

La corretta esecuzione di un singolo compito, può essere fatta in molti modi [9], [10] con differenti azioni operative in funzione dello stato e delle condizioni esterne del dominio su cui si opera. La esecuzione di ogni azione viene supportata dal sistema tramite le funzioni inserite nei diversi strumenti del Tool-Kit operativo. Gli strumenti diventano attivi e possono essere utilizzati dall'operatore tramite la selezione della relativa icona sulla barra di menù del Tool-Kit stesso. Come già menzionato precedentemente, uno stesso strumento può essere associato a diversi compiti durante la esecuzione di una certa procedura operativa.

In una visione orientata agli oggetti, ai singoli strumenti operativi sono associati degli attributi (che definiscono le modalità di utilizzo dello strumento), e dei metodi (che definiscono le modalità di esecuzione delle azioni). Il sistema contiene attualmente sia strumenti generali (non associati ad un particolare domino operativo), che strumenti più specifici (associati ad un particolare dominio operativo).

Sono riportati qui di seguito i principali strumenti operativi definiti nel Tool-Kit.

Supporta l'operatore nella fase di attivazione dell'allarme generale

ALLARME GENERALE

Supporta l'operatore nella scelta di una strategia atta ad ottimizzare l'utilizzo delle risorse

GESTIONE ANTINCENDIO

Supporta l'operatore nella scelta di una strategia atta a minimizzare il tempo di svuotamento dei

GESTIONE CONNESSIONI

Supporta l'operatore nel reperire i numeri telefonici delle Autorità esterne da allertare

STRUMENTO TELEFONO

Supporta l'operatore nel ricercare il Personale in turno con competenze adeguate

GESTIONE PERSONALE

Essi possono essere facilmente inseriti o rimossi dalla barra degli strumenti operativi.

3.4. Funzioni di "What If" e di supporto attivo

Come visualizzato nella figura 3, gli strumenti operativi utilizzano un insieme di funzioni di supporto attivo, ed in particolare:

1) Modelli numerici del processo fisico (per predire i livelli di irraggiamento termico e l'andamento delle temperature sugli oggetti a rischio in presenza di incendi);
2) Regole qualitative preposte alla ottimizzazione della gestione delle risorse antincendio;
3) Algoritmi di minimizzazione dei costi di evacuazione dei prodotti a rischio(atti a ridurre il tempo, i rischi e i danni economici durante le fasi di svuotamento rapido di serbatoi fortemete irraggiati).

Le seguenti metodologie sono state utilizzate per rendere più attive le funzioni sopradescritte:

- Generazione di regole Neuro-Fuzzy: ciò significa migliorare l'efficacia e l'adattabilità di un insieme di regole specifiche, inizialmente acquisite con l'aiuto di un esperto di dominio, attraverso una *analisi off-line* di dati forniti da un simulatore di processo. Una rete neurale è stata utilizzata per apprendere ed estrarre i *comportamenti termici* dominanti da un set di predefinite configurazioni tipiche di un evento di tipo incendio su un insieme di oggetti di rischio (serbatoi); gli algoritmi basati sulla logica fuzzy sono stati usati per *discretizzare* i comportamenti termici e produrre regole fuzzy da utilizzarsi *on-line*.
- Modelli Genetici: questi modelli sono stati utilizzati per ricercare soluzioni efficienti di svuotamento dei serbatoi a rischio su altri serbatoi o linee sufficientemente lontane dal luogo dell'incendio. Essi tendono a minimizzare il tempo di svuotamento considerando un set di vincoli come la disponibilità di serbatoi targhet e la minimizzazione del danno economico.

4. PROGETTAZIONE TOP-DOWN E SVILUPPO AGENT-BASED DI UN IDSS

Rispetto alla progettazione di tipo bottom-up la realizzazione top-down di un sistema di supporto alla gestione delle emergenze si sviluppa nel seguente modo:

- ***in modo top-down*** dalla analisi del processo di decision-making (per una certa classe di emergenze), si ***decompongono*** le fasi di gestione in sotto-attività modellabili in modo formale,
- si ***modellano*** e si ***formalizzano*** le attività stesse,
- si definiscono le ***interfacce utente*** necessarie alla loro esecuzione automatica, rispetto ad un scenario generico di decision-making.

E' stata impiegata a questo scopo la ontologia TOGA (Top-down, Object-based, Goal-oriented approach). Il framework concettuale TOGA di una decomposizione top-down viene presentata nei papers [11], [3]. Per i nostri scopi correnti, è necessario richiamare alcune definizioni chiave e spiegazioni.

Dato un ambiente a rischio, nel quale si vuole introdurre un sistema di supporto alla gestione delle emergenze, **obiettivo di progetto** è stato dell'ambiente che si vuole raggiungere con il supporto del sistema da progettare.
Funzioni sono quelle proprietà del sistema, che sono necessarie per raggiungere l'obbiettivo di progetto.
Processi elaborativi sono processi eseguiti da algoritmi di calcolo (software) ed unicamente descritti da metodi computazionali.
E' generalmente possibile realizzare la stessa funzione usando differenti processi di calcolo, per esempio, usando differenti tecnologie software.
Architettura (struttura del sistema) è una struttura di relazioni, invarianti durante il ciclo di vita del sistema, fra i componenti del sistema stesso.
Obiettivo di intervento è uno stato aspettato(finale) del dominio, che attiva nel sistema un processo decisionale orientato alle azioni in uno specifico stato iniziale del dominio stesso.
Il **Compito** definisce i cambiamenti necessari sul dominio per raggiungere l'obiettivo di intervento. Esso è la *proprietà* (astratta) di una **azione** necessaria a questo scopo.

Naturalmente, in diversi stati del dominio e con diversi mezzi, lo stesso task può essere eseguito da azioni diverse.

L'obiettivo di progetto di un sistema uomo-macchina viene decomposto in sotto obiettivi collegati con funzioni, e le funzioni sono proprietà dei processi elaborativi. I processi sono realizzati dal sistema software, utilizzando la struttura architetturale del sistema stesso.

Nel sistema GEO, l'obiettivo generico di gestione della emergenza è rappresentato come N obiettivi di progetto alternativi. Ogni obiettivo è definito da una classe di stati del dominio di emergenza (dipendenti dalle specifiche condizioni iniziali). Il raggiungimento degli N obiettivi richiede la realizzazione di N funzioni ed ogni funzione è realizzata da una predefinita procedura. Le procedure richiedono la esecuzione di una sequenza ordinata di funzioni da parte del sistema uomo-macchiana.

4.1. Applicazione del sistema GEO per il supporto alla gestioone della emergenza

Come visualizzato in fig. 3, il sistema GEO realizza tre funzioni principali:

Configurazione di scenari di emergenza
Supporto on-line nella realizzazione di procedure di intervento and
Simulazione dello scenario dell'emergenza.

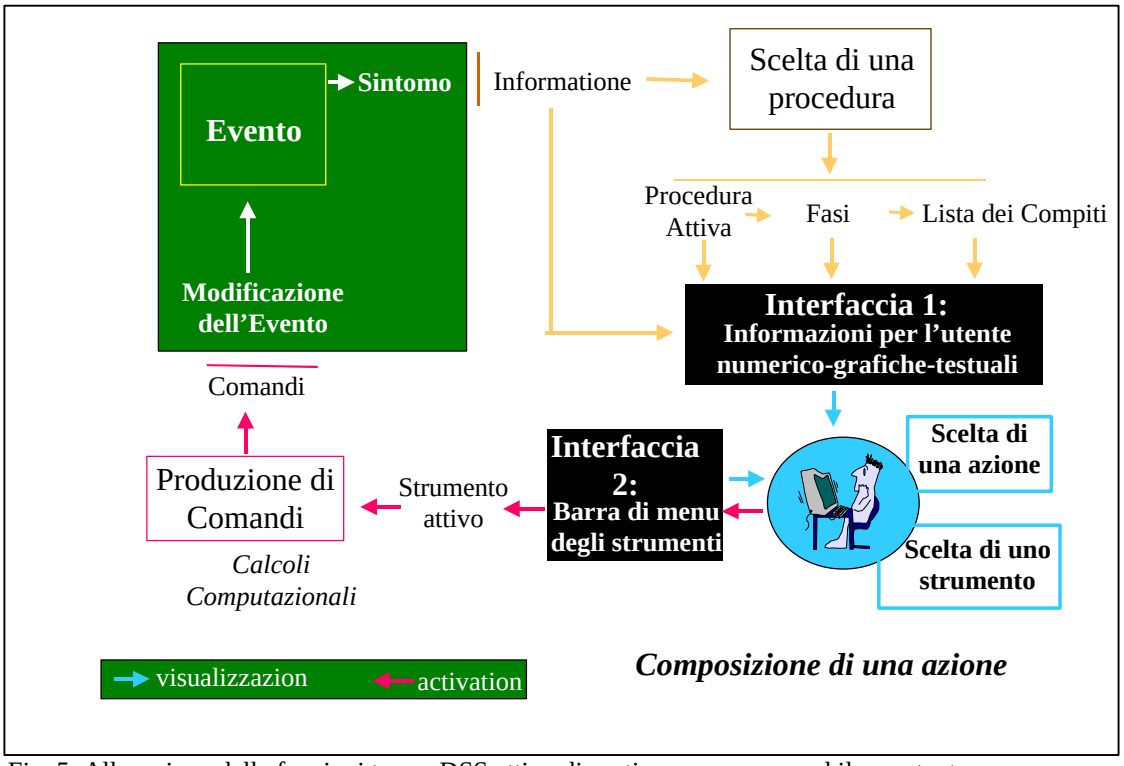

Fig. 5 Allocazione delle funzioni tra un DSS attivo di gestione emergenza ed il suo utente

La funzione di *supporto on-line* di GEO è decomposta ed e allocata nell'ambito di una architettura generale definita in modo top-down, come illustrato in fig 5.

Il presente scenario rappresenta questo tipo di allocazione.

F1. **Sistema**: sceglie la procedura adeguata, da una Base di Procedure strutturata, basandosi su una serie di informazioni iniziali e su un sintomo corrente.
F2: **Sistema**: illustra le fasi della procedura e dei compiti, dove l'ordine di esecuzione dei compiti dipende (almeno parzialmente) dalla scelta dell'utente.
F3. **Utente**: seleziona o sceglie un compito.
F4. **Sistema:** mostra le icone degli strumenti che possono essere utilizzati per eseguire il compito
F5. **Utente**: sceglie una delle azioni messe a disposizione dallo strumento, in quanto il compito può essere in generale eseguito mediante azioni diverse. La scelta dell'azione è un processo mentale che non è stato modellato.
F6. **Sistema**: esecuzione della azione mediante la produzione e la realizzazione di comandi che modificano gli attributi degli oggetti nel dominio.

Avvertenza: In questo caso, la complessità degli strumenti non la cosa più importante ai fini dello sviluppo top-down dell'intero sistema; utilizzando un approccio bottom-up, ogni strumento può essere sviluppato sulla base di conoscenza operativa dettagliata circa un certo tipo di intervento ed in accordo con le possibilità di calcolo fornite da metodi anche avanzati, come ad esempio algoritmi di tipo genetico o neuro-fuzzy.

E’ stata verificata la possibilità di configurare/integrare GEO con una architettura IDSS di tipo multi-agent, migliorando l’autonomia e la flessibilità del protocollo di comunicazione uomo-macchina. In un tale contesto le funzioni svolte dagli strumenti possono essere anche supportate da agenti autonomi. In relazione alla complessità dei loro compiti gli agenti possono essere più o meno “intelligenti”

4.2. Agenti Intelligenti nel progetto di un IDSS

Applicare metodi degli agenti intelligenti al progetto di un IDSS significa applicare la metafora [12] che vede l’IDSS come un “lavoratore” equipaggiato di vari strumenti adeguati al proprio ruolo all’interno di un determinato scenario di intervento. Egli può interagire con gli oggetti fisici ma, principalmente, “egli” comunica con diversi tipi (attivi o passivi) di sorgenti di informazioni al fine di realizzare i suoi compiti. L’attività dell’agente intelligente è principalmente rivolta a:

- riconoscere/valutare i propri compiti all’interno di uno scenario;
- scegliere/pianificare azioni adeguate ai compiti e allo scenario;
- realizzare le azioni scegliendo degli strumenti da utilizzare in modo opportuno.

L’architettura generale IPK(Information, Preference, Knowledge) e la funzionalità di un agente astratto intelligente sono descritte in [13][14][11][3][15]. Altre architetture ad agente possono essere considerate, analizzando ad esempio [16][17][18][19], ove l’ultima referenza si riferisce ad agenti BDI (Belief, Desire, Intention). Comunque, indipendentemente dal tipo di struttura ad agente assunta, è utile distinguere i seguenti componenti funzionali di un agente intelligente:

1. Strato comunicativo (interfaccia esterna)
   1.1 zona percettiva (comunicazione con apparati fisici)
   1.2 zona attuativa (comunicazione con apparati fisici)
   1.3 zona comunicativa (comunicazione con altri agenti)

2. Strato di ragionamento di dominio
   2.1 strato associativo (ragionamento neuro-fuzzy/subsimbolico)
   2.2 strato comprensivo (basato su ontologie e modelli concettuali)
   2.3 strato delle preferenze e di selezione degli obiettivi
   2.4 strato della conoscenza operativa (sui piani e sulle azioni)

3. Strato di meta-ragionamento;
   Esso include le strategie di generazione degli obbiettivi, di scelta dei modelli da usare, delle regole da applicare etc. I risultati ottenuti nello strato di meta-ragionamento forniscono dei parametri, preferenze e conoscenze che vengono usati nello starto di ragionamento di dominio cambiandone il comportamento.

In accordo con la precedente struttura, gli agenti senza lo strato di meta-ragionamento non sono intelligenti. Nello strato di meta ragionamento si realizza l’*apprendimento* dell’agente che modifica il suo modo di comportarsi a livello di strato di dominio.

Questa struttura è, per quanto possibile, indipendente dal tipo di agente. Essa richiede una identificazione dei seguenti concetti: *informazione*, *preferenze*, *conoscenze*. Per ***informazione*** si intendono tutti i dati acquisibili dal dominio di attività degli agenti, le ***preferenze*** sono regole che organizzano i possibili stati del dominio in forma di grafi orientati i cui archi suno appunto le preferenze, la ***conoscenza*** è data da ogni struttura concettuale del dominio che permetta il processamento della informazione[20].

Durante la progettazione di un IDSS, in dipendenza della complessità del sistema, le funzioni possono essere allocate ad uno o più agenti intelligenti. La fig.6 illustra una situazione in cui la definizione di una azione e la scelta di uno strumento (ciò significa la pianificazione di una azione), viene allocata ad un agente intelligente. La pianificazione è eseguita in accordo con dei vincoli predefiniti relativi al dominio ed al ruolo dell’operatore. In situazioni in cui occorre prendere una decisione, un certo strumento del Tool-Kit operativo viene suggerito all’operatore, dall’agente software.

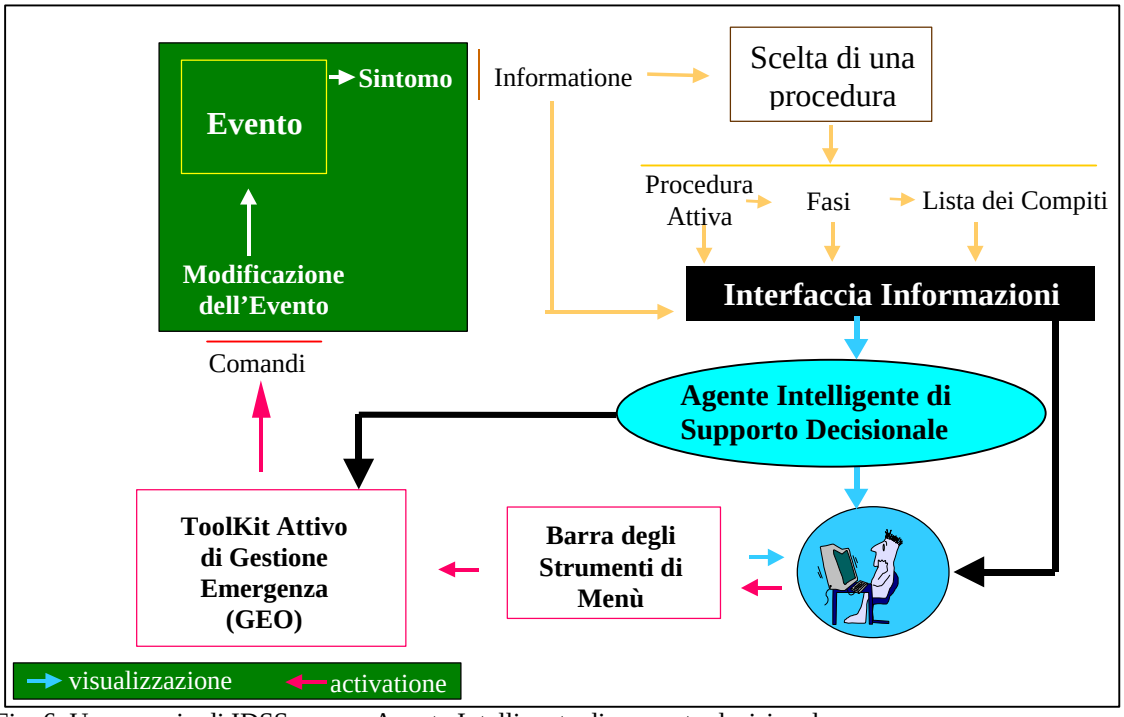

Fig. 6 Un esempio di IDSS con un Agente Intelligente di supporto decisionale.

5. CONCLUSIONI

Una applicazione dei metodi bottom-up a top-down supportati dalla tecnologia multi-agente, permette di affrontare più efficacemente il problema principale nello sviluppo di IDSS innovativi a supporto dei gestori della emergenza: la pianificazione e la gestione del progetto stesso.

Un IDSS è un sistema basato sulla conoscenza e, per questa ragione, richiede un *ciclo di vita* specifico, che differisce dai sistemi software tradizionali. Ciò viene affrontato in modo esteso nella letteratura scientifica. Comunque, indipendentemente dalla tecnologia software adottata, la realizzazione di un IDSS deve essere decomposta nelle seguenti passi principali:

1. Analisi preliminare dei possibili requisiti-utente e vincoli di progetto.
2. Scelta del dominio dell'emergenza (con le classi di eventi) e ruoli degli end-users.
3. Prima fase di acquisizione di conoscenza per modellare una classe selezionata di situazioni di emergenza e le relative funzioni necessarie.
4. Allocazione delle funzioni modellate in parte all'utente umano ed in parte al sistema IDSS.
5. Seconda fase di acquisizione di conoscenza per modellare uno scenario più generico e generalizzare le funzioni definite nella prima fase.
6. Allocazione delle nuove funzioni modellate all'utente umano e a vari componenti del sistema IDSS.
7. Progettazione degli agenti intelligenti.
8. Sviluppo degli agenti e degli strumenti operativi.
9. Integrazione del sistema.
10. Test e validazione usando casi di emergenza.
11. Modifica e miglioramento del sistema.
12. Training utente.

Poiché un sistema IDSS è fortemente eterogeneo, il progetto deve essere supportato da:

- Piattaforme di sviluppo software; linguaggi agent-oriented di tipo FIPA[21] [17];
- Vocabolario orientato al progetto(continuamente aggiornato);
- Definizione di una ontologia relativa alla gestione della emergenza, nonché una meta-ontologia di riferimento per il team di sviluppatori, quale TOGA, UML[22] o National Rose [23];

Questi strumenti sono indispensabili per una buona realizzazione del progetto. In ogni caso, la adozione di un approccio top-down/bottom-up orientato agli agenti potrebbe fortemente facilitare l'impegno e ridurre i tempi del progetto. Questo approccio rende infatti più trasparente l'architettura funzionale del del progetto stesso, e, allo stesso tempo, permette di realizzare in parallelo una identificazione top-down e bottom-up delle funzioni e della loro allocazione a specifici agenti software.

REFERENCES

[1] GEMINI96,http://wwwerg.casaccia.enea.it/ing/tispi/gadomski/gad-gemi.htm, http://wwwerg.casaccia.enea.it/ing/tispi/gadomski/gad-prog.html

[2] S. Bologna, A.M.Gadomski. Managerial Intelligent Node of Decision Support for Emergency Supervisory (MINDES) Program. The proceedings of the GEMINI Annual International Meeting Roma, 3-6 settembre 1996.

[3] A.M.Gadomski , S. Bologna, G. Di Costanzo. Intelligent Decision Support for Cooperating Emergency Managers: the TOGA based Conceptualization Framework. The Proceedings of "TIEMEC 1995: The International Emergency Management and Engineering Conference", J.D. Sullivan, J.L. Wybo, L. Buisson (Eds), Nice, May, 1995.

[4] M.Sepielli, A.M.Gadomski, G.Brenna, Information-Flow Model for Emergency Management System, ENEA ,TERM-MEP, Internal Report, C.R.E. Casaccia, May,1989.

[5] C.Balducelli, S. Bologna,G. Di Costanzo,, A.M. Gadomski,G. Vicoli. Computer aided training for cooperating emergency managers: same results of MUSTER Project. Proceedings of the Membrain '95 Conference,Olso, 18-21 Giugno 1995.

[6] A.M.Gadomski, G. Di Costanzo. Intelligent Decision Support System for Industrial Accident Management. The Proceedings of 8th European Simulation Symposium , SCS, Genoa, Oct.,1996.

[7] C.Balducelli, G.Vicoli, N.Arcidiacono. "EMAT (Emergency Management Active Tool-Kit) An Active Decision Support and Training System based on the Configuration and Simulated Execution of Emergency Scenarios", Halden Project Workshop Meeting on "Intelligent Decision Support Systems for Emergency Management" Halden, 20th-21st October, 1997

[8] A.M.Gadomski. The Nature of Intelligent Decision Support Systems. Intelligent Decision Support Systems for Emergency Management. The HALDEN Project Workshop, Oct. 20-21,1997. The transparent sheets: http://tispi.casaccia.enea.it/activities/emergency/eme-idss/sld001.htm

[9] B. Azvine, N. Azarmi, K.C. Tsui. An Introduction to Soft Computing – A Tool for Building Intelligent Systems. Lecture Notes in Artificial Intelligence 1991, "Software Agents and Soft Computing", Nwana- Azarmi Eds. pp. 191-210.

[10] G.J. Yaverbaum, M.A. Reynolds. Managerial Problem Identifier (MPI): Integrating Expert System and Decision Support Technologies. Expert Systems with Application, Vol. 3, pp. 507-515, 1991.

[11] A.M.Gadomski, J.M.Zytkow. Intelligent Agents: Paradigms, Foundations and Expectations. The Proceedings of the Second International Round-Table on Abstract Intelligent Agent, AIA94, Printed by ENEA, Rome, 1995.

[12] M.J.Wooldridge,N.R.Jennings (Eds.), Intelligent Agents, Springer Verlag,1995.

[13] A.M. Gadomski, TOGA: A Methodological and Conceptual Pattern for Modeling Abstract Intelligent Agent. Proceedings of the First International Round-Table on Abstract Intelligent Agent (AIA '93). Rome., Jan. 23-27, 1993. , A.M. Gadomski (editor). Printed by ENEA, Feb.1994.

[14] C.Balducelli, A.M. Gadomski. Intelligent Agents in Computer Supported Training for Emergency Management. In the Proceeding of the First International Round-Table on Abstract Intelligent Agent, Agent (AIA '93). Rome., Jan. 23-27, 1993. , A.M. Gadomski (editor). Printed by ENEA, Feb.1994.

[15] G.Di Costanzo, A.M.Gadomski, A Prototype of an Active Decision Support System, based on an Abstract Intelligent Agent Architecture, TIEMEC 1997: Proceedings of "The International Emergency Management and Engineering Conference", 1997.

[16] M.P.Singh, Multiagent Systems: A Theoretical Framework for Intentions, Know-How, and Communications. Springer-Verlag, 1994.

[17] G.M.P. O'Hare, N.R.Jennings. Foundation of Distributed Artificial Intelligence, J.Wiley&Sons, 1996.

[18] K.Sycara at al., Distributed Intelligent Agents, In IEEE Expert: Intelligent Systems & Their Applications. Vol.11, N.6, Dec 1996.

[19] B.Dunin-Klepicz, J.Treur "Compositional Formal Specification of Multi-Agent Systems", In Intelligent Agents, M.J.Wooldridge, N.R.Jennings (Eds.), Springer Verlag, 1995.

[20] A.M.Gadomski: IKP: Information, Knowledge, Preferences. White paper,1998, http://wwwerg.casaccia.enea.it/ing/tispi/gadomski/gad-dict.htm
[21] FIPA, Agent Communication Language. The FIPA document. http://drogo.cselt.stet.it/fipa/spec/fipa7412.htm
[22] UML; White paper: Unified Modeling Language for Real-Time Systems Design", http://www.rational.com/uml/hot and http://www.rational.com/uml/1.1.
[23] Rational Rose; http://www.rational.com/demos/rose4demo/wlkthr_rse.html