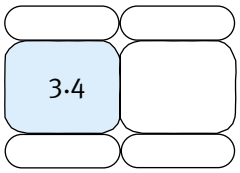
GRID: LA ROADMAP DEI PROGETTI ITALIANI ED EUROPEI

Mirco Mazzucato

Si presenta una roadmap dell'evoluzione della Grid: l'infrastruttura d'avanguardia per la condivisione di sistemi di comunicazione, storaggio e calcolo. L'idea nasce in America, ma si sviluppa immediatamente anche in Italia che, a livello europeo, assume ruoli di primissimo ordine sia nello sviluppo che nella realizzazione di e-Infrastrutture (Internet e Grid) per tutti i settori delle scienze moderne che coinvolgono nella produzione di sapere e conoscenze team e risorse distribuiti sul territorio.



1. INTRODUZIONE

La prima grid nazionale in Europa nasce, nel febbraio del 2000, all'interno dell'Istituto Nazionale di Fisica Nucleare (INFN), l'Ente pubblico italiano che promuove, coordina e sviluppa ricerche sperimentali e teoriche di base nell'ambito della fisica nucleare e sub-nucleare, da sempre all'avanguardia nello sviluppo di tecnologie avanzate. Il progetto INFN-Grid (http://grid.infn.it) coinvolge da allora le strutture di calcolo di 20 sedi localizzate nelle principali Università italiane e dei 5 laboratori nazionali. Se pur focalizzato allo sviluppo dell'infrastruttura di calcolo per il *Large Hadron Collider* (LHC), il nuovo acceleratore del CERN (*Consiglio Europeo per la Ricerca Nucleare*) di Ginevra, il progetto parte, fin dall'inizio, con l'obiettivo di sviluppare una soluzione generale aperta alle esigenze di altre scienze e dell'industria. Il progetto INFN Grid anticipa di quasi un anno il programma d'investimento di 250 M£ compiuto in due fasi dal governo inglese per l'e-Science che rapidamente si è focalizzato sullo sviluppo delle Grid in UK.
La grid è la nuova tecnologia che permetterà agli scienziati di collaborare a grandi obiettivi internazionali di ricerca raggiungibili solo mettendo in comune le centinaia di migliaia di PC e i grandi super-computers delle Università, degli Enti di Ricerca, dei Centri di Calcolo di tutta l'Europa e del mondo, come se fossero un'unica grande risorsa.
Inoltre, come il WEB sviluppato al CERN agli inizi degli anni '90 ha permesso di rivoluzionare l'accesso all'informazione disponibile in domini di gestione diversi e distribuiti geograficamente, così il **middleware** di Grid rivo-

Il termine **middleware** - ossia (soft)ware intermedio - denomina la classe di programmi che consentono di collegare tra loro applicazioni diverse. Si tratta, in sostanza, di interfacce logiche con cui moduli eterogenei vengono a costituire un insieme funzionale in grado di elaborare e scambiare dati tra i differenti livelli di software. In uno schema a strati di un sistema informativo, il *middleware* si colloca tra il sistema operativo e i programmi applicativi o di produttività. In sostanza, il *middleware* costituisce una specie di infrastruttura del sistema in cui sono raggruppati e messi a fattor comune funzionalità condivisibili dalle varie applicazioni.

luzionerà lo sfruttamento dell'enorme mole d'informazioni digitali che le moderne società producono sempre più abbondantemente e renderà fruibili a tutti risorse computazionali indipendentemente dalla loro localizzazione permettendo lo sviluppo di nuove applicazioni in ogni settore.
Per raggiungere questo scopo il nuovo Middleware di Grid affianca al servizio HTTP del WEB una nuova serie di servizi che consentono, da una parte, di accedere in modo trasparente ad ogni tipo d'informazione digitale: immagini di satelliti, dati da acceleratori come LHC del CERN, basi di dati della genomica e proteomica, immagini mediche da TAC, RMN, PET, disegni tecnici da CAD ..., indipendentemente dal dominio geografico o di gestione in cui si trovano, e, dall'altra, di sfruttare una qualunque risorsa computazionale per estrarre da questi dati grezzi i preziosi germi di conoscenza di cui la società ha bisogno per progredire[1]. Tutto questo è ottenuto in modo sicuro grazie ad un'infrastruttura di sicurezza distribuita, basata su certificati personali di tipo X509 rilasciati da un insieme d'autorità di certificazione legate tra loro da un rapporto di mutua confidenza e ad un sistema d'autorizzazione che permette ai possessori di mantenere un completo controllo locale su chi e quando può usare le proprie risorse e, nello stesso tempo, alle organizzazioni virtuali di stabilire dinamicamente in modo centralizzato delle politiche generali per regolare le priorità e l'uso delle stesse risorse.

2. CENNI STORICI

L'idea della Grid nasce negli USA alla fine degli anni '90 come risultato finale dell'elaborazione collettiva della comunità scientifica internazionale sul calcolo distribuito, iniziata agli albori di quel decennio.
È, infatti, nel 1989-90 che comincia all'INFN, al CERN e nei maggiori Centri di Calcolo avanzato in Europa e negli USA, la rivoluzione che si affianca a quella del WEB e di Internet, che porterà, nel giro di pochi anni, alla sostituzione dei grandi supercalcolatori mainframe con cluster di workstation e PC personali. I mainframe, costruiti su architetture speciali sviluppate per pochi grandi sistemi, richiedono tempi e costi di progettazione e realizzazione che rapidamente non riescono più a tenere il passo con lo sviluppo dei processori "commodity" adottati da milioni di utenti. I semplici PC (che tutti possono trovare e gestire) e i dischi poco costosi a questi collegati, assieme alle interfacce di rete standard e agli standard backbones per le reti locali (Ethernet), diventano componenti elementari per costruire sistemi di calcolo e memoria davvero ragguardevoli. Le prestazioni di queste componenti da allora migliorano, seguendo la ben nota legge di Moore, di un fattore x2 ogni 18 mesi a parità di costo e le loro dimensioni si miniaturizzano tanto che oggi si arriva ad alloggiare centinaia di CPU e dischi in un rack standard di 60×80 cm^2.
L'INFN è stato all'avanguardia in questa trasformazione.
Nel 1989 realizza, infatti, al CERN, in un comune pionieristico progetto di ricerca e sviluppo con Digital, uno dei primi cluster di workstations basato su processori commodity, noto come "INFN Farm". Esso mostra al mondo scientifico come questa tecnologia può essere utilizzata di routine dall'esperimento DELPHI per le proprie produzioni con costi che, per le applicazioni di quell'esperimento, a parità di potenza erogata, sono inferiori di circa un ordine di grandezza rispetto a quelli del grande mainframe della Computing Division.
Negli anni '90 questa trasformazione si completa. I modelli di calcolo "centralisti" basati sui grandi supercomputers (IBM, Cray...), attorno ai quali sono nati i grandi Centri di Calcolo con migliaia di utenti negli USA e in Europa, vengono progressivamente sostituiti da modelli distribuiti che possono sfruttare i clusters di PC, i quali, attualmente, sono disponibili in quasi tutte le Università e Centri di Ricerca.
L'ultimo passo importante per le Grid viene dalla riduzione dei costi per l'uso della rete geografica. Grazie alle liberalizzazioni intervenute in tutto il mondo a metà degli anni '90

[1] Per un inquadramento del Grid Computing, si veda, per esempio: Migliardi M., "Grid computing: da dove viene e che cosa manca perché diventi una realtà?". *Mondo Digitale*, Anno III, n. 10, giugno 2004, p. 21-31.

nel settore delle telecomunicazioni, i costi cominciano a decrescere ancora più rapidamente di quanto previsto dalla legge di Moore per CPU e dischi.
Alla fine degli anni '90 sono quindi disponibili, su una rete a banda larga che ormai collega le università e i centri di ricerca di tutto il pianeta con velocità di trasmissione sempre più elevata e costi sempre più ridotti, un numero crescente di risorse computazionali e di memoria (Figura 1). Si pone quindi con urgenza il problema dello sviluppo di una nuova tecnologia che permetta alla comunità scientifica di sfruttare e condividere in modo dinamico queste risorse distribuite per accelerare i processi innovativi ed aumentare la propria efficienza nella produzione di nuova conoscenza.
Il concetto di grid, presentato per la prima volta in un famoso libro edito da Ian Foster e Karl Kesselmann nella primavera del 1999 [1], per risolvere questo problema propone una strategia che è rapidamente adottata da tutta la comunità scientifica internazionale e che si basa sullo sviluppo di servizi e protocolli standard per la condivisione di risorse distribuite che nascondono all'utente l'eterogeneità delle risorse stesse (Figura 2).

FIGURA 1
"Farm" di processori e dischi

3. LO SVILUPPO DELLE GRID IN US ED EUROPA

3.1. La fase pionieristica (1999-2000)

Negli Stati Uniti il gruppo Globus di Foster e Kesselman inizia già a metà del 1998 a sviluppare alcuni servizi di base che cercano di realizzare in pratica il concetto di Grid. Questi sono rapidamente resi disponibili come Open Source attraverso il Globus Toolkit (www.globus.org) che diviene un primo standard internazionale de facto per l'accesso e la condivisione di risorse computazionali distribuite.
In Europa nel 2000 è l'INFN, che già aveva approvato il proprio programma di Grid nazionale, assieme a PPARC (Inghilterra) che stava

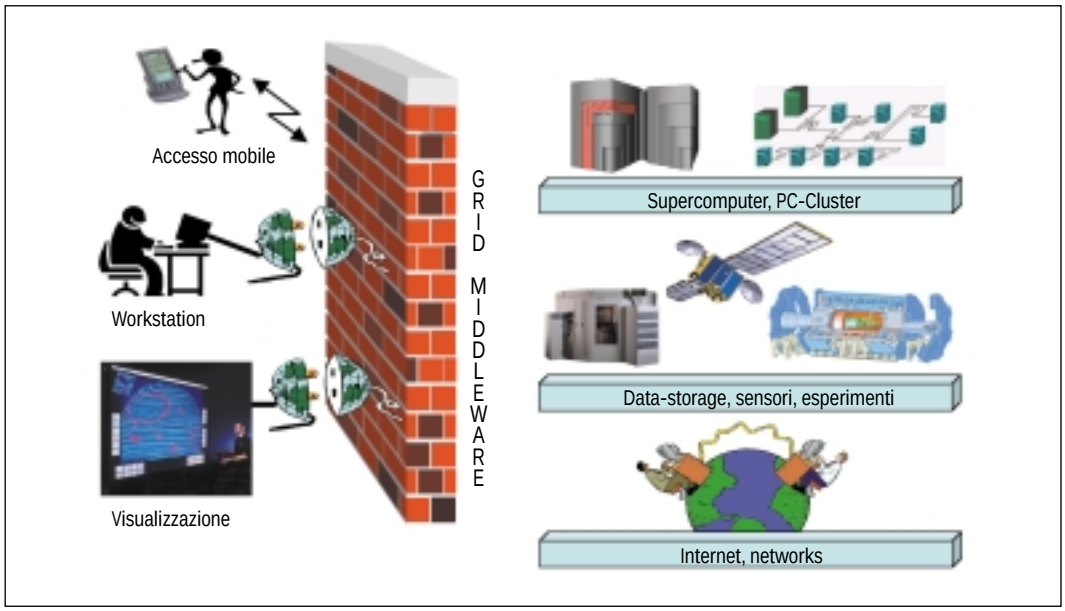


FIGURA 2
La metafora Grid

dando il via all'iniziativa sull'e-Science ad associarsi con il CERN per promuovere la costituzione del primo grande progetto Grid europeo, DataGrid (http://www.eu-datagrid.org), che parte nel 2001. DataGrid, finanziato dall'Unione Europea con 10 Milioni di Euro raccoglie una ventina di partners di molti paesi europei e comprende soggetti di molte discipline scientifiche (quali la Fisica delle Alte Energie, la Bio-medicina e le Osservazioni Terrestri) e dell'industria.
Gli obiettivi principali di DataGrid includono:

- Lo sviluppo di nuove componenti di middleware per l'accesso a dati disponibili in domini di gestione diversi e distribuiti a livello geografico;
- l'ottimizzazione della gestione dei carichi di lavoro su risorse computazionali distribuite a livello geografico;
- la gestione delle sicurezze e delle Organizzazioni Virtuali;
- la realizzazione di un primo "testbed" Europeo e internazionale che permetta l'inizio di effettive attività utili per la comunità scientifica.

In DataGrid l'INFN collabora fin da subito con la Datamat SpA per lo sviluppo del middleware e con NICE per la realizzazione del portale GENIUS (Grid Enabled web environment for site Independent User job Submission). GENIUS https://genius.ct.infn.it. permette all'utente, con semplici click, di accedere alla grid in modo sicuro, di eseguire le proprie applicazioni, e di accedere in modo trasparente ai dati della comunità di cui fa parte.
Sempre con il CERN e altri partners europei l'INFN avvia nel 2001 il progetto DataTAG (http: // www.cern.ch/datatag) che affronta il problema dell'interoperabilità con le Grid in sviluppo negli USA e nei paesi dell'area Asia-Pacifico e stabilisce uno stretto legame di collaborazione con i principali progetti Grid USA, come Globus e Condor, per lo sviluppo d'interfacce comuni e di standard internazionali anche all'interno della nuova organizzazione mondiale che viene allora a crearsi per questo scopo, il Global Grid Forum.

3.2. La fase di consolidamento (2001-2003)

Nei due anni seguenti, in Europa, un numero crescente d'attività di ricerca e sviluppo sulle Grid è finanziato da quasi tutti i Paesi e dalla Comunità che già nel Quinto Programma Quadro (2001-2003) approva una ventina di progetti per un totale di ~45 M€ di finanziamenti. Di questi l'Italia ottiene ~10% a riprova del suo alto livello di competitività in questo campo.
Negli Stati Uniti la National Science Foundation (NSF) e il Department Of Energy (DOE) finanziano in questa prima fase progetti per ~100 M$ tra cui spicca TeraGrid che ha come obiettivo la costruzione di un'infrastruttura nazionale di supercalcolo e che ottiene inizialmente ~50M €.
Progetti nazionali con finanziamenti rilevanti (vari M$) partono anche in Giappone, Corea, Taiwan e Cina.
In Italia a INFN Grid si affiancano altri progetti nazionali, grazie ai fondi governativi FIRB e PON, come il progetto nazionale Grid.it da circa 11M€ che coinvolge molte Istituzioni di ricerca e università, il progetto di Grid per la finanza EGRID, il progetto di Grid per il supercalcolo al sud S-PACI, il progetto di Grid Inter-dipartimentale a Napoli e altri progetti minori. Il finanziamento complessivo raggiunge ~60 M€ che, in Europa, è inferiore solo al programma UK di e-Science. Tutti i maggiori Enti di ricerca quali INFN, CNR, INAF, INGV, ASI, ENEA... e molte università sono progressivamente coinvolti nelle attività su Grid.

3.3. La maturità (2003-2006)

Nel successivo Sesto Programma Quadro (FP6) della Comunità Europea le Grid ottengono un posto di primo piano con un finanziamento complessivo di 225 M€ che si aggiunge ai ~100 M€ destinati allo sviluppo della rete europea per la ricerca Geant.
Ottiene il via libera il nuovo progetto Europeo EGEE con un finanziamento di 32 M€ per due anni (2004-2005), rinnovabile per il biennio successivo. Il progetto parte il 1 aprile 2004, ha durata di 2 anni, e realizzerà la prima Grid europea per la scienza, aperta all'industria, al commercio e alla società. EGEE è l'acronimo di *Enabling Grids for E-sciencE* e può essere considerato il successore di DataGrid e DataTAG.
La costruzione della prima Grid Europea di produzione da parte di EGEE sarà un'impre-

sa storica, coordinata dal CERN di Ginevra, a cui parteciperanno più di 70 Enti e Istituzioni scientifiche appartenenti a 26 Paesi Europei, organizzati in 9 grandi Federazioni, che forniranno le risorse di calcolo e storage, le applicazioni, i servizi operativi e di gestione necessari per far funzionare quest'enorme infrastruttura che non ha eguali nel mondo. Un sistema di "accounting" terrà conto dell'uso delle risorse mentre un robusto e sicuro sistema d'autenticazione e autorizzazione garantirà ad un numero sempre più vasto d'utenti scientifici, dell'ordine di decine di migliaia, appartenenti a varie organizzazioni e comunità scientifiche, la sicurezza e la riservatezza necessaria allo svolgimento del proprio lavoro.

EGEE svilupperà anche un middleware Grid Open Source più robusto ed affidabile, costruito con stretti criteri d'ingegneria del software e in grado di durare nel tempo. Questo sostituirà gradualmente quello esistente e farà passare definitivamente l'Europa dalla fase di "sperimentazione" a quella di "produzione". Si baserà sui nuovi standard come WSRF (*Web Service Resource Framework*) per la costruzione di WEB e Grid services, definiti a livello mondiale da W3C e OASIS e dal *Global Grid Forum* (GGF), organizzati in una logica di *Open Grid Services Architecture* (OGSA). Collaboreranno, come nel passato, con EGEE i gruppi americani di Globus (Università di Chicago, Argonne National Laboratory e ISI California) e di Condor (Università di Wisconsin).

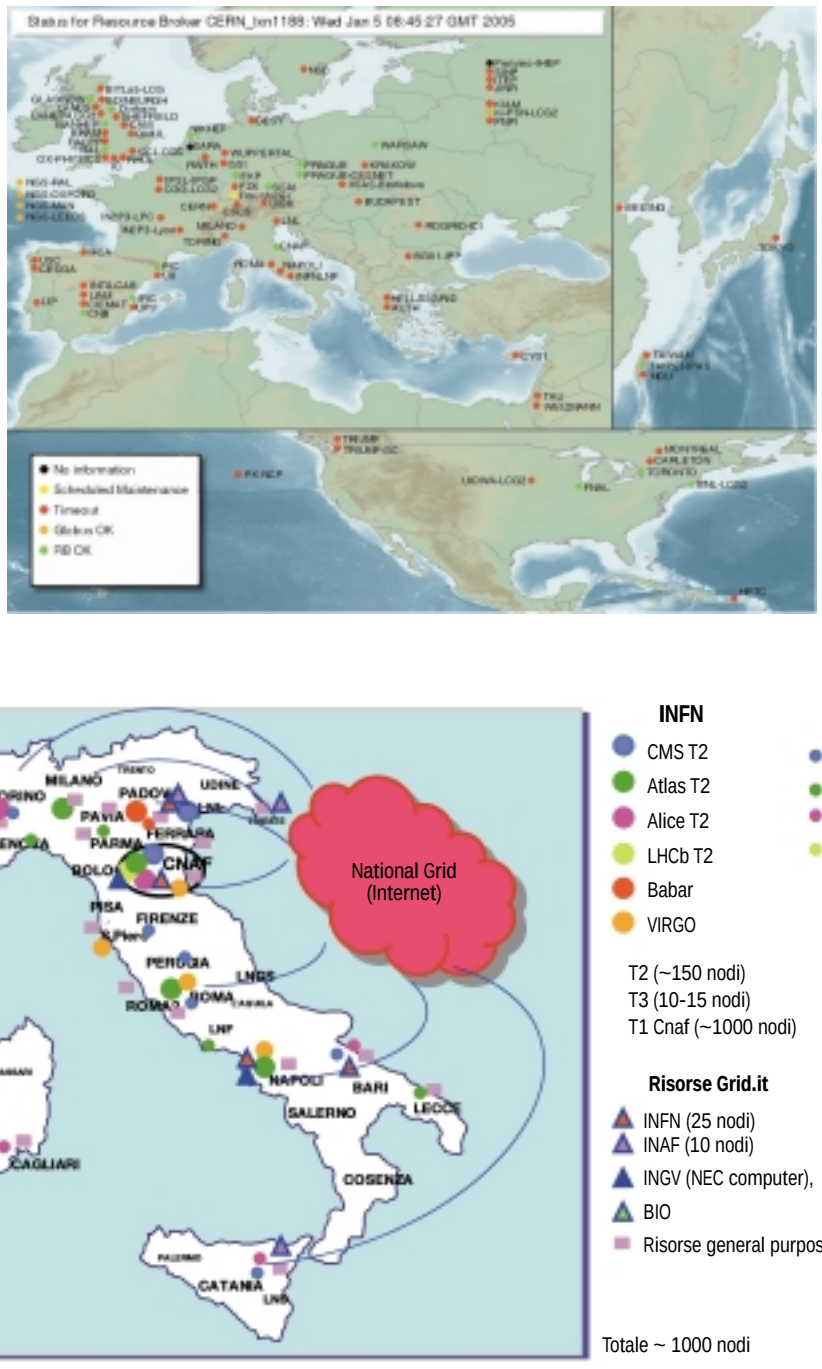
EGEE rappresenta una grande sfida vinta dalla comunità scientifica europea chiamata ad organizzarsi in tempi brevi in un grande progetto pionieristico di dimensione competitiva a livello mondiale. EGEE integrerà tutte le esistenti infrastrutture grid nazionali con le loro strutture tecniche e operative in una grande e-Infrastruttura (Internet e Grid) di scala europea. EGEE si collegherà alla Cyber-Infrastruttura americana proposta dalla National Science Foundation e alle Grid asiatiche in costruzione in Cina e Giappone. È un passo decisivo verso la costruzione di quella grid mondiale, o più probabilmente nel medio periodo, di quella federazione di grids richiesta dalla necessità delle moderne società di mettere la conoscenza alla base d'ogni nuovo sviluppo. Si tratta di una svolta epocale dal punto di vista scientifico e tecnologico, poiché le Grid di produzione cambieranno il modo di fare ricerca sia per gli enti pubblici sia per le aziende private.

L'Italia partecipa a tutte le aree d'attività di EGEE con un finanziamento complessivo di 4.7 M€ sul totale dei 32M€ del progetto, il più alto dopo quello del CERN, coordinatore del progetto, e con un ruolo d'estremo rilievo. L'INFN coordina la federazione italiana a cui partecipano le tre Università del consorzio S-PACI (*Southern Partnership for Advanced Computational Infrastructures*) Calabria, Lecce, Milano e Napoli, l'ENEA e le industrie Datamat SpA e NICE.

In FP6 l'Italia ottiene la leadership di nuovi progetti come Diligent (6M€), mirato allo sviluppo di Librerie Digitali distribuite su un'e-infrastruttura Grid come quella di EGEE e coordinato dall'ISTI CNR di Pisa che esprime un livello d'eccellenza mondiale in questo campo, Gridcc (4 M€), coordinato dall'INFN, che vuole estendere le funzionalità del middleware in modo da soddisfare le esigenze delle attività real time, come il controllo remoto d'apparati, e GridCoord, coordinato dall'Università di Pisa che si propone come attività di coordinamento dei programmi nazionali di ricerca sulle Grid. L'Italia, con il CINECA, ha una partecipazione di primo piano in DEISA, il progetto Europeo per una Grid dei Centri di supercalcolo europei. Molte università ed enti di ricerca ottengono ruoli di primo piano in altri progetti come Core Grid, che ha come obiettivo la ricerca sulle grid di prossima generazione ed e-Legi, un progetto di Grid per l'e-Learning. Complessivamente in FP6 l'Italia riceve finanziamenti pari a ~10% di quelli finora erogati (125 M€).

Ancora una volta l'Italia dunque interpreta un ruolo d'eccellenza in questo settore grazie al ruolo pionieristico svolto dall'INFN e da altri enti, come le Università di Lecce e della Calabria e il CNR, nella sperimentazione e nello sviluppo delle GRID in Europa e nel mondo.

Negli Stati Uniti NSF in parallelo a FP6 propone un vasto programma da ~ 1000 M$ in 5 anni per la creazione di una Cyber-infrastructure nazionale che è ora in corso d'approva-



Infrastruttura INFN-Grid

FIGURA 3

Grid map del progetto EGEE-LHC e della infrastuttura italiana

zione da parte del Congresso e il Giappone approva il programma di Grid nazionale da 130 M$ e il progetto Naregi.
Il programma UK e-Science è portato nel 2004 a 250 M£.
Grazie a questi progetti e a quelli ormai terminati, infrastrutture grid d'interesse generale per svariate discipline scientifiche cominciano a divenire operanti in Italia, in Europa e in USA con funzionalità costantemente incrementate dalla serie di progetti sopraccitati: INFN-Grid, UK e-Science, EGEE, Globus, Condor (Figura 3).
In particolare in Italia i massicci investimenti

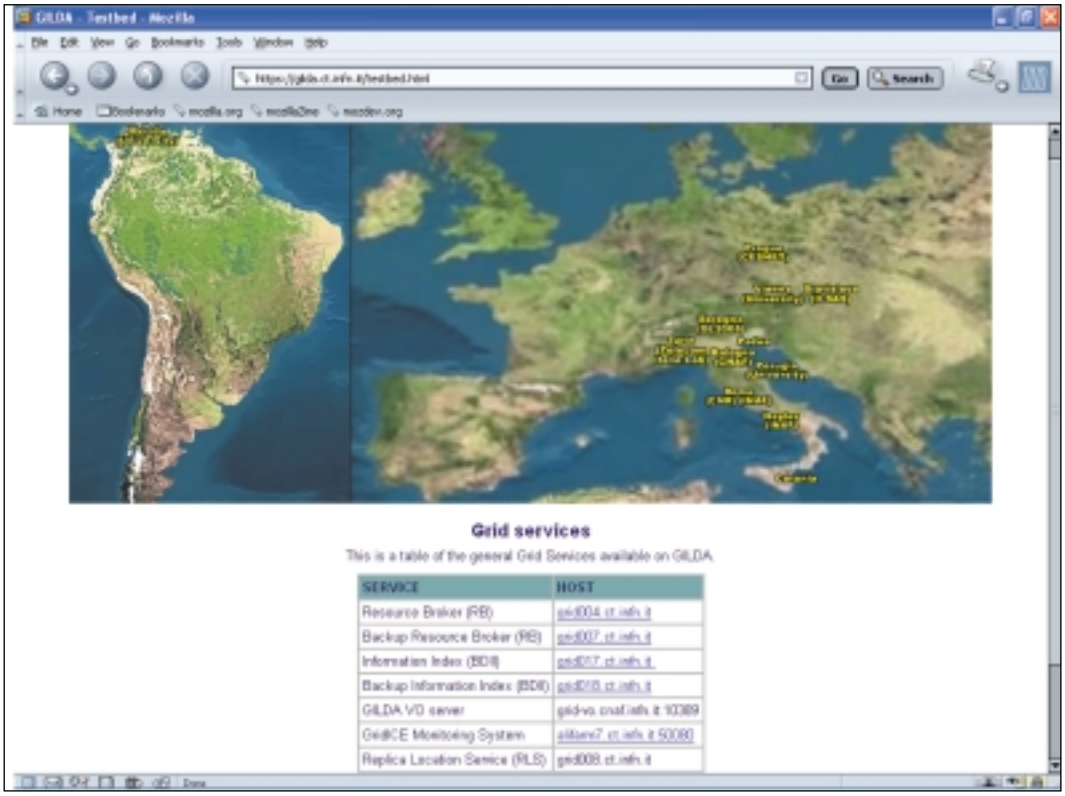


FIGURA 4
Testbed di GILDA

nel progetto INFN grid e nell'infrastruttura di Calcolo per LHC, fatti in anticipo rispetto al resto degli altri Paesi Europei, unitamente ai finanziamenti nazionali MIUR sui fondi FIRB ed europei nell'ambito del 5° e 6° Programma Quadro, stanno rendendo possibile la creazione di un'infrastruttura nazionale e-Science condivisa da molti settori di ricerca come la Fisica, l'Astrofisica, la Geofisica, la Biologia, la Chimica Computazionale, l'Osservazione della Terra e che si pone all'avanguardia nel mondo. Numerosi sono i progressi effettivamente realizzati per la diffusione di questa tecnologia in Italia. Sono stati completamente automatizzati gli strumenti per l'installazione del midleware e lo sviluppo di quelli per il controllo e il management operativo procede a ritmi incessanti. Gli utenti Grid possono già, di fatto, installare e aggiornare il loro middleware con un semplice click, e dispongono del portale Genius che consente l'uso trasparente dei servizi della grid. Gli utenti possono provare direttamente questa funzionalità tramite il Testbed (piattaforma di prova) GILDA che l'INFN ha messo a punto proprio per questo, ed è utilizzato da EGEE per le attività di divulgazione (http://grid-it.cnaf.infn.it/), (Figura 4). Alcuni esempi delle applicazioni sono mostrati nelle figure 5 A - D.

4. L'E-INFRASTRUCTURE REFLECTION GROUP (E-IRG)

La rilevanza italiana nel settore è stata ulteriormente enfatizzata in occasione del workshop internazionale organizzato a Roma sotto l'egida del MIUR e della Presidenza Italiana della Commissione EU "*e-Infrastructures (Internet and Grids): The new foundation for knowledge-based Societies*" durante il quale si è iniziato a discutere sulla necessità della costituzione di un quadro politico-amministrativo di riferimento, capace di contemplare meccanismi e regole in grado di abbattere le barriere nazionali relative alla realizzazione e all'uso delle e-Infrastrutture in Europa e nel resto del mondo. Durante il workshop si è costituito l'e-Infrastructure Reflection Group" (e-IRG), con delegati nominati dai Ministri della Scienza o della Ricerca di tutti i Paesi europei.
L' e-IRG si prefigge di armonizzare le singole iniziative nazionali suggerendo ai diversi governi locali le linee-guida politiche e possibili regole da adottare in merito alle pro-

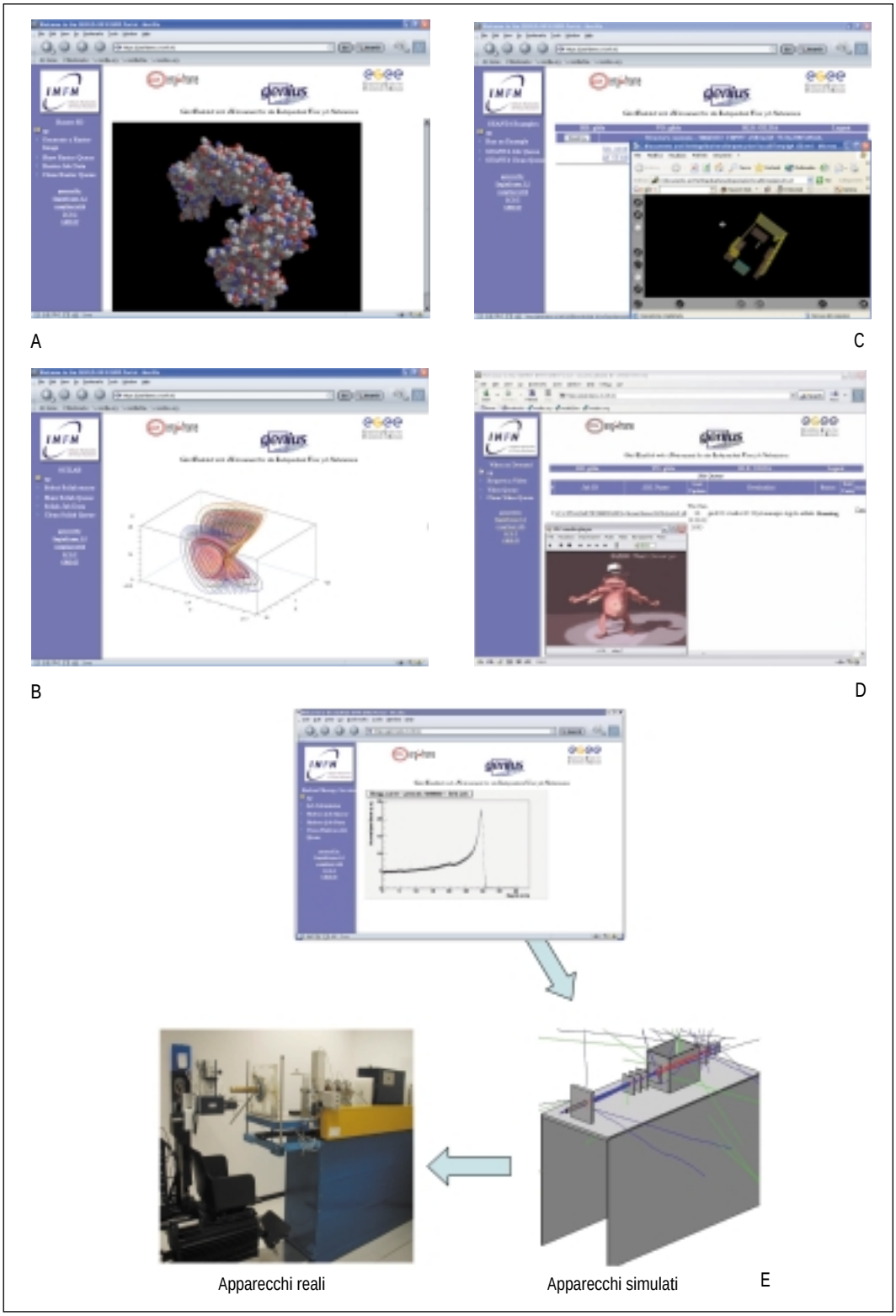


FIGURA 5

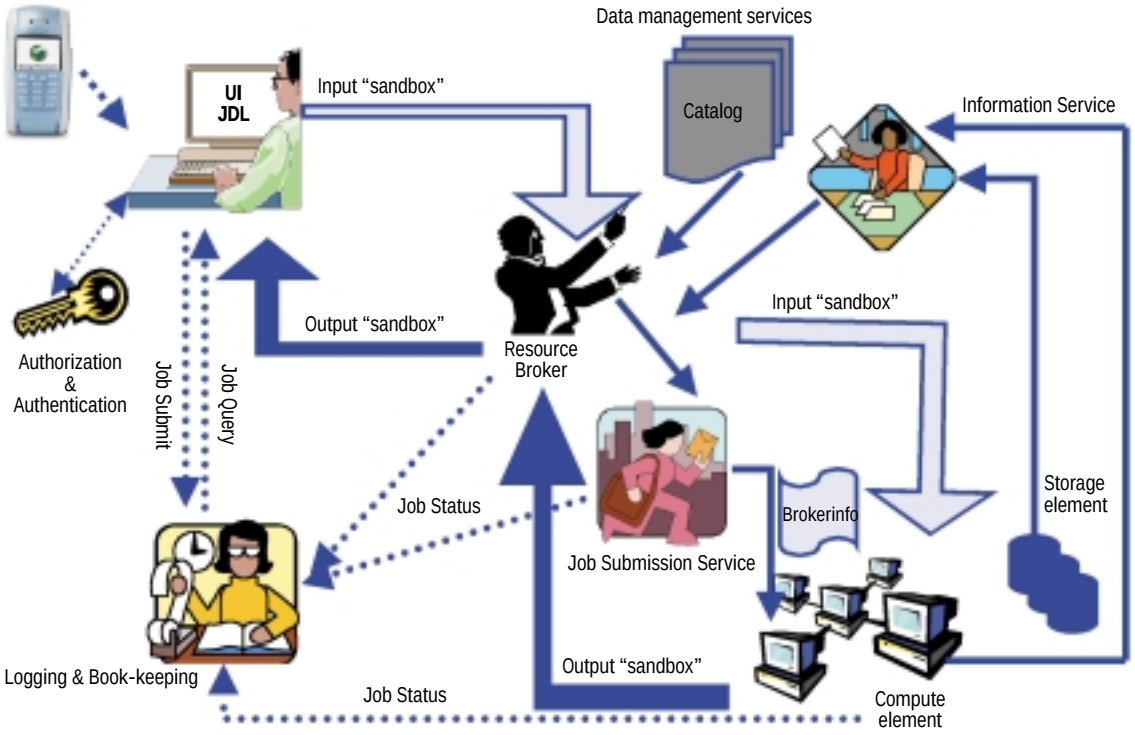
Esempi di applicazioni utilizzabili in GILDA: A presentazione tridimensionale di atomi e/o molecole; B struttura per computazioni numeriche per applicazioni scientifiche ed ingegneristiche; C simulazione di possibile evento in fisica delle alte energie; D video su richiesta fornito e implementato dalla Grid; E simulazione di fascio di protoni utilizzato per scopi medico-terapeutici, in particolare per la cura del tumore dell'occhio

blematiche sulla sicurezza delle e-Infrastrutture sia europee che di quelle condivise internazionalmente, all'interoperabilità su scala mondiale, oltre a quelle relative ai sistemi d'Autorizzazione, Accounting, Cost sharing e Business model che dovrebbero avere una base comune per poter permettere la diffusione della tecnologia.
Queste sono pagine ancora tutte da scrivere e l'Italia durante l'evento romano ha gettato le basi per l'avvio di una discussione consapevole e coordinata su questi argomenti che è poi continuata sotto la Presidenza Irlandese e Olandese.

5. LA SFIDA ATTUALE: DALLA FASE DI R&D ALLO SFRUTTAMENTO

5.1. Introduzione

Le attività di ricerca e sviluppo di questi anni hanno portato ad una vasta produzione di componenti di middleware in gran parte disponibili come software Open Source che permettono di certificare ed autorizzare gli utenti, elaborare dati digitali distribuiti, sostenere estesi processi di modellizzazione e condividere in modo trasparente risorse computazionali distribuite (Figura 6). Molti di



Scenario tipico del ciclo di un job:
L'utente in generale dispone di un certificato, ottenuto dall'Autorità di certificazione della sua Organizzazione, che lo identifica univocamente. Per l'INFN vedi http://grid-it.cnaf.infn.it/

1. L'utente sottomette alla grid un job tramite lo User Interface (UI), che è configurato per contattare l'appropriato Resource Broker (RB) incaricato di trovare le risorse con le caratteristiche richieste e archiviare la richiesta al Logging e Book-keeping (LB). A questo punto il Job si può considerare in "stato di sottomissione".
2. Mentre il job è stato accettato dal RB, si dice che è in "stato di attesa".
3. Quando un Computing Element (CE) che soddisfi le richieste specifiche viene trovato dal RB, il job viene trasferito al Job Submission System (JSS) e diventa "pronto".
4. A questo punto JSS sottomette al CE il job che passa quindi allo stato di "SCHEDULATO".
5. Dallo stato di schedulato tipicamente il job passa prima a quello di "running".
6. In seguito a quello "dell'Eseguito" (se l'esecuzione del job può essere conclusa).
7. Successivamente, passa
   - *a.* allo stato di "esecuzione terminata del job" (ovvero DONE+RB sono in grado di recuperare l'output del "sandbox") oppure
   - *b.* a quello di "aborted" (se l'esecuzione del job è conclusa).
8. L'utente può in ogni istante conoscere lo stato del suo job semplicemente inviando un comando al RB.

FIGURA 6
Esempio di sottomissione di un job

questi servizi, grazie al loro utilizzo in varie infrastrutture del mondo della ricerca, Datagrid, LHC Computing Grid, EGEE, cominciano a possedere livelli di robustezza e qualità da poter fornire una soluzione operativa per altri settori della società.
Obiettivi primari della fase attuale per l'Italia e l'Europa sono:

- Costruire un framework (Piattaforma Tecnologica a livello EU) per il coordinamento delle attività su Grid svolte a livello nazionale, europeo ed internazionale.
- Pianificare efficientemente il passaggio da una fase dominata dalla Ricerca e Sviluppo ad una dominata dallo sfruttamento innovativo della tecnologia Grid in nuove e-Infrastrutture e a livello industriale. Questo richiede di:
- Consolidare le componenti di middleware, sviluppate finora in modo non coordinato da progetti di R&D indipendenti, in una piattaforma di servizi Grid coerenti e interoperabili sempre più aderenti a Standard Internazionali e resi disponibili come sofware Open Source.
- Avviare una serie di progetti pilota per lo sfruttamento di questa piattaforma nella Ricerca, nell'Industria e nei Servizi.

Le Grid possono avere un impatto dirompente sull'evoluzione tecnologica futura del mondo della ricerca, di quello dell'industria, dell'ICT in particolare e dei servizi e possono essere in grado di sostenere lo sviluppo di nuovi settori produttivi, com'è stato per il WEB nel passato.

5.2. Lo sfruttamento della piattaforma Grid in Italia

In una riunione convocata recentemente a Roma presso la sede della Presidenza, l'INFN ha proposto che l'Italia si doti di un'organizzazione in grado di coinvolgere le maggiori Istituzioni di ricerca attive nel campo, le Industrie e i Servizi nel promuovere e sostenere lo sfruttamento puntuale di quanto finora prodotto, fornendo al tempo stesso, continuità, solidità e fondamento alle future evoluzioni di questa piattaforma e agli interventi a livello internazionale. Il Consorzio per l'Open Middleware Enabling Grid Applications (c-OMEGA) permetterà all'Italia di conservare il suo attuale livello di eccellenza internazionale rispetto alle recentissime iniziative adottate dagli altri due Paesi con cui si confronta in questo campo:

- L'*Open Middleware Initiave* (OMII) in UK http://www.omii.ac.uk/
- La *New Middleware Initiative* (NMI) in US http://www.nsf-middleware.org/

I principali obiettivi del consorzio c-OMEGA sono:

- Diventare un punto di riferimento nazionale per la creazione, lo sviluppo, il supporto e la diffusione della piattaforma tecnologica Grid in Italia e in Europa, lavorando anche in stretto coordinamento con gli USA e i paesi dell'Asia-Pacifico.
- Sfruttare creativamente le componenti di middleware e gli ambienti Grid sviluppati da progetti di R&D indipendenti e in generale i prodotti disponibili come software Open Source, per costruire in Italia delle releases di servizi coerenti e interoperanti basati sugli Standard emergenti dagli Organismi Internazionali per Grid e Service-oriented architectures. (Per esempio specifiche OGSA del Global Grid Forum, WSRF di OASIS, security di W3C ecc.), compatibilmente con le modalità e le tipologie di licenze open source.
- Far coesistere la missione e gli obiettivi del mondo della ricerca e accademico, quelli del mondo industriale, in particolare del settore ICT e dei grandi servizi pubblici nazionali (Ospedali, Scuole, Amministrazioni pubbliche).
- Estendere a livello di tutto il paese, con attività d'informazione, formazione e progetti mirati, lo sfruttamento delle tecnologie Grid in modo da far nascere nuove opportunità di crescita e di occupazione aumentando nello stesso tempo la competitività globale del Paese.

Hanno finora dato la propria adesione e sono attivamente coinvolte nell'iniziativa per la costituzione del consorzio OMEGA, oltre alle Industrie IT Datamat SpA e Nice srl che dal 2000 collaborano con l'INFN nello sviluppo delle Grid, i rappresentanti delle maggiori Istituzioni pubbliche di Ricerca: INFN, CNR, INAF, ICTP, CHEMGRID, il Centro Europeo per Studi Teorici in Fisica Nucleare e Aree Collegate (ECT), l'Istituto Trentino di Cultura (ITC-irst), l'Università di Messina e altre Università italiane; vari consorzi IT: S-PACI, CRMPA Consorzio Pisa Ricerche; grandi Industrie come Elasis SpA, società del

gruppo FIAT, Engineering Ingegneria Informatica SpA e grandi banche nazionali oltre a numerose PMI: euriX, Create-Net, Avanade –Italy srl, Synapsis srl e l'Associazione Italiana per La Telemedicina e Informatica Medica (@TIM). L'IBM sta fornendo un notevole contributo di idee ed esperienze già avviate all'estero.

Riferimenti Bibliografici

Foster I., Kesselman C., Kaufmann Morgan: T*he Grid 2: Blueprint for a New Computing Infrastructure*, *2nd Edition*. 2003.

Foster I., Kesselman C., Nick J., Tuecke S.: *Grid Services for Distributed System Integration*. Computer, 2002.

Foster I., Kesselman C., Tuecke S.: The Anatomy of the Grid: Enabling Scalable Virtual Organizations. *International J. Supercomputer Applications*, 2001.

Siti Web

BIGEST, http://www.pd.infn.it/bigest/

CONDOR, http://www.cs.wisc.edu/condor/

COREGRID, http://grid.infn.it/index.php?coregrid

DATAGRID, http://www.eu-datagrid.org

DATAMAT, http://www.datamat.it

DATATAG, http://datatag.web.cern.ch/

EGEE, http://www.cern.ch/egee

FIRB GRID-IT, http://www.grid.it/

GARR, http://www.garr.it/

GEANT, http://www.geant.net/server.php?show=nav.007

GENIUS, https://genius.ct.infn.it

GILDA, https://gilda.ct.infn.it/

GLOBUS, http://www.globus.org/

GRIPHYN, http://www.griphyn.org/index.php

INFN, http://www.infn.it

INFN-GRID, http://grid.infn.it

LCG, http://lcg.web.cern.ch/LCG/

NICE, http://www.nice-italy.com

NMI, http://www.nsf-middleware.org/

OMII, http://www.omii.ac.uk/

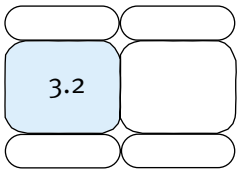
PPDG, http://www.ppdg.net/

MIRCO MAZZUCATO è dirigente di ricerca dell'INFN, direttore del Centro INFN del CNAF di Bologna e del progetto nazionale INFN Grid. È da anni impegnato nello studio per la creazione, lo sviluppo e l'applicazione di nuove tecnologie computazionali. Ha diretto e coordinato commissioni nazionali ed internazionali di progetti innovativi e pionieristici, tra cui l'iniziativa per lo sviluppo della tecnologia Grid che ha promosso a livello nazionale ed Europeo fin dal 1999. È delegato italiano del MIUR presso il Comitato Europeo IST per PQ6 nel cui ambito è membro del Comitato esecutivo del progetto EGEE, per l'implementazione di una comune infrastruttura grid europea.
È stato promotore e Chairman, del Grid Deployment Board del progetto internazionale LHC Computing Grid del CERN. Attualmente coordina una iniziativa per la realizzazione in Italia del Consorzio per l'Open Middleware Enabling Grid Applications (c-OMEGA) con l'obbiettivo di rilasciare, certificare e supportare un insieme di servizi grid Open Source, aderenti a Standard Internazionali, su cui costruire soluzioni verticali per applicazioni specifiche e un progetto per costituire in Italia ed in Europa una Piattaforma Tecnologica Grid.
mirco.mazzucato@pd.infn.it

ALGORITMI EVOLUTIVI: CONCETTI E APPLICAZIONI

Andrea G. B. Tettamanzi

Gli algoritmi evolutivi sono una famiglia di tecniche stocastiche per la risoluzione di problemi che fa parte della più ampia categoria dei "modelli a metafora naturale". Essi trovano la loro ispirazione in biologia e, in particolare, si basano sull'imitazione dei meccanismi della cosiddetta "evoluzione naturale". Nel corso degli ultimi 25 anni queste tecniche sono state applicate ad un elevato numero di problemi di grande rilevanza pratica ed economica. L'articolo presenta una rassegna di queste tecnologie e alcuni esempi di applicazione.



1. CHE COSA SONO GLI ALGORITMI EVOLUTIVI?

Guardando agli esseri viventi, tra cui l'Uomo e ad alcuni dettagli dei loro organi, alla loro complessità e perfezione, viene quasi da chiedersi come sia possibile che soluzioni tanto sofisticate possano essersi evolute autonomamente. Eppure esiste una teoria, proposta inizialmente da Charles Darwin e perfezionata in seguito da numerosi altri naturalisti, biologi e genetisti, che è in grado di spiegare in modo soddisfacente la maggior parte di tali fenomeni biologici, partendo dallo studio dei meccanismi di adattamento delle specie ad ambienti mutevoli e complessi. Questa teoria è supportata da importanti evidenze e non è stata ancora falsificata da alcun dato sperimentale. Secondo la teoria darwiniana, tali prodotti mirabili altro non sarebbero che il risultato di un processo evolutivo che procede senza uno scopo, guidato da una parte da una componente casuale e, dall'altra, dalla legge della sopravvivenza del più adatto: l'evoluzione naturale.
Se un processo simile è stato capace di produrre degli artefatti tanto sofisticati come l'occhio, il sistema immunitario e il nostro stesso cervello, pare soltanto logico cercare di imitarlo, simulandolo con gli elaboratori elettronici di cui disponiamo, al fine di risolvere i problemi più complicati che la realtà ci pone. È questa l'idea che sta alla base dello sviluppo degli algoritmi evolutivi.

1.1. La metafora di fondo

Gli algoritmi evolutivi sono dunque delle tecniche informatiche ispirate dalla biologia che si basano su una metafora, illustrata schematicamente nella tabella 1: come un individuo di una popolazione di organismi deve essere adattato all'ambiente che lo circonda per sopravvivere e riprodursi, così una possibile soluzione deve essere adatta a risolvere il suo problema. Il problema è l'ambiente in cui una soluzione vive, all'interno di una popolazione di altre possibili soluzioni; le soluzioni differiscono tra loro per qualità, cioè per costo o merito, che si riflettono nella valutazione della funzione obiettivo, così come gli individui di una popolazione di organismi differiscono

TABELLA 1

Illustrazione schematica della metafora su cui si basano gli algoritmi evolutivi

Evoluzione	Problem Solving
Ambiente	Problema da risolvere
Individuo	Possibile soluzione
Adattamento	Qualità della soluzione

Un po' di storia

L'idea di usare selezione e mutazione casuale per un compito di ottimizzazione risale almeno agli anni cinquanta, con il lavoro dello statistico George E. P. Box, noto per la massima "tutti i modelli sono sbagliati, ma alcuni sono utili", che tuttavia non fece uso dell'elaboratore elettronico. Box giunse a formulare una metodologia statistica che sarebbe divenuta di largo uso nell'industria e che egli battezzò *evolutionary operation* [1]. Più o meno negli stessi anni, altri studiosi concepirono l'idea di simulare l'evoluzione sull'elaboratore elettronico: Barricelli e Fraser utilizzarono simulazioni al calcolatore per studiare i meccanismi dell'evoluzione naturale, mentre al biomatematico Hans J. Bremermann va dato il credito di avere per primo riconosciuto nell'evoluzione biologica un processo di ottimizzazione [2].

Come spesso accade per molte idee pionieristiche, questi primi sforzi incontrarono uno scetticismo considerevole. Ciononostante, i tempi evidentemente erano maturi perché queste idee, che ormai erano nell'aria, venissero sviluppate. Probabilmente un fattore determinante per cui ciò avvenne fu l'aumento, oltre una certa soglia critica, della potenza computazionale degli elaboratori elettronici, allora disponibili nelle migliori università, che rese finalmente possibile la messa in pratica del calcolo evoluzionistico. Gli algoritmi evolutivi, in quelle che oggi riconosciamo come le loro varianti originarie, furono inventati indipendentemente e praticamente allo stesso tempo, a metà degli anni Sessanta, nel seno di tre distinti gruppi di ricerca: in America, Lawrence Fogel e colleghi, dell'Università di California a San Diego, posero le basi della programmazione evolutiva (*evolutionary programming*) [3], mentre presso l'Università del Michigan ad Ann Arbor John Holland proponeva i primi algoritmi genetici (*genetic algorithms*) [4]; in Europa, invece, furono Ingo Rechenberg e colleghi, allora studenti presso il Politecnico di Berlino, a ideare quelle che battezzarono "strategie evolutive" (*Evolutionsstrategien*) [5]. Per i successivi 25 anni questi tre filoni si svilupparono essenzialmente ciascuno per conto suo, finché nel 1990 non venne messo in atto uno sforzo organizzato per farli convergere: la prima edizione del congresso PPSN (*Parallel Problem Solving from Nature*), che si tenne quell'anno a Dortmund. Da allora i ricercatori interessati al calcolo evoluzionistico (*evolutionary computation*) formano un'unica, anche se articolata, comunità scientifica.

tra di loro per grado di adattamento all'ambiente, chiamato dai biologi *fitness*. Se la selezione naturale permette a una popolazione di organismi di adattarsi all'ambiente che la circonda, sarà anche in grado, applicata a una popolazione di soluzioni a un problema, di far evolvere soluzioni sempre migliori ed eventualmente, con il tempo, ottime.

In base a questa metafora, il modello computazionale prende in prestito dalla biologia alcuni concetti e i relativi termini: ogni soluzione è codificata in uno o più *cromosomi*; i *geni* sono i pezzi della codifica responsabili di uno o più *tratti* di una soluzione; gli *alleli* sono le possibili configurazioni che un gene può assumere; lo scambio di materiale genetico tra due cromosomi si chiama *crossover*, mentre ci si riferisce alla perturbazione della codifica di una soluzione con il termine *mutazione*.

Sebbene il modello computazionale introduca delle semplificazioni drastiche rispetto al mondo naturale, gli algoritmi evolutivi si sono rivelati capaci di far emergere strutture sorprendentemente complesse e interessanti. Ogni individuo può essere la rappresentazione, secondo un'opportuna codifica, di una particolare soluzione di un problema, di una strategia per affrontare un gioco, di un piano, di un'immagine o addirittura di un semplice programma per calcolatore.

La **storia** e il **funzionamento** degli algoritmi genetici sono riassunti nei due riquadri.

1.2. Gli ingredienti di un algoritmo evolutivo

Fatta questa premessa di tipo concettuale, vediamo ora in che cosa consiste, praticamente, un algoritmo evolutivo.

Un algoritmo evolutivo è una tecnica stocastica di ottimizzazione che procede in modo iterativo, mantenendo una popolazione (che in questo contesto significa un multiinsieme, ovvero una collezione di elementi non necessariamente tutti distinti tra loro) di individui che rappresentano possibili soluzioni per il problema che deve essere risolto (il problema oggetto) e facendola evolvere mediante l'applicazione di un certo numero, di solito abbastanza ridotto, di operatori stocastici: *mutazione, ricombinazione e selezione*.

La mutazione può essere qualsiasi operatore che perturbi casualmente una soluzione; gli operatori di ricombinazione decompongono due o più individui distinti e quindi mescolano le loro parti costitutive per formare un certo numero di nuovi individui; la selezione crea delle repliche degli individui che rappresentano le soluzioni migliori all'interno della popolazione ad un tasso proporzionale alla loro *fitness*.

La popolazione iniziale può provenire da un campionamento casuale dello spazio delle soluzioni oppure da un nucleo di soluzioni iniziali trovate da semplici procedure di ricer-

Come funziona un algoritmo genetico

Proviamo ad osservare da vicino il funzionamento di un algoritmo genetico servendoci di un esempio. Supporremo di dover risolvere un problema, che chiameremo *maxuno*, il quale consiste nel cercare, tra tutte le stringhe binarie di lunghezza *l*, quella che contiene il numero massimo di "1". A prima vista questo potrebbe sembrare un problema banale, per il semplice motivo che conosciamo in anticipo la soluzione: la stringa di tutti "1". Tuttavia, se si immaginasse di dover compiere *l* scelte binarie per risolvere un problema e che la qualità della soluzione fosse proporzionale al numero di scelte corrette effettuate, ecco che avremmo un problema di difficoltà equivalente e per nulla facile. Il fatto di supporre che le scelte corrette corrispondano tutte a un "1" è solo un artificio per rendere l'esempio più facile da seguire. Definiamo dunque la *fitness* di una soluzione come il numero di "1" presenti nella sua codifica binaria, fissiamo $l = 10$, che è un numero abbastanza piccolo da essere gestibile, e proviamo ad applicare a questo problema l'algoritmo genetico.

Per prima cosa dobbiamo stabilire la dimensione della popolazione: una buona scelta, tanto per cominciare, potrebbe essere 6 individui. A questo punto, è necessario generare una popolazione iniziale: lo faremo lanciando 60 volte (6 individui per 10 cifre binarie) una monetina non truccata e scrivendo 0 se esce "testa" e 1 se esce "croce". La popolazione iniziale così ottenuta è quella della tabella A. Notiamo che la media della *fitness* nella popolazione iniziale è di 5,67.

Numero	Individuo	Fitness
1	1111010101	7
2	0111000101	5
3	1110110101	7
4	0100010011	4
5	1110111101	8
6	0100110000	3

TABELLA A *La popolazione iniziale dell'algoritmo genetico per risolvere il problema maxuno, con la fitness corrispondente a ciascun individuo*

Il ciclo evolutivo ora può cominciare: per applicare la selezione proporzionale alla *fitness* il metodo più semplice consiste nel simulare il lancio di una pallina in una *roulette* speciale, con tanti settori quanti sono gli individui della popolazione (in questo caso 6) , ciascuno avente un'ampiezza che sta alla circonferenza quanto la *fitness* dell'individuo corrispondente sta alla somma delle *fitness* di tutta la popolazione (in questo caso 34). Perciò, quando lanceremo la pallina, questa avrà una probabilità di 7/34 di fermarsi nel settore dell'individuo 1, 5/34 di fermarsi nel settore dell'individuo 2, e così via. Dovremo compiere esattamente 6 lanci per formare una popolazione intermedia di 6 stringhe per la riproduzione. Supponiamo che i lanci diano il seguente esito: 1, 3, 5, 2, 4 e ancora 5. Significa che verranno usate per la riproduzione due copie dell'individuo 5 e una copia degli altri individui ad eccezione dell'individuo 6, che non lascerà discendenti. Il successivo operatore ad essere applicato è la ricombinazione: si formano le coppie, il primo estratto con il secondo, il terzo con il quarto, e così via; per ciascuna delle coppie, decidiamo con una certa probabilità, diciamo 0,6, se effettuare il *crossover*. Supponiamo che il *crossover* venga effettuato solo sulla prima e sull'ultima coppia, con punti di taglio scelti a casi rispettivamente dopo la seconda cifra e dopo la quinta.

Per la prima coppia, avremo:

11.11010101 che diventa 11.10110101
11.10110101 " 11.11010101

Notiamo che, siccome le parti a sinistra del punto di taglio sono identiche, il *crossover* non ha alcun effetto. L'eventualità è più comune di quanto si possa immaginare, specialmente quando, avanti con le generazioni, la popolazione sia piena di individui tutti buoni e quasi identici tra di loro.

Invece, per la terza coppia, avremo:

01000.10011 che diventa 01000.11101
11101.11101 " 11101.10011

Non resta che applicare la mutazione alle sei stringhe risultanti dalla ricombinazione, decidendo per esempio con probabilità di 1/10 per ogni cifra binaria se invertirla. Avendo in tutto 60 cifre binarie, ci aspetteremo in media 6 mutazioni distribuite casualmente in tutta la popolazione. La nuova popolazione, dopo l'applicazione di tutti gli operatori genetici, potrebbe essere quella mostrata nella tabella B, dove le cifre binarie mutate sono state evidenziate in grassetto.

Numero	Individuo	Fitness
1	11101**0**0101	6
2	1111**1**1010**0**	7
3	11101**0**111**1**	8
4	0111000101	5
5	0100011101	5
6	11011000**0**1	6

TABELLA B *La popolazione dell'algoritmo genetico per risolvere il problema maxuno dopo una generazione, con la fitness corrispondente a ciascun individuo*

In una generazione, la *fitness* media della popolazione è passata da 5,67 a 6,17, con un incremento dell'8,8%. Iterando lo stesso processo più volte, si arriva ben presto a un punto in cui fa la sua comparsa un individuo di tutti "1", la soluzione ottima del problema.

ca locale, se disponibili, o determinate da un esperto umano.
Gli operatori stocastici, applicati e composti secondo le regole che definiscono il particolare algoritmo evolutivo, determinano un operatore stocastico di trasformazione di popolazioni, in base al quale è possibile modellare il funzionamento di un algoritmo evolutivo come una catena di Markov i cui stati sono le popolazioni. È possibile dimostrare che, se sono soddisfatte alcune ipotesi tutto sommato ragionevoli, tale processo stocastico converge in probabilità all'ottimo globale del problema [16].
Si parla spesso, con riferimento agli algoritmi evolutivi, di *parallelismo implicito*. Questo termine si riferisce al fatto che ciascun individuo può essere pensato come un rappresentante di una moltitudine di *schemi* di soluzione, cioè di soluzioni parzialmente specificate, di modo che, elaborando un singolo individuo, l'algoritmo evolutivo starebbe in realtà elaborando implicitamente allo stesso tempo (cioè in parallelo) tutti gli schemi di soluzione di cui quell'individuo è un rappresentante. Non bisogna confondere questo concetto con il *parallelismo inerente* degli algoritmi evolutivi, derivante dal fatto che essi conducono una ricerca basata su una popolazione, il quale fa sì che, sebbene il loro funzionamento sia esprimibile per comodità mediante una descrizione sequenziale, la loro realizzazione su hardware parallelo risulti particolarmente naturale e vantaggiosa.

1.3. Algoritmi genetici

Il modo migliore per capire come funzionano gli algoritmi evolutivi consiste nel considerare una delle loro versioni più semplici: gli algoritmi genetici [6]. Negli algoritmi genetici, le soluzioni sono rappresentate come stringhe binarie di lunghezza fissa. Questo tipo di rappresentazione è sicuramente il più generale, tuttavia, come vedremo più avanti, non sempre è anche il più conveniente: infatti, qualsiasi struttura dati, per quanto complessa e articolata, sarà sempre codificata in alfabeto binario nella memoria dell'elaboratore elettronico. Ora, una sequenza di due simboli, 0 e 1, da cui è possibile ricostruire una soluzione, ricorda moltissimo un filamento di DNA, costituito da una sequenza di quattro basi, A, C, G e T, da cui è possibile ricostruire un organismo vivente! In altre parole, possiamo considerare una stringa binaria come il DNA di una soluzione del problema oggetto.
Un algoritmo genetico è composto di due parti:
1. una procedura che genera (casualmente o utilizzando qualche euristica) la popolazione iniziale;
2. un ciclo evolutivo, che, ad ogni iterazione (o *generazione*), crea una nuova popolazione applicando gli operatori genetici alla popolazione precedente.

Il ciclo evolutivo degli algoritmi genetici può essere schematizzato mediante lo pseudocodice di tabella 2. A ciascun individuo viene assegnato un particolare valore di *fitness*, che dipende dalla qualità della soluzione che esso rappresenta. Il primo operatore ad essere applicato è la selezione, il cui scopo è simulare la legge darwiniana della sopravvivenza del più adatto. Nella versione originale degli algoritmi genetici, questa legge è implementata per mezzo della cosiddetta selezione proporzionale alla *fitness*: per creare una nuova popolazione intermedia di *n* individui "genitori", vengono effettuate *n* estrazioni indipendenti di un individuo dalla popolazione esistente, con probabilità per ogni individuo di essere estratto direttamente proporzionale alla sua *fitness*. Di conseguenza, gli individui al di sopra della media verranno in media estratti più volte, mentre quelli al di sotto della media andranno incontro all'estinzione.
Una volta estratti gli *n* genitori come descritto, gli individui della generazione successiva

```
generation = 0
Inizializza popolazione
while not <condizione di terminazione>
do
        generation = generation + 1
        Calcola fitness di ciascun individuo
        Selezione
        Crossover(p_cross)
        Mutazione(p_mut)
end while
```

TABELLA 2
Pseudocodice che illustra un tipico algoritmo genetico semplice

saranno prodotti mediante l'applicazione di un certo numero di operatori di riproduzione, i quali possono coinvolgere un solo genitore (simulando quindi una sorta di riproduzione asessuata), nel qual caso si parla di *mutazione*, o più di un genitore, normalmente due (riproduzione sessuata), nel qual caso si parla di *ricombinazione*. Negli algoritmi genetici sono utilizzati due operatori di riproduzione: crossover e mutazione.

Per applicare il *crossover*, gli individui genitori vengono accoppiati a due a due; quindi, con una certa probabilità p_{cross}, chiamata "tasso di *crossover*", che è un parametro dell'algoritmo, ciascuna coppia subisce il *crossover* vero e proprio, che consiste nell'allineare le due stringhe binarie, tagliarle in un punto estratto a caso, e scambiarne le metà destre, ottenendo così due nuovi individui, che ereditano parte dei loro caratteri da un genitore e parte dall'altro.

Dopo il *crossover*, tutti gli individui subiscono la mutazione, il cui scopo è quello di simulare l'effetto di errori casuali di trascrizione che possono avvenire con una probabilità molto bassa p_{mut} ogniqualvolta un cromosoma venga duplicato, e che consiste nel decidere di invertire ciascuna singola cifra binaria, indipendentemente dalle altre, con probabilità p_{mut}. In altre parole, ciascuno zero ha una probabilità p_{mut} di diventare un uno e viceversa.

Il ciclo evolutivo, per come è concepito, potrebbe andare avanti all'infinito. Nella pratica, però, bisogna decidere quando arrestarlo, in base a qualche criterio di terminazione specificato dall'utente. Alcuni esempi di criteri di terminazione possono essere:

- il passaggio di un numero prefissato di generazioni o di una certa quantità di tempo;
- il rinvenimento di una soluzione soddisfacente secondo qualche misura;
- la mancanza di miglioramenti per un certo numero prefissato di generazioni.

1.4. Strategie evolutive

Le strategie evolutive affrontano l'ottimizzazione di una funzione obiettivo reale di variabili reali in uno spazio a l dimensioni. La rappresentazione utilizzata per le variabili indipendenti della funzione (la soluzione) è quella più diretta, cioè un vettore di numeri reali. Oltre a codificare le variabili indipendenti, tuttavia, le strategie evolutive includono nell'individuo anche delle informazioni sulla distribuzione di probabilità da utilizzare per la loro perturbazione (operatore di mutazione): a seconda delle versioni, queste informazioni possono andare dalla semplice varianza, valida per tutte le variabili indipendenti, all'intera matrice di varianza e covarianza $\boldsymbol{C}$ di una distribuzione normale congiunta; in altre parole, la dimensione di un individuo può andare da $l + 1$ a $l\,(l + 1)$ numeri reali.

Nella sua forma più generale, l'operatore di mutazione perturba un individuo in due passi:

1. perturba la matrice $\boldsymbol{C}$ (o, meglio, una matrice equivalente di angoli di rotazione da cui la matrice $\boldsymbol{C}$ può essere agevolmente calcolata) con una distribuzione di probabilità identica per tutti gli individui;
2. perturba il vettore dei parametri che rappresenta la soluzione al problema di ottimizzazione con una probabilità normale congiunta con media $\mathbf{0}$ e matrice di varianza e covarianza la $\boldsymbol{C}$ perturbata.

Questo meccanismo di mutazione permette all'algoritmo di far evolvere autonomamente i parametri della sua strategia di ricerca mentre va in cerca della soluzione ottima: il processo che ne deriva, denominato *autoadattamento*, è uno degli aspetti più potenti e interessanti di questo tipo di algoritmo evolutivo.

La ricombinazione nelle strategie evolutive può assumere diverse forme: quelle utilizzate più di frequente sono la ricombinazione *discreta* e quella *intermedia*. Nella ricombinazione discreta, ciascuna componente dell'individuo figlio è presa da uno dei genitori a caso; nella ricombinazione intermedia, invece, ciascuna componente è ottenuta mediante combinazione lineare, con un parametro casuale, delle componenti corrispondenti dei genitori.

Esistono due schemi di selezione alternativi, che definiscono due classi di strategie evolutive: (n, m) e $(n + m)$. Nelle strategie (n, m), da una popolazione di n individui vengono prodotti $m > n$ individui figli e gli n migliori vanno a costituire la popolazione della generazione successiva. Nelle strategie $(n + m)$, invece, anche gli n individui genitori partecipano alla selezione e, di questi $n + m$ individui, soltanto i migliori n entrano a far parte della popolazione della generazione successiva. Si noti che,

in entrambi i casi, la selezione è deterministica e funziona "per troncamento", cioè scartando gli individui peggiori: in questo modo, non è necessario definire una *fitness* non negativa per gli individui e l'ottimizzazione considera direttamente la funzione obiettivo, che può essere, a seconda dei casi, massimizzata o minimizzata.

1.5. Programmazione evolutiva

L'evoluzione, naturale o artificiale, in sé non ha nulla di "intelligente" nel senso letterale del termine: infatti non capisce quello che sta facendo, né deve capirlo. L'intelligenza, invece, ammesso che si possa definire, può essere un fenomeno "emergente" dell'evoluzione, nel senso che l'evoluzione può giungere a produrre organismi o soluzioni dotati di una qualche forma di "intelligenza".
La programmazione evolutiva nasce come approccio all'intelligenza artificiale, alternativo rispetto alle tecniche basate sul ragionamento simbolico. Il suo scopo è di far evolvere, piuttosto che definire a priori, comportamenti intelligenti, rappresentati per mezzo di automi a stati finiti. Nella programmazione evolutiva, quindi, il problema oggetto determina l'alfabeto di ingresso e di uscita di una famiglia di automi a stati finiti, e gli individui sono opportune rappresentazioni di automi a stati finiti che operano su tali alfabeti. La rappresentazione naturale di un automa a stati finiti consiste nella matrice che definisce le due funzioni di transizione di stato e di uscita. La definizione degli operatori di mutazione e ricombinazione è leggermente più complessa che nel caso degli algoritmi genetici o delle strategie evolutive, in quanto deve tenere conto della struttura degli oggetti che tali operatori devono manipolare. La *fitness* di un individuo può essere calcolata mettendo alla prova su un insieme di casi del problema l'automa a stati finiti che esso rappresenta: per esempio, se si desidera far evolvere individui capaci di modellare una serie storica, si selezioneranno un certo numero di pezzi della serie passata, li si daranno in ingresso a un individuo e se ne osserveranno i simboli prodotti, interpretandoli come previsioni e confrontandoli con i dati effettivi per misurarne l'accuratezza.

1.6. Programmazione genetica

La programmazione genetica [7] è una branca degli algoritmi evolutivi relativamente nuova, che si pone come obiettivo un vecchio sogno dell'intelligenza artificiale: la programmazione automatica. In un problema di programmazione, una soluzione è rappresentata da un programma in un dato linguaggio di programmazione. Nella programmazione genetica, quindi, gli individui rappresentano programmi.
Qualsiasi linguaggio di programmazione, almeno in linea di principio, potrebbe essere adottato; tuttavia, la sintassi della maggior parte dei linguaggi renderebbe la definizione di operatori genetici che la rispettassero particolarmente goffa ed onerosa, ragione per cui i primi sforzi in questa direzione trovarono in una sorta di LISP ristretto un mezzo ideale di espressione. Il LISP ha il pregio di possedere una sintassi particolarmente semplice, oltre a permettere di manipolare dati e programmi in modo uniforme. In pratica, per ogni problema di programmazione che si voglia risolvere, si stabilisce un insieme di variabili, costanti e funzioni adatto a risolverlo, limitando così lo spazio di ricerca che altrimenti sarebbe spropositato. Le funzioni scelte saranno quelle che *a priori* si riterranno utili ai bisogni; inoltre, di solito, si cerca di fare in modo che ciascuna funzione accetti come argomenti i risultati prodotti da ognuna delle altre, così come ogni variabile e costante predefinita. Di conseguenza, lo spazio dei possibili programmi, all'interno del quale si cerca quello che risolve il problema, sarà costituito da tutte le possibili composizioni di funzioni che possano essere formate ricorsivamente a partire dall'insieme delle funzioni, delle variabili e delle costanti predefinite.
Per semplicità, e senza perdita di generalità, si possono considerare gli individui della programmazione genetica come alberi di *parsing* di altrettanti programmi, come illustrato nella figura 1. La ricombinazione di due programmi, in questo contesto, consiste nell'estrarre a caso un nodo dell'albero di ciascuno dei due genitori e nello scambiare i sottoalberi che hanno tali nodi come radici, come illustrato schematicamente nella figura 2. L'operatore di muta-

zione riveste un'importanza limitata nella programmazione genetica, in quanto la ricombinazione da sola è in grado di generare una diversità sufficiente a far procedere l'evoluzione.
Il calcolo della *fitness* di un individuo procede in un modo non troppo dissimile dal *testing* di un programma: deve essere dato, come parte integrante della descrizione del problema da risolvere, un insieme di casi di *test*, cioè di coppie (dati di ingresso, risultato corretto), che verranno usati per testare i programmi generati dall'algoritmo in questo modo: per ciascun caso, il programma viene eseguito sui dati di ingresso; il risultato ottenuto viene confrontato con quello corretto, e l'errore misurato; infine, la *fitness* è ottenuta in funzione dell'errore totale accumulato.
Un approccio ancora più recente alla programmazione genetica è costituito dalla cosiddetta *evoluzione grammaticale* [8], la cui idea di fondo è semplice ma potente: data la grammatica di un linguaggio di programmazione (che questa volta è completamente arbitrario, senza limitazioni derivanti dalla particolare sintassi), costituita da un certo numero di regole di produzione, un programma in questo linguaggio è rappresentato da una stringa di cifre binarie. La rappresentazione viene decodificata partendo dal simbolo non terminale obiettivo della grammatica e leggendo le cifre binarie da sinistra a destra, ogni volta in numero sufficiente a scegliere quale delle regole di produzione applicabili debba essere effettivamente applicata; si considera la stringa come se fosse circolare, in modo che il processo di decodifica non si trovi mai a corto di cifre binarie; il processo termina quando nessuna regola di produzione è più applicabile e si è ottenuto quindi un programma ben formato, che può essere compilato ed eseguito in ambiente controllato.

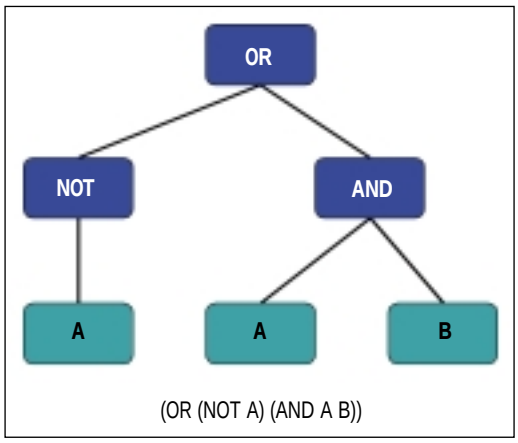


FIGURA 1
Un esempio di programma LISP con il suo albero di parsing associato

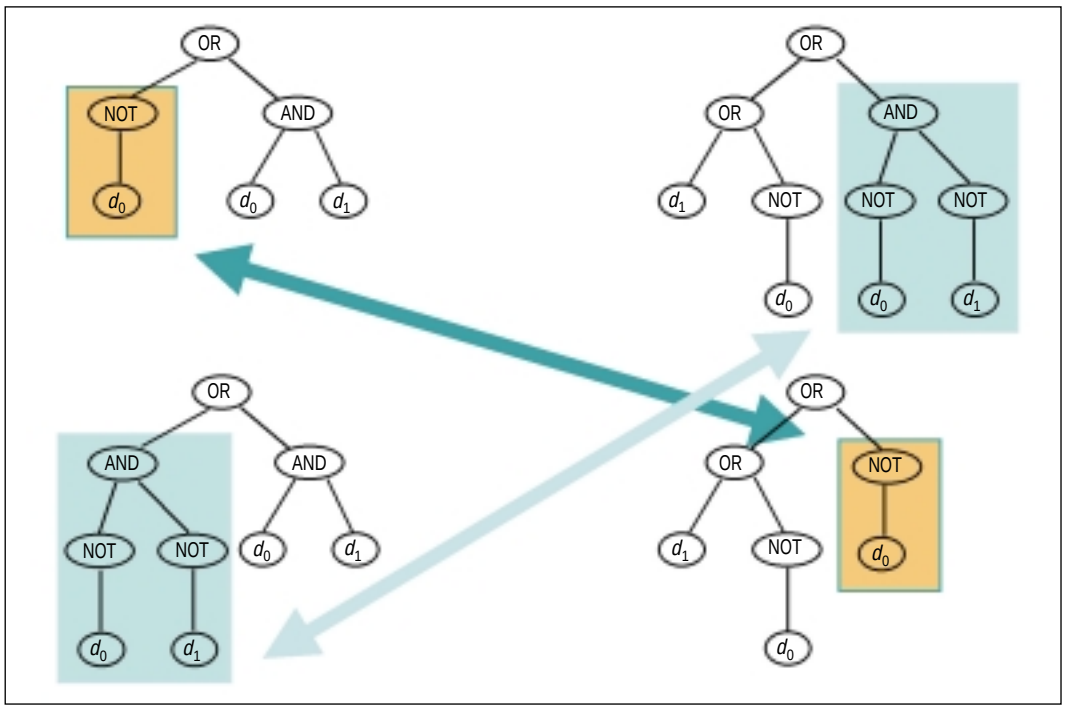


FIGURA 2
Illustrazione schematica della ricombinazione nella programmazione genetica

2. ALGORITMI EVOLUTIVI "MODERNI"

Dagli inizi degli anni ‘80 ad oggi gli algoritmi evolutivi sono stati applicati con successo a molti problemi del mondo reale, studiati dalla ricerca operativa e difficili o impossibili da trattare con metodi esatti e si sono guadagnati un posto di tutto rispetto nella cassetta degli attrezzi del risolutore di problemi. Chiaramente, questo quarto di secolo ha assistito a una maturazione delle varie tecniche evolutive e a una loro fertilizzazione incrociata, oltre che a una progressiva ibridazione con altre metodologie.

Se dovessimo identificare una linea di tendenza principale in questo processo di sviluppo, potremmo senza dubbio riconoscere un progressivo distacco dalle rappresentazioni eleganti, a stringhe binarie, dei primi algoritmi genetici, così suggestivamente vicine alla fonte d’ispirazione biologica e una maggiore propensione ad adottare rappresentazioni più vicine alla natura del problema oggetto, che si mappino in modo più diretto sugli elementi costitutivi delle soluzioni, permettendo così di sfruttare tutte le informazioni a disposizione per “aiutare”, per così dire, il processo evolutivo a trovare la strada migliore [9].

Adottare rappresentazioni più vicine al problema significa anche necessariamente progettare operatori di mutazione e ricombinazione che manipolano gli elementi di una soluzione in modo esplicito, in modo informato; da un lato questi operatori finiscono per essere meno generali, ma dall’altro i vantaggi in termini di prestazioni sono spesso notevoli e ripagano del maggior sforzo di progettazione.

Chiaramente, la richiesta di soluzioni efficienti fa perdere di vista l’unitarietà del modello genetico.

2.1. Gestione dei vincoli

I problemi del mondo reale, quelli cioè che si incontrano nell’industria, nel commercio, nella finanza e nella pubblica amministrazione, la cui risoluzione ha spesso un impatto economico rilevante e che formano l’obiettivo principale della ricerca operativa, hanno tutti in comune la caratteristica di avere vincoli complessi e difficili da trattare.

Nei primi lavori sul calcolo evoluzionistico, non era molto chiaro come affrontare nella maniera ottimale la problematca della gestione dei vincoli. Con il tempo, gli algoritmi evolutivi da una parte hanno cominciato ad essere apprezzati come metodi approssimati per la ricerca operativa e, dall’altra, hanno potuto beneficiare di tecniche ed accorgimenti messi a punto nell’ambito della ricerca operativa per altri metodi approssimati. Da questa fertilizzazione incrociata sono emerse tre tecniche principali che all'occorrenza possono essere combinate tra di loro, per tenere conto di vincoli non banali in un algoritmo evolutivo:

- l’uso di funzioni di penalizzazione;
- l’uso di algoritmi di decodifica o di riparazione;
- la progettazione di codifiche e operatori genetici specializzati.

Le funzioni di penalizzazione sono associate a ciascun vincolo del problema e misurano il grado di violazione del loro vincolo da parte di una soluzione. Come suggerisce il loro nome, queste funzioni vengono combinate con la funzione obiettivo in modo da penalizzare la *fitness* degli individui che non rispettano qualche vincolo. Sebbene le funzioni di penalizzazione costituiscano un approccio molto generale e facile da applicare a qualsiasi problema, il loro utilizzo non è privo di insidie: se le funzioni di penalizzazione non sono accuratamente pesate, l’algoritmo potrebbe sprecare la maggior parte del suo tempo elaborando soluzioni non ammissibili, o addirittura finire per convergere su un ottimo apparente che in realtà non è realizzabile. Per esempio, in un problema di trasporto, in cui sono dati *n* stabilimenti di produzione ed *m* acquirenti a cui una data quantità di merce deve essere consegnata, ed è noto quanto costi trasportare un’unità di merce tra ciascuno stabilimento e ciascun acquirente, una soluzione che minimizza il costo complessivo in modo imbattibile è quella che non trasporta assolutamente nulla! Se la violazione dei vincoli che impongono che ad ogni acquirente sia consegnato il quantitativo di merce che ha ordinato non viene penalizzata a sufficienza, la soluzione assurda di non consegnare alcuna merce potrebbe risultare migliore di tutte quelle che soddisfano gli ordini degli acquirenti. In alcuni problemi, chiamati *problemi di ammissibilità*,

trovare una soluzione che non viola alcun vincolo è tanto difficile quanto trovare la soluzione ottima, o poco meno: in questo genere di problemi, le funzioni di penalizzazione devono essere progettate con cura, altrimenti l'evoluzione non riuscirà mai a trovare alcuna soluzione ammissibile.
Gli algoritmi di decodifica, o decodificatori, sono algoritmi di ottimizzazione basati su un'euristica parametrizzabile che cercano di costruire una soluzione ottima partendo da zero e compiendo un certo numero di scelte. L'idea allora, qualora uno di questi algoritmi sia disponibile, è quella di codificare negli individui trattati dall'algoritmo genetico i parametri dell'euristica, piuttosto che direttamente la soluzione, e utilizzare il decodificatore per ricavare dai parametri la soluzione corrispondente. Si ha cioè quella che potremmo chiamare una rappresentazione *indiretta* delle soluzioni.
Gli algoritmi di riparazione sono operatori che, in base a qualche euristica, prendono una soluzione non ammissibile e la "riparano" forzando il rispetto prima di un vincolo violato, poi di un altro, fino ad ottenere una soluzione ammissibile. Applicati al risultato degli operatori genetici di mutazione e ricombinazione, questi riparatori possono garantire che l'algoritmo evolutivo elabori in ogni momento solamente soluzioni ammissibili. Tuttavia, l'applicabilità di questa tecnica è limitata, in quanto per molti problemi la complessità computazionale del riparatore è tale da vanificare i vantaggi derivanti da una sua eventuale adozione.
La progettazione di codifiche e operatori genetici specializzati sarebbe la tecnica ideale, ma anche quella più complicata da applicare in tutti i casi. L'idea di fondo consiste nel cercare di progettare una rappresentazione delle soluzioni che, per costruzione, sia in grado di codificare solo tutte le soluzioni ammissibili e operatori specifici di mutazione e ricombinazione che preservino l'ammissibilità delle soluzioni a cui sono applicati. Come si può immaginare, questo esercizio, al crescere della complessità e del numero dei vincoli, diventa ben presto formidabile ed eventualmente impossibile. Tuttavia, quando questa strada può essere percorsa, è sicuramente quella ottimale, in quanto garantisce che l'algoritmo evolutivo elabori soltanto soluzioni ammissibili e quindi, di fatto, riduce lo spazio di ricerca al minimo indispensabile.

2.2. Combinazione con altre tecniche di soft computing

Gli algoritmi evolutivi fanno parte, insieme alla logica *fuzzy* e alle reti neurali, di quello che potremmo chiamare *soft computing*, per contrapposizione alla computazione tradizionale, *hard*, basata su criteri come la precisione, il determinismo e il contenimento della complessità. Il *soft computing* si distingue dalle tecniche convenzionali (*hard computing*) per il fatto di tollerare l'imprecisione, l'incertezza e le verità parziali. Il suo principio guida è quello di sfruttare questa tolleranza per ottenere trattabilità, robustezza e contenimento delle risorse computazionali necessarie per risolvere i problemi affrontati.
Il *soft computing* non è semplicemente una mistura degli ingredienti che lo compongono, ma una disciplina nella quale ciascuna metodologia completa le altre intervenendo sull'aspetto del problema che meglio le si adatta [10]. Così gli algoritmi evolutivi possono essere impiegati per progettare e ottimizzare sistemi *fuzzy*, come insiemi di regole *fuzzy* o alberi di decisione *fuzzy*, ma anche per migliorare le caratteristiche di apprendimento delle reti neurali, arrivando anche a determinarne la topologia ottimale; d'altro canto, la logica *fuzzy* può essere utilizzata per controllare il processo evolutivo agendo in modo dinamico sui parametri dell'algoritmo, in modo da accelerare la convergenza all'ottimo globale e sfuggire dagli ottimi locali, ma anche per "*fuzzificare*" alcuni elementi dell'algoritmo, come la *fitness* degli individui o la loro codifica, mentre le reti neurali possono essere affiancate a un algoritmo evolutivo per ottenere una stima approssimata della *fitness* degli individui per problemi in cui il calcolo della *fitness* richieda simulazioni molto pesanti dal punto di vista computazionale, riducendo in tal modo il tempo macchina e migliorando le prestazioni.
Le combinazioni degli algoritmi evolutivi con altre tecniche di *soft computing* costituiscono un campo di ricerca affascinante e una

delle grandi promesse di questa gamma di tecniche computazionali.

3. LE APPLICAZIONI

Gli algoritmi evolutivi sono stati applicati con successo a problemi in un grande numero di domini. A puro scopo illustrativo e senza pretendere di proporre una classificazione significativa, potremmo dividere il campo di applicazione di queste tecniche in cinque vasti domini:

- pianificazione, che include tutti quei problemi in cui si richiede di scegliere, tra diversi modi alternativi di impiegare un insieme finito di risorse, quello a minor costo o a più alte prestazioni: fanno parte di questo dominio la pianificazione di rotta di una flotta dei veicoli, il problema del trasporto, la pianificazione della traiettoria di un robot, la pianificazione della produzione di un impianto industriale, la confezione di orari, la determinazione del carico ottimale di un mezzo di trasporto ecc.;
- progettazione, che include tutti quei problemi in cui si richiede di determinare una disposizione ottimale di elementi (componenti elettroniche o meccaniche, elementi architettonici ecc.) al fine di soddisfare una serie di requisiti funzionali, estetici e di robustezza: ricadono in questo dominio, quindi, vari problemi di disegno di circuiti elettronici, di strutture ingegneristiche, di progettazione di sistemi informativi ecc.;
- simulazione e identificazione, che consiste, dato un progetto o un modello di un sistema, nel determinare come tale sistema si comporterà: in alcuni casi ciò deve essere fatto perché non si è sicuri del comportamento del sistema, altre volte il comportamento è noto ma si vuole valutare l'accuratezza di un modello. I sistemi studiati possono essere chimici (determinazione della struttura tridimensionale di una proteina, dell'equilibrio di una reazione chimica), economici (simulazione delle dinamiche della concorrenza in un'economia di mercato), medici, e così via;
- controllo, che include tutti quei problemi in cui è richiesto di stabilire una strategia di controllo per un dato sistema;
- classificazione, modellazione e apprendimento automatico, dove, a partire da un insieme di osservazioni, si richiede di costruire un modello del fenomeno sottostante: a seconda dei casi questo modello può consistere nella semplice determinazione di appartenenza di ciascuna osservazione a una di due o più classi, oppure nella costruzione (o apprendimento automatico) di un modello più o meno complesso, spesso da utilizzare per scopi di previsione. Fa parte di questo dominio anche il *data mining*, che consiste nello scoprire regolarità invisibili "ad occhio nudo" in mezzo a enormi quantità di dati.

Naturalmente i confini tra questi cinque domini applicativi non sono nettamente definiti e i domini stessi possono in alcuni casi sovrapporsi in qualche misura. Tuttavia si noterà che essi comprendono tutta una serie di problemi di grande rilievo economico, oltre che di enorme difficoltà.

Nel seguito, cercheremo di dare un'idea di che cosa significhi applicare algoritmi evolutivi a problemi di rilevanza pratica descrivendo tre esempi di applicazioni in domini molto diversi tra loro: la confezione di orari scolastici, la progettazione di circuiti elettronici e la modellazione del comportamento di un cliente.

3.1. Confezione di orari scolastici

Il problema dell'orario (*timetable problem*) consiste nella pianificazione di un certo numero di incontri (esami, lezioni, partite ecc.) che coinvolgano un gruppo di persone (studenti, insegnanti, giocatori ecc.) per un certo periodo e richiedano un insieme di risorse (aule, laboratori, campi di gioco ecc.) secondo le loro rispettive disponibilità e rispettando tutta una serie di altri vincoli accessori. Questo problema è noto essere NP-completo: questo è il motivo principale per cui non può essere affrontato in modo soddisfacente (dal punto di vista delle prestazioni) con algoritmi di tipo esatto ed è da tempo una palestra per tecniche alternative, come gli algoritmi evolutivi. Il problema di confezionare orari scolastici, in particolare per le scuole medie superiori italiane, molte delle quali distribuite su più sedi, è ulteriormente complicato dalla presenza di vincoli molto stringenti, che lo rendono molto prossimo a un vero e proprio problema di ammissibilità.

Un'istanza di questo problema è costituita dalle seguenti entità e dalle loro relazioni:

- aule, caratterizzate per tipo, capacità e localizzazione;

- materie di insegnamento, identificate dal tipo di aula che richiedono;
- insegnanti, caratterizzati dalle materie che insegnano e dai loro orari di disponibilità;
- classi, cioè gruppi di studenti che seguono lo stesso *curriculum*, assegnate a una data sede, e con un orario di presenza a scuola;
- lezioni, una relazione <*t*, *s*, *c*, *l*>, dove *t* è l'insegnante, *s* è la materia, *c* è la classe e *l* è la durata temporale espressa in *periodi* (per esempio, ore); in certi casi, ad una lezione possono partecipare più di un insegnante e più di una classe, nel qual caso si parlerà di lezioni *raggruppate*.

I vincoli del problema sono molteplici, divisi in vincoli rigidi e vincoli flessibili, ma lo spazio ristretto non ci consente di passarli in rassegna; fortunatamente, chiunque abbia frequentato una scuola media superiore in Italia dovrebbe averne almeno un'idea.
Questo problema è stato affrontato mediante un algoritmo evolutivo che sta alla base di un prodotto commerciale, *EvoSchool* [11]. L'algoritmo adotta una rappresentazione "diretta" delle soluzioni, che consiste in un vettore le cui componenti corrispondono alle lezioni che devono essere programmate, e il valore (intero) di ciascuna componente indica il periodo in cui la lezione corrispondente deve avere inizio. La funzione che associa ad ogni orario la sua *fitness*, uno dei punti critici dell'algoritmo, consiste in pratica solo in una combinazione di funzioni di penalizzazione ed assume la seguente forma:

$$f(\mathbf{x}) = 1/\Sigma_i\, \alpha_i\, h_i + \gamma/\Sigma_j\, \beta_j\, s_j$$

dove h_i è la penalizzazione associata alla violazione dell'*i*-esimo vincolo rigido (*hard*), s_j è la penalizzazione associata alla violazione del *j*-esimo vincolo flessibile (*soft*), mentre le α_i e β_j sono opportuni pesi associati a ciascun vincolo; infine, γ è un indicatore che vale uno quando tutti i vincoli rigidi sono soddisfatti e zero altrimenti. In sostanza, ciò vuol dire che i vincoli flessibili vengono presi in considerazione solo dopo che tutti i vincoli rigidi siano stati soddisfatti.
Tutti gli altri ingredienti dell'algoritmo evolutivo utilizzato sono abbastanza standard, tranne la presenza di due operatori di perturbazione, mutuamente esclusivi, chiamati dall'operatore di mutazione ciascuno con la sua probabilità:

- mutazione intelligente;
- operatore di miglioramento.

La mutazione intelligente, pur mantenendo una natura stocastica, è rivolta ad effettuare modifiche che non diminuiscano la *fitness* dell'orario a cui sono applicate; in particolare, se l'operatore agisce sull'*i*-esima lezione, propagherà la sua azione anche a tutte le altre lezioni che coinvolgono la stessa classe, insegnante o aula. La scelta del "raggio d'azione" di questo operatore è casuale, con una certa distribuzione di probabilità. In pratica, l'effetto di questo operatore è di muovere casualmente alcune lezioni collegate tra loro in modo da ridurre le violazioni di vincoli.
L'operatore di miglioramento, invece, compie una ristrutturazione radicale di un orario, scegliendo a caso una lezione di partenza e concentrandosi sugli orari parziali della classe, dell'insegnante e dell'aula in essa coinvolti. La ristrutturazione consiste nel compattare le lezioni presenti, liberando uno spazio sufficiente per sistemare senza conflitti la lezione selezionata.
L'interazione accuratamente bilanciata di questi due operatori è il segreto dell'efficacia di questo algoritmo evolutivo, che si è dimostrato in grado di generare in poche ore, su PC non particolarmente potenti come quelli in uso nei licei e negli istituti tecnici, orari di alta qualità per complessi scolastici con migliaia di lezioni da programmare in diverse sezioni staccate disperse sul territorio.

3.2. Progettazione di circuiti elettronici digitali

Uno dei problemi che hanno ricevuto considerevole attenzione da parte della comunità internazionale del calcolo evoluzionistico è la progettazione di filtri digitali con risposta all'impulso finita. Questo interesse è giustificato dalla presenza di questo tipo di componente in un gran numero di dispositivi elettronici presenti su altrettanti prodotti di largo consumo, come telefoni cellulari, dispositivi di rete ecc.
Le metodologie tradizionali per la progettazione di circuiti elettronici hanno come criterio principale la minimizzazione dei *transistor* impiegati e dunque del costo di produzione. Tuttavia, un altro criterio molto signifi-

TABELLA 3

Operazioni primitive per la rappresentazione di filtri digitali. Il formato delle primitive è fisso. Gli interi n e m si riferiscono agli ingressi al ciclo t – n e t – m rispettivamente

Operazione	Codice	Operando 1	Operando 2	Descrizione
Ingresso	I	non usato	non usato	Copia l'ingresso
Ritardo	D	*n*	non usato	Ritardo di *n* cicli
Scorrimento a sx	L	*n*	*p*	moltiplica per 2^p
Scorrimento a dx	R	*n*	*p*	divide per 2^p
Sommatore	A	*n*	*m*	somma
Sottrattore	S	*n*	*m*	differenza
Complemento	C	*n*	non usato	inverte l'ingresso

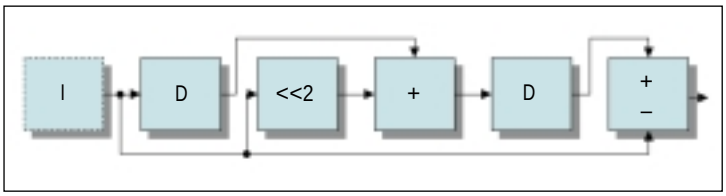


FIGURA 3

Diagramma schematico di un esempio di circuito ottenuto per composizione di 6 operazioni primitive

cativo è quello dell'assorbimento di potenza, che è funzione del numero di transizioni logiche subite dai nodi del circuito. La progettazione di filtri digitali ad assorbimento minimo di potenza è stata affrontata con successo mediante un algoritmo evolutivo [12].
Un filtro digitale può essere rappresentato come composizione di un numero molto ridotto di operazioni elementari, come le primitive elencate in tabella 3. Ciascuna operazione primitiva è codificata per mezzo del suo codice (un carattere) e due numeri interi, che rappresentano l'*offset* relativo (calcolato rispetto alla posizione corrente) dei due operandi. Quando tutti gli *offset* sono positivi, il circuito non presenta retroazioni e la struttura risultante è quella di un filtro con risposta a impulso finito. Per esempio, l'individuo

(I 0 2) (D 1 3) (L 2 2) (A 2 1) (D 1 0) (S 1 5)

corrisponde al diagramma schematico di figura 3.
La funzione di *fitness* è a due stadi: nel primo stadio, penalizza le violazioni delle specifiche della risposta in frequenza del filtro, rappresentate mediante una "maschera" nel grafico della risposta in frequenza; nel secondo stadio, che entra in azione nel momento in cui la risposta in frequenza è in maschera, la *fitness* è inversamente proporzionale all'attività del circuito, la quale dal canto suo è direttamente proporzionale all'assorbimento di potenza.
L'algoritmo evolutivo che risolve questo problema richiede molta potenza computazionale; per questo motivo è stato realizzato in modo distribuito, su un *cluster* di elaboratori elettronici, secondo un modello a *isole*, in cui la popolazione è divisa in un certo numero di isole, residenti su macchine distinte, che evolvono indipendentemente salvo scambiarsi, di tanto in tanto, degli individui "emigranti", che permettono di realizzare una circolazione di materiale genetico mantenendo allo stesso tempo la banda di comunicazione necessaria al funzionamento dell'algoritmo piccola a piacere.
Un risultato sorprendente di questo approccio evolutivo alla sintesi di circuiti elettronici è stato che i filtri digitali scoperti dall'evoluzione, oltre ad avere, come richiesto, un assorbimento di potenza nettamente inferiore rispetto ai filtri corrispondenti ottenuti con tecniche di sintesi tradizionali, mostrano una riduzione dal 40% al 60% del numero di elementi logici, e quindi anche dell'area e della velocità. La diminuzione dei consumi, in altre parole, non è stata ottenuta a spese del costo di produzione e della velocità, ma si è accompagnata con un generale aumento di efficienza rispetto ai metodi tradizionali di progettazione.

3.3. Data mining

Un fattore critico di successo per ogni azienda è la sua capacità di utilizzare le informazioni (e la conoscenza che se ne può estrarre) in modo efficace. L'uso strategico dei dati può derivare dalle opportunità offerte dalla

scoperta di fatti nascosti, precedentemente non riscontrati e spesso estremamente preziosi relativi ai consumatori, ai rivenditori e ai fornitori, all'andamento delle attività commerciali. Conoscendo queste informazioni, un'organizzazione è in grado di formulare strategie efficaci di marketing e vendita, focalizzare le azioni promozionali, scoprire e penetrare nuovi mercati e ottenere sul mercato una posizione di vantaggio rispetto ai concorrenti. L'attività di vaglio delle informazione al fine di ottenere un siffatto vantaggio competitivo è nota come *data mining* [13]. Da un punto di vista tecnico, il *data mining* può essere definito come la ricerca di correlazioni, schemi e tendenze non percepibili "ad occhio nudo" per mezzo del vaglio approfondito di grandi masse di dati immagazzinati in grandi banche dati e *data warehouse*, sfruttando metodi statistici, di intelligenza artificiale, di apprendimento automatico e di *soft computing*. Sono molte le grandi aziende e organizzazioni, come banche, assicurazioni, catene di grande distribuzione ecc., che dispongono di un'enorme quantità di informazioni sul comportamento dei loro clienti. La possibilità di sfruttare queste informazioni per inferire modelli del comportamento dei propri clienti attuali e futuri in relazione a prodotti specifici o a classi di prodotti è una prospettiva molto attraente per queste organizzazioni. I modelli così ottenuti potranno poi essere utilizzati per prendere decisioni strategiche e per meglio focalizzare le azioni di *marketing*, a patto che essi siano accurati, comprensibili e informativi.

Chi scrive ha partecipato, negli ultimi cinque anni, alla definizione, messa a punto e validazione di un potente motore per il *data mining*, sviluppato da Genetica Srl e da Nomos Sistema SpA (oggi un'azienda del gruppo Accenture) in collaborazione con l'Università degli Studi di Milano, nel quadro di due progetti Eureka finanziati dal Ministero dell'Istruzione e dell'Università (ex M.U.R.S.T.), che si basa sull'utilizzo di un algoritmo genetico per la sintesi di modelli predittivi del comportamento dei clienti, espressi mediante insiemi di regole *fuzzy* di tipo SE ... ALLORA. Questo approccio, tra l'altro, è un chiaro esempio dei vantaggi che possono essere conseguiti mediante la combinazione degli algoritmi evolutivi con la logica *fuzzy*, di cui si è parlato sopra.

L'approccio parte dalla disponibilità di un *data set*, cioè di un insieme, grande a piacere, di record che rappresentano osservazioni o registrazioni dei comportamenti passati di clienti. A dire il vero, il campo di applicabilità sarebbe ancor più vasto, considerando questi record come osservazioni puntuali di un qualche fenomeno, non necessariamente economico o commerciale, come per esempio la misurazione mediante radar di elettroni liberi nella ionosfera [14].

Un record è composto da m attributi, cioè valori di variabili che descrivono il cliente. Tra questi, si suppone che ne esista almeno uno che misura l'aspetto del comportamento dei clienti che si desidera modellare. Senza perdita di generalità, si può assumere che esista un solo attributo di questo genere: infatti, se fossimo interessati a modellare diversi aspetti del comportamento, potremmo sviluppare altrettanti modelli distinti. Potremmo chiamare questo attributo "predittivo", perché ci serve per predire il comportamento di un cliente. In questo quadro concettuale, un modello è una funzione con $m - 1$ parametri che esprime il valore dell'attributo predittivo in base ai valori degli altri attributi.

Il modo in cui si sceglie di rappresentare questa funzione è critico: l'esperienza dimostra che l'utilità e l'accettabilità di un modello non derivano soltanto dalla sua accuratezza, che, beninteso, è una condizione necessaria, ma anche e soprattutto dalla sua intellegibilità da parte dell'esperto che necessariamente lo dovrà valutare prima di autorizzarne l'utilizzo. Una rete neurale o un programma LISP, che altri autori hanno scelto come "linguaggi" per esprimere i modelli, può fornire risultati imbattibili quanto ad accuratezza, ma qualsiasi organizzazione sarà riluttante a "fidarsi" dei suoi risultati se non potrà comprendere e spiegare come essi siano stati ottenuti. Questa è la motivazione di fondo che ha determinato l'adozione degli insiemi di regole *fuzzy* SE ... ALLORA come linguaggio per l'espressione dei modelli. Tali insiemi di regole, infatti, costituiscono probabilmente quanto di più prossimo esista al modo intuitivo di esprimere la propria conoscenza da parte di

un esperto, grazie all'utilizzo di regole che esprimono relazioni tra variabili linguistiche (aventi cioè valori linguistici del tipo BASSO, MEDIO, ALTO). Inoltre, le regole *fuzzy*, sfumate, possiedono la proprietà molto desiderabile di comportarsi in modo *interpolativo*, cioè non saltano mai da una conclusione a quella opposta per colpa di un piccolo cambiamento nel valore di una condizione, come invece succede per le regole di tipo netto.
La codifica adottata per rappresentare un modello all'interno dell'algoritmo genetico è abbastanza complicata, ma rispecchia da vicino la struttura logica di un insieme di regole *fuzzy* e consente la definizione di operatori specifici di mutazione e ricombinazione che operano in modo informato sui suoi blocchi costitutivi. In particolare, l'operatore di ricombinazione è progettato in modo tale da preservare la legalità sintattica dei modelli; un modello figlio si ottiene combinando le regole dei due modelli genitori: ogni regola del modello figlio può essere ereditata dall'uno o dall'altro genitore con eguale probabilità e, quando viene ereditata, porta con sé tutte le definizioni dei valori linguistici (insiemi *fuzzy*) presenti nel genitore da cui proviene, che contribuiscono a determinarne la semantica.
I modelli sono valutati applicandoli a una parte del *data set* e ottenendo un valore di *fitness* che misura la loro accuratezza; la parte restante del *data set* viene utilizzata, come è prassi nell'apprendimento automatico, per monitorare le capacità di generalizzazione dei modelli ed evitare così il fenomeno dell'*overfitting*, che si ha quando un modello apprende uno per uno gli esempi che ha "visto", invece di catturare le regole generali che possono essere applicate anche a casi mai visti in precedenza.
Il motore che utilizza questo approccio è stato applicato con successo alla valutazione del credito in ambito bancario, alla stima della redditività dei clienti in campo assicurativo [15] e al recupero dei crediti al consumo.

4. CONCLUSIONI

Questa breve rassegna sugli algoritmi evolutivi ha cercato di fornire una panoramica completa, anche se per ovvi motivi di spazio non esaustiva, sui vari filoni tradizionali in cui si dividono (algoritmi genetici, strategie evolutive, programmazione evolutiva e programmazione genetica). Ha inoltre fornito alcuni elementi sulle problematiche più significative che riguardano l'applicazione pratica del calcolo evoluzionistico a problemi di rilevanza industriale ed economica, come la rappresentazione delle soluzioni e la gestione dei vincoli, per le quali la ricerca ha fatto negli ultimi anni sostanziali progressi. Infine, ha completato la trattazione con un'illustrazione più approfondita, pur se non appesantita da eccessivi dettagli tecnici, di tre esempi di applicazioni a problemi "del mondo reale", selezionati in domini il più possibile distanti tra loro, in modo tale da fornire tre visioni complementari sulle criticità e sulle problematiche che si incontrano nel realizzare, a partire dall'idea di fondo, un sistema *software* che "funzioni", ma anche da far apprezzare al lettore la versatilità e le enormi potenzialità di queste tecniche che, a quasi quarant'anni dalla loro prima introduzione, si trovano ancora nella loro adolescenza. Se c'è una lacuna in questa rassegna, è nei fondamenti teorici del calcolo evoluzionistico, che comprendono la teoria degli schemi (con l'ipotesi cosiddetta dei *building block*) e la teoria della convergenza, argomenti che sono stati volutamente tralasciati perché avrebbero richiesto un livello di formalismo inopportuno per un lavoro di rassegna: il lettore interessato potrà tuttavia colmare questa lacuna autonomamente, facendo riferimento alla bibliografia citata. Un altro aspetto che è stato trascurato perché non rappresenta una vera e propria "applicazione", ma che nondimeno riveste un grande interesse scientifico è quello dell'impatto che il calcolo evoluzionistico ha avuto sullo studio dell'evoluzione stessa e dei sistemi complessi in generale: si veda per esempio il lavoro di Axelrod sull'evoluzione spontanea di comportamenti cooperativi in un mondo di agenti egoisti [18].
Il lettore che volesse avvicinarsi al calcolo evoluzionistico può consultare alcuni eccellenti libri introduttivi [6, 9, 17, 19] o trattazioni più approfondite [20, 21], nonché visitare i siti Internet indicati nel riquadro **"Algoritmi evolutivi in Internet"**.

Algoritmi evolutivi in Internet
I seguenti sono alcuni siti web selezionati in cui il lettore può trovare materiale introduttivo o avanzato sugli algoritmi evolutivi:
- "http://www.isgec.org/": portale della International Society for Genetic and Evolutionary Computation;
- "http://evonet.lri.fr/": portale della rete di eccellenza europea sugli algoritmi evolutivi;
- "http://www.aic.nrl.navy.mil/galist/": GA Archives, nati come archivi della listi di distribuzione "GA-List", che oggi si chiama "EC Digest"; contiene informazioni aggiornate sugli eventi più importanti nel campo e link ad altre pagine web;
- "http://www.fmi.uni-stuttgart.de/fk/evolalg/index.html": EC Repository, mantenuto presso l'Università di Stoccarda.

Bibliografia

[1] Box George E.P., Draper N.R.: *Evolutionary Operation: Statistical Method for Process Improvement*. John Wiley & Sons, 1969.

[2] Bremermann Hans J.: *Optimization through Evolution and Recombination*. In: Yovits M.C., Jacobi G.T., Goldstein G.D. (a cura di), *Self-Organizing Systems 1962*. Spartan Books, Washington D. C., 1962.

[3] Fogel Lawrence J., Owens A. J., Walsh M.J.: *Artificial Intelligence through Simulated Evolution*. John Wiley & Sons, New York, 1966.

[4] Holland John H.: *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[5] Rechenberg Ingo: *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog, Stoccarda, 1973.

[6] Goldberg David E.: *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, 1989.

[7] Koza John R.: *Genetic Programming*. MIT Press, Cambridge, Massacusets, 1992.

[8] O'Neill Michael, Ryan Conor: *Grammatical Evolution*. Evolutionary automatic programming in an arbitrary language. Kluwer, 2003.

[9] Michalewicz Zbigniew: Genetic *Algorithms + Data Structures = Evolution Programs*. 3rd Edition. Springer, Berlino, 1996.

[10] Tettamanzi Andrea G.B., Tomassini Marco: *Soft Computing. Integrating evolutionary, neural, and fuzzy systems*. Springer, Berlino, 2001.

[11] Di Stefano Calogero, Tettamanzi Andrea G.B.: *An Evolutionary Algorithm for Solving the School Time-Tabling Problem*. In: Boers E. et al., *Applications of Evolutionary Computing*. EvoWorkshops 2001, Springer, 2001, p. 452-462.

[12] Erba Massimiliano, Rossi Roberto, Liberali Valentino, Tettamanzi Andrea G.B.: *Digital Filter Design Through Simulated Evolution*. Atti di ECCTD'01 - European Conference on Circuit Theory and Design, 28-31 agosto 2001, Espoo, Finlandia.

[13] Berson Alex, Smith Stephen J.: *Data Warehousing, Data Mining & OLAP*. McGraw Hill, New York, 1997.

[14] Beretta Mauro, Tettamanzi Andrea G.B.: *Learning Fuzzy Classifiers with Evolutionary Algorithms*. In: Bonarini A., Masulli F., Pasi G., (a cura di), *Advances in Soft Computing*. Physica-Verlag, Heidelberg, 2003, p. 1-10.

[15] Tettamanzi Andrea G.B., et al.: *Learning Environment for Life-Time Value Calculation of Customers in Insurance Domain*. In: Deb K., et al. (a cura di), Proceedings of the Genetic and Evolutionary Computation Congress (GECCO 2004), S. Francisco, 26–30 giugno 2004, p. 1251-1262.

[16] Rudolph Günter: Finite Markov Chain Results in Evolutionary Computation: A Tour d'Horizon. *Fundamenta Informaticae*, Vol. 35, 1998, p. 67-89.

[17] Mitchell Melanie: *An Introduction to Genetic Algorithms*. Bradford, 1996.

[18] Axelrod Robert: *The Evolution of Cooperation*. Basic Books, 1985.

[19] Fogel David B.: *Evolutionary Computation: Toward a new philosophy of machine intelligence*. 2nd Edition. Wiley-IEEE Press, 1999.

[20] Bäck Thomas: *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, 1996.

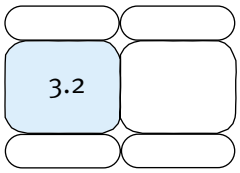
[21] Bäck Thomas, Fogel David B., Michalewicz Zbigniew (a cura di): *Evolutionary Computation* (2 volumi). IoP, 2000.

Andrea Tettamanzi è professore associato presso il Dipartimento di Tecnologie dell'Informazione dell'Università degli Studi di Milano. Laureato in Scienze dell'Informazione nel 1991, ha conseguito il Dottorato di Ricerca in Matematica Computazionale e Ricerca Operativa nel 1995, anno in cui ha fondato Genetica Srl, un'azienda specializzata nelle applicazioni industriali degli algoritmi evolutivi e del *soft computing*. Attivo nella ricerca sugli algoritmi evolutivi e sul *soft computing*, ha sempre cercato di coniugare gli aspetti teorici con quelli pratici e applicativi.
andrea.tettamanzi@unimi.it

PROGRAMMAZIONE ORIENTATA AGLI ASPETTI: SCENARI DI ADOZIONE INDUSTRIALE

Filippo Diotalevi

Il grande successo della programmazione orientata agli oggetti non ha limitato la ricerca di nuovi paradigmi e tecnologie che possano incrementare la produttività degli sviluppatori e la qualità del software. In questi ultimi anni, la programmazione orientata agli aspetti sta conoscendo una crescente diffusione. Questo articolo è sia un'introduzione a questo nuovo paradigma di programmazione che una guida alla sua graduale adozione nello sviluppo di applicazioni aziendali.



1. INTRODUZIONE

L'evoluzione dell'informatica ha portato all'affermarsi della programmazione orientata agli oggetti [1] come principale strumento per lo sviluppo di applicazioni aziendali. Complementariamente a questa affermazione si è assistito al nascere e allo svilupparsi di metodologie, linguaggi di analisi e design e di strumenti orientati agli oggetti che hanno svolto un ruolo fondamentale nel permettere di costruire sistemi informativi sempre di maggior complessità, fino ad arrivare ai moderni software distribuiti, multipiattaforma e caratterizzati dai più alti standard di scalabilità e sicurezza.
La realizzazione di applicazioni che risolvono tali problematiche ha però comportato un aumento di pari portata della complessità del software; infatti, le tematiche e le tecnologie che i professionisti del settore devono padroneggiare per poter rispondere adeguatamente alle richieste del mercato si sono moltiplicate, rendendo di conseguenza il processo di produzione del software più critico e complicato.
Se da un lato questa crescente complessità è implicita nella richiesta di funzionalità evolute ed alti livelli di servizio, è stato osservato più volte [2] che essa è anche conseguenza del fatto che il paradigma di programmazione orientato agli oggetti non si è dimostrato capace negli anni di modellare in modo adeguato tutte le problematiche emerse. La scoperta di queste limitazioni ed inefficienze ha stimolato di conseguenza la nascita di nuove teorie e proposte, collettivamente indicate come *Post Object Programming* (POP); tra queste particolare successo sta riscuotendo negli ultimi cinque anni la programmazione orientata agli aspetti (*Aspect Oriented Programming*, AOP) che sta ricevendo un notevole impulso anche in ambito industriale grazie alla comparsa di tool open source affidabili ed accessibili che permettono di applicarne i principi anche all'interno del più tradizionale sviluppo orientato agli oggetti.
Questo articolo vuole essere un'introduzione alla programmazione orientata agli aspetti. Esso si divide in tre parti: la prima parte (Paragrafo 2) fornirà una breve presentazione

delle problematiche che hanno portato alla formulazione di questo nuovo paradigma, soffermandosi sulla nuova terminologia introdotta e sulle sue caratteristiche innovative; nella seconda parte (Paragrafo 3) si cercherà di capire quali siano le criticità che ancora limitano l'adozione di AOP nel processo di produzione di software in grande scala, e quali siano le necessità di evoluzione della ricerca e dell'industria in questo senso; infine, nella terza parte (Paragrafo 4) verranno delineati alcuni scenari concreti in cui è possibile applicare allo sviluppo del software, in modo efficace e produttivo, le idee della programmazione orientata agli aspetti.

2. PROGRAMMAZIONE ORIENTATA AGLI ASPETTI: UN'INTRODUZIONE

David Parnas, in un articolo del 1972 [4], sottolinea come una delle maggiori conquiste nell'ambito della programmazione sia stato lo sviluppo di tecniche, paradigmi e strumenti che permettono una chiara e netta modularizzazione del prodotto software. L'articolo in questione affronta quindi la tematica centrale di tutto il processo di progettazione, ovvero come sia possibile modularizzare il software in modo tale da poter garantire l'indipendenza tra i componenti e la loro completa riusabilità.
In termini più moderni, si usa definire come "*concern*" una funzionalità, un requisito che può essere pensato e sviluppato come entità autonoma; data questa definizione, il lavoro dell'architetto, sottolinea Parnas, consiste nell'individuare i concern che l'applicazione dovrà indirizzare e guidare il loro sviluppo cercando di mantenerne l'indipendenza: si parla quindi di *separation of concerns* (SoC) come uno degli obiettivi primari da raggiungere nel processo di sviluppo. A trent'anni di distanza la *separation of concerns* è ancora una tematica di grande attualità, e il fermento di ricerche e di proposte in questo ambito dimostra chiaramente che ancora non si è trovata una soluzione ottimale a questo problema.
Una delle caratteristiche che ha contribuito all'affermazione del paradigma di programmazione orientato agli oggetti è stata proprio la sua capacità di modellare come componenti autoconsistenti, dotati di chiaro scopo e responsabilità, le entità principali di business, dando origine a sistemi più modulari, riusabili e verificabili.
Tuttavia l'aumentare della complessità dei requisiti (e del software di conseguenza) ha mostrato, nella pratica, come non sempre la modularità riesce ad essere gestita e implementata così come sarebbe auspicabile sulla carta. Il problema centrale risiede nel fatto che in ogni componente responsabile di eseguire una certa funzionalità vengono ad interagire tipicamente molti concern; di questi, uno o pochi sono le competenze fondamentali per quell'ambito (dette funzionalità di business) mentre molti altri sono caratteristiche accessorie, dettate spesso dalla tecnologia stessa o da requisiti non funzionali.
Si faccia riferimento per esempio a quanto è rappresentato nella figura 1; il componente competente per "inserimento cliente nel database" si trova a dover avere a che fare con concern complementari, per esempio problemi di autorizzazione (l'utente è autorizzato ad inserire nuovi clienti?), di tracciatura, di politiche di connessione a database. Tutte competenze per cui il modulo stesso non è responsabile, e che sono utilizzate diffusamente in varie parti dell'applicazione. Ci si riferisce a queste funzionalità che toccano molti componenti di una applicazione con il nome di "*crosscutting concern*".
Ciò che si verifica è che la programmazione ad oggetti, con il suo modello basato sull'ereditarietà e sulle dipendenze, non riesce a modellare efficacemente i *crosscutting concern*. Infatti, se da una parte è vero che attra-

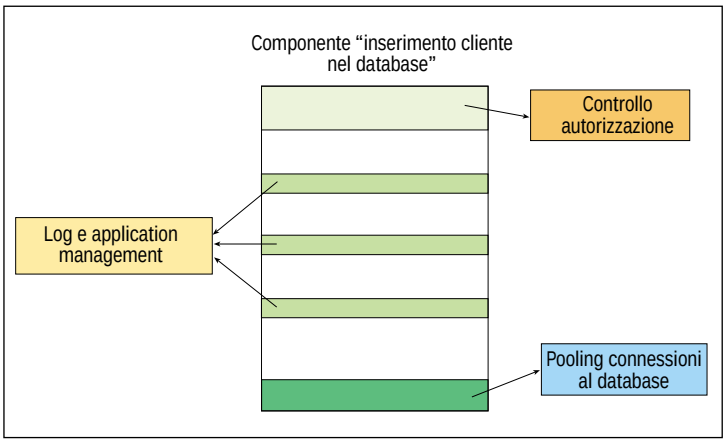


FIGURA 1
Compresenza di più concern all'interno di un singolo componente

verso questo paradigma è possibile disegnare moduli separati per la gestione di queste funzionalità di uso generale, è altrettanto vero che il meccanismo delle dipendenze fa sì che tutti gli altri moduli dell'applicazione siano intrinsecamente legati a questi. Questo fatto si manifesta chiaramente analizzando come nel codice che implementa una determinata logica di business si faccia esplicito e frequente riferimento a moduli esterni non direttamente legati a tale funzionalità.
Lo scopo con cui nasce la programmazione orientata agli aspetti è quella di fornire una base teorica, un linguaggio e degli strumenti per catturare e definire in modo sintetico i crosscutting concern e per permettere il loro utilizzo all'interno delle applicazioni minimizzando le dipendenze tra moduli diversi.

2.1. Terminologia della programmazione orientata agli aspetti

La programmazione orientata agli aspetti mette a disposizione due metodologie per indirizzare il problema dei crosscutting concerns, denominate rispettivamente *dynamic crosscutting e static crosscutting*.
Con *dynamic crosscutting* si intende la possibilità di aggiungere nuovi comportamenti ad un software esistente nel momento in cui questo viene eseguito. Questo significa fornire rispettivamente una codifica del comportamento aggiuntivo (il concern) e un set di punti dell'applicazione in cui esso deve essere eseguito.
In questo contesto, esattamente come la classe è il concetto fondamentale per la modellazione di una entità nella programmazione ad oggetti, un crosscutting concern è modellato completamente attraverso un *aspetto*.
L'aspetto si compone fondamentalmente di due parti:
- l'*advice*, che fornisce una completa implementazione del concern;
- il *pointcut*, che definisce una famiglia di punti nell'esecuzione del programma in cui l'advice deve essere eseguito

Il *pointcut* a sua volta è esprimibile come una combinazione di entità fondamentali, dette *join point*, che esprimono i punti caratteristici individuabili all'interno dell'esecuzione di un programma come l'invocazione di un metodo, di un costruttore, l'accesso ad un attributo o l'occorrenza di una eccezione.
L'utilizzo contemporaneo di una applicazione modellata ad oggetti (per tutte le funzionalità di business) e ad aspetti (per i crosscutting concern) rende il software molto più modulare e di semplice comprensione. Gli oggetti che compongono l'applicazione si occupano solamente della logica applicativa, senza fare alcun riferimento esplicito a problematiche di tipo tecnologico come quelle illustrate nella figura 1.
Ogni aspetto, d'altro canto, modella uno specifico crosscutting concern: al suo interno, l'advice specifica l'implementazione del crosscutting concern, mentre il pointcut tutto l'insieme dei punti dell'applicazione in cui esso va applicato. Sarà compito dell'infrastruttura (che sia essa la macchina virtuale Java, un framework specifico o il middleware) eseguire al momento opportuno gli aspetti all'interno del normale flusso di esecuzione dell'applicazione.
Come si può immediatamente intuire, l'utilizzo della tecnica dei pointcut si rivela uno strumento potente per migliorare la modularità di un'applicazione, perchè permette di esprimere sinteticamente all'interno dell'aspetto tutti i punti in cui dovranno essere aggiunte le funzionalità descritte dall'advice.
Ritornando all'esempio presentato in precedenza, la situazione si configura ora come illustrato nella figura 2. I tre concern non fondamentali del componente (controllo autorizzazione, log e pool di connessione al databa-

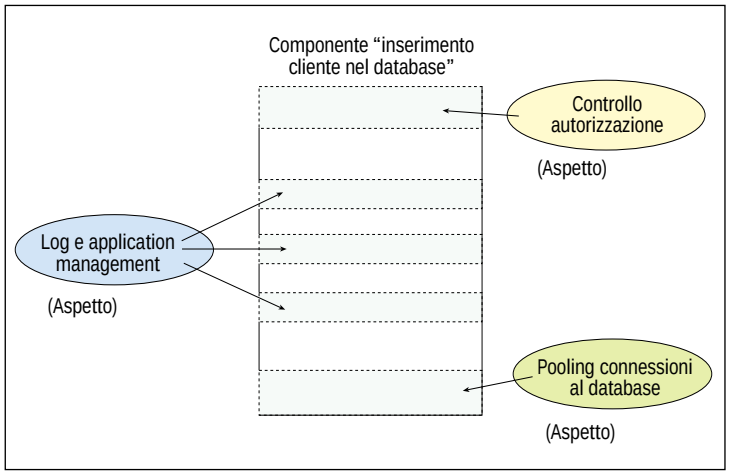


FIGURA 2
Applicazione degli aspetti all'interno di un componente

se) sono modellati come aspetti, e contengono al loro interno la definizione delle famiglie di punti del software in cui devono essere applicati. Ciò che si verifica è una inversione delle dipendenze: il componente "inserimento cliente nel database" ora non interagisce più esplicitamente con moduli esterni ma è influenzato da questi al momento dell'esecuzione. Questo significa altresì che nello sviluppo del componente non sarà più necessario occuparsi dei concern non fondamentali.
Lo *static crosscutting* si differenzia dal precedente perchè consente di alterare la struttura stessa di una famiglia di oggetti attraverso l'introduzione (si parla infatti di *introduction*) di nuovi attributi, metodi o interfacce. Anche in questo caso, l'alterazione della struttura determinata dai tool AOP non è visibile nel codice dell'applicazione, ma si manifesta solamente al momento dell'esecuzione.
Sebbene sia meno utilizzato e conosciuto rispetto al dynamic crosscutting, lo static crosscutting consente di snellire le gerarchie degli oggetti presenti nell'applicazione così come di aggirare le limitazioni di alcuni linguaggi, come Java, che non gestiscono ereditarietà multiple.

3. SFIDE E PROBLEMI NELL'ADOZIONE DELL'AOP NEL PROCESSO DI PRODUZIONE SOFTWARE

Pur rimanendo finora confinata alla popolazione dei ricercatori e dei pionieri delle nuove tecnologie, l'attenzione che sta montando attorno alla programmazione orientata agli aspetti è chiaramente misurabile considerando il numero sempre crescente di tool che nascono per permettere di utilizzare questo nuovo approccio con i linguaggi di programmazione più diffusi.
In particolare il linguaggio Java, grazie alla sua vasta base di sviluppatori, offre un'ampia e qualificata scelta in questo senso: dopo la nascita, nel 1996, del progetto precursore AspectJ [13], si è assistito negli ultimi anni al proliferare di tool analoghi, spesso finanziati e sponsorizzati dalle maggiori multinazionali del settore informatico. Tra questi particolare seguito hanno JBossAop [14], implementazione che è parte integrante dell'application server open source JBoss, e AspectWerkz [15], prodotto open source che gode della sponsorizzazione di BEA Systems.
Sul fatto che AOP possa un giorno essere adottato su grande scala le opinioni sono ancora constrastanti. Accade così che mentre IBM nella primavera di quest'anno dichiara che "l'Aspect Oriented Programming ha raggiunto livelli di affidabilità tali da permetterne un utilizzo commerciale"[6], James Gosling, uno dei creatori storici del linguaggio di programmazione Java, a soli pochi mesi di distanza osserva che "AOP è un po' troppo complicato per me, perchè promette bene nella teoria... ma il modo in cui viene messo in pratica tende ad essere troppo pericoloso" [7].
Non c'è dubbio che in queste dichiarazioni di verso opposto si nascondono i pregi e le problematiche di cui una futura adozione di AOP su scala industriale dovrà tenere conto. Da un lato, il nuovo paradigma di programmazione mette a disposizione nuove possibilità e strumenti per indirizzare in modo più efficiente problemi e temi rilevanti; d'altro canto rende necessario un ulteriore "paradigm shift" (cambio di paradigma) nelle competenze dei programmatori. E non c'è dubbio che quest'ultimo tema, già noto alle aziende che hanno dovuto riconvertire parte del loro personale dalle tecniche della programmazione strutturata a quelle della programmazione ad oggetti, rappresenti un'incognita che Gosling non manca di sottolineare.
Da questo punto di vista l'introduzione della programmazione orientata agli aspetti è agevolata dal fatto che questa si affianca e non sostituisce il paradigma più tradizionale ad oggetti, agendo quindi più come complemento che come avversario delle metolodogie di analisi, design e sviluppo attualmente in uso. Ciò rende possibile la creazione di tool che, consentendo di utilizzare congiuntamente entrambe le modalità di lavoro, possono rappresentare un acceleratore notevole per l'adozione di questo nuovo approccio.
Un ultimo aspetto che deve essere sottilineato è che la programmazione orientata agli aspetti manca tuttora di uno standard, che è un requisito forte per attirare gli investimenti delle aziende. Infatti il periodo a cui stiamo assistendo è ancora caratterizzato dalla nascita di nuovi progetti fortemente innovativi, che tentano di portare nuove idee e nuove

tecniche all'attenzione degli sviluppatori, e di conseguenza di guadagnare maggior popolarità, utilizzatori e finanziamenti. Si sente ancora la mancanza di un forte movimento atto a stabilire una linea comune in questo settore; su questa via si sta muovendo, a passi lenti, l'AOP Alliance [8], i cui standard sono ancora poco accolti e soprattutto non ancora molto sponsorizzati dai grandi gruppi multinazionali.

4. SCENARI DI ADOZIONE IN AMBITO INDUSTRIALE

Quando si parla di adozione in ambiente di produzione della programmazione orientata agli aspetti il primo esempio sempre citato è legato alle funzionalità di *logging* (tracciatura e registrazione di dati rilevanti durante l'esecuzione dell'applicazione). Si tratta infatti di un caso tipico di crosscutting concern, una operazione pervasiva che tocca tutti i moduli di una applicazione pur essendo un'aspetto marginale nel contesto di applicazioni complesse.
L'utilizzo di AOP per il logging rappresenta tuttavia solo un primo passo verso l'adozione di questa tecnologia, sicuramente necessario per cominciare ad conoscere ed approfondire i benefici che può portare, ma che non ne rivela appieno tutte le potenzialità.
Nella figura 3 è delineata una possibile evoluzione nell'utilizzo della programmazione orientata agli aspetti che, al crescere della maturità dei tool e della confidenza degli sviluppatori, può consentire di esplorarne appieno tutti i possibili utilizzi.
La curva di adozione ipotizzata nella figura 3 è composta di quattro fasi.
Nella prima, la programmazione orientata agli aspetti interviene come componente secondario di una applicazione già esistente, portando alcune nuove funzionalità interessanti ma non fondamentali, come il *logging* ed il *design by contract* (si veda il paragrafo 4.3. per una breve illustrazione di questa tecnica). In questa fase gli aspetti introdotti possono essere rimossi dall'applicazione, perchè non ne contribuiscono sostanzialmente alla funzionalità.
In una seconda fase, si assiste all'introduzione di aspetti ancora non fondamentali dal punto di vista funzionale, ma che comportano notevoli vantaggi all'applicazione. Rientrano in questa categoria tutte le tecniche per l'ottimizzazione delle prestazioni e la gestione dell'applicazione. La rimozione di questi aspetti non pregiudica il funzionamento dell'applicazione, ma possono essere avvertiti dall'utente finale (sotto forma di peggioramento dei tempi di risposta del prodotto) o dai gestori dell'applicazione.
Nella terza fase di adozione, AOP è utilizzata per modellare componenti responsabili di funzionalità applicative fondamentali, come la registrazione dei dati su database (persistenza) e l'applicazione di politiche di sicurezza e di autorizzazione. Tali aspetti fanno parte a tutti gli effetti della applicazione e non possono essere rimossi.
Infine, nella fase finale, si può ipotizzare che il paradigma di programmazione orientato agli aspetti diverrà parte integrante, assieme a quello orientato agli oggetti, del processo di progettazione del software. Ciò significa che già in fase di stesura del modello concettuale del software potranno essere identificati i principali concern ed assegnati rispettivamente ad oggetti e ad aspetti, secondo criteri formalizzati. Inoltre, la disponibilità di librerie di aspetti riutilizzabili potrà garantire un processo di sviluppo più snello ed efficiente.
Considerando ora il presente e l'immediato futuro, nella parte rimanente di questa sezione saranno delineati brevemente alcuni scenari di adozione tipici.

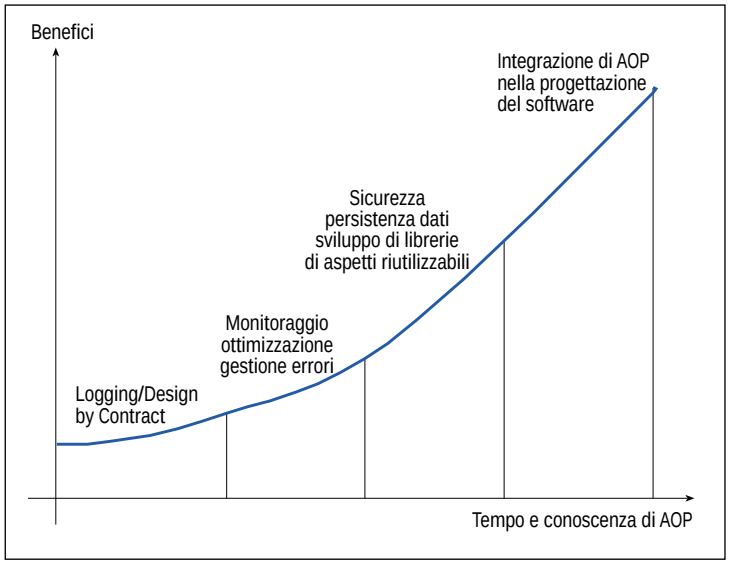


FIGURA 3
Curva di adozione di AOP

4.1. Monitoraggio dell'applicazione

Si tratta sostanzialmente di un'estensione e generalizzazione del logging che sfrutta la caratteristica della programmazione orientata agli aspetti di poter intercettare determinati eventi nel ciclo di esecuzione del software e di poter eseguire logiche opportune in queste circostanze.
Nel caso specifico è possibile in modo immediato valutare i tempi di esecuzione di determinate procedure [9] e il loro numero di invocazioni o effettuare un controllo sul numero di istanze di oggetti create. Rispetto ad un approccio più tradizionale AOP consente di variare in modo flessibile la definizione di quali parti dell'applicazione sono soggette a monitoring e quali controlli effettuare senza per questo richiedere alcuna modifica del codice applicativo oggetto dell'analisi.

4.2. Caching e ottimizzazione delle prestazioni

L'ottimizzazione delle prestazioni è una tematica spesso critica nello sviluppo di applicazioni aziendali, a causa di due fattori concomitanti: in primo luogo, l'esigenza di introdurre miglioramenti nelle performance emerge spesso nelle fasi avanzate di sviluppo, se non perfino ad applicazione già in produzione, a causa del manifestarsi di problemi di carico o di eccessivi tempi di risposta; inoltre, le tecniche utilizzate in questo ambito sono spesso complesse, soggette ad errori e di conseguenza oggetto di continui miglioramenti ed aggiustamenti.
L'utilizzo della programmazione orientata agli aspetti ha in questa ottica il grande pregio di permettere di disaccoppiare nettamente il codice responsabile della logica applicativa da quello relativo alla ottimizzazione delle prestazioni , consentendo di applicare senza problemi tali tecniche anche a posteriori, direttamente in ambiente di produzione e senza modificare il codice applicativo.

4.3. Design by Contract

I problemi di integrazione tra applicazioni diverse spesso portano al manifestarsi di errori inaspettati la cui causa è difficile da determinare proprio per la stretta interazione tra moduli differenti, spesso sviluppati da diversi fornitori, a volte anche costruiti con tecnologie differenti. In questo contesto risulta di grande utilità l'adozione di una tecnica nota come *Design by Contract* [10] che permette di verificare la correttezza dell'interfacciamento tra moduli diversi andando a controllare la conformità a determinati requisiti (detti *contratti*) dei dati di ingresso e di uscita da un'applicazione.
La programmazione orientata agli aspetti consente di applicare il controllo di aderenza a specificati contratti tra applicazioni in modo flessibile [11]; è possibile porre sotto controllo ogni singolo metodo di un'applicazione, e definire politiche di monitoraggio diverse a seconda dell'ambiente sotto osservazione (sviluppo, test, collaudo, produzione).

4.4. Gestione della sicurezza

La tematica della sicurezza è anch'essa centrale nello sviluppo di applicazioni aziendali; anche in questo caso la programmazione orientata agli oggetti consente di progettare un modulo indipendente per la gestione della sicurezza, ma è normalmente compito degli sviluppatori interfacciarsi correttamente ad esso nello scrivere le funzionalità di business.
Grazie alla programmazione orientata agli aspetti è possibile mettere in atto una più netta *separation of concerns*, facendo in modo che un esperto di sicurezza sviluppi indipendentemente un modulo dedicato, dichiarando esplicitamente (all'interno di specifici *aspetti* di sicurezza) quali regole vadano applicate ed in quali punti dell'applicazione; grazie a questo approccio, gli sviluppatori delle funzionalità di business sono in grado scrivere il loro codice senza mai trattare esplicitamente le problematiche di sicurezza.
L'applicazione di AOP consente alla sicurezza non solo di verificare i diritti di un certo profilo utente ad eseguire determinati servizi, metodi, costruttori, ma anche di gestire correttamente il verificarsi di errori critici, applicare algoritmi di crittografia a dati sensibili oppure segnalare tentativi di accesso anomali al sistema [12].

5. CONCLUSIONI

La programmazione orientata agli aspetti rappresenta una grossa opportunità per imprimere un'ulteriore evoluzione agli strumenti ed alle modalità con cui viene sviluppato il software. Il suo maggiore punto di forza consiste nel fatto che è possibile introdurre questo nuovo paradigma a fianco del tradizionale processo di sviluppo in maniera graduale, cominciando ad apprezzarne i benefici senza sostenere investimenti iniziali eccessivi. Per sfruttare appieno le sue caratteristiche innovative è comunque necessario investire in un cambio di paradigma che consentirà di integrare meglio queste nuove teorie nel processo di progettazione del software, in modo da poter identificare già in fase di analisi quale sia l'approccio più efficace per indirizzare ogni *concern* relativo ad una applicazione.
Non mancano tuttavia le incognite che ne possono rallentare l'adozione: la mancanza di uno standard unitario innanzitutto e di conseguenza anche il non perfetto supporto di queste nuove tecnologie all'interno delle piattaforme software (application server, middleware) sviluppate dalle più importanti società.

Bibliografia

[1] Succi Giancarlo: L'evoluzione dei linguaggi di programmazione: analisi e prospettive. *Mondo Digitale*, Anno II, n. 8, Dicembre 2003, p. 39-52.

[2] Kiczales Gregor, Lamping John, Mendhekar Anurag, Maeda Chris, Lopes Cristina, Loingtier Jean-Marc, Irwin John: *Aspect Oriented Programming*. Atti della European Conference on Object-Oriented Programming (ECOOP), Finland. Springer-Verlag LNCS 1241, 1997.

[3] Elrad Tzilla, Filman Robert E., Bader Atef: Aspect Oriented Programming. *Communication of the ACM*, Vol. 44, Ottobre 2001.

[4] Parnas David L.: On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, December 1972

[5] Laddad Ramnivas: *AspectJ in Action – Practical Aspect Oriented Programming*. Manning Publication Co., 2003

[6] *IBM to make aspect-oriented development a reality*. CNET News.com, http://news.com.com/2100-1008_3-5178164.html?tag=nefd_top

[7] *Sun's Gosling: New Java Flavors Brewing*. eWeek, http://www.eweek.com/article2/0,1759,1624844,00.asp

[8] *AOP Alliance*, http://aopalliance.sourceforge.net/

[9] Krishnamurthy *Ramchandar: Performance Analysis of J2ee Applications Using AOP Techniques*. ONJava.com, http://www.onjava.com/pub/a/onjava/2004/05/12/aop.html

[10] Meyer Bertrand: *Object Oriented Software Construction*. Prentice Hall, 1997.

[11] Diotalevi Filippo: *Contract enforcement with AOP*. DeveloperWorks, http://www-106.ibm.com/developerworks/java/library/j-ceaop/

[12] Viega John, Bloch J.T., Chandra Pravir: Applying Aspect-Oriented Programming to Security. *Cutter IT Journal*, Vol.14, n. 2, p. 31-39, 2001.

[13] Home page del progetto AspectJ, http://www.aspectj.org

[14] JBossAop, un tool alternativo ad AspectJ, http://www.jboss.org/developers/project/jboss/aop

[15] AspectWerkz, un tool alternativo ad AspectJ, http://aspectwerkz.codehaus.org

FILIPPO DIOTALEVI lavora come IT Specialist presso IBM Italia a Milano, occupandosi di tematiche di progettazione e sviluppo di applicazioni su architettura J2ee. È autore di articoli tecnici e divulgativi di argomento informatico apparsi su riviste e siti italiani ed internazionali.
filippo.diotalevi@it.ibm.com

SEGNALI TELEVISIVI DIGITALI: STANDARD DI CODIFICA E TRASMISSIONE

Alberto Morello
Vittoria Mignone
Paola Sunna

La diffusione di programmi televisivi all'utente finale avviene tramite l'utilizzo di diversi sistemi diffusivi (satellite, cavo, digitale terrestre ecc.). I sistemi diffusivi si basano sul compromesso di garantire un'elevata qualità di servizio, utilizzando *bit-rate* e potenze limitate. Le tecniche per la riduzione del *bit-rate* sono identificate come la *codifica di sorgente* e *la codifica di canale*. Questo articolo presenta una rassegna degli standard di codifica e trasmissione consolidati e introduce le più recenti novità del settore.

3·4

1. PREMESSA

La digitalizzazione del segnale televisivo e la disponibilità di nuovi mezzi trasmissivi ha creato, rispetto al passato, un'estesa gamma di servizi per la diffusione di programmi televisivi all'utente finale; in particolare tra i sistemi diffusivi oggi disponibili vale la pena di ricordare quelli via satellite, via cavo (con scarso interesse per l'Italia ma usati in molti altri Paesi), i recenti sistemi di *broadcasting* terrestre e i più recenti sistemi in ADSL e fibra che si appoggiano sul protocollo IP, ma che non sono stati finora completamente standardizzati.
L'ottimizzazione di un sistema diffusivo si basa essenzialmente sul compromesso di garantire un'elevata qualità di servizio (bassa probabilità di errore sul segnale al ricevitore) pur utilizzando *bit-rate* e potenze limitate. Nella teoria delle comunicazioni le tecniche per la riduzione del *bit-rate* sono identificate come *codifica di sorgente*: il loro scopo è eliminare nel modo più efficiente possibile tutte le ridondanze presenti nel segnale audio-video originale, in particolare le *ridondanze spaziali* a livello di singolo quadro e le *ridondanze temporali* tra quadri successivi di una sequenza televisiva. Per questa ragione gli *standard di codifica* basati sull'eliminazione della ridondanza del segnale originario vengono spesso denominati *standard di compressione*.
L'ottimizzazione del sistema trasmissivo propriamente detto si basa, invece, sulla scelta di un'opportuna *codifica di canale* (con codici correttori di errori che aumentano leggermente la ridondanza per ridurre drasticamente la probabilità di errore) e del *sistema di modulazione* più appropriato alle caratteristiche di propagazione del mezzo trasmissivo.
Gli studi condotti nell'ambito della codifica di canale e di sorgente hanno originato e continuano a dar vita a *standard* caratterizzati da prestazioni sempre migliori che non sempre però sono adottati a livello universale e tanto meno garantiscono la compatibilità con terminali di utente già esistenti.
Va sottolineato che, nella pratica corrente, s'identifica col *nome di standard di codifica* tutta la famiglia degli standard di codifica di sorgente e con il nome di *standard di trasmissione* la famiglia degli standard per i di-

versi mezzi trasmissivi (e le loro evoluzioni) con riferimento alle scelte effettuate sui sistemi di modulazione e di codifica di canale.
Il presente articolo, oltre ad un richiamo storico e ad una rassegna dei più importanti standard ormai consolidati nei due campi sopra accennati, introduce le più recenti novità nell'ambito degli standard di codifica e degli standard di trasmissione.
Per i sistemi di codifica si fa riferimento essenzialmente a quelli studiati dall'Organismo di standardizzazione MPEG (*Motion Picture Expert Group*) che sono riusciti ad ottenere un consenso a carattere universale sia per quanto riguarda diversi settori merceologici (informatica, telecomunicazioni, broadcasting e consumer) sia a livello di copertura geografica.
Per i sistemi trasmissivi si considerano quelli studiati in ambito dell'organismo DVB (*Digital Video Broadcasting*) che, pur avendo caratteristiche essenzialmente europee, è riuscito a diffondere i propri standard in tutto il mondo (esiste, infatti, un analogo organismo anche negli Stati Uniti).

2. LA STORIA DEGLI STANDARD DI CODIFICA E TRASMISSIONE

Per comprendere gli sviluppi attuali della televisione digitale, è utile ripercorrere brevemente la storia, dagli esordi ad oggi, dei cambiamenti conseguenti alla digitalizzazione.
All'inizio degli anni '80 sembrava che l'evoluzione tecnologica del sistema radiotelevisivo negli Stati Uniti, in Europa e Giappone dovesse avviarsi verso un modello che privilegiava il miglioramento qualitativo dell'immagine, in termini di definizione e di resa cromatica. Notevoli investimenti venivano effettuati già da qualche anno in Europa e Giappone nella direzione dell'EDTV (*Enhanced Definition TV*); il Giappone, inoltre, da circa un decennio aveva impegnato enormi risorse nello sviluppo della TV ad alta definizione HDTV (*High Definition TV*) che sembrava lo strumento ideale per offrire all'utente la sensazione di partecipare all'evento (*effetto presenza*) combinando una visione perfetta su uno schermo panoramico con l'audio stereofonico surround.
Gli Stati Uniti, dal canto loro, avevano intravisto nel lancio dell'HDTV una favorevole occasione per rivitalizzare l'industria nazionale dell'elettronica di consumo, mentre l'Europa, rimasta pressoché assente dal comparto dell'informatica, non intendeva lasciarsi sfuggire di mano anche quello della televisione. Le soluzioni *full digital* erano allora ai primi passi, con la standardizzazione dei formati video di studio (è degli anni '80 la raccomandazione ITU-R BT.601) e dei sistemi di codifica video di tipo PCM, richiedenti velocità di trasmissione molto elevate (dell'ordine di 200 Mbit/s) anche per segnali a definizione convenzionale.
La diffusione digitale all'utente sembrava ancora un'utopia e le proposte, al cui sviluppo lavorava l'industria - MUSE (*MUltiple Subsampling Encoding*) in Giappone, MAC (*Media Access Control*) e HDMAC (*High Definition MAC*) in Europa - erano, di fatto, sistemi analogici assistiti da flussi di informazione digitale (fra cui l'audio con qualità CD).
La svolta radicale si ebbe nel 1990 (dopo tre anni di studi e sperimentazioni) quando un consorzio formato da RAI, Telettra, RTVE (l'allora Ente Pubblico radiotelevisivo spagnolo) e Politecnico di Madrid nell'ambito del progetto europeo EUREKA 256 dimostrò la fattibilità di un sistema di trasmissione interamente digitale di TV ad alta definizione, durante i campionati mondiali di calcio in Italia del 1990. Oltre 16 partite (tre da Milano, quattro da Napoli, due da Torino, una da Firenze, una da Bari e le restanti da Roma, compresa la finale) furono trasmesse in diretta, attraverso il satellite sperimentale italiano Olympus, in salette appositamente attrezzate presso le sedi RAI, con circa 800 spettatori ciascuna, ottenendo un grande consenso dal pubblico presente. Fu possibile anche collegarsi con Barcellona, limite di copertura del satellite, e trasmettere via fibra ottica il segnale dei campionati del mondo fino a Madrid dove era stata allestita una sala apposita. In contemporanea il sistema fu portato negli Stati Uniti in concorrenza ad un sistema presentato dalla NEC, riscotendo un notevole interesse.
Il rivoluzionario sistema ("siamo riusciti a far passare un cammello dalla cruna di un ago...!", disse allora il Direttore Generale della Telettra nella conferenza stampa di presentazione) era basato sull'utilizzo della trasformata matematica DCT (*Discrete Cosine Transform*) che avrebbe costituito, quattro anni più tardi, la base essenziale dello standard MPEG-2, accettato an-

che dagli Stati Uniti [1]. Il sistema studiato permetteva di trasmettere i segnali HDTV sulla stessa banda satellitare (70 Mbit/s ridotti l'anno dopo a 34 Mbit/s) richiesta dai sistemi analogici MUSE e HDMAC, ma offriva una qualità video di gran lunga superiore. Inoltre, a parità di risoluzione dell'immagine, permetteva di ridurre di circa una decade la potenza trasmessa dal satellite, mandando di fatto in pensione sul nascere i grandi e costosi satelliti nazionali per la radiodiffusione.

Se, nel 1990, l'evento sembrò ignorato dalla comunità dei radiodiffusori e dell'industria elettronica di consumo europea, tuttavia in breve tempo la televisione digitale per l'utente domestico divenne una realtà.

Il passo successivo fu compiuto con la *standardizzazione del sistema di codifica video ISO-MPEG-2*, sotto la guida di un altro Centro Ricerche Italiano, lo CSELT. Questo sistema, orientato al mercato di massa, permetteva ulteriori riduzioni della banda trasmissiva (circa 5 Mbit/s per programma televisivo a definizione convenzionale e circa 19 Mbit/s per programma HDTV) e *concentrava la complessità sul codificatore* per ridurre i costi dei ricevitori. Quando i chip per la ricezione MPEG-2 furono disponibili sul mercato, fu l'operatore americano *DirecTV* a lanciare un servizio a pagamento di televisione digitale via satellite, abbandonando l'idea dell'alta definizione. In tal modo la compressione del segnale video nata per trasmettere programmi ad alta qualità, si avviò rapidamente ad essere impiegato per moltiplicare il numero di programmi trasmissibili nella larghezza di banda di un canale in cui era allocato, in precedenza, un solo programma analogico.

L'Europa reagì rapidamente, creando nel 1992 il *Digital Louncing Group*, che diventò in breve il progetto DVB (*Digital Video Broadcasting*). Dopo aver speso alcuni mesi per studiare un sistema di TV/HDTV terrestre, il Gruppo per lo studio del *DVB* comprese la grande opportunità di business della TV digitale via satellite, dietro la spinta degli operatori della TV a pagamento: l'idea dell'alta definizione fu abbandonata anche a causa della mancanza di televisori commerciali di grandi dimensioni, a favore della possibilità di trasmettere su un unico canale molti programmi a definizione convenzionale.

Fu ancora il Centro Ricerche RAI a svolgere un ruolo fondamentale ed ad assumere il coordinamento dell'attività di definizione del primo standard di trasmissione per TV digitale: ingegneri del Centro presiedettero il gruppo di specialisti del consorzio DVB che in sei mesi, da giugno a dicembre 1993, definì lo *standard di trasmissione DVB-S* [2], una pietra miliare per la diffusione satellitare su base mondiale.

Il sistema per la televisione digitale terrestre DTT (*Digital Terrestrial TV*) europea fu introdotto con lo standard denominato *DVB-T* [3] e nacque circa due anni dopo, in diretta concorrenza con il sistema americano ATSC (*Advanced Television Systems Committee*): il primo a definizione normale e multicanale, particolarmente adatto alla ricezione in condizioni critiche (anche portatile), il secondo ad alta definizione, a programma singolo e meno robusto dal punto di vista trasmissivo. È dal trasmettitore Rai di Torino Eremo che fu diffuso nel 1998 il primo segnale DTT in Italia. La televisione digitale terrestre può oggi rappresentare un'importante frontiera per i *broadcaster* italiani di servizio pubblico e commerciale. Il 2004 ha visto la partenza dei servizi dei grandi operatori nazionali e di alcune emittenti locali e molti nuovi programmi digitali appositamente studiati per tale applicazione cominciano a raggiungere gli utenti. Una novità, oltre all'aumento del numero dei programmi, è costituita dalle applicazioni di TV interattiva basate sulla piattaforma a standard MHP (*Multimedia Home Platform*): servizi di pubblica utilità per il cittadino, *super-teletext*, votazione a distanza all'interno dei programmi televisivi, pubblicità interattiva, giochi, servizi bancari. La televisione potrà anche gradualmente offrire all'utente la navigazione Internet, senza perdere tuttavia le proprie caratteristiche di elettrodomestico di facile utilizzo per tutti.

Ma la tecnologia della televisione digitale è giunta veramente al capolinea? Certamente no e questo articolo intende sintetizzare i recenti progressi tecnologici, individuando i temi principali su cui si stanno impegnando i laboratori di tutto il mondo nell'ambito dello studio dei nuovi sistemi di codifica e di trasmissione, sempre sotto il coordinamento di MPEG e del DVB.

3. GLI STANDARD DI CODIFICA DEL SEGNALE VIDEO CON PARTICOLARE RIFERIMENTO AI NUOVI SISTEMI IN VIA DI DEFINIZIONE

3.1. Richiamo degli standard esistenti di codifica

La trasmissione completa dell'informazione contenuta nel segnale televisivo numerico, descritto nella Raccomandazione ITU-R BT.601, richiederebbe la generazione di un flusso binario caratterizzato da un *bit-rate* estremamente elevato, con un notevole svantaggio rispetto alle diffusioni in tecnica analogica. Si hanno, infatti, 720×576 campioni (*pixel*) per quadro per la componente di luminanza e 360×576 campioni per quadro per ciascuna delle due componenti di crominanza nonché 25 quadri al secondo e 8 bit per campione, il *bit-rate* necessario risulta perciò di:

$$720 \times 576 + 2\,(360 \times 576) \times 25 \times 8 = 166 \text{ Mbit/s}$$

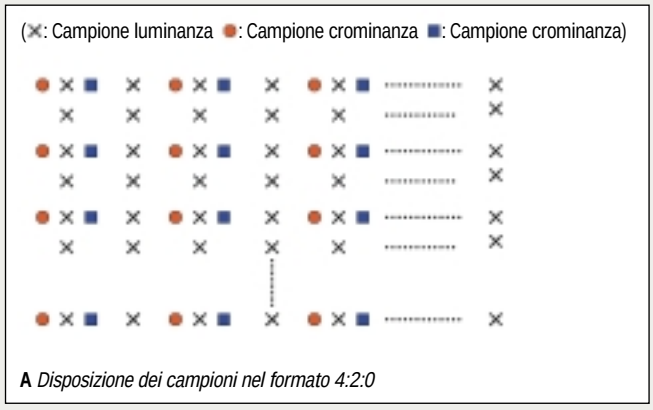
Indipendentemente dal tipo di modulazione adottato, la banda occupata dal suddetto segnale sarebbe notevolmente superiore alla capacità di un canale di trasmissione tradizionale da cui la necessità imprescindibile di adottare sistemi in grado di comprimere l'informazione originaria al fine di ridurre la banda occupata.

Allo scopo di definire un sistema standard per la codifica delle immagini in movimento, nel gennaio 1988 fu costituito l'Organismo MPEG (*Motion Picture Expert Group*) come Gruppo di esperti dell'ISO/IEC che fu il vero e proprio Organismo di standardizzazione. Il primo standard prodotto fu MPEG-1 la cui applicazione tipica era prevista in campo multimediale per la codifica video e audio e la memorizzazione su CD-ROM. Attualmente MPEG-1 è caduto in disuso. Lo standard successivamente approvato in MPEG fu MPEG-2 che supporta la codifica del video nel **formato** 4:2:2 e 4:2:0.

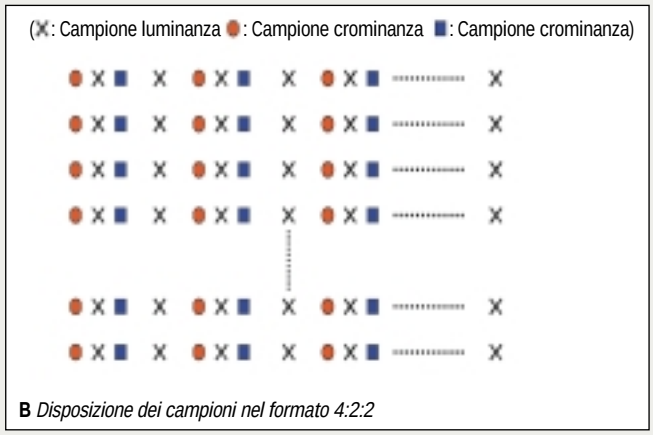
L'algoritmo MPEG-2 effettua la compressio-

Formati di crominanza. Come già noto dall'esperienza televisiva analogica a colori il sistema visivo umano risulta essere più sensibile alla definizione della luminanza rispetto alla definizione della crominanza; in ambito analogico questa caratteristica aveva indotto gli sperimentatori a limitare la banda disponibile per le informazioni di crominanza rispetto a quella disponibile per le informazioni di luminanza: il sistema PAL analogico assegna infatti una banda di 5 MHz al segnale di luminanza e 1,3 MHz ai segnali di crominanza. In ambito numerico questa scelta si traduce nell'esecuzione di un filtraggio della sequenza di campioni di crominanza allo scopo di ridurne il numero rispetto a quello dei campioni di luminanza introducendo, in tal modo, una perdita di informazione che non pregiudica eccessivamente la qualità percepita dall'utente. Il campionamento delle componenti di crominanza rispetto a quelle di luminanza dà origine ai cosiddetti *formati di crominanza*. Due formati di crominanza comunemente utilizzati in ambito televisivo sono il 4:2:0 ed il 4:2:2. Nel formato 4:2:0 le matrici di crominanza *Cr* e *Cb* associati a ciascun quadro hanno dimensioni pari a metà della corrispondente matrice *Y* di luminanza sia orizzontalmente che verticalmente, ossia le componenti di crominanza del segnale analogico originario sono state campionate orizzontalmente e verticalmente con frequenze pari a metà di quella di luminanza; come si può osservare dalla figura **A**, orizzontalmente i campioni di crominanza sono associati a campioni alterni di luminanza mentre verticalmente esse occupano righe alterne.

Nel formato 4:2:2, le matrici *Cr* e *Cb* associate a ciascun quadro hanno dimensione pari a metà della corrispondente matrice *Y* di luminanza soltanto orizzontalmente mentre verticalmente hanno la stessa dimensione, ossia le componenti di crominanza del segnale analogico originario sono campionate a frequenza spaziale orizzontale pari a metà della frequenza della luminanza e a frequenza spaziale verticale pari a quella di luminanza; come si osserva dalla figura **B**, infatti, i campioni di crominanza sono associati a campioni alterni di luminanza orizzontalmente mentre verticalmente non viene saltata nessuna riga.



A *Disposizione dei campioni nel formato 4:2:0*



B *Disposizione dei campioni nel formato 4:2:2*

ne dell'informazione video secondo quanto riportato qui di seguito:

- compressione senza perdita di informazione basata sullo sfruttamento della ridondanza spaziale (correlazione tra *pixel* adiacenti nel quadro), della ridondanza temporale (correlazione tra quadri/semiquadri nel tempo) e *sull'utilizzo di codici a lunghezza variabile*, VLC (*Variable-Length Code*);
- compressione con eliminazione dell'irrilevanza, ossia di quell'informazione non più ricostruibile dal decodificatore, ma non percepibile dal sistema visivo umano (codifica psico-visiva);
- compressione con perdita di informazione che si verifica nel caso in cui ridondanza e irrilevanza non siano sufficienti a ottenere la riduzione di *bit-rat*e desiderata. L'informazione scartata, non più ricostruibile dal ricevitore, è percepita dal sistema visivo umano come un degradamento dell'immagine.

L'algoritmo MPEG-2 utilizza la trasformata DCT (*Discrete Cosine Transform*) per la *riduzione della ridondanza spaziale*; l'applicazione della DCT, infatti, abbassa notevolmente la correlazione spaziale tra i pixel del quadro e rende non uniforme la distribuzione statistica dei livelli dei campioni favorendo la successiva *codifica con codici a lunghezza variabile*, dopo la quantizzazione. Poiché i coefficienti DCT sono legati al contenuto spettrale dell'immagine e poiché la sensibilità dell'occhio non è uniforme per tutte le frequenze e per tutte le direzioni rispetto all'orizzontale, la codifica psico-visiva, in MPEG-2, è realizzata predisponendo opportune matrici di quantizzazione. I valori sono assegnati in modo da provocare una quantizzazione più grossolana dei coefficienti che corrispondono alle alte frequenze e alla direzione diagonale per le quali la sensibilità dell'occhio umano è inferiore.

Per quanto riguarda lo *sfruttamento della ridondanza temporale*, MPEG-2 migliora l'approccio predittivo tramite la *compensazione del movimento* che consente di determinare lo spostamento "locale" in termini di ampiezza, direzione e verso delle singoli porzioni costituenti l'immagine. In MPEG-2, la predizione può essere "pura" se avviene tra immagini successive una rispetto all'altra o "interpolativa" se viene utilizzata sia un'immagine precedente che una successiva a quella corrente.

Lo standard MPEG-2 è strutturato secondo profili e livelli, per ciascuno dei quali viene specificato il *bit-rate* massimo che il decodificatore deve essere in grado di elaborare (Figura 1). Riferendosi all'esempio numerico riportato all'inizio del paragrafo, si può notare, per esempio che scegliendo il MainProfile@MainLevel il flusso video originario può essere co-

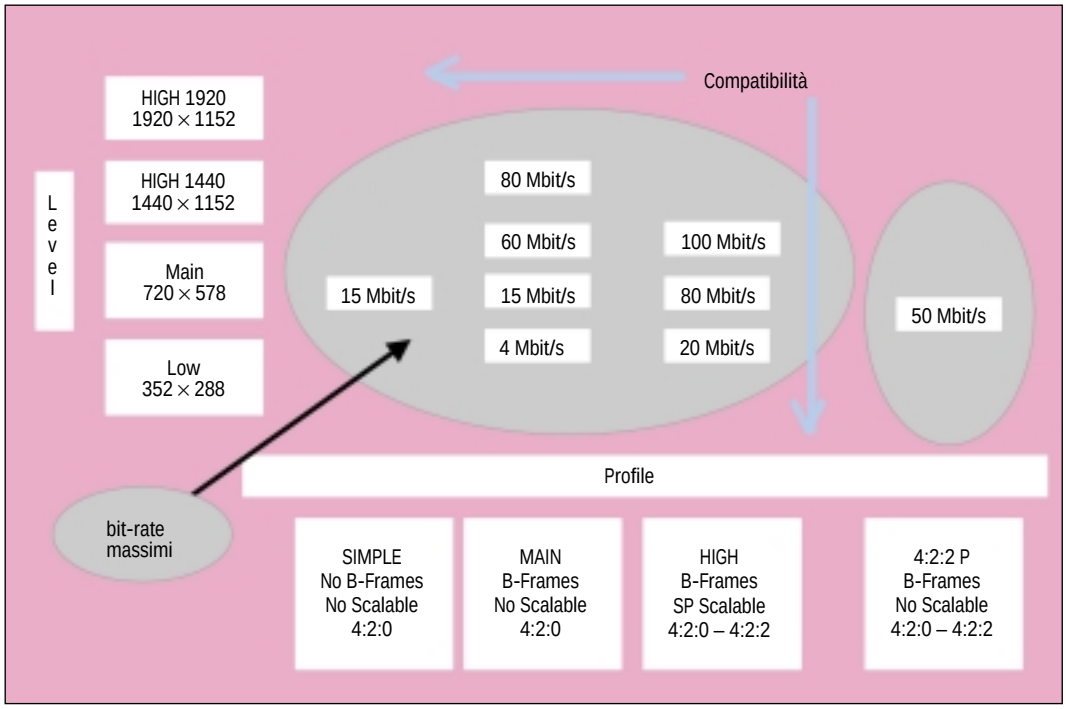


FIGURA 1
Profili e livelli in MPEG-2

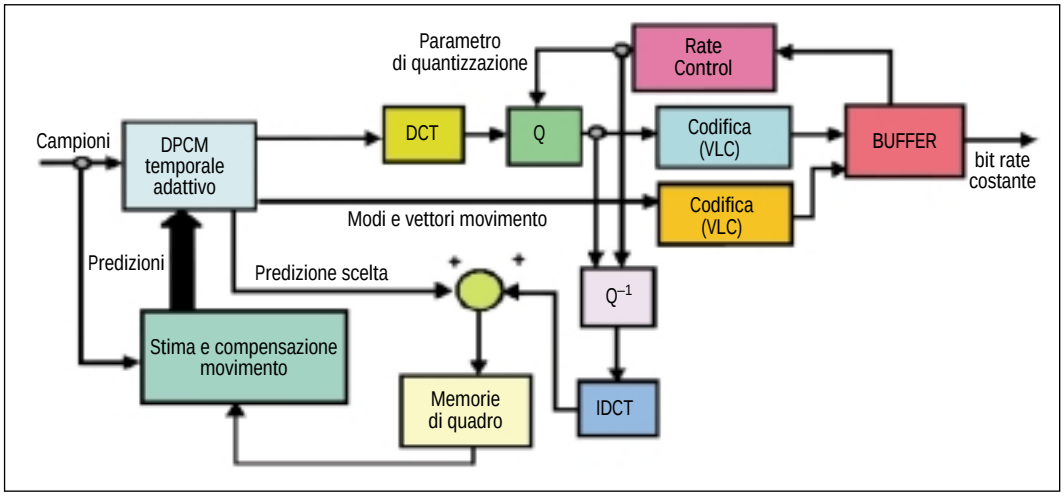


FIGURA 2
Schema a blocchi di un codificatore MPEG-2

dificato fino ad un *massimo* di 15 Mbit/s che corrisponde ad un fattore di compressione pari a 11. Codifiche a *bit-rate* inferiori corrispondono a compressioni maggiori.
La figura 2 riporta invece lo schema di principio di un codificatore MPEG-2.
Il blocco denominato DPCM temporale adattativo calcola gli errori di predizione come differenza tra l'immagine sorgente corrente e l'immagine predetta in base all'immagine precedente (*predizione pura*) o all'immagine precedente e a quella successiva (*predizione interpolativa*). L'adattatività del processo consiste nel fatto che il codificatore può scegliere di codificare i coefficienti sorgente invece degli errori di predizioni se questi sono eccessivamente elevati; il blocco predittivo è completato da stima e compensazione del movimento.La presenza delle memorie di quadro è pertanto necessaria al codificatore per memorizzare i riferimenti per le predizioni. Sui coefficienti della matrice da codificare sono eseguite le operazioni di trasformazione DCT, di quantizzazione e di codifica a lunghezza variabile. La parte retroattiva dello schema serve a ricostruire i riferimenti per le predizioni, analogamente a quanto avviene sul lato ricevitore. Il buffer ha un'importanza fondamentale in quanto consente di ottenere all'uscita del processo di codifica un flusso a *bit-rate* costante; il controllo del *bit-rate* è attuato tramite lo la *variazione scalare* dei valori di quantizzazione in funzione del riempimento del buffer.
Dal momento in cui è stato normalizzato, il sistema MPEG-2 ha avuto una rapida penetrazione (DVB, ATSC, DVD Forum ...) ed è alla base della maggior parte dei sistemi a livello mondiale di diffusione del segnale video sia per applicazioni *broadcasting* che *webcasting* (diffusione su Internet).
Un cenno a parte merita anche lo standard MPEG-4[1] Parte 2 pubblicato dall'Ente di standardizzazione ISO nel 1999. Come nel caso di MPEG-2, l'efficienza di codifica di questo algoritmo è strettamente dipendente dalle caratteristiche del materiale sorgente e dall'implementazione sul lato codificatore. MPEG-4 è stato studiato per applicazioni legate alla codifica multimediale di contenuti audio-video a bassi *bit-rate*, ma in seguito il campo di applicazione è stato esteso anche al *broadcasting*; le valutazioni soggettive eseguite allo scopo di individuare l'efficienza di codifica di MPEG-4 Parte 2 rispetto a MPEG-2 hanno evidenziato un guadagno del primo sul secondo dell'ordine del 15-20%, ma per il Grup-

[1] Possono anche essere ricordati gli standard MPEG-7 e MPEG-21 che però non si riferiscono alla compressione dei contenuti audio-video, ma piuttosto ne definiscono le modalità di utilizzo. In particolare, MPEG-7 fornisce gli strumenti per la descrizione dei contenuti multimediali audio e video allo scopo di facilitarne estrazione, indicizzazione e gestione, principalmente sui motori di ricerca mentre MPEG-21 è orientato verso la creazione di un "Multimedia Framework" ed opera sull'integrazione di strumenti e linguaggi diversi per la definizione di un ambiente, nel quale interagiscono compressione, protezione dei diritti e possibilità di riconoscere e reperire i contenuti.

po DVB l'efficienza di MPEG-4 non è stata ritenuta tale da giustificare un'eventuale sostituzione di MPEG-2 dato che MPEG-4 non è compatibile con MPEG-2.
Lo standard MPEG-2, come la maggior parte degli standard di codifica, definisce, esclusivamente la sintassi del *bit-stream*, e quindi del decodificatore, lasciando pertanto ai costruttori diversi gradi di libertà nell'implementazione dell'algoritmo presente nel codificatore. I codificatori MPEG-2 esistenti sul mercato presentano pertanto prestazioni differenti a seconda, ad esempio, del tipo di algoritmo utilizzato per la stima del movimento, dei valori nelle matrici di quantizzazione, del tipo di controllo effettuato sul *bit-rate* in funzione della complessità spazio-temporale del segnale originario, ma producono tutti un flusso dati compatibile con un ricevitore MPEG-2.
Se un costruttore agisse, invece, su elementi come il tipo di trasformata, la dimensione dei blocchi elementari in fase di codifica, il tipo di codifica con codici a lunghezza variabile, VLC (*Variable-Length Code*), il numero di quadri utilizzati per la compensazione del movimento produrrebbe un *bit-stream* e quindi un nuovo sistema di compressione non più compatibile con un decoder MPEG-2 e con prestazioni differenti da quelle di MPEG-2.
È proprio allo scopo di creare un nuovo sistema di compressione caratterizzato da un'elevata efficienza di codifica che, nel 2001, gli Organismi di standardizzazione ISO[2]/IEC (MPEG) e ITU[3], e in particolare per quest'ultimo il VCEG (*Video Coding Expert Group*), costituirono il JVT (*Joint Video Team*) e cioè un gruppo di lavoro congiunto per la definizione di un sistema avanzato di codifica, denominato AVC (*Advanced Video Coding*).

3.2. I nuovi sistemi di codifica H.264 e VC.9

Il sistema AVC specifica la codifica del video, VCL (*Video Coding Layer*) ed il formato con cui organizzare i dati video per il trasporto e la memorizzazione, NAL (*Network Abstraction Layer*). Nel 2003 l'AVC è stato integrato come Parte 10 dello standard MPEG-4 ISO/IEC 14496-10 (pur essendo totalmente diverso come finalità dall'MPEG-4 originale) e con il nome di H.264 in ITU [4], denominazione che è conveniente utilizzare per evidenziarne le caratteristiche innovative. L'approvazione finale congiunta da parte di ISO e ITU era prevista per giugno 2003 ma è slittata a ottobre 2004. Lo standard AVC, così come avviene nel caso di MPEG-2, definisce la sintassi del flusso dati ed il metodo di decodifica.
Come detto precedentemente, lo standard AVC non produce un *bit-stream* compatibile con MPEG-2. L'adozione di AVC richiederà quindi l'utilizzo di nuovi apparati sia per la codifica che per la decodifica.
Il nuovo standard prevede quattro profili, rivolti ad applicazioni differenti:
- *Baseline Profile*, destinato ad applicazioni a basso ritardo *end-to-end*, applicazioni mobili, videotelefonia;
- *eXtended Profile*, per applicazioni mobili e per *streaming*;
- *Main Profile*, rivolto ad applicazioni diffusive SDTV (*Standard Definition TeleVision*).
- FREXt (*Fidelity Range Extensions*)Profiles per applicazioni professionali di contribuzione, editing in studio e HDTV (*High Definition Tlevision*).

Lo schema a blocchi di riferimento per il codificatore H.264 è riportato nella figura 3. Come

[2] ITU (*International Telecommunication Union*): è un Organismo internazionale che ha il compito di governare e coordinare tutte le attività e i servizi attinenti alle tecnologie di telecomunicazione. Costituita nel lontano 1865 come Convenzione Telegrafica Internazionale da 20 nazioni e dal 1934 mutatasi in International Telecommunication Union, ITU seguì tutta l'evoluzione delle telecomunicazioni, dal telegrafo al telefono, fino alle trasmissioni radio, nell'etere, su cavo o i recenti sistemi ottici e satellitari. Dal 1947 ITU è un'Agenzia specializzata delle Nazioni Unite. Oggi ai lavori dell'ITU partecipa la quasi totalità dei Paesi i del mondo.

[3] ISO (*International Standardization Organization*): costituita nel 1947, l'ISO è una Federazione non governativa che abbraccia oltre 130 Enti normatori di altrettante nazioni a livello mondiale.L'ISO promuove lo sviluppo e l'unificazione normativa per consentire e facilitare lo scambio dei beni e dei servizi. Coordina l'ambiente scientifico, tecnologico ed economico e fissa riferimenti vincolanti per una vastità di settori quali informatica, meccanica, elettrica ...I lavori dell'ISO sono il risultato di lunghi accordi internazionali e danno luogo a *International Standard*. I Paesi aderenti all'accordo, tramite i singoli comitati di standardizzazione nazionali, s'impegnano a introdurre gli "International Standard" nelle corrispondenti norme nazionali.

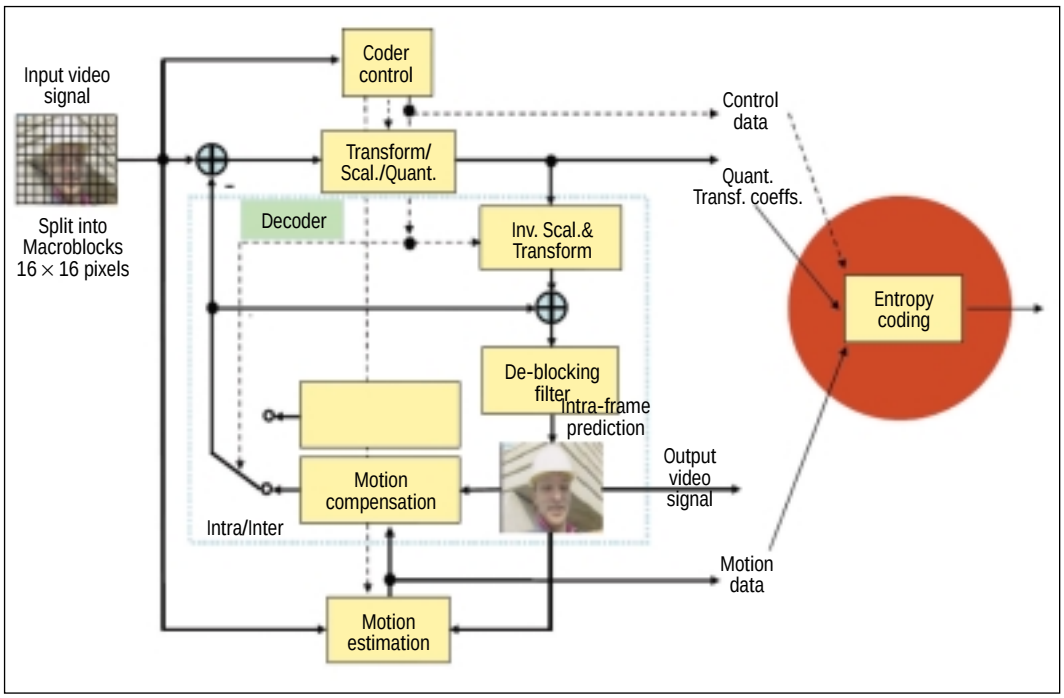


FIGURA 3
Schema a blocchi di un codificatore H.264
(Fonte: Heinrich Hertz Institut di Berlino)

si può notare la struttura è molto simile a quella riportata nella figura 2 per il codificatore MPEG-2 se si esclude la presenza del blocco *de-blocking filter*. Nonostante la somiglianza, l'efficienza di codifica di un dispotico realizzato secondo la norma H.264 è nettamente superiore a quella di MPEG-2 per le differenze riportate qui di seguito.

1. *Compensazione del movimento*

- H.264 utilizza blocchi di dimensione e forma variabile rispetto al blocco di dimensione fissa 16 × 16 di MPEG-2 realizzando, in questo modo, un risparmio di *bit-rate* che può arrivare fino al 15%.
- La precisione nella stima dei vettori movimenti in H.264 è più precisa che in MPEG-2 (¼ di *pixel* in H.264 contro ½ in MPEG-2) e consente di ridurre il *bit-rate* necessario alla codifica fino al 20%.
- H.264 utilizza fino a un massimo di cinque quadri per la stima del movimento contro i due di MPEG-2 con un guadagno di *bit-rate* compreso tra il 5 e il 10%.

2. *Riduzione della ridondanza spaziale*

H.264 utilizza una trasformata intera invece della DCT allo scopo di ridurre la perdita di precisione in seguito alla trasformata inversa.

3. *Quantizzazione*

H.264 utilizza un maggiore numero di livelli di quantizzazione: 52 contro i 31 di MPEG2.

4. *Codifica entropica*

H.264 utilizza tecniche più complesse quali CAVLC (*Context-Adaptive Variable Length Code*) e CABAC (*Context-based Adaptive Binary Arithmetic Coding*), ma più efficienti rispetto all'uso di tabelle VLC (*Variable Lenght Code*) statiche di MPEG-2.

5. *De-blocking filter* [5]

Lo standard H.264/MPEG-4 AVC, a differenza di quello MPEG-2, utilizza per il deblocking un filtro adattativo che consente di migliorare le aberrazioni visive (presenza di una struttura a blocchi che permea tutta l'immagine) dovute alla perdita di dati per effetto di un processo di compressione molto spinto, quale quello effettuato in questo caso sul segnale video.

H.264, a differenza di MPEG-2, utilizza questo filtro adattativo allo scopo di ridurre l'effetto della blocchettizzazione sulla sequenza decodificata, effetto che, com'è noto, pregiudica drasticamente la qualità percepita dall'utente finale.

La maggiore efficienza (definibile come la riduzione di *bit-rate* ottenibile a parità di qualità soggettiva) di H.264 rispetto a MPEG-2 si paga però in termini di aumento della complessità sia del codificatore che del decodificatore, come risulta evidente dalle indicazioni riportate in tabella 1.

Profilo	Stima preliminare dell'efficienza rispetto a MPEG-2	Aumento della complessità stimata per il decodificatore H.264*
Baseline	Circa 1,5	Circa 2,5 volte
Extended	Circa 1,75	Circa 3,5 volte
Main	Circa 2	Circa 4 volte

* Il codificatore è circa otto volte più complesso

TABELLA 1
Aumento di complessità e efficienza passando da MPEG-2 a H.264/MPEG-4 AVC (Fonte: www.m4if.org)

L'elevata efficienza di compressione fa sì, inoltre, che gli ambiti di applicazione del sistema H.264 siano estremamente versatili; ad esempio, nelle reti mobili, Organizzazioni come il 3GPP (3rd *Generation Partnership Project*) hanno affiancato ai sistemi di codifica H.263 e MPEG-4 anche l'H.264 nel profilo *Baseline*, data la limitata disponibilità di banda che caratterizza le reti di terza generazione.
L'H.264 si pone come candidato interessante anche per applicazioni in reti a larga banda (xDSL, fibra ottica) in cui il segnale video è fruibile in modalità *streaming, download o video on demand* su un PC o su un televisore.
Per quanto riguarda la diffusione televisiva con definizione standard, l'efficienza del 50% rispetto a MPEG-2 ha fatto sì che l'H.264 venisse preso in considerazione anche in ambito DVB. Al momento della redazione di questo articolo, infatti, un gruppo di lavoro DVB sta redigendo le Raccomandazioni tecniche per l'inclusione dello standard H.264 come sistema opzionale per la diffusione televisiva digitale affianco a MPEG-2. Sempre lo stesso Gruppo sta lavorando alle specifiche per la trasmissione del flusso codificato H.264 sia su TS (*Transport Stream* formato di trasporto di dati definito dallo standard ISO/IEC 13818) che su IP (*Internet Protocol*). L'adozione di H.264 consentirebbe infatti di aumentare la capacità in termini di numero di programmi disponibili nei *bouquet* dei sistemi diffusivi DVB a fronte però di nuovi investimenti legati alla necessità di sostituire l'attuale parco di STB (*Set Top Box*) MPEG-2 con nuovi apparati in grado di decodificare entrambi i sistemi di compressione.
Infine, la diffusione presso l'utente finale dei display a schermi piatti (Plasma, LCD,....) ha riaperto tutta una serie di problematiche inerenti alla trasmissione della HDTV. In particolare, i *bit-rate* necessari ad ottenere una qualità trasparente per l'HDTV si aggirano, nel caso di codifica MPEG-2, intorno ai 18-20 Mbit/s un solo canale HDTV occuperebbe perciò l'intero canale trasmissivo dei sistemi di trasmissione DVB. Le prestazioni dello standard H.264 risultano pertanto interessanti nel caso in cui il *broadcaster* decidesse di affiancare l'HDTV alla SDTV.
Anche il gruppo di lavoro DVB-H, per lo studio dello standard di diffusione televisiva verso i sistemi mobili e di cui si dirà nel seguito, sta considerando la possibilità di utilizzare per la *mobile television* l'H.264 sia il profilo *baseline* che il *main* data la maggiore banda disponibile rispetto ai sistemi 3G. È importante completare questa panoramica ricordando che sul mercato si sono affacciati sistemi di codifica proprietari e concorrenti dello standard H.264, come per esempio *Real Video, On2, Sorenson*.
Un cenno a parte merita il codificatore WM9 (*Windows Media 9*) [6], basato su tecnologia Microsoft, che fino agli inizi del 2004 apparteneva alla categoria dei sistemi di codifica proprietari. Nel marzo del 2004, Microsoft ha presentato all'Organismo SMPTE (*Society for Motion Picture and Television Engineers*) il documento *Proposed SMPTE Standard for Television: VC-9 compressed video bitstream format and decoding Process* per la standardizzazione del proprio sistema di codifica, di cui WM9 non è altro che un'implementazione.
L'approvazione finale dello standard è prevista per dicembre 2004. La tabella 2 riporta il confronto tra MPEG-2, H.264 e VC9 in termini di caratteristiche degli algoritmi di compressione. Dall'analisi delle caratteristiche riportate nella tabella 2, il VC9 e l'H.264 sembrano molto simili. Al momento della redazione di questo articolo non si dispone di dati che forniscano un'idea dell'efficienza e della complessità del sistema VC9 rispetto ad H.264. La Microsoft ha comunque affermato che le *prestazioni* di VC9 sono confrontabili con quelle dello standard H.264 mentre la *complessità* di VC9 è inferiore a quella di H.264 e confrontabile con quella di MPEG-2.

	MPEG-2	H.264	VC9
Profili	❑ Simple ❑ Main ❑ High ❑ 4:2:2	❑ Baseline ❑ Main ❑ Extended ❑ FREXt	❑ Simple ❑ Main ❑ Advanced
Input	❑ Interlacciato ❑ Progressivo	❑ Interlacciato ❑ Progressivo	❑ Interlacciato ❑ Progressivo
Tipo di algoritmo	Ibrido (ridondanza spaziale/temporale)	Ibrido (ridondanza spaziale/temporale)	Ibrido (ridondanza spaziale/temporale)
Struttura di codifica	Gerarchica	Gerarchica	Gerarchica
Tipi di immagine	I, P, B	I, P, B, SP, SI	I, P, B, BI
Struttura di macroblocco	❑ 16 × 16 *Y* ❑ 8 × 8 *Cb* ❑ 8 × 8 *Cr*	❑ 16 × 16 *Y* ❑ 8 × 8 *Cb* ❑ 8 × 8 *Cr*	❑ 16 × 16 *Y* ❑ 8 × 8 *Cb* ❑ 8 × 8 *Cr*
Accuratezza nella stima del movimento	Fino a $^1/_2$ pixel	Fino a $^1/_4$ pixel	Fino a $^1/_4$ pixel
Dimensioni del blocco minimo su cui è applicata la compensazione del movimento	8 × 8	4 × 4	4 × 4
Loop *filter* per ridurre l'effetto della blocchettizzazione	Assente	Presente	Presente
Tipo di trasformata	DCT	Intera	Intera
Codifica entropica	VLC	❑ CAVLC ❑ CABAC	VLC

TABELLA 2
Confronto tra le caratteristiche di compressione degli algoritmi VC9, AVC e MPEG-2

4. GLI STANDARD TRASMISSIVI E LA LORO ARCHITETTURA CON PARTICOLARE RIFERIMENTO AI NUOVI SISTEMI IN VIA DI DEFINIZIONE

4.1. Architettura dei sistemi diffusivi DVB

I sistemi DVB per la diffusione televisiva (satellite DVB-S, via cavo DVB-C, terrestre DVB-T), sono caratterizzati da una struttura comune, schematicamente rappresentata in figura 4. In realtà con la denominazione DVB si intende la struttura completa impiegata per la diffusione dei segnali che include la codifica di sorgente e la multiplazione: tuttavia, poiché gli standard di codifica adottati sono quelli MPEG, il lavoro del Gruppo DVB si è concentrato essenzialmente sugli standard di trasmissione.
La *codifica di sorgente e la multiplazione si basano sullo standard* MPEG-2, che genera in uscita un segnale di multiplazione di trasporto con pacchetti di lunghezza fissa di 188 byte (1 byte di sincronismo, 3 di prefisso - contenenti gli identificatori di pacchetto – e 184 byte utili). Il multiplex è flessibile e consente di convogliare in un singolo flusso numerico segnali relativi a un gran numero di programmi televisivi, ciascuno comprendente le relative informazioni video, audio e dati.
Canali di servizio aggiuntivi sono inoltre previsti per indicare i vari programmi inseriti all'interno del multiplex (*Service Information,* SI), per attuare l'accesso condizionato (*Conditional Access,* CA), per fornire una guida elettronica dei programmi (*Electronic Program Guide*, EPG).
L'adattamento al canale trasmissivo (in par-

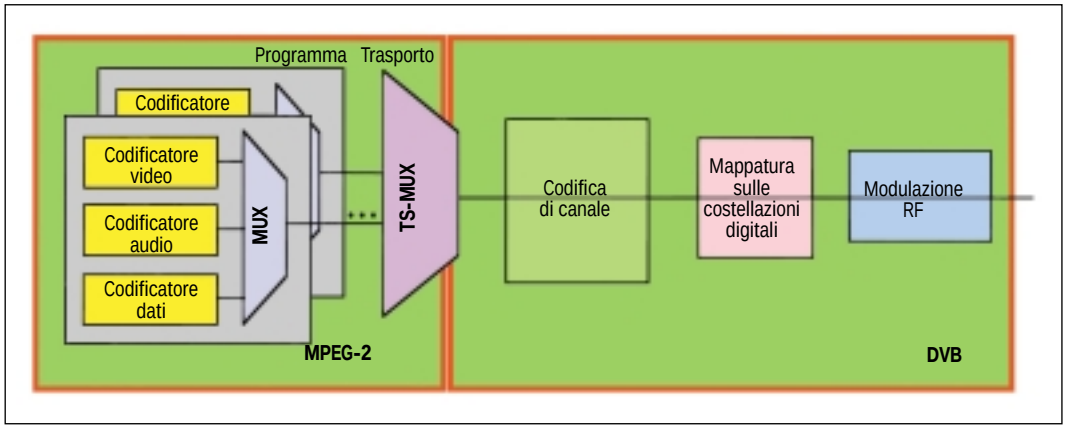


FIGURA 4
Schema a blocchi generale di un sistema di diffusione DVB

ticolare la scelta della codifica di canale e del tipo di modulazione da adottare) è invece stato sviluppato dal Gruppo di studio DVB, ed è specifico di ogni mezzo per adeguarsi al tipo di disturbi da contrastare. Ciononostante il DVB ha voluto mantenere la massima comunanza possibile fra i tre principali sistemi diffusivi standardizzati e in particolare:

- per i sistemi DVB–S e DVB-T la protezione contro gli errori è realizzata mediante la concatenazione di un *codice esterno a blocco* di tipo Reed-Solomon (che opera sui pacchetti MPEG-2 da 188 byte) e di un *codice interno convoluzionale* (con possibilità di scegliere tra diversi rapporti di codifica da 1/2 a 7/8); il sistema DVB-C, molto meno critico, adotta il solo codice esterno Reed-Solomon;
- le costellazioni digitali adottate per la modulazione sono: per il DVB-S la QPSK (*Quadrature Phase Shift Keying*), costituita da quattro punti posti su un cerchio, molto robusta e caratterizzata da un inviluppo costante (e quindi particolarmente idonea per gli amplificatori non lineari usati nei satelliti); per il DVB-T la QPSK assieme alle modulazioni 64QAM e 16QAM (*Quadrature Amplitude Modulation*), costituite da punti posti su una griglia regolare, con efficienza spettrale crescente con l'aumentare del numero di punti; per il DVB-C la QPSK, con le modulazione 16QAM, 32QAM e 64QAM.
- la modulazione a radiofrequenza del segnale è invece fortemente dipendente dal canale di trasmissione: i sistemi DVB-S e DVB-C sono a singola portante, il sistema DVB-T è invece **multiportante** COFDM per contrastare la propagazione multicammino (*multipath*) del segnale in ambiente terrestre (come i codici correttori con opportune strategie proteggono dagli errori *a burst*, così la modulazione multicanale protegge dai fading selettivi che possono "distruggere" porzioni dello spettro occupato dal segnale).

4.2. Il sistema di trasmissione via satellite di seconda generazione DVB-S2

4.2.1. Generalità sul DVBS-2

Nel 2003, a dieci anni dalla nascita del DVB, è stato standardizzato il sistema di seconda generazione per la trasmissione via satellite, DVB-S2 [7], erede del sistema di diffusione della televisione digitale da satellite DVB-S, attualmente usata dalla maggior parte degli operatori satellitari nel mondo.

Il sistema DVB-S2 è stato progettato per varie applicazioni satellitari a larga banda: servizi diffusivi di TV a definizione standard (SDTV) e ad alta definizione (HDTV), applicazioni interattive per l'utenza domestica e professionale, compreso l'accesso a Internet, servizi professionali di contribuzione TV e SNG (*Satellite News Gathering*), distribuzione di segnali TV a trasmettitori digitali terrestri VHF/UHF, distribuzione di dati e di siti Internet (*Internet trunking*).

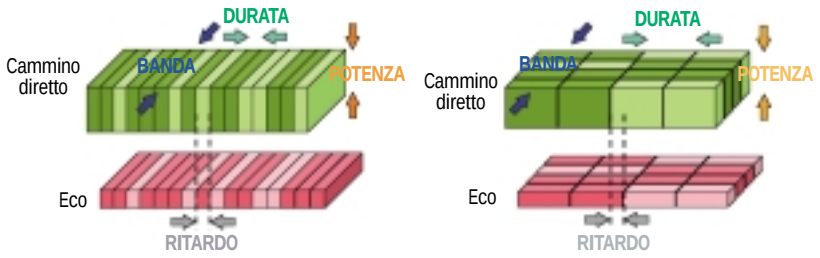
Sono tre i concetti chiave in base cui lo standard DVB-S2 è stato definito: *maggiore capacità trasmissiva* rispetto ai sistemi di prima generazione ed in particolare al DVB-S, *totale flessibilità e ragionevole complessità del ricevitore*.

Per ottenere il bilanciamento tra prestazioni

La modulazione **multiportante** COFDM (*Coded Orthogonal Frequency Division Multiplexing*) permette di suddividere il flusso di informazione a velocità di R_s in un numero elevato n di flussi a velocità R_s/n, e anteporre al simbolo OFDM un *intervallo di guardia* temporale (con durata T_g pari ad una frazione di quella di simbolo utile T_u) che separa simboli OFDM adiacenti. L'intervallo di guardia è la continuazione ciclica della parte utile T_u del simbolo e viene scartato dal ricevitore, cosicché gli echi che raggiungono il ricevitore con un ritardo τ inferiore a T_g non generano interferenza intersimbolica (*Inter Symbol Interference*, ISI).

In aggiunta all'intervallo di guardia, il sistema *COFDM* fa uso di un potente schema di *correzione degli errori* che permette il recupero dell'informazione trasportata da quelle portanti che sono state attenuate dal canale di trasmissione.

La mutua ortogonalità è garantita dalla spaziatura in frequenza $1/T_u$ tra le portanti pari alla velocità di simbolo R_s. Il processo OFDM è attuato per mezzo di una I-FFT (*Inverse Fast Fourier Trasform*).



Confronto fra la modulazione a singola portante e quella multiportante

Il sistema DVB-T è dotato di due modalità operative: 2K, con FFT su 2008 portanti di cui 1705 attive per le reti di diffusione convenzionali multifrequenza (MFN, *Multy Frequency Network*) e durata utile del simbolo di 224 μs, e 8 K, con FFT su 8192 portanti di cui 6817 portanti attive e durata utile del simbolo di 896 μs per operare anche su reti a singola frequenza (SFN, *Single Frequency Network*) e consentire un'utilizzazione ottimale dello spettro. Quattro diverse opzioni sono previste per l'intervallo di guardia, precisamente 1/32, 1/16, 1/8, 1/4 della durata utile di simbolo, con valori compresi tra 7 e 224 μs.

La modulazione COFDM permette quindi di contrastare echi con ritardi molto elevati, indipendentemente dal livello di potenza del segnale principale ed è idonea per il canale terrestre. Anche il canale via cavo è caratterizzato da echi, che però sono generalmente molto meno critici, sia come durata che come potenza relativa. In tal caso le modulazioni classiche a singola portante con equalizzazione al ricevitore sono sufficienti per contrastare gli echi presenti sul canale di trasmissione. L'equalizzatore è generalmente una linea a prese di ritardo di lunghezza pari ad alcune volte il massimo ritardo che si vuole equalizzare, e funziona bene in presenza di echi corti (qualche μs), pressoché stabili nel tempo e attenuati rispetto al segnale principale (almeno 5-6 dB).

e complessità, il DVB-S2 si avvale dei più recenti sviluppi nella codifica di canale e nella modulazione. La codifica di canale è basata sui codici LDPC (*Low Density Parità Check*), una famiglia di codici a blocco molto semplici, con struttura algebrica molto limitata, scoperti nel 1960, ma soltanto oggi utilizzabili nei prodotti consumer, grazie ai progressi della tecnologia dei microcircuiti[4]. Quattro sono i tipi di modulazione presenti nella norma DVB-S2: QPSK, 8 PSK, 16 APSK (*Amplitude Phase Shift Keying*), 32 APSK[5].

L'adozione nel DVB-S2 di queste tecniche innovative di codifica e modulazione garantisce un aumento di capacità dell'ordine del 30 per cento rispetto al DVB-S operando in modo CCM (*Constant Coding & Modulation*, letteralmente Modulazione e Codifica Costanti), ossia con parametri di trasmissione fissi: il DVB-S non prevedeva, infatti, di poter cambiare i parametri durante la trasmissione.

Nelle applicazioni punto-punto, come ad esempio l'*IP Unicast*, il guadagno del DVB-S2 rispetto al DVB-S può essere ancora maggiore. La funzione ACM (*Adaptive Coding & Modulation*, letteralmente Modulazione e Codifica Adattative) permette infatti di variare lo schema di modulazione ed i livelli di protezione dagli errori per ogni nuovo blocco elementare di codifica, ottimizzando il sistema di trasmissione alle condizioni di ricezione dell'utente. Per comunicare al tra-

[4] La massima complessità ammessa per il *decoder* era fissata in 14 mm^2 di silicio con tecnologia 0,13 μm, e la velocità di simbolo di riferimento di 55 MBaud. Dal 2004 per i prodotti elettronici consumer è disponibile una tecnologia di 0,09 μm, che dovrebbe ulteriormente aumentare il numero di componenti integrabili riducendo la complessità dell'intero ricevitore.

[5] I parametri delle modulazioni 16 APSK e 32 APSK sono state ottimizzati per operare su un trasponditore non lineare, collocando i vari punti sul bordo di cerchi; le prestazioni fornite da un modulatore su un canale lineare sono paragonabili rispettivamente con quelle delle modulazioni 16 QAM e 32 QAM.

smettitore le condizioni di ricezione del singolo utente, il sistema deve operare "ad anello chiuso", utilizzando un canale di ritorno per esempio via telefono o satellite.Il DVB-S2 è così flessibile da adattarsi a tutti i tipi di trasponditori satellitari esistenti, grazie ad un'ampia varietà di efficienze spettrali e di rapporti segnale/rumore C/N (*Carrier to Noise*), richiesti. Esso, inoltre, è progettato per trattare una grande varietà di formati audio-video e di dati, dall'MPEG-2 oggi utilizzato negli standard DVB, a quelli che il progetto DVB sta ora definendo per le applicazioni future: H264 e VC9 (si veda il paragrafo 3). Il sistema DVB-S2 si adatta a qualunque formato di flusso di dati in ingresso, compresi flussi digitali MPEG-TS (*Transport Stream*), singoli o multipli, o flussi in formato IP e ATM. Questo fa sì che anche se in futuro saranno definiti altri formati, essi potranno essere impiegati senza bisogno di modificare il sistema.
Il nuovo sistema DVB-S2 non è purtroppo compatibile con i ricevitori oggi esistenti. Per permettere ai *broadcaster* di attuare una transizione graduale, mantenendo in esercizio i sistemi DVB-S attuali[6] e contemporaneamente aumentando la capacità trasmissiva per i servizi dedicati ai nuovi ricevitori DVB-S2, lo standard DVB-S2 definisce, in maniera opzionale, modalità trasmissive compatibili con il sistema DVB-S. In particolare, potranno essere presenti due flussi di dati TS, il primo ad alta priorità (*High Priority*, HP), compatibile con i ricevitori DVB-S attuali e con i nuovi ricevitori DVB-S2, il secondo a bassa priorità (*Low Priority*, LP), compatibile soltanto con i ricevitori DVB-S2. Questi flussi vengono inviati su un singolo canale satellitare, multiplati in modo sincrono a livello di simbolo di modulazione su una costellazione 8PSK non uniforme, con livelli diversi di protezione dagli errori (modulazione gerarchica). Poiché il segnale risultante ha un inviluppo quasi-costante, esso può essere trasmesso su un singolo *transponder*, portato quasi alla saturazione.
In alternativa lo standard propone (senza definirne le modalità trasmissive) l'utilizzo delle modulazioni stratificate (*Layered Modulation*), dove un segnale DVB-S2 e uno DVB-S sono combinati in modo asincrono sul canale a radio-frequenza, con il segnale DVB-S trasmesso a un livello di potenza assai più elevato del DVB-S2. Poiché il segnale risultante mostra sensibili variazioni di inviluppo, in questo caso esso deve essere trasmesso su un trasponditore quasi-lineare, lontano dalla saturazione[7].
Il sistema DVB-S2 è strutturato come una "scatola di attrezzi" (in inglese *tool-kit*), e cioè come un insieme di tecniche che permettono di coprire tutte le aree applicative, realizzabili in *single-chip* con complessità ragionevole, per permettere di utilizzare prodotti destinati al mercato di massa anche per applicazioni professionali.
Esso è composto da una sequenza di blocchi funzionali, come illustrato nella figura 5 e descritto in dettaglio in [6].
Il blocco identificato come *Adattatore di modo e di flusso* fornisce l'interfaccia per il flusso di ingresso, strumenti opzionali richiesti per l'ACM (per esempio per la sincronizzazione e la cancellazione dei pacchetti nulli nel caso di flussi di ingresso del tipo TS) e inserisce la codifica CRC (*Cyclic Redundancy Check*) per permettere al ricevitore di rivelare la presenza di errori nel flusso ricevuto. Inoltre, nel caso di ingressi multipli, esso unisce i flussi di ingresso (*merger*) per poi suddividerli (*slicer*) in blocchi del codice FEC (*Forward Error Correction*). Questi ultimi sono composti da bit presi da una sola porta di ingresso da trasmettere in modo omogeneo (con la stessa modulazione e con il codice FEC).

[6] Il successo ottenuto dallo standard DVB-S tradizionale ha portato a un'elevata diffusione dei STB (*Set Top Box*) DVB-S per la ricezione a casa dell'utente.

[7] Come alternativa, per meglio sfruttare le risorse di potenza del satellite, i segnali HP e LP possono essere trasmessi indipendentemente sulla tratta in salita del collegamento satellitare (*up-link*), amplificati ciascuno da un amplificatore da satellite indipendente (HPA, *High Power Amplifier*) portato vicino alla saturazione, ed essere combinati sulla tratta di discesa (*down-link*). Ciò richiede però la progettazione ed il lancio di una nuova generazione di satelliti.

Si inserisce poi l'intestazione di *banda base* (80 bit) davanti al campo dati per informare il ricevitore del formato del flusso di ingresso e del tipo di "adattamento" utilizzato. Nel caso i dati di utente disponibili per la trasmissione non siano sufficienti a riempire completamente il BBFRAME, blocco di banda base, si provvederà a completarlo con bit di riempimento. In ultimo, nel blocco denominato *stream adapter* il BBFRAME viene moltiplicato per una sequenza pseudocasuale (*scrambler*), che uniformemente distribuisce gli zeri e gli uno del BBFRAME, evitando la presenza di sequenze critiche per il codice FEC.
Il blocco *Codifica FEC* effettua la codifica concatenata del codice esterno BCH e del codice interno LDPC. I rapporti di codifica del codice LDPC interno sono 1/4, 1/3, 2/5, 1/2, 3/5, 2/3, 3/4, 4/5, 5/6, 8/9, 9/10, da scegliersi congiuntamente allo schema di modulazione in base ai requisiti del sistema. A seconda dell'area di applicazione i blocchi di codice FEC (FECFRAME), possono avere una lunghezza di 64,8 o 16,2 kbit. L'introduzione di due possibili valori è stata dettata da due esigenze opposte: le prestazioni in funzione del rapporto C/N migliorano al crescere della lunghezza dei blocchi di codifica, ma, al contempo, aumenta molto il ritardo globale della catena trasmissiva. Quindi, per applicazioni non critiche per i ritardi (come per esempio la diffusione di programmi), sono preferibili i blocchi lunghi, mentre per le applicazioni interattive un blocco più corto può essere più efficiente: infatti un pacchetto di informazione "corto" è immediatamente messo in onda dalla stazione trasmittente. La modulazione e il codice FEC sono costanti all'interno del FECFRAME, e possono cambiare in differenti FECFRAME nelle modalità VCM (*Variable Coding & Modulation*) e *ACM*. Il segnale trasmesso può anche contenere FECFRAME corti e normali.
Il blocco *Mapping* associa i bit alla costellazione: QPSK, 8PSK, 16APSK o 32APSK a seconda dell'applicazione. Tipicamente, per applicazioni *broadcast* vengono proposte le costellazioni QPSK e 8PSK, poiché sono di fatto modulazioni ad inviluppo costante e possono essere usate su *transponder* da satellite non lineari portati vicino alla saturazione. Le modalità 16APSK e 32APSK sono invece principalmente orientate ad applicazioni professionali; possono anche essere impiegate per il *broadcasting*, ma richiedono la disponibilità di un più elevato livello di C/N al ricevitore e l'adozione di metodi avanzati di pre-distorsione nella stazione di *up-link* per attenuare gli effetti di non-linearità del *transponder*. Sebbene non permettano efficienze di potenza analoghe agli schemi ad inviluppo costante, queste costellazioni offrono però una maggiore capacità trasmissiva.
Il blocco di *Generazione della trama PL* (*Physical Layer*), sincrono con i FECFRAME, gestisce l'inserzione dell'intestazione di livello fisico e dei simboli pilota opzionali (2,4% di perdita di capacità), di PLFRAME fittizi (*dummy frame*) in assenza di dati utili pronti per la trasmissione, e la moltiplicazione per

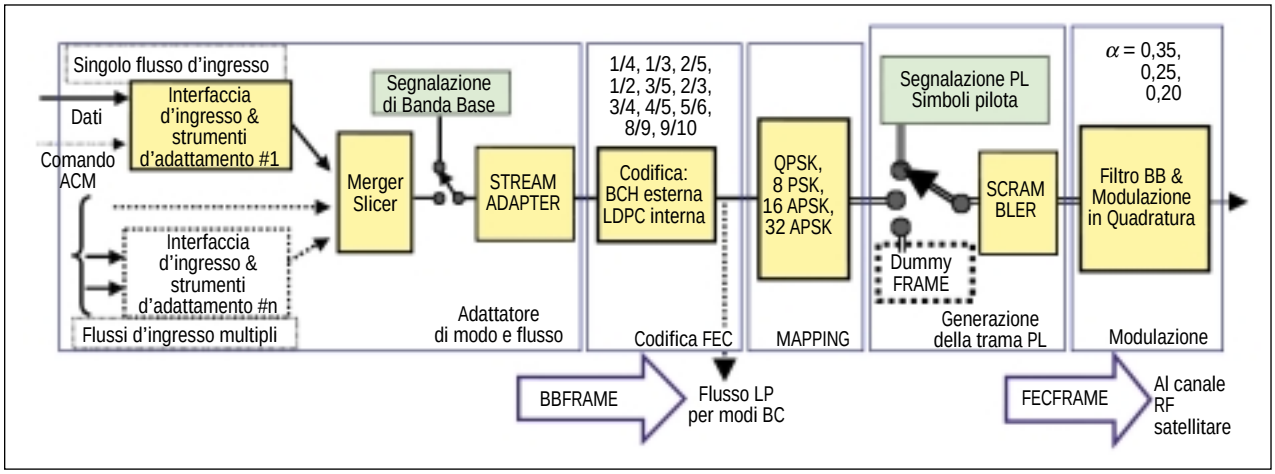


FIGURA 5

Schema a blocchi funzionale del sistema DVB-S2

una sequenza pseudo-casuale (*scrambler*) per la dispersione dell'energia.
Il *filtraggio* in banda base e la *modulazione* in quadratura si applicano per modellare lo spettro del segnale e per generare il segnale a radio frequenza. Il filtro usato in trasmissione è la radice quadrata del filtro a coseno rialzato con tre possibili coefficienti di *roll-off* α: 0,35 per continuità con il DVB-S, 0,25 e 0,20 per i casi in cui si abbiano maggiori limitazioni di banda.

4.2.2. Le prestazioni del sistema DVBS-2

Il DVB-S2 permette di selezionare lo schema di modulazione ed il tasso di codifica a seconda dei requisiti del servizio e delle caratteristiche del *transponder* per satellite impiegato. L'efficienza spettrale va da 0,5 bit/s/Hz, usando la modulazione QPSK 1/4, a 4,5 bit/s/Hz, usando la configurazione 32 APSK 9/10, mentre il rapporto segnale rumore passa da –2.4 dB a 16 dB (per canale gaussiano e demodulazione ideale), per garantire una ricezione QEF (*Quasi Error Free*, quasi priva di errori), definita per il DVB-S2 come la ricezione di meno di un pacchetto errato per ora di trasmissione per un servizio video a 5 Mbit/s, corrispondente a un tasso d'errore sul pacchetto (*Packet Error Rate*, PER) TS circa pari a 10^{-7}. Il grafico di figura 6 mostra l'efficienza spettrale del sistema DVB-S2 in funzione del rapporto C/N richiesto per ricezione QEF, riferito alla potenza media del segnale; esso è stato ottenuto attraverso simulazioni al calcolatore, su un canale affetto da rumore additivo Gaussiano bianco AWGN, e nel caso di una demodulazione ideale. Le linee tratteggiate si riferiscono al limite teorico (*limite di Shannon*), variabile in funzione del tipo di modulazione. Sulle ordinate è riportato il rapporto tra il *bit rate* utile R_u, e la velocità di trasmissione di simbolo R_s. Le curve non tengono conto delle distorsioni e dei degradamenti introdotti dal canale di trasmissione satellitare.

4.2.3. Esempi di possibili usi del sistema DVBS-2

4.2.3.1. Diffusione televisiva

L'utilizzo del DVB-S2 per la diffusione televisiva a definizione convenzionale SDTV in modalità *CCM*, come già visto, offre un guadagno in termini di capacità del DVB-S2 rispetto al DVB-S, dell'ordine del 30%. Il guadagno diventa ancora maggiore se lo si combina con la sostituzione della codifica MPEG-2 con quella H.264, riducendo drasticamente il costo per canale della capacità trasmissiva del satellite.
Il DVB-S2, inoltre grazie alla sua flessibilità, permette di differenziare la protezione contro gli errori per ogni multiplex (con modalità *VCM*): un'applicazione tipica è la trasmissione di un multiplex molto protetto contro gli errori per la televisione SDTV e di un multi-

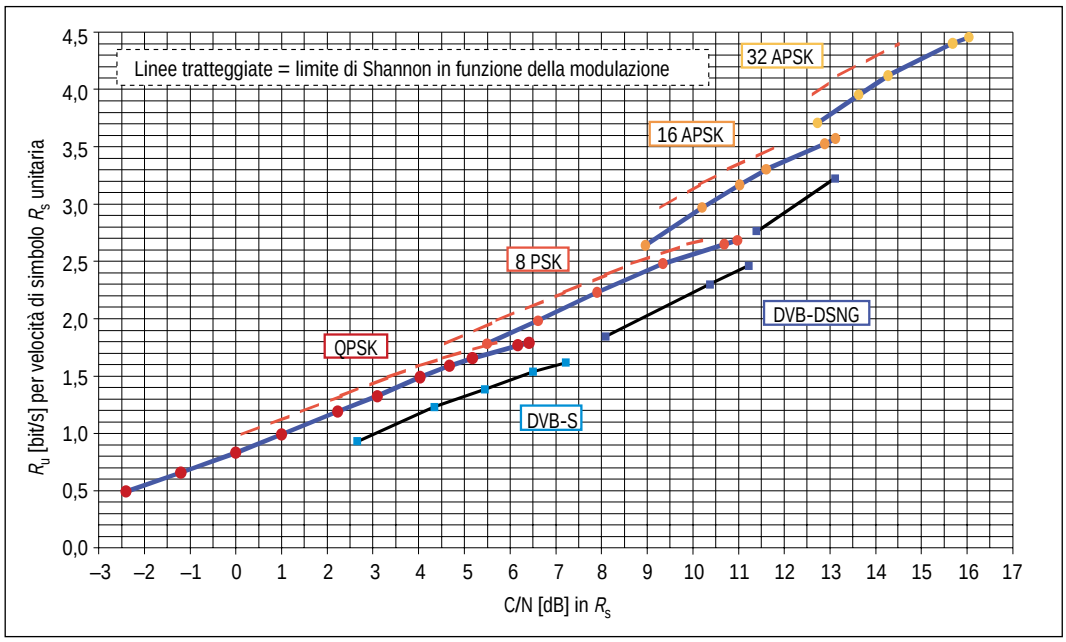


FIGURA 6
Efficienza spettrale in funzione del rapporto C/N richiesto su canale AWGN

plex meno protetto per servizi televisivi ad alta definizione HDTV.

4.2.3.2. Distribuzione del multiplex MPEG ai trasmettitori DTT

Molti Paesi stanno introducendo la televisione digitale terrestre (*Digital Terrestrial Television*, DTT) ed il satellite è uno dei mezzi candidati a distribuire i flussi MPEG ai trasmettitori digitali terrestri. I sistemi attualmente operativi si basano sul sistema DVB-S, che però permette la trasmissione di un singolo multiplex MPEG per segnale. Il risultato è che per la distribuzione di n multiplex MPEG, dovrebbero essere trasmesse n portanti per trasponditore satellitare, richiedendo perciò un elevato OBO (*Output Back Off*) dell'amplificatore satellitare, per un funzionamento quasi lineare (in alternativa all'uso di n trasponditori). L'adozione del sistema DVB-S2 permette di distribuire più multiplex MPEG, usando una configurazione a singola portante per trasponditore, ottimizzando così l'efficienza in potenza attraverso la saturazione dell'amplificatore del satellite.

4.2.3.3. Codifica e modulazione adattativa ACM per servizi punto-punto

Nelle applicazioni interattive punto-punto l'utilizzo della modalità adattativa ACM del DVB-S2 permette di recuperare il cosiddetto "margine a cielo chiaro" (tipicamente da 4 a 8 dB), inutilizzati[8] per la maggior parte del tempo nei collegamenti satellitari convenzionali impieganti schemi CCM (*Constant Coding and Modulation*), raddoppiando o addirittura triplicando così la capacità media del satellite e riducendo drasticamente i costi del servizio. Inoltre il guadagno dell'ACM rispetto al sistema a parametri fissi CCM aumenta in condizioni critiche di propagazione: quindi un sistema adattativo di tipo ACM è perciò fondamentale per le bande di frequenza più elevate (come per esempio la banda Ka) o per l'impiego in zone climatiche tropicali.

Il modulatore DVB-S2 ACM (Figura 7) opera ad una velocità di simbolo costante, poiché si assume costante la larghezza di banda del transponder. L'ACM è implementato dal modulatore DVB-S2 attraverso la trasmissione di una sequenza in multiplazione a divisione di tempo (*Time Division Multiplexing*, TDM) di sequenze di *frame* del livello fisico DVB-S2, dove il formato di codifica e modulazione possono cambiare ad ogni nuovo *frame*. La continuità di servizio è ottenuta, durante i periodi con forti attenuazioni da pioggia, riducendo il *bit rate* d'utente, e contemporaneamente aumentando la ridondanza del codice correttore FEC e/o la robustezza della modulazione.

Nei collegamenti punto-punto, dove un singolo segnale è inviato ad un'unica stazione ricevente -per esempio a una SNG (*Satellite News Gathering*), l'ACM permette di proteggere i pacchetti di dati seguendo le variazioni della qualità del collegamento tra la postazione trasmittente e quella ricevente, valutata in termini del rapporto C/(N + I) tra la potenza del segnale e quella del rumore e dei segnali interferenti sul canale satellitare. Analogamente sono impostabili

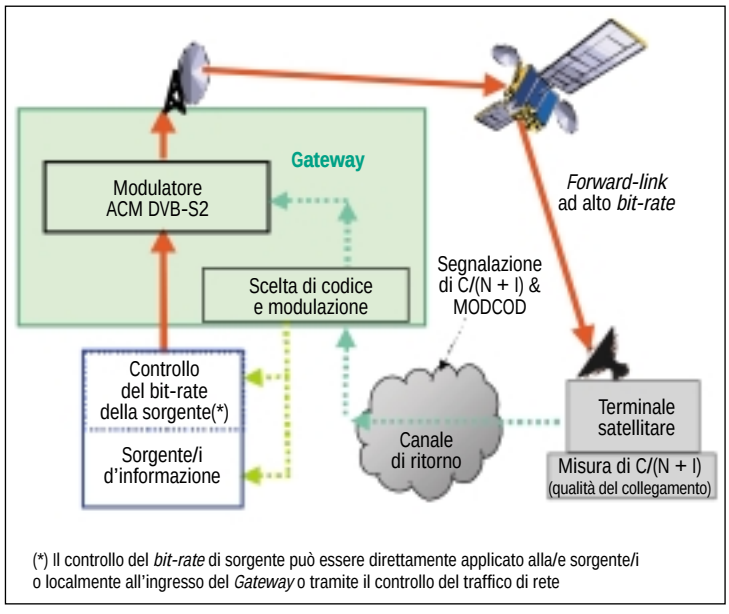


FIGURA 7

Diagramma a blocchi di un collegamento DVB-S2 in modalità ACM

[8] I margini del rapporto segnale-rumore C/N non producono aumenti di qualità del servizio (*Quality of Service*: QoS,) per l'utente, poiché la ricezione QEF (*Quasi Error Free*) è garantita anche alla soglia del rapporto C/N, ma essi servono a garantire la ricezione del servizio anche in presenza di perturbazioni atmosferiche sul collegamento.

in modalità ACM i collegamenti *IP Unicast*, con la differenza che la configurazione di protezione dagli errori deve essere scelta per ognuno degli utenti del servizio, tenendo conto che il numero di utenti può essere molto ampio (anche di centinaia di migliaia). I servizi dati interattivi possono trarre vantaggio dall'uso del DVB-S2 grazie alla possibilità di avere una protezione dagli errori non uniforme mediante l'ACM e livelli di servizio differenziati, come priorità nelle code di consegna, *bit-rate* minimo garantito. In questo caso sono però necessarie strategie di allocazione delle risorse di rete tra i vari utenti, per evitare sovraccarichi o ritardi di trasmissione.

4.3. Lo standard DVB-H per la ricezione mobile portatile terrestre di segnali televisivi

4.3.1. Generalità sul DVB-H

Negli ultimi anni il processo di convergenza tra servizi *broadcast* e *telecom* si sta espandendo con successo a tutti i settori delle telecomunicazioni, fisse e mobili, ed in modo particolare a quello dalla telefonia cellulare. L'avvento dei nuovi sistemi 2,5G/3G ha creato le basi per fare arrivare la TV all'utente sul telefonino, seppure i loro costi (insieme con altri problemi più di carattere tecnico, quali l'autonomia delle batterie, le dimensioni fisiche ed ergonomiche dei terminali, le dimensioni delle antenne, la protezioni ai disturbi) per il momento frenino il successo del servizio.
Le reti cellulari, infatti, sono caratterizzate da un elevato numero di stazioni base con copertura su aree di piccole dimensioni, per garantire all'utente alte capacità di traffico personalizzato bidirezionale. Sono perciò intrinsecamente più costose (anche per un elevato costo delle licenze) delle reti di diffusione terrestri radiofoniche e televisive, che usano trasmettitori di potenza elevata per coprire aree di servizio vaste, potenzialmente suddividendo su un grande numero di utenti il costo della rete.
In quest'ottica, il DVB ha istituito un Gruppo *ad hoc*, denominato DVB-H (dall'inglese *Handheld*, portatile), per studiare la possibilità di utilizzare il DVB-T per fornire servizi radiotelevisivi su terminali mobili portatili.
Il sistema DVB-T per la diffusione terrestre della televisione digitale era stato concepito per la ricezione statica (fissa e portatile). I progetti europei *Motivate* [8, 9] prima, *MCP* e *Drive* successivamente, hanno poi dimostrato la possibilità di estendere la ricezione all'ambiente mobile, utilizzando modi di funzionamento molto robusti (come per esempio le modulazioni QPSK o 16 QAM ed i rapporti di codifica 1/2 o 2/3) del sistema. Unico neo per la ricezione su portatili è l'elevato consumo di potenza del ricevitore, fattore non critico per la ricezione fissa e mobile in auto, che rende il sistema non idoneo alla ricezione con terminali portatili con alimentazione a batteria. Ciò è dovuto alla complessità degli algoritmi di codifica video e di protezione del segnale, ma soprattutto al fatto che la trasmissione secondo lo standard DVB-T avviene in modo continuo ed il ricevitore DVB-T deve rimanere sempre attivo ed elaborare tutto il segnale ricevuto per poter estrarre il servizio richiesto dall'utente.
Bisogna tra l'altro tenere presente che il DVB-T si basa su MPEG-2 per la codifica video, essendo dedicato alla visualizzazione su schermi televisivi molto più grandi di quelli dei terminali portatili, troppo onerosa per la qualità richiesta ai servizi DVB-H.
Il Gruppo ha pertanto prodotto uno standard [10] per la diffusione di servizi IP in formato compatibile con lo standard DVB-T per la ricezione portatile mobile, con i seguenti requisiti:

- minore consumo energetico rispetto al DVB-T per terminali alimentati a batteria;
- definizione di procedure di *hand-over*, per permettere all'utente di muoversi all'interno di una rete senza perdere la ricezione del servizio;
- flessibilità e scalabilità dei parametri del livello fisico del sistema, per ricezione nei vari ambienti (interni ed esterni agli edifici, urbano, rurale) ed a varie velocità (da pochi chilometri all'ora del pedone fino a centinaia di chilometri all'ora per i treni ad alta velocità);
- possibilità di utilizzare i sistemi più innovativi di codifica audio/video, descritti nel paragrafo 3.

4.3.2. Architettura di sistema DVB-H

Con riferimento alla pila protocollare OSI di figura 8, il sistema DVB-H il cui schema è riportato in figura 9 è caratterizzato dall'intro-

duzione al livello di connessione (*link*) di:

- un codice correttore d'errore denominato MPE-FEC[9] che rappresenta un livello di protezione aggiuntivo, di tipo Reed Solomon, sui dati in formato IP a livello MPE (*MultiProtocol Encapsulation*, standard EN 301 192);
- la **tecnica time slicing** – una suddivisione cioè dell'asse dei tempi in intervalli assegnati in modo predefinito ai vari servizi, per permettere al terminale ricevente di spegnersi quando i dati trasmessi, non appartengono al servizio richiesto.

Il DVB-H prevede inoltre, al livello fisico (*Physical Layer*), l'estensione opzionale dei modi di funzionamento del DVB-T con i seguenti elementi, propriamente definiti per il DVB-H:

- la segnalazione (con le portanti DVB-T dedicate alla segnalazione dei parametri di trasmissione), dei parametri DVB-H, tra cui l'identificativo di cella, per accelerare il rilevamento del servizio ed il processo di handover per i terminali in movimento;
- l'introduzione del nuovo modo di funzionamento 4K, compromesso fra le due soluzioni (2K e 8K) previste dal DVB-T, per migliorare la ricezione mobile;
- l'utilizzo dell'interallacciatore temporale (*time interleaving*) del modo 8K anche per i modi 2K e 4K, per aumentarne la tolleranza al rumore impulsivo.

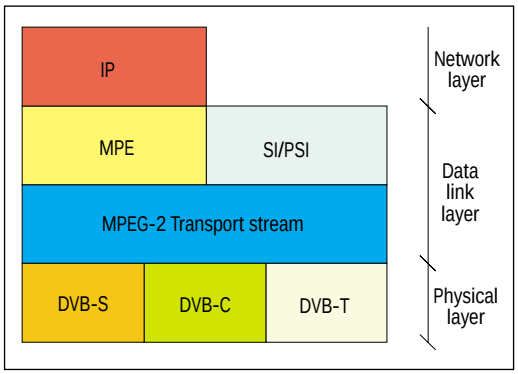


FIGURA 8
Pila protocollare OSI per la distribuzione di IP su DVB - livelli 1 ÷ 3

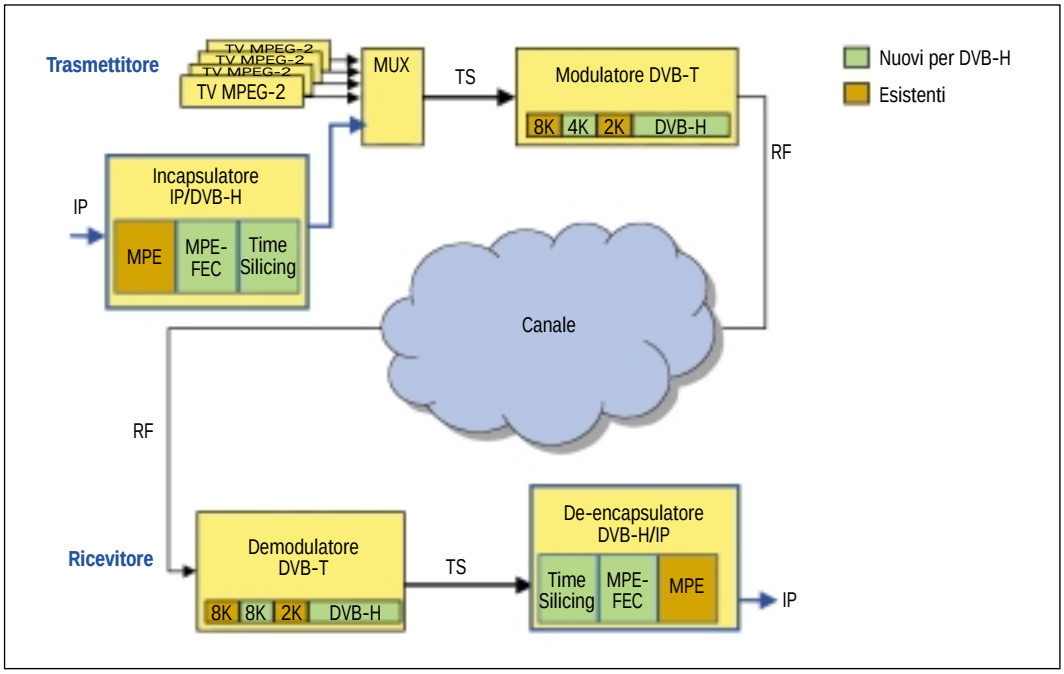
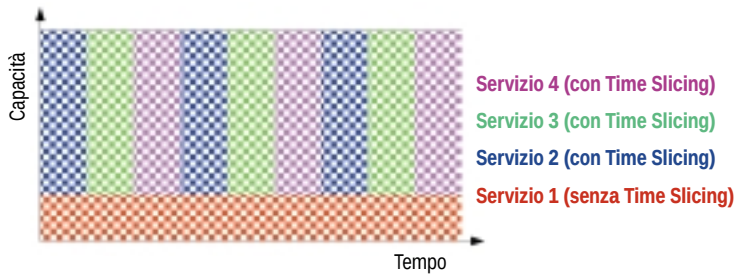


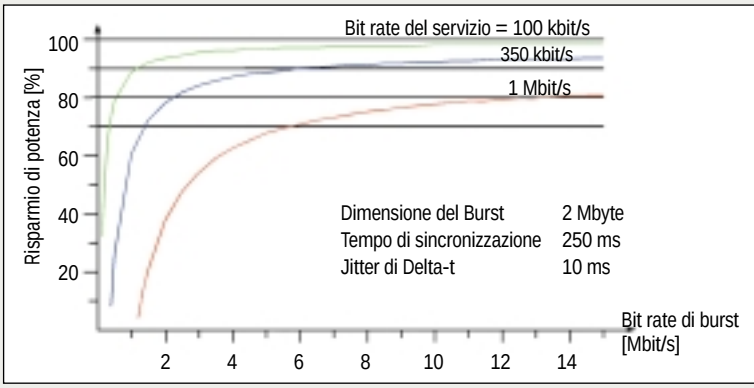
FIGURA 9
Schema a blocchi di principio del sistema DVB-H

[9] L'MPE-FEC (*MultiProtocol Encapulation Forward Error Correction*) permette di migliorare le prestazioni del sistema DVB-T in ambiente mobile e di ridurne la sensibilità all'interferenza impulsiva. L'informazione di parità calcolata dal codice RS (255,191,64) sui datagrammi IP, viene trasmessa in sezioni separate denominate MPE-FEC, per poter essere ignorata dai ricevitori non in grado di decodificarla. Il codice ha un'efficienza del 75% ed è in grado di correggere fino a 64 byte errati su 255 ricevuti nella configurazione base; massima flessibilità è però lasciata agli operatori di *aumentarne o ridurne l'efficienza agendo sui bit di informazione o parità*. Maggiori informazioni sull'impiego di MPE-FEC nelle applicazioni broadcasting si trovano in [11, 12].

La **tecnica di time-slicing** (Figura **A**) consiste nella trasmissione dei dati relativi ad un servizio raggruppati a *burst*, con una velocità istantanea del flusso molto più elevata (anche di una decade) di quella media richiesta per trasmettere il servizio secondo i metodi tradizionali. All'interno del flusso dati viene segnalato al ricevitore l'intervallo di tempo Ćt che intercorre prima dell'inizio del *burst* successivo. Nell'intervallo tra due burst successivi dello stesso servizio, la banda disponibile è utilizzata per altri servizi secondo lo stesso principio.
Contemporaneamente ai servizi, in *time-slicing* possono essere trasmessi anche servizi a flusso continuo. Il *time-slicing* permette al ricevitore di rimanere attivo per la sola frazione del tempo necessaria a ricevere i servizi richiesti (il trasmettitore è invece sempre attivo, essendo la trasmissione continua). L'assorbimento di potenza dipende dal *bit-rate* medio del servizio e dal *bit-rate* di picco e può essere ridotto anche del 90% rispetto ad un ricevitore tradizionale (Figura **B**).



A *Allocazione temporale dei servizi nei sistemi DVB-T e DVB-H*



B *Risparmio percentuale di potenza offerto dalla funzione di time-slicing del DVB-H*

MPE-FEC e *time-slicing* agiscono ai livelli OSI 2 e 3, e quindi non modificano lo standard DVB-T, permettendo ai ricevitori DVB-T tradizionali di interpretare correttamente il segnale, semplicemente ignorando la segnalazione DVB-H. Questo funzionamento va naturalmente a scapito di una perdita di efficienza trasmissiva, dell'ordine del 25%, variabile a seconda della configurazione DVB-H. Il modo 4K non può invece essere usato, se si vuole mantenere la compatibilità con il DVB-T.

4.3.3. Configurazioni di rete del sistema DVB-H

Lo standard prevede tre diverse tipologie di configurazione di rete (Figura 10):
a. l'intero multiplex può essere dedicato alla trasmissione secondo lo standard DVB-H: in assenza di vincoli di compatibilità con i ricevitori DVB-T, possono essere utilizzate anche le opzioni del DVB-H che modificano lo standard DVB-T (per esempio. modo 4K);
b. il multiplex può essere condiviso tra servizi DVB-T e DVB-H, assegnando una banda complessiva costante al flusso DVB-H: per mantenere la compatibilità con i ricevitori attuali le funzioni DVB-H che modificano il livello fisico DVB-T non possono essere utilizzate;
c. sullo stesso canale a radio frequenza può essere trasmessa una modulazione gerarchica[10], con servizi DVB-H sul flusso ad alta priorità (per la ricezione mobile è richiesta maggiore robustezza) e DVB-T sul flusso a bassa priorità (con maggiore disponibilità di banda). Anche in questo caso il livello fisico DVB-T non può essere modificato.
Le coperture del DVB-H nel caso di multiplex dedicato sono paragonabili a quelle ottenibili con il sistema di radiofonia digitale DAB (*Digital Audio Broadcasting*) - anche se potrebbero essere necessari 2-3 dB di potenza in più - ma il numero dei programmi audio a parità di banda può essere quasi raddoppiato (essa ri-

[10] La modulazione gerarchica, così come prevista dallo standard DVB-T, è effettuata suddividendo il flusso di informazione, prima della mappatura dei bit sulle costellazioni bidimensionali, in due rami, l'uno ad alta priorità, l'altro a bassa priorità. I bit sul ramo ad alta priorità, che devono essere maggiormente protetti, scelgono il quadrante, quelli a bassa priorità discriminano i punti all'interno del quadrante.

sulta da sei a otto volte maggiore[11] per un'occupazione di banda quattro volte superiore).
Nei due casi di multiplex dedicato o condiviso poi, le coperture DVB-T e DVB-H sono molto disomogenee. Il DVB-T è pianificato per offrire una copertura fissa garantendo buona qualità di ricezione con trasmissioni a *bit rate* elevati (per esempio 24 Mbit/s con raggio di copertura di circa 38-53 km per potenza di trasmissione ERP pari ad 1 kW). Per garantire la copertura DVB-H mobile bisogna invece utilizzare modi trasmissivi più robusti, riducendo così drasticamente il *bit rate* (si può per esempio pianificare la copertura a 12 Mbit/s, con una rete molto più densamente occupata, costituita da più del triplo di trasmettitori, oppure con un incremento di ERP di 13 dB).
Con le modulazioni gerarchiche è possibile bilanciare parzialmente le coperture DVB-T e DVB-H; tuttavia per una copertura mobile continua, occorre un numero di trasmettitori da tre a cinque volte maggiore rispetto alla copertura DVB-T utilizzata per la ricezione fissa (con una riduzione del *bit rate* totale del multiplex da 24 a 20,5 Mbit/s).

5. CONCLUSIONI

Nonostante gli enormi progressi fatti dalla televisione dall'avvento del digitale fino ai giorni nostri, non si può affatto affermare che il processo innovativo si sia arrestato. Al contrario, nuovi scenari si sono aperti con l'affermazione del digitale e con la spinta della convergenza tra il mondo *telecom* e quello *broadcast*, offrendo la possibilità di offrire nuovi servizi agli utenti. I nuovi sistemi di codifica video e di trasmissione vanno in questa direzione.
Tuttavia la migrazione dai sistemi di diffusione attuali a quelli più avanzati non è esente da problemi: ad esempio il consorzio DVB non prevede una sostituzione a breve termine del DVB-S con il DVB-S2, e dell'MPEG-2 con l'H.264, in quanto operano con successo milioni di decodificatori in tutto il mondo con tecnologia tradizionale, ai quali gli operatori della radiodiffusione devono garantire la continuità del servizio.

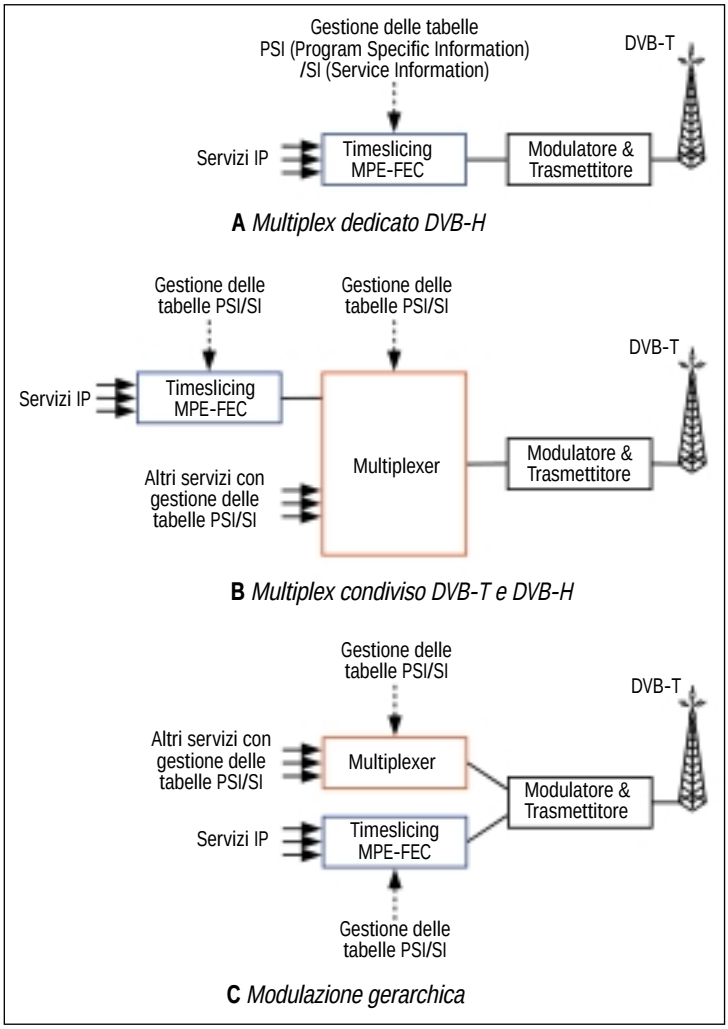


FIGURA 10
Configurazioni di rete DVB-H

La diffusione del DVB-S2 e dell'H.264 potrà invece riguardare servizi completamente nuovi non forniti dagli attuali ricevitori: la televisione ad alta definizione, il video su IP su canali limitati in banda ed i nuovi servizi che utilizzano l'Internet veloce.
Per fare previsioni relative all'introduzione del DVB-H bisogna tenere presente lo scenario attuale della televisione terrestre in Italia, dove i *broadcaster* sono già impegnati nell'introduzione del sistema digitale DVB-T, superando le difficoltà della scarsità di frequenze disponibili per il servizio. Quando questo processo sarà giunto a maturazione e cominceranno a essere spente le connessioni con *frequenze analogiche*, si apriranno migliori prospettive per il loro riutilizzo per i servizi mobili tipo DVB-H, e si sfrutteranno le sinergie con i sistemi di telefonia mobile avanzati.

[11] Da notare che un'eccessiva capacità del multiplex può essere un problema per le emittenti locali.

Bibliografia

[1] Information Technology-Generic Coding of Moving Pictures and Associated Audio Information: Video. JTC 1/SC 29; ISO/IEC 13818-2:2000, p. 208, http://www.iso.org/iso/en/prods-services/ISOstore/store.html

[2] Cominetti M., Morello A.: Il sistema europeo (DVB) per la diffusione televisiva da satellite. *Elettronica e Telecomunicazioni*, Anno XLIII, n. 3, 1994.

[3] Mignone V., Morello A., Visintin M.: Lo standard DVB-T per la televisione digitale terrestre. *Elettronica e Telecomunicazioni*, Anno LI, n. 1, aprile 2002, http://www.crit.rai.it/eletel/2002-1/21-4.pdf

[4] Schäfer R., Wiegand T., Schwarz H.: The emerging H264/AVC standard. *EBU Technical Review*, January 2003, http://www.ebu.ch/trev_293-contents.html

[5] List P., Joch A., Lainema J., Bjontegaard G., Karczewicz M.: Adaptive Deblocking Filter. I*EEE Transactions on Circuits and Systems for Video Technology*, Vol. 13, n. 7, July 2003.

[6] Ribas-Corbera J.: Windows Media 9 Series — a platform to deliver compressed audio and video for Internet and broadcast applications. *EBU Technical Review*, January 2003, http://www.ebu.ch/trev293-contents.html

[7] Morello A., Mignone V.: Il sistema DVB-S2 di seconda generazione per la trasmissione via satellite e Unicast. *Elettronica e Telecomunicazioni*, Anno LII, n. 3, dicembre 2003, http://www.crit.rai.it/eletel/2003-3/33-1.pdf

[8] Bertella A., Rossini M., Sunna P., Vignaroli L.: *Mobile DVB-T reception: quality of streaming over IP of audiovisual services*. IBC'02 Conference Amsterdam, 11-15 September 2002, http://www.broadcastpapers.com/tvtran/IBCRAIMobileDVB-To1.htm

[9] Pogrzeba P., Burow R., Faria G., Oliphant A.: *Lab & Field tests of mobile applications of DVB-T*. Montreux Symposium '99 Records, June 1999, p. 649-656, http://www.broadcastpapers.com/tvtran/DVBLab&FieldTestsMobile%20-%20print.htm

[10] *DVB Transmission system for handheld terminals*. DVB Document A081, June 2004, http://www.dvb.org/documents/white-papers/a081.dEN302304.sb1333.tm3037.tmh0202r4.pdf

[11] http://www.digitalradiotech.co.uk/fec_coding.htm

[12] Henriksson J., Talmola P.: Coach potato (Television on mobile home). *IEE Communications Engineer*, June 2004.

Alberto Morello si è laureato in Ingegneria Elettronica al Politecnico di Torino nel 1982 e ha ottenuto il titolo di "Dottore di Ricerca in Telecomunicazioni" nel 1987.
Dal 1999 è Direttore del Centro Ricerche e Innovazione Tecnologica della RAI di Torino. Dal 1984 si occupa di trasmissioni digitali dei segnali radio-televisivi e multimediali su canali via satellite, via cavo e via etere. È stato presidente di importanti gruppi tecnici che hanno definito gli standard DVB-S, DVB-DSNG e DVB-S2 e contribuisce regolarmente a riviste tecniche internazionali e congressi.
a.morello@rai.it

Paola Sunna consegue la laurea in Ingegneria Elettronica nel 1997 presso il Politecnico di Torino discutendo una tesi sulla valutazione oggettiva della qualità di segnali video compressi con MPEG-2. Dal 1997 lavora al Centro Ricerche e Innovazione Tecnologica, svolgendo test per la valutazione della qualità del segnale video nell'ambito di applicazioni broadcasting, webcasting e 3G. Dal 2002 presiede il gruppo EBU B-VIM (Video in Multimedia) che ha eseguito una campagna di test per la misurazione delle prestazioni di codificatori per applicazioni multimediali.
p.sunna@rai.it

Vittoria Mignone si è laureata in Ingegneria Elettronica presso il Politecnico di Torino nel 1990. Nel 1991, in collaborazione con il dipartimento di Ingegneria Elettronica del Politecnico di Torino, ha effettuato studi sulla trasmissione via satellite commissionati dal Consiglio Nazionale delle Ricerche. Dal 1992 lavora presso il Centro Ricerche RAI, dove ha collaborato agli studi per la definizione dei sistemi europei DVB per la trasmissione di televisione digitale via satellite, via cavo, terrestre, e per servizi di tipo DSNG (*Digital Satellite News Gathering*) e di contributo.
v.mignone@rai.it

DENTRO LA SCATOLA

Rubrica a cura di

Fabio A. Schreiber

Il Consiglio Scientifico della rivista ha pensato di attuare un'iniziativa culturalmente utile presentando in ogni numero di Mondo Digitale un argomento fondante per l'Informatica e le sue applicazioni; in tal modo, anche il lettore curioso, ma frettoloso, potrà rendersi conto di che cosa sta "**dentro la scatola**". È infatti diffusa la sensazione che lo sviluppo formidabile assunto dal settore e di conseguenza il grande numero di persone di diverse estrazioni culturali che - a vario titolo - si occupano dei calcolatori elettronici e del loro mondo, abbiano nascosto dietro una cortina di nebbia i concetti basilari che lo hanno reso possibile.
Il tema scelto per il 2004 è stato: "***Perché gli anglofoni lo chiamano computer***, ovvero: ***introduzione alle aritmetiche digitali***". Per il 2005 il filo conduttore della serie sarà: "***Ma ce la farà veramente?***, ovvero: ***introduzione alla complessità computazionale e alla indecidibilità***" e il suo intento è di guidare il lettore attraverso gli argomenti fondanti dell'Informatica e alle loro implicazioni pratiche e filosofiche. La realizzazione degli articoli è affidata ad autori che uniscono una grande autorevolezza scientifica e professionale a una notevole capacità divulgativa.

Potenza e limiti del calcolo automatico: le radici teoriche dell'informatica

Dino Mandrioli

INTRODUZIONE

Questo primo articolo della serie dedicata agli aspetti più concettuali e profondi dell'informatica ha un taglio decisamente storico: mostreremo come, contrariamente al creder comune, l'informatica non sia affatto "nata ieri" e non consista soltanto in un agglomerato inestricabile di circuiti elettronici e software; che non offra solo "servizi web", giochi elettronici, impianti satellitari ecc., ma affondi le sue radici nelle stesse origini del pensiero umano e abbia contatti e intersezioni fortissimi con discipline "nobili e astratte" come la matematica e la filosofia.
Dopo un breve richiamo al concetto di algoritmo e alla sua essenza indissolubilmente legata all'esecuzione mediante uno strumento automatico ne investigheremo non solo la potenza ma anche i limiti in termini dei problemi che attraverso questo strumento possono essere affrontati e risolti. Sottolineeremo come potenza e limiti del calcolo automatico abbiano un fondamentale impatto sia in termini pratici sulla nostra attività quotidiana sia sulla natura stessa del pensiero umano.
Il successivo articolo di questa serie, pur mantenendo uno stile divulgativo, entrerà nel merito delle tecniche utilizzate per arrivare ai risultati fondamentali qui enunciati: ciò darà ulteriore evidenza alle connessioni tra informatica, matematica e filosofia.

ALGORITMI; STRUMENTI E MODELLI DI CALCOLO AUTOMATICO

Tra le tante formulazioni del concetto di algoritmo facciamo riferimento alla seguente:

> *"Un algoritmo è un processo di elaborazione di informazioni, basato su una codifica rigorosa e precisa delle medesime e costituito da una sequenza di passi elementari definita in maniera altrettanto precisa e rigorosa, tanto da non lasciare alcun dubbio all'ente preposto alla sua esecuzione."*

Esempi universalmente noti e applicati di algoritmi sono gli algoritmi per il calcolo delle operazioni aritmetiche, per la ricerca di un elemento all'interno di un insieme, per mettere in ordine una sequenza di elementi ecc..
A sottolineare il fatto che questo fondamentale pilastro dell'informatica ha radici storiche millenarie, ricordiamo il famoso algoritmo di Euclide per il calcolo del massimo comun divisore (MCD) tra due numeri naturali x e y. Esso è basato sulla

constatazione che se $x = y$, allora evidentemente MCD $(x, y) = x = y$; altrimenti esso coincide con il MCD $(x - y, y)$, se $x > y$ o il MCD $(y - x, x)$ se $y > x$. Di conseguenza, continuando a sottrarre il minore tra i due all'altro e sostituendo il risultato della differenza al maggiore, si ottiene una sequenza che termina con il MCD (che sarà 1 nel caso degenere di due numeri primi tra loro). Per esempio, partendo da 15 e 9 si ottiene la sequenza <6, 9>, <3, 6>, <3, 3> e quindi MCD (15, 9) = 3.
Questa formulazione ed esemplificazione del concetto di algoritmo ci permette una prima serie di commenti e delucidazioni:

- Ogni "problema informatico", anche il più complesso come la gestione di varie forme di commercio elettronico o il monitoraggio e controllo di una rete elettrica nazionale, consiste nella costruzione di nuova informazione a partire dall'informazione esistente. Di conseguenza, ogni "progetto informatico" altro non è, in ultima analisi, che un complesso algoritmo, il quale al suo interno racchiuderà altri algoritmi per la soluzione di sottoproblemi (anche un algoritmo per il calcolo della moltiplicazione può sfruttare un sottoalgoritmo per il calcolo della somma), e così via fino alla scomposizione in un enorme numero di operazioni elementari che, nella tecnologia elettronica digitale, consistono nell'elaborazione di singoli bit.
- L'esecuzione di un algoritmo non richiede "intelligenza" ma soltanto precisione: quando sbagliamo la soluzione di un problema applicando un algoritmo, diciamo che "abbiamo sbagliato i conti" e, giustamente, tendiamo a valutare l'errore commesso come "non grave", rispetto ad errori come la cattiva progettazione dell'algoritmo o addirittura la cattiva comprensione stessa del problema.
- Proprio la precedente osservazione ha generato, fin da subito, il desiderio di usare una macchina per eseguire algoritmi: sono ben noti, infatti, i vari "prototipi di calcolatori" costruiti nei secoli da diversi "pionieri" come Blaise Pascal, Charles Babbage e vari altri.
- È altresì noto che per molti secoli il calcolo meccanico sia rimasto un sogno e divenne realtà pratica solo relativamente di recente con l'avvento della tecnologia elettronica digitale.

Traendo spunto da quest'ultima osservazione, è importante notare che "la fantasia umana ha largamente anticipato l'evoluzione tecnologica"; in altri termini *modelli di calcolatori*, ossia *macchine astratte* per l'esecuzione di algoritmi, sono stati concepiti e studiati a fondo ben prima che i moderni calcolatori elettronici facessero il loro ingresso nel mondo industriale. Tra i tanti, la *Macchina di Turing*, che deve il nome al suo inventore, è forse il modello più semplice e affascinante di calcolatore. Come suggerito dalla figura 1, essa consiste in due parti:

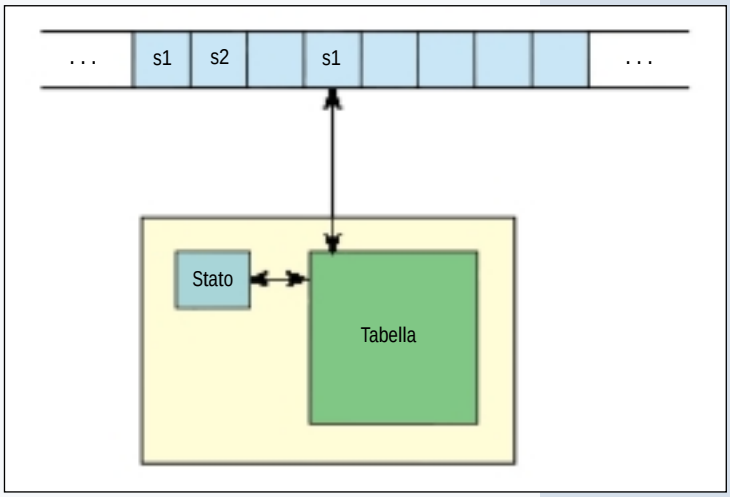


FIGURA 1
Una macchina di Turing

- un *nastro* infinito, suddiviso in celle, ciascuna contenente un *singolo simbolo* (come una lettera dell'alfabeto o una cifra decimale). Il nastro funge da *supporto per l'inserimento dei dati* (contiene i dati da elaborare), da *supporto di memoria* (è possibile leggere e cambiare il contenuto delle celle), e da *supporto di uscita* (il risultato del calcolo è parte di quello che rimane sul nastro quando l'esecuzione ha termine);
- un'unità di controllo, che in ogni istante si trova in uno e un solo *stato*, appartenente ad un prefissato insieme finito.

Un'operazione elementare di una macchina di Turing consiste nella:

- lettura di un dato dal nastro per mezzo di una *testina di lettura-scrittura* che mette in comunicazione l'organo di controllo con il nastro;
- lettura dello stato dell'organo di controllo.

In base alle letture effettuate la macchina:

- scrive un nuovo simbolo sul nastro al posto di quello letto;
- modifica lo stato dell'organo di controllo;
- sposta la testina di lettura-scrittura di una posizione a destra o a sinistra, o la lascia dove si trova.

In alternativa, la macchina viene posta in stato di *halt*, in cui cioè non è in grado di eseguire altri movimenti; a questo punto l'elaborazione è terminata.

L'algoritmo da eseguire viene comunicato alla macchina di Turing attraverso una tabella rettangolare in cui le righe indicano i simboli che possono essere letti dal nastro, le colonne indicano i possibili stati, e ogni cella indica l'operazione da eseguire in funzione del simbolo letto e dello stato dell'organo di controllo. Una cella contiene perciò una terna di elementi <simbolo da scrivere sul nastro, nuovo stato, spostamento della testina>; una cella vuota indica che la macchina si deve arrestare.

La figura 2 descrive, a titolo di esempio, una semplice macchina il cui alfabeto del nastro consiste in due soli simboli: "|" e "–", e il cui insieme di stati è dato da s1 e s2. Nell'ipotetico stato iniziale s1, il nastro contiene un numero *n* di "|" consecutivi, tutte le altre celle contengono "–", e la testina è posta sulla prima barra a sinistra. È facile constatare che, alla fine dell'elaborazione, ci sono *n* + 1 barre sul nastro; in altri termini, la macchina calcola *n* + 1 a partire da n sulla base di una codifica unaria.

Altri modelli matematici formalizzano la costruzione del linguaggio (le *Grammatiche formali* elaborate dai linguisti-matematici sono una "versione matematica" del concetto normalmente associato a questo termine), o addirittura la formulazione stessa del pensiero: in ultima analisi la sorgente primaria di informazione è proprio il nostro pensiero. Non a caso l'origine storica della *logica matematica* risiede nella logica aristotelica.

Nella prossima sezione scopriremo gli affascinanti legami tra i disparati modelli astratti di elaborazione dell'informazione e le macchine reali che, sparse per il mondo, realizzano tale elaborazione.

Stato \ s	–	|
s1	<|, s2, R>	<|, s1, R>
s2		

FIGURA 2

Tabella di una macchina di Turing per il calcolo di n + 1 in codifica unaria

La tesi di Church e la (in)calcolabilità

Una volta elaborato il concetto di algoritmo, sviluppati i primi algoritmi per la soluzione di vari problemi, concepiti e realizzati i primi strumenti per eseguire algoritmi, sorge spontanea e impellente la domanda: "*Quanti e quali problemi è possibile risolvere mediante il calcolo automatico (ossia mediante l'esecuzione di algoritmi)*?"

A prima vista, una lettura pignola ma superficiale di questa domanda potrebbe portare a ritenerla addirittura mal posta: la definizione di algoritmo è tutt'altro che matematicamente precisa e fa riferimento a quelli che possiamo definire "passi elementari". Chiaramente, tali passi, possono variare molto a seconda che l'esecutore dell'algoritmo sia un essere umano, o un calcolatore tra i tantissimi che sono stati costruiti o saranno costruiti, o un modello astratto come la macchina di Turing. Inoltre la nozione stessa di problema può essere codificata in diverse maniere: per esempio un numero può essere rappresentato in forma unaria (mediante aste o bastoncini come abbiamo imparato alle elementari e ricordato nella sezione precedente) o in base 10 o in base 2 ecc.; chi ci garantisce che un problema che non sappiamo risolvere se formulato in una certa maniera, non diventi invece risolvibile cambiandone formalizzazione? Una semplice analisi ci aiuterà a sgombrare il campo da questi dubbi, pur giustificati.

In primo luogo osserviamo che, anche se è vero che uno stesso problema può essere formulato in diverse maniere, è sempre possibile tradurre una rappresentazione in un'altra: questa stessa traduzione è un problema di elaborazione dell'informazione e può a sua volta essere eseguita mediante opportuni algoritmi. Per esempio, sono ben noti gli algoritmi per trasformare la rappresentazione di un numero da una base ad un'altra. Quindi, se riusciamo a risolvere un problema mediante una certa formalizzazione, possiamo risolverlo anche in una formalizzazione diversa mediante un'opportuna conversione.

Ancor più affascinante, e di enorme impatto concettuale e pratico, è la constatazione che i numerosissimi modelli di calcolo nonché tutti i calcolatori costruiti finora *hanno la medesima capacità di calcolo*; in altri termini, se si trova un algoritmo per risolvere un certo problema che sia eseguibile da un calcolatore *X*, si può essere certi di poter "programmare" il medesimo algoritmo, e quindi risolvere lo stesso problema, me-

diante un calcolatore *Y*: la semplicissima macchina di Turing descritta nella sezione precedente è in grado di risolvere gli stessi problemi che possono essere affrontati mediante un costosissimo calcolatore moderno. Una breve riflessione dimostra che questa affermazione non è così incredibile: basta infatti constatare che il contenuto dell'intera memoria di un qualsiasi calcolatore reale può essere "trasportato" nel nastro della macchina di Turing e che ogni istruzione del calcolatore può essere simulata da un opportuno numero di operazioni elementari (una sorta di sottoprogramma) della medesima.
Lo stesso vale per altri modelli formali come le grammatiche e, soprattutto, per ogni altro modello di calcolo, astratto o reale, che si possa inventare in futuro. Quest'ultima affermazione, ovviamente non può essere enunciata come un teorema poiché non si possono definire i modelli di calcolo ancora non inventati; essa è però supportata da tutta la storia pregressa e da un'analisi attenta del concetto generale di "operazione elementare". Affascinante, da un punto di vista storico, è il fatto che questa enunciazione sia dovuta ad Alonso Church e ad altri precursori della teoria della computazione, tra cui lo stesso Turing, che la formularono negli anni 30 del XX secolo, ben prima che vedesse la luce il moderno calcolatore elettronico. Per questo motivo, questo fondamentale pilastro della teoria della computazione è anche noto come *Tesi di Church*[1].
Armati di questo fondamentale risultato possiamo dunque tornare alla domanda di partenza, riformulandola in maniera matematicamente più precisa ma non per questo meno generale:
"*Quanti e quali problemi sono risolvibili mediante macchine di Turing*?"
La risposta comporta una sostanziale limitazione nella potenza del calcolo automatico: a dispetto di un'infinità di algoritmi già noti per la soluzione di problemi di ogni tipo e di tanti altri che verranno elaborati in futuro, possiamo affermare che *una altrettanto grande - anzi, una ben "maggiore - infinità" di problemi non è risolvibile mediante calcolo automatico*. Nel presente articolo ci limitiamo ad affermare questo fondamentale risultato; nei paragrafi seguenti esamineremo alcuni tipici problemi che non possono essere risolti algoritmicamente, e ne sottolineeremo l'impatto sia pratico che concettuale. Introdurremo invece le principali tecniche matematiche che portano a dimostrare questo ed altri risultati della teoria della computazione nell'articolo seguente di questa serie. Anche in questo caso evidenzieremo gli strettissimi rapporti e comunanze tra i vari filoni dell'evoluzione del pensiero umano: dalla matematica alla filosofia, all'informatica.

Impatto pratico dei limiti teorici alla calcolabilità

Storicamente, il problema più classico non calcolabile, ossia non risolvibile mediante un algoritmo, riguarda proprio una proprietà del calcolo automatico, ossia la sua terminazione. In termini teorici esso è noto come "*il problema dell'halt della macchina di Turing*: data una macchina M e un dato di partenza x, scritto sul nastro di M, la computazione di M avrà prima o poi termine?". Grazie alla tesi di Church questa domanda è del tutto equivalente alla seguente, dal "sapore" decisamente più pratico: "Ho scritto un programma in C e ne ho lanciato l'esecuzione fornendo in ingresso certi dati; dopo diversi secondi/minuti/ore il programma non mi ha ancora dato risposta: ne devo concludere che "è andato in loop" e quindi spegnere il calcolatore o abortire l'esecuzione o devo pazientare ulteriormente sperando che in un prossimo futuro giunga la risposta?".
Molti di noi, anche se non "informatici professionisti" saranno probabilmente passati almeno una volta attraverso questa frustrante esperienza ed avranno perciò constatato quanto grave sia la mancanza di un "messaggio diagnostico" da parte del compilatore che ci avverta a priori che se proviamo ad eseguire il programma testé scritto con certi dati esso è destinato a non terminare mai. Perché invece, il compilatore ci avverte, per esempio, che nel nostro programma abbiamo dimenticato di chiudere una parentesi e magari ci suggerisce anche dove e come eseguire la correzione? Perché il compilatore è a sua volta un algoritmo e il problema di stabilire se un'espressione è ben parentetizzata è decidibile[2], mentre

[1] Il lettore non si lasci fuorviare da certe affermazioni apparse in pubblicazioni a carattere divulgativo tendenti a sostenere che "la tesi di Church è ormai superata": si tratta di affermazioni ad effetto del tutto prive di rigore teorico.

[2] Il termine "decidibile" è sinonimo del termine "calcolabile" laddove il problema da risolvere sia espresso in termini booleani, ossia abbia una soluzione binaria.

il problema della terminazione dell'esecuzione di un algoritmo non lo è.
Come dicevamo, molti altri problemi sono sfortunatamente non calcolabili. Alcuni tipici esempi nell'ambito della programmazione sono i seguenti:

- L'occorrenza di una divisione per 0: durante l'esecuzione del mio programma, potrà capitare che si verifichi un tale errore?
- Accesso ad un array con un indice il cui valore è fuori dai limiti imposti dalla dichiarazione del medesimo array.
- Accesso ad una variabile non inizializzata, ossia cui non è stato ancora assegnato alcun valore durante l'esecuzione.

Chiunque abbia un po' di esperienza di programmazione sa bene che errori del genere sono molto frequenti e spesso hanno conseguenze anche gravissime (e gli hackers ne approfittano!). Perciò, l'impossibilità di individuarli mediante un opportuno algoritmo costituisce un grave impedimento alla produzione di software di qualità e pone maggiori responsabilità sulle spalle del programmatore.
In ambito matematico è ben noto che alcune equazioni ammettono soluzione mentre altre no; in taluni casi però esistono algoritmi per il calcolo della soluzione di un'equazione mentre in altri casi non ne esistono. Un esempio classico in tal senso è il famoso decimo problema di Hilbert, enunciato dal noto matematico all'inizio del XX secolo: "dato un polinomio a coefficienti interi in un numero qualsiasi di variabili, esistono radici intere del dato polinomio?". Solo alla fine degli anni 60 è stata dimostrata l'indecidibilità di questo problema: non esistono algoritmi per rispondere a questa domanda.
Si noti tuttavia che se ci si limita a considerare il caso di polinomi in una sola variabile, il problema diventa decidibile: è abbastanza facile costruire un algoritmo che, dato un polinomio in una variabile, stabilisca se ne esistono radici intere e, in caso positivo, le calcoli; questa circostanza ha una valenza generale: spesso accade che un problema non sia risolvibile in una sua formulazione generale ma lo diventi se opportunamente ristretto a casi particolari.
È necessario sottolineare che in questo contesto il termine "problema non risolvibile", non significa che non è possibile trovarne la soluzione in assoluto ma significa che non esistono algoritmi per la sua soluzione. Per esempio, in molti casi un'attenta analisi di un programma potrà far capire se esso è esposto al rischio della non terminazione o di un qualsiasi altro errore del tipo di quelli summenzionati. Questa analisi però, non sarà il risultato dell'esecuzione di un algoritmo, bensì il frutto di immaginazione, esperienza e, perché no, fortuna, doti queste, tipicamente umane e "non meccanizzabili"[3].

Impatto filosofico dei limiti teorici alla calcolabilità

Un importante settore applicativo dell'informatica è la dimostrazione automatica di teoremi. A dispetto del "suono decisamente matematico" di questo termine, esso ha notevoli risvolti pratici. Infatti, le formule matematiche altro non sono che una notazione precisa e astratta per descrivere problemi di vario tipo: così come un'equazione differenziale può descrivere la traiettoria di un missile e il problema della sua intercettazione, un'altra formula può esprimere il fatto che un impianto nucleare è sicuro, ossia garantisce dal rischio di esplosioni, perdite radioattive ecc. In questo contesto, perciò, dimostrare un teorema equivale a dimostrare la sicurezza di un impianto, la correttezza di un programma ecc.
A questo punto non dovrebbe sorprendere che la dimostrazione di teoremi è un altro problema non calcolabile: non esistono algoritmi generali per decidere se da certe ipotesi si possono trarre certe tesi. Questa nuova constatazione di limite alle possibilità del calcolo automatico ha risvolti non solo pratici, ma anche filosofici. Essa ci suggerisce infatti il passaggio dall'investigazione dei limiti del calcolo automatico all'investigazione dei limiti dello stesso ragionamento umano. Senza pretesa di sviluppare un affascinante ma complesso e delicato argomento in modo approfondito, cercheremo in questa sezione di far capire come da certi risultati dell'informatica teorica si possa risalire ai grandi temi filosofici che da sempre hanno sfidato il pensiero umano.
Come accennato in precedenza, è naturale la tendenza a descrivere il ragionamento umano in forma matematica: il tipico sillogismo aristotelico è un esempio di come il concetto di teorema formalizzi il ricavare una nuova verità a partire da altre verità assunte come ipotesi. In generale chiunque è disposto ad accettare come verità un

[3] Fatte salve alcune considerazioni proposte nella sezione seguente.

teorema ben dimostrato. La dimostrazione di teoremi risulta perciò un importante strumento per ricavare "verità", qualsiasi sia il contesto in cui tali verità sono ricercate (l'aritmetica, le operazioni finanziarie, il controllo di impianti, strategie belliche ecc.). Se però è del tutto accettabile - sotto opportune ipotesi - che un teorema possa essere assunto come verità, il problema inverso relativo alla "ricerca di verità" mediante dimostrazione di teoremi è tutt'altro che scontato. Infatti, sempre nella prima metà del XX secolo, Kurt Goedel ha dimostrato i suoi fondamentali *teoremi di incompletezza*, che, in qualche maniera, sintetizzano millenni di studi sulla capacità e sui limiti del ragionamento umano. L'essenza filosofica dei teoremi di Goedel potrebbe essere formulata nel modo seguente:

> *"Qualsiasi formalizzazione del ragionamento umano non è in grado di catturare in modo completo tutte le verità che sono conseguenze di certe ipotesi"*

Dalla filosofia alla religione il passo è breve e così più d'uno, incluso lo stesso Goedel in tarda età, ha ritenuto di poter ricavare da questa constatazione una "dimostrazione" dell'esistenza di un'Entità Superiore, capace di catturare tutte quelle verità che sfuggono ai limiti umani; ciò costituisce un errore logico altrettanto grave dell'affermazione opposta[4].

Prima di chiudere questo articolo dedichiamo un breve cenno al settore dell'Intelligenza Artificiale. Questo termine, spesso abusato, include studi ed obiettivi che coprono aspetti di grande valenza pratica, come la robotica, e di profondo studio filosofico, elemento accomunante essendo il desiderio di ottenere dalla macchina risultati comparabili o superiori a quelli ottenuti dall'intelligenza umana (per esempio, nelle partite a scacchi, nella traduzione del linguaggio naturale ecc.). A prima vista, quanto affermato in precedenza potrebbe portare alla conclusione che necessariamente "il cervello umano non è un semplice esecutore di algoritmi" e quindi il termine "intelligenza" include capacità diverse da ciò che si può affrontare in modo algoritmico. In realtà, però, è ammissibile anche l'ipotesi opposta, sposata da diversi pensatori, tra cui lo stesso Turing. L'apparente contraddizione potrebbe essere spiegata con l'ipotesi che, quando noi risolviamo un problema non calcolabile stiamo certamente eseguendo un algoritmo, ma un algoritmo che non ci garantisce di giungere al risultato cercato: in un certo senso, stiamo "tentando un strada algoritmica", ma diversamente da quando risolviamo un problema calcolabile, non sappiamo se si tratti della strada giusta né se questa ci porterà mai ad un risultato. Alcune tecniche illustrate nell'articolo successivo potranno forse aiutarci a capire meglio questa congettura.

[4] Scusandoci per l'inevitabile banalizzazione di un tema così profondo e delicato, commette un errore logico chi ritiene di aver dimostrato un'affermazione per il fatto di non trovare altro modo di spiegare un fenomeno, così come chi dichiara falsa un'affermazione per il fatto che non ne esista dimostrazione.

Bibliografia

Essendo impossibile rendere completa giustizia ai tanti "pionieri" dell'informatica (che a nostro modo di vedere includono matematici e filosofi greci e arabi di millenni trascorsi) ci limitiamo qui a citare alcune pietre miliari della teoria della computabilità ([2] e [4]), della linguistica matematica [1]) e della logica [3].

[1] Chomsky N.: Three Models for the Description of Languages. *IRE Transactions on Information Theory*, Vol. 2, n. 3, 1956, p. 113-124.

[2] Church A.: An Unsolvable Problem of Elementary Number Theory. *American Journal of Mathematics*, Vol. 58, 1936, p. 345-363.

[3] Goedel K.: *On Undecidable Propositions of Formal Mathematical Systems*. Princeton University Press, 1934.

[4] Turing A.: On Computable Numbers, with an Application to the Entscheidungsproblem. *Proceedings London Mathematical Society*, Vol. 42, p. 230-265 e Vol. 43, p. 544-546, 1936-1937.

Dino Mandrioli è professore ordinario di Informatica Teorica presso il Politecnico di Milano. I suoi interessi di ricerca sono principalmente nei settori dell'informatica teorica, dell'ingegneria del software e dei sistemi critici in tempo reale. Ha pubblicato oltre 80 articoli scientifici su riviste ed atti di convegni internazionali. È coautore di vari libri, fra cui *Theoretical Foundations of Computer Science* (J. Wiley & Sons), *Fundamentals of Software Engineering* (Prentice-Hall), *The Art and Craft of Computing* (Addison Wesley). Dino Mandrioli è stato membro del Program Committee di diverse conferenze internazionali, Associate Editor di diverse riviste internazionali, program co-chairman della conferenza Formal Methods 2003.

ICT E DIRITTO

Rubrica a cura di

Antonio Piva, David D'Agostini

Scopo di questa rubrica è di illustrare al lettore, in brevi articoli, le tematiche giuridiche più significative del settore ICT: dalla tutela del *domain name* al *copyright* nella rete, dalle licenze software alla *privacy* nell'era digitale. Ogni numero tratterà un argomento, inquadrandolo nel contesto normativo e focalizzandone gli aspetti di informatica giuridica.

La firma digitale tra crittografia e diritto

Maurizio Blancuzzi

Introduzione

La disciplina normativa della firma digitale rappresenta con molta probabilità il punto più elevato di intersezione tra il mondo giuridico e quello delle scienze informatiche e matematiche.
Secondo la definizione tratta dal Decreto Presidente Repubblica 28 dicembre 2000 n.445 la "firma digitale é un particolare tipo di firma elettronica qualificata basata su un sistema di chiavi asimmetriche a coppia, una pubblica e una privata, che consente al titolare tramite la chiave privata e al destinatario tramite la chiave pubblica, rispettivamente, di rendere manifesta e di verificare la provenienza e l'integrità di un documento informatico o di un insieme di documenti informatici".
Il legislatore, pertanto, ha impiegato il concetto di chiave nell'accezione propria della scienza crittografica, la cui conoscenza si rivela di fondamentale importanza per approfondire l'argomento trattato.

La crittografia

Tramite la *crittografia* è possibile scambiarsi informazioni confidenziali in formato elettronico in modo sicuro, rendendo quindi difficilmente accessibili tali informazioni a utenti non autorizzati. In particolare, con il processo di *cifratura* il dato in chiaro viene codificato in una sequenza comprensibile solamente al destinatario; mentre dal testo cifrato si può risalire al testo in chiaro tramite il procedimento inverso, vale a dire la *decifratura*. Tali processi fanno uso di chiavi e la procedura di decifratura di un testo può avvenire solo conoscendo la chiave apposita. Senza entrare troppo nel dettaglio si può affermare che la "forza" della moderna crittografia si basa proprio sulla segretezza delle chiavi piuttosto che su quella degli algoritmi. La principale suddivisione può essere fatta tra sistemi *simmetrici* e sistemi *asimmetrici*.
I primi (esempio DES, IDEA, AES ecc.) utilizzano un'**unica chiave** per codificare e decodificare le informazioni. Tali algoritmi sono più veloci e di più facile realizzazione rispetto quelli asimmetrici ma presentano una debolezza intrinseca: sia il mittente che il destinatario devono essere a conoscenza della chiave. Infatti, se per poter scambiare messaggi riservati entrambi gli interlocutori devono essere a conoscenza di un segreto, allora aumenta la probabilità che tale chiave possa essere scoperta (si pensi al problema della distribuzione delle chiavi).
Per superare questo problema sono stati adottati algoritmi che utilizzano una **coppia di chiavi**, legate tra loro in maniera univoca mediante precise relazioni matematiche. In questo caso le chiavi sono dette, una *privata* (segreta) e l'altra *pubblica*: ciò che viene cifrato con una chiave viene decifrato con l'altra. La chiave privata viene mantenuta segreta dal mittente, mentre quella pubblica viene diffusa a terzi. La validità di un sistema di questo tipo consiste nella difficoltà di ricostruire la chiave privata a partire da quella pubblica ("rottura del cifrario", attività che richiede un tempo esponenziale all'aumentare della lunghezza delle chiavi).
Esistono molti algoritmi a chiavi asimmetriche: il primo risale al 1976 con l'algoritmo Diffie-Helman, ma il più diffuso è senza dubbio RSA, svi-

luppato nel 1977, che prende il nome dalle iniziali dei suoi tre ideatori (Rivest, Shamir e Adleman). La normativa italiana, parlando di firme digitali, indica esplicitamente l'algoritmo RSA e la lunghezza delle chiavi deve essere di almeno 1024 bit.

Nel processo di generazione della firma digitale rivestono particolare importanza gli algoritmi di *hash*, che servono per calcolare l'*impronta* di un documento informatico (file), in altre parole per ridurne la dimensione ad un valore ben preciso (esempio 160 bit secondo la legge italiana). Esistono molti algoritmi di hash; quelli ufficialmente identificati per la firma digitale sono SHA-1 (*Secure Hash Algorithm*) e RIPEMD-160. Questi algoritmi sono tali che:

- è impossibile risalire al documento originale partendo dalla sua impronta;
- è (praticamente) impossibile ottenere la stessa impronta partendo da due documenti differenti.

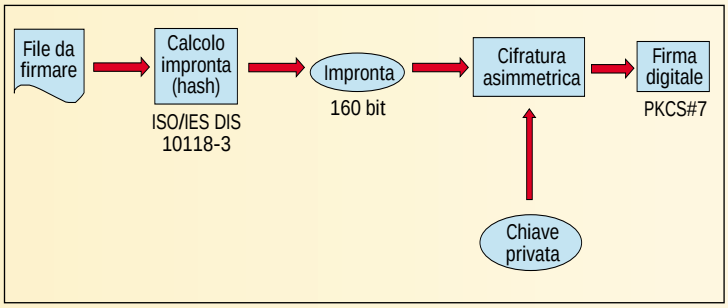


FIGURA 1
Generazione della firma digitale

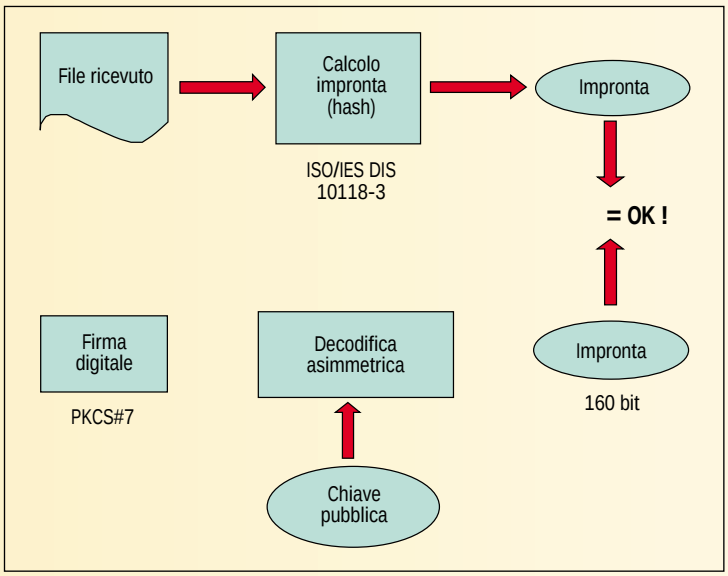


FIGURA 2
Verifica di una firma digitale

Solo l'impronta (e non l'intero documento) viene cifrata utilizzando la chiave privata; questo velocizza molto l'operazione di firma (cifratura) che di per sé è un'operazione complessa e pesante in termini di elaborazione.

Generazione e verifica

Per apporre una firma digitale a un documento informatico (non necessariamente testuale) è, quindi, necessario disporre di una coppia di chiavi digitali asimmetriche: la chiave privata, disponibile solo per il titolare, è utilizzata per sottoscrivere, mentre quella pubblica è invece utilizzata per verificare l'autenticità della firma.

Generazione della firma digitale

1. Dal file originario viene calcolata l'impronta di 160 bit usando un algoritmo di hashing che assicura la corrispondenza univoca tra l'impronta e il file.
2. L'impronta viene cifrata usando la chiave privata. Per fare questo viene utilizzata la smart card, contenente la coppia di chiavi e il certificato digitale, previa digitazione di un P.I.N.; il risultato di tale operazione, generata all'interno della smart card, è la *firma digitale*.
3. Viene quindi costruito un file in formato crittografico standard PKCS#7 costituito da: file originale, impronta cifrata e certificato digitale, con la chiave pubblica, del firmatario (Figura 1).

Verifica della firma digitale

1. L'impronta cifrata viene decifrata tramite la chiave pubblica del firmatario.
2. Viene calcolata nuovamente l'impronta applicando l'algoritmo di hash al file originale.
3. Si confrontano l'impronta estratta dalla firma con quella ricalcolata: se coincidono il messaggio è autentico (Figura 2).

Per l'apposizione o la verifica di una firma digitale vengono utilizzati opportuni programmi software che con semplici interventi dell'utente effettuano tutte le operazioni (hashing, cifratura, decodifica, confronto) in maniera completamente automatica[1].

[1] Sul sito www.cnipa.gov.it sono gratuitamente disponibili vari applicativi software per la generazione/verifica della firma digitale conformi alla circolare n. AIPA/CR/24 del 19 giugno 2000 sull'interoperabilità.

La certificazione

Gli obiettivi da raggiungere con la sottoscrizione digitale sono:
- *integrità*: intesa come la sicurezza che il documento informatico non sia stato modificato dopo la sua sottoscrizione;
- *autenticazione*: la certezza dell'identità del sottoscrittore di un documento;
- *non ripudio*: colui che firma un messaggio non può disconoscerne la paternità.

Tali caratteristiche si possono considerare realizzate solamente se vi è la sicurezza che la chiave pubblica del firmatario sia realmente associata al firmatario stesso. A tal fine è necessaria un'attività di *certificazione* il cui scopo principale è proprio quello di realizzare la corrispondenza biunivoca tra il titolare e la sua coppia di chiavi (pubblica e privata). Tale corrispondenza si realizza previa identificazione diretta del titolare, o indirettamente tramite una *Local Registration Authority* (LRA).
L'attività di certificazione viene svolta da una terza parte fidata, il *Certificatore* che svolge quindi una funzione di *certification authority*. L'attività principale di un Certificatore è l'emissione di certificati digitali che garantiscono l'identità del titolare della coppia di chiavi. Un certificato è quindi l'associazione tra i dati identificativi del titolare e la propria chiave pubblica. Inoltre tale certificato è pubblicamente accessibile e consultabile al fine della verifica dell'identità del firmatario; ogni certificato digitale è a sua volta sottoscritto digitalmente dal Certificatore mediante un'opportuna chiave di "certificazione".
Tra le varie attività "istituzionali" svolte dal Certificatore vi è anche la pubblicazione delle liste di revoca e di sospensione dei certificati (*Certication Revocation List* o CRL e *Certication Suspension List* o CSL).
Le informazioni tipicamente contenute in un certificato in standard ITU X.509 sono:

Versione	esempio V3
Numero di serie	
Algoritmo di firma	esempio sha1With RSAEncryption
Periodo di validità	solitamente da uno a tre anni
Titolare della chiave pubblica	
Valore della chiave pubblica	
Utilizzo del certificato	esempio Nonrepudiation
Impronta del certificato	

La competenza per la tenuta dell'elenco pubblico dei certificatori spetta al Centro Nazionale per l'Informatica nella Pubblica Amministrazione. Tale elenco è reso disponibile telematicamente tramite il sito web www.cnipa.gov.it. e contiene per ogni certificatore abilitato, le seguenti informazioni:
- Ragione o denominazione sociale
- Sede legale
- Rappresentante legale
- Nome X.500
- Indirizzo Internet
- Lista dei certificati delle chiavi di certificazione
- Manuale operativo
- Data di accreditamento volontario
- Data di cessazione e certificatore sostitutivo

Firma digitale e firma autografa

La firma digitale è il risultato di un procedura (cifratura) effettuata su un documento (file) specifico ed è quindi unica e caratteristica per ogni file elaborato. Non è quindi riproducibile ne è possibile apporre una determinata firma digitale ad un documento diverso da quello cui si riferisce.
La firma digitale fornisce un livello di garanzia dell'autenticità del documento anche superiore a quello offerto dalla firma autografa, purché si disponga di strumenti di sottoscrizione sufficientemente sicuri (*crypto-card*) e di un sistema di gestione dei certificati (*Public Key Infrastructure*) efficiente e affidabile. La differenza tra firma autografa e firma digitale è che la prima è legata alla grafia, caratteristica fisica del firmatario, mentre la seconda al possesso di uno strumento informatico (il token di firma) e alla conoscenza di una password.
Inoltre la firma digitale, essendo univocamente connessa al documento sul quale viene calcolata, cambia da un documento all'altro circostanza che ne rende impensabile la falsificazione o l'imitazione come può avvenire per la sottoscrizione vergata di pugno su un foglio di carta.
La firma digitale, infine, non può essere apposta su un documento "in bianco" in quanto l'assenza di un documento comporta l'impossibilità di ricavare l'impronta mediante l'algoritmo di hash (Tabella 1).

Firma autografa	È sempre uguale a sé stessa	Non garantisce l'integrità del testo	Non garantisce l'autenticità	Può essere apposta in bianco
Firma digitale	Varia in relazione al documento	Garantisce l'integrità del testo	Garantisce l'autenticità	Non può essere apposta in bianco

TABELLA 1
Differenze tra firma autografa e digitale

Aspetti giuridici

La firma digitale in Italia è un meccanismo tecnico-normativo in continua evoluzione, sia perché la diffusione delle tecnologie e il conseguente sviluppo della società dell'informazione sono processi inarrestabili, sia per la difficoltà di riconoscere il giusto valore legale ai documenti informatici; la smaterializzazione degli atti giuridici rende necessario predisporre nuove regole del diritto che affianchino quelle tradizionali.
Va riconosciuto al nostro paese il merito di essere stato il primo in Europa ad affrontare questo problema e, a partire dall'art. 15 comma 2 Legge 59/97, ad attribuire ai documenti informatici piena validità e rilevanza a tutti gli effetti di legge.
Il legislatore italiano, peraltro, ha dovuto ben presto fare i conti prima con esigenze di coordinamento che hanno portato all'approvazione del Decreto del Presidente della Repubblica n. 445/2000; poi con l'obbligo di recepimento della direttiva 1999/93/CE che è stata attuata mediante il Decreto Legislativo 23 gennaio 2002, n.10.
Nell'attuale quadro normativo, pertanto, al concetto di firma digitale si è aggiunto quello di firma *elettronica*, suddivisa in varie categorie in relazione al livello di sicurezza richiesto.
Le diverse definizioni ai sensi dell'art. 2 del D.lgs.10/02 sono:

- ***firma elettronica***: *l'insieme dei dati in forma elettronica, allegati oppure connessi tramite associazione logica ad altri dati elettronici, utilizzati come metodo di autentificazione informatica.*

Di fatto si tratta di una definizione generica (e omnicomprensiva) che identifica dati utilizzati per autenticare altri dati elettronici. Si può definire firma elettronica ogni sistema che può funzionare come chiave per accedere a un dato informatico (per es. una password, un P.I.N., un sistema a chiavi asimmetriche). Informalmente è definita anche come firma "debole" per differenziarla dalla firma digitale.

- ***firma elettronica avanzata***: *la firma elettronica ottenuta attraverso una procedura informatica che garantisce la connessione univoca al firmatario e la sua univoca identificazione, creata con mezzi sui quali il firmatario può conservare un controllo esclusivo e collegata ai dati ai quali si riferisce in modo da consentire di rilevare se i dati stessi siano stati successivamente modificati.*

Come per la firma elettronica "generica" non conta la tecnologia utilizzata, ma è importante che la sottoscrizione avanzata identifichi il firmatario, al quale deve essere ricondotta in maniera univoca; inoltre, deve garantire che l'atto non sia stato modificato.
Ha la stessa validità della firma elettronica generica, ma con il vantaggio che è più sicura: aumenta quindi la possibilità di utilizzarla come prova.

- ***firma elettronica qualificata***: *la firma elettronica avanzata che sia basata su un certificato qualificato e creata mediante un dispositivo sicuro per la creazione della firma.*

Si tratta di un ambito ancora più ristretto che richiede un certificato qualificato e deve essere creata mediante un dispositivo sicuro. Le firme elettroniche qualificate costituiscono prova, fino a querela di falso, della volontà del sottoscrittore del documento. È detta anche "firma forte" e corrisponde alla firma digitale con cifratura a doppia chiave asimmetrica, il metodo attualmente più sicuro per trasmettere documenti con la sicurezza dell'integrità, della riservatezza e del non ripudio. Ha quindi lo stesso valore della sottoscrizione tradizionale.
Per poter utilizzare queste differenti tipologie di firme elettroniche serve un nuovo tipo di certificatore: a quelli accreditati si affiancano, quindi, i certificatori di base o notificati. Di conseguenza l'attività che il soggetto certificatore deve compiere viene dimensionata in base al tipo di certificati che intende rilasciare. Queste nuove figure potranno emettere certificati di firma con standard di sicurezza che possono non corrispondere ai criteri fissati dalla legge, come invece dev'essere per i certificatori accreditati. Per questi ultimi i certificati emessi hanno i più elevati requisiti di sicurezza e qualità e su di essi il controllo sarà molto più rigido.

Le firme "forti" possono essere associate a certificati emessi solo da certificatori accreditati, mentre quelle "deboli" da tutti gli altri certificatori.
Il documento sottoscritto con la firma elettronica soddisfa comunque il requisito legale della forma scritta ed è valutabile come prova. In ogni caso non può essergli a priori negata rilevanza giuridica né ammissibilità come mezzo di prova. Quindi il documento sottoscritto elettronicamente è comunque assimilabile a quello cartaceo ed è utilizzabile come prova, seppure dietro valutazione del grado di affidabilità della firma adottata.

Conclusioni

L'interesse per le firme elettroniche appare elevato, come dimostra anche il numero di certificatori accreditati attivi, nonostante gli investimenti economici necessari per esercitare tale attività siano notevoli e non sia previsto un ritorno economico in tempi brevi.
Per quanto riguarda la certificazione, il recepimento delle disposizioni europee porta sì dei vantaggi in un'ottica di liberalizzazione e razionalizzazione, ma introduce anche delle complicazioni e contraddizioni rispetto al precedente sistema che prevedeva solamente certificatori accreditati.
Al di là di problemi tecnici ancora da risolvere e di alcune controverse interpretazioni normative, per poter dare avvio alla "rivoluzione" della firma digitale sembra di fondamentale importanza lo sviluppo dei servizi applicativi che ne sfruttino a pieno la reale natura di sistema abilitante, soprattutto nella gestione in via telematica del rapporto tra il cittadino e la Pubblica Amministrazione (per esempio pratiche e documenti amministrativi *on line*) su cui si basa il concetto di *e-government*.
Proprio in riferimento agli enti pubblici, è stato recentemente approvato un nuovo decreto legislativo intitolato "Codice dell'amministrazione digitale" che, entrando in vigore il 1° gennaio 2006, dovrebbe risolvere le attuali incertezze interpretative sull'efficacia delle firma elettronica e sugli aspetti probatori del documento informatico.

Bibliografia

[1] Cammarata M., Maccarone E.: *La firma digitale sicura*. Giuffrè, 2003.

[2] Finocchiaro G.: *Firma digitale e firme elettroniche*. Giuffrè, 2003.

[3] Rognetta G.: *La firma digitale e il documento informatico*. Ed. Simone, 1999.

[4] Ziccardi G.: *Crittografia e diritto*. Giappichelli, 2003.

[5] IN RETE: *Bollettino informativo del Centro Tecnico*. Presidenza Consiglio dei Ministri.

[6] www.cnipa.gov.it: Centro Nazionale per l'Informatica nella Pubblica Amministrazione.

[7] Il Sole-24 Ore: Guida agli Enti Locali.

[8] Il Sole-24 Ore: Norme e Tributi.

Antonio Piva laureato in Scienze dell'Informazione, Presidente, per il Friuli - Venezia Giulia, dell'ALSI (*Associazione Nazionale Laureati in Scienze dell'Informazione ed Informatica*) e direttore responsabile della Rivista di Informatica Giuridica.
Docente a contratto di Informatica giuridica all'Università di Udine.
Consulente sistemi informatici, valutatore di sistemi di qualità ISO9000 e ispettore AICA per ECDL base e advanced.
antonio_piva@libero.it

David D'Agostini avvocato, ha conseguito il master in informatica giuridica e diritto delle nuove tecnologie, fornisce consulenza e assistenza giudiziale e stragiudiziale in materia di *software, privacy* e sicurezza, contratti informatici, *e-commerce*, nomi a dominio, computer crime, firma digitale. Ha rapporti di partnership con società del settore ITC nel Triveneto.
Collabora all'attività di ricerca scientifica dell'Università di Udine e di associazioni culturali.
david.dagostini@adriacom.it

Maurizio Blancuzzi, laureato in Scienze dell'Informazione.
Responsabile settore ICT ed e-government nella PA locale. Membro della commissione informatica giuridica dell'ALSI (*Associazione Nazionale Laureati in Scienze dell'Informazione e Informatica*).
Ha svolto anche attività didattica e di consulenza in materia di ICT ed informatica giuridica.
maublanc@katamail.com