

HIGH PERFORMANCE COMPUTING

STATO E PROSPETTIVE

L'High Performance Computing (HPC) gioca un ruolo importante nella soluzione di problemi computazionali complessi. Il parallelismo ha affascinato i ricercatori per oltre 30 anni e oggi si assiste a un rinnovato interesse in questo settore. Nell'articolo, dopo una breve introduzione alle scienze computazionali e ad alcune applicazioni scientifiche di frontiera che si affrontano con strumenti HPC, viene presentata l'evoluzione delle architetture per il supercalcolo e se ne evidenziano le prospettive. Infine si sottolinea l'importanza dell'HPC come tecnologia abilitante a supporto della ricerca.

1. INTRODUZIONE ALLE SCIENZE COMPUTAZIONALI

Nella ricerca scientifica la formulazione di nuove teorie non può prescindere da una rigorosa formulazione di modelli matematici e da una loro verifica e sperimentazione basata sempre più spesso su simulazioni numeriche condotte al calcolatore.

Tali simulazioni, che si avvalgono di strumenti informatici avanzati, permettono di indagare sistemi fisici complessi e di estendere le teorie fondamentali della scienza moderna.

La simulazione numerica su calcolatore permette di estendere o di ridurre a piacere la scala del tempo e dello spazio per arrivare a rappresentare fenomeni molto grandi (come avviene per esempio in meteo-climatologia, in astrofisica e in geofisica) o molto piccoli (come nel caso della sperimentazione di nuovi farmaci, nella genomica, nella progettazione e validazione di dispositivi elettronici) oppure fenomeni complessi (fisica delle particelle elementari, dinamica dei fluidi), ma anche fenomeni pericolosi o costosi da gestire

con metodi tradizionali (come per esempio la simulazione di guasti di impianti industriali critici, la crash analysis e così via).

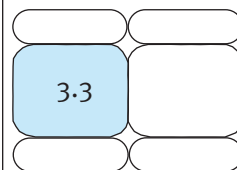
Si parla di scienze computazionali per intendere quelle discipline scientifiche che utilizzano in modo sistematico metodi matematici e strumenti informatici avanzati nella loro metodologia di indagine.

L'affermazione delle *scienze computazionali* come metodologia distinta dell'indagine scientifica, affiancata alla teoria e alla sperimentazione (Figura 1), è anche il risultato della crescita continua e sostanziale delle prestazioni dei sistemi di supercalcolo. Oggi le scienze computazionali sono in grado di indirizzare i problemi scientifici più complessi e permettono di indagare fenomeni non immaginabili anche solo un decennio fa, proprio grazie alla crescita di potenza che ha caratterizzato i sistemi di elaborazione in questi anni.

L'importanza di questa metodologia è stata sancita ufficialmente nel 1998 quando venne assegnato il premio Nobel per la chimica a John A. Pople, per il contributo dato allo svi-



Giovanni Erbacchi



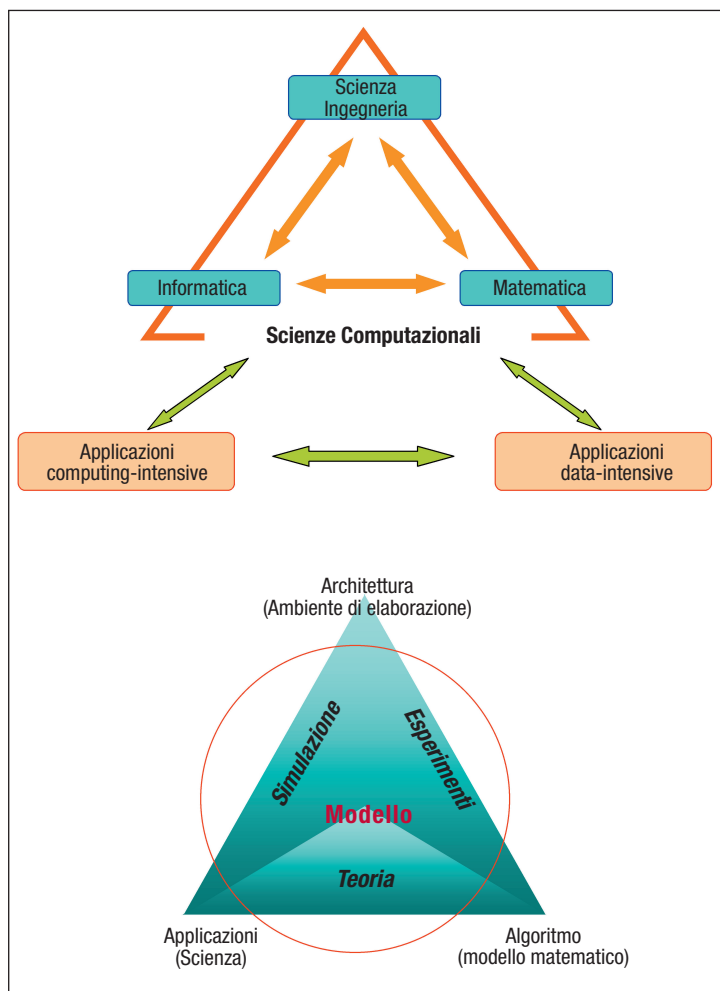


FIGURA 1
Relazioni tra scienze computazionali, informatica, matematica e applicazioni

luppo di metodi computazionali nella chimica quantistica [1].

Così come le epoche scientifiche sono definite in termini di crescita di un ordine di grandezza della capacità di osservazione sperimentale (per esempio in termini di risoluzione degli strumenti) altrettanto le epoche computazionali sono definite in termini di crescita di ordini di grandezza nelle prestazioni dei calcolatori, sia in termini di potenza di elaborazione che di capacità di memoria.

A metà degli anni '70, quando fecero la loro comparsa i primi supercomputer vettoriali, la potenza dei supercomputer più potenti era dell'ordine dei Mflop/s (10^6 operazioni in virgola mobile al secondo) [2]. Alla fine degli anni '90 si arrivò al traguardo dei Gflop/s (10^9 operazioni al secondo), attualmente

siamo nell'era dei Tflop/s (10^{12}) e prima del 2010 saranno disponibili sistemi in grado di erogare una potenza sostenuta di oltre un Pflop/s (10^{15} operazioni al secondo).

Il primo supercomputer di rilevanza per la comunità scientifica è stato il *Cray-1*, progettato da Seymour Cray. Il primo esemplare, installato al Los Alamos National Laboratory (USA) nel 1976 al costo di 8,8 milioni di dollari, aveva un processore vettoriale con una frequenza di 80 MHz ed era dotato di una memoria centrale di 8 Mbyte. La potenza di picco era di 160 Mflop/s e in un particolare kernel computazionale raggiunse una prestazione sostenuta record di 133 Mflop/s.

Oggi il calcolatore più potente è installato a poca distanza, al Lawrence Livermore National Laboratory. Si tratta di un sistema IBM BlueGene/L con 212.992 processori e 73 Tbyte di memoria centrale. La potenza di picco è 596 Tflop/s, mentre quella sostenuta raggiunge i 478 Tflop/s.

In un arco temporale di 30 anni si è assistito a un incremento in termini di prestazioni di oltre un fattore 10^6 , sia in termini di prestazioni del sistema che di capacità di memoria.

La complessità dei processori, in termini di densità di transistori continua a crescere e, anche se con qualche rallentamento, ancora segue la legge di Moore¹. L'incremento di potenza è dovuto alle prestazioni del singolo processore ma, soprattutto, alla capacità di integrare più processori in un'architettura scalabile dove la potenza computazionale della macchina può essere espansa con l'aggiunta di più moduli.

In Italia, il primo supercomputer fu installato al Centro Interuniversitario CINECA nel 1984. Era un supercomputer vettoriale *Cray X-MP 12* con una potenza di picco di 160 MFlop/s e una memoria di 16 Mbyte. Attualmente, il supercomputer più potente in Italia è installato sempre in CINECA: si tratta di un cluster linux con 5120 processori e una potenza di picco di 61 TFlop/s (Figura 2).

¹ Gordon Moore, co-fondatore di Intel, predisse nel 1965 che la densità dei transistori nei circuiti integrati sarebbe raddoppiata circa ogni 18 mesi. Moore nel 1975 modificò poi tale previsione in 24 mesi.

**FIGURA 2**

A) il Supercalcolatore Cray X-MP48 installato in CINECA a metà degli anni '80 (4 CPU vettoriali e 64 Mbyte di RAM, potenza di picco 0.94 GFlop/s. B) il supercomputer più potente oggi in Italia, installato sempre in CINECA (Cluster Linux con 5120 core Xeon a 3.0 GHz, 20 Tbyte di RAM e una potenza di picco di 61.4 TFlop/s)

2. APPLICAZIONI SCIENTIFICHE E PROBLEMI DI FRONTIERA

Molte applicazioni scientifiche simulano un fenomeno fisico nel dominio bi-dimensionale (2D) o tri-dimensionale (3D). Il fenomeno viene studiato su un certo dominio temporale per osservarne l'evoluzione, per esempio movimenti o cambiamenti nelle forme e delle proprietà degli oggetti. Per tradurre il fenomeno fisico in un modello matematico si usano sistemi, spesso accoppiati, di equazioni differenziali che non si risolvono analiticamente e richiedono uno schema di soluzione numerica che implica calcoli pesanti su sistemi HPC.

Il problema deve essere quindi discretizzato nello spazio e nel tempo. La discretizzazione spaziale significa che il dominio è sostituito da una griglia (o mesh) 2D o 3D e per ogni punto di questa griglia si individuano un certo numero di parametri rilevanti per la computazione (per esempio, temperatura, velocità, pressione ecc.). La discretizzazione spaziale è spesso regolare, nel senso che la distanza tra i punti della mesh è costante su tutto il dominio e in tutte le dimensioni. Tipicamente, più è piccola la distanza, più accu-

rata risulta la computazione, ma questo comporta un costo in quanto aumenta sia il tempo di calcolo che la memoria richiesta. Se certe aree del dominio fisico richiedono una maggiore accuratezza nella computazione, si introducono griglie irregolari con distanze differenti tra i punti della mesh.

La discretizzazione temporale significa che la simulazione ripete i calcoli sui diversi elementi della mesh per una sequenza di passi temporali, che varia a seconda del fenomeno da simulare: dai femto-secondi, per esempio, per le simulazioni di dinamica molecolare, agli anni, per esempio, per la simulazione delle variazioni climatiche.

Pertanto, il tempo totale per la simulazione diventa non solo proporzionale al numero di punti nella mesh ma anche al numero dei passi temporali.

Fino a un decennio fa, solo le discipline scientifiche classiche, come la chimica e la fisica, si avvalevano dei supercalcolatori per indirizzare problemi computazionali puntuali ma, nel corso degli ultimi anni, le simulazioni numeriche sono diventate sempre più una metodologia di indagine comune nei più diversi campi scientifici e indirizzano problemi

su larga scala, in modo globale. Alcuni esempi riguardano la simulazione biomolecolare applicata a proteine e ad altri sistemi complessi (Figura 3); la turbolenza in fluidodinamica; il sequenziamento in genomica; la simulazione globale dell'ecosistema geofisico terrestre ottenuto integrando diversi sottosistemi (atmosfera, oceani, flussi di calore, riscaldamento chimico, magnetismo terrestre); la simulazione di sistemi e processi biologici, multi-componenti, fino ad arrivare alla possibilità di descrivere interi organismi e addirittura popolazioni; le nanotecnologie, che affrontano lo studio di nano-dispositivi con specifiche funzionalità [3].

Per comprendere meglio la complessità dei problemi che si possono affrontare, introduciamo con maggior dettaglio due applicazioni in un settore non tradizionale dell'HPC come quello della Scienza della Terra.

2.1. Terremoti virtuali

Negli ultimi anni la teoria e le applicazioni della propagazione delle onde acustiche hanno permesso di indirizzare nuove metodologie computazionali in campi quali la sismologia, l'oceanografia, la meteorologia l'acustica, ma anche l'ingegneria, le scienze dei materiali le scienze mediche ecc..

La simulazione del fenomeno tridimensionale completo della propagazione delle onde

su strutture realistiche, con un adeguato livello di dettaglio, permette di studiare sistemi complessi in campi diversi e a diverse scale come, per esempio, la prospezione geofisica a diverse scale (analisi e gestione dei giacimenti) o il monitoraggio del rischio sismico e vulcanico.

Nella sismologia computazionale, metodi di tomografia ad alta risoluzione permettono di simulare la propagazione delle onde sismiche a varie scale, da quella locale (dimensione di una città) a quella globale, includendo man mano fenomeni complessi quali l'evoluzione della sorgente sismica e l'interazione con le strutture geologiche. Questo permette una valutazione del rischio sismico in ambiti specifici fino allo studio della struttura interna della terra.

Questi problemi diventano vere e proprie sfide computazionali quando si aumenta la complessità dei modelli coinvolti poiché in tutti i metodi attualmente in uso (differenze finite, elementi finiti, metodi spettrali ecc.) la risoluzione è funzione della scala introdotta. Inoltre, all'interno dell'analisi del rischio sismico, assume un peso considerevole lo studio probabilistico degli effetti locali, che comporta la definizione di tutti i possibili, o quantomeno probabili, scenari di rischio; questi metodi richiedono usualmente diverse centinaia di simulazioni per raggiungere un buon grado di attendibilità.

Una sfida ulteriore è rappresentata dalla grande quantità di dati coinvolti; un esempio concreto della dimensione del problema è dato dalla modellazione del bacino di Los Angeles, il cui scenario completo in quattro dimensioni è dell'ordine di 40 Tbytes.

Progetti di questo tipo, che riguardano modelli realistici, richiedono risorse importanti sia in termini di innovazione algoritmica (per migliorare la complessità computazionale in funzione dell'accuratezza) che di supercalcolo. Esempi notevoli sono i tre grandi progetti attivi al SCEC, (*Southern California Earthquake Center*): TeraShake per la simulazione della propagazione delle onde sismiche, CyberShake, una piattaforma computazionale per modellare la forma d'onda 3D e sviluppare le curve probabilistiche di rischio sismico di prossima generazione, ed infine DynaShake per la simulazio-

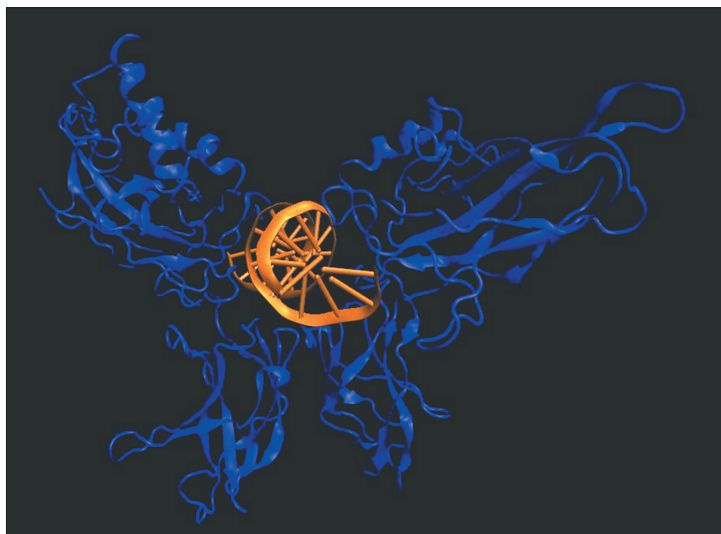


FIGURA 3

Simulazione della dinamica molecolare di un complesso proteico (blu) con il DNA (arancio). La simulazione per un sistema di circa 150.000 atomi, che dura 13 ns, ha richiesto circa 10.000 h di CPU su un cluster Linux

ne delle rotture dinamiche a livello sismico e la parametrizzazione cinematica degli epicentri². L'esecuzione di questi progetti richiede potenze di calcolo considerevoli quali l'utilizzo di un sistema IBM BlueGene a 40.000 processori.

2.2. Vulcani simulati

Un esempio più vicino a noi è rappresentato dal progetto Exploris³. Nell'ambito di questo progetto è stato formulato il modello matematico 3D delle eruzioni esplosive e della dispersione di ceneri vulcaniche. Per la prima volta al mondo sono state realizzate le simulazioni tridimensionali dei processi di dispersione delle ceneri nell'atmosfera, del collasso della colonna vulcanica e della formazione di colate piroclastiche lungo le pendici del vulcano [4].

Tali modelli, utilizzando notevoli potenze computazionali, hanno consentito di rappresentare i fenomeni vulcanici in modo molto accurato nonché di quantificare meglio le azioni pericolose ad essi associate. Assieme alla temperatura, le simulazioni hanno permesso di studiare anche molte altre grandezze che caratterizzano l'eruzione, come la densità dei piroclasti e dei gas, la pressione, la velocità e la direzione del flusso piroclastico. Le simulazioni più accurate hanno avuto come soggetto l'eruzione esplosiva più probabile per il Vesuvio, per la quale è stato utilizzato un dominio di simulazione comprendente un'area di 12 km di lato (Vesuvio più territorio circostante) e che si estende fino alla quota di 8 km; il dominio è stato discretizzato su una griglia cartesiana di dimensione 200³ a risoluzione variabile (dai 20 m in prossimità del cratere a 100 metri per le celle più lontane).

L'eruzione è stata seguita per 30 min di tempo

reale utilizzando una discretizzazione temporale di 0.1 s. Per portare a compimento questa simulazione sono stati utilizzati 450 processori del sistema IBM SP Power5 del CINECA per un totale di 180 h (80.000 h/cpu), equivalenti a 10 anni di calcolo utilizzando un solo processore [5].

Le simulazioni su grande scala, come quelle sopra citate, producono un'enorme mole di dati, circa 0.5 Tbyte per ogni simulazione, che devono essere analizzati e visualizzati per essere comprensibili agli esperti. Per la visualizzazione sono stati realizzati due strumenti: uno dedicato all'analisi quantitativa, che si interfaccia direttamente con i dati della simulazione (pressione, temperatura, concentrazione e velocità delle particelle e dei gas); l'altro, dedicato alla analisi qualitativa ed alla integrazione con altre tipologie di dati provenienti da sorgenti diverse. Tale strumento consente di integrare in un'unica visualizzazione: i dati sui flussi piroclastici provenienti dalla simulazione; un modello fotorealistico del suolo (*orthofoto*) e il suo profilo topografico (DEM - *Digital Elevation Model*); e le informazioni geografico-urbanistiche (GIS). Grazie a questa integrazione è possibile, ad esempio, visualizzare contemporaneamente la densità abitativa o la rete viaria, sovrapposta alla temperatura del flusso piroclastico, come si vede nella figura 4.

3. GRANDI QUANTITÀ DI DATI

Come mostrano gli esempi precedenti, oggi le simulazioni riguardano problemi reali complessi che vengono affrontati globalmente con approcci multidisciplinari. È distante ormai il caso in cui lo studio della aerodinamica di un veicolo non poteva essere gestito in modo globale perché la simulazione era troppo onerosa (sia in termini di tempi di calcolo che di memoria richiesta) e si analizzava solo una metà del veicolo, sfruttando la probabile simmetria dell'altra metà.

Oggi, si possono affrontare simulazioni complesse sia perché sono presenti sistemi di supercalcolo potenti, ma anche perché sofisticati dispositivi di acquisizione dati quali sensori, radar, satelliti, TAC, scan-

² <http://www.scec.org/>

³ EXPLOsive Eruption RISk and Decision Support for EU Populations Threatened by Volcanoes. È un progetto triennale finanziato dalla Comunità Europea e coordinato dalla sezione di Pisa dell'Istituto Nazionale di Geofisica e Vulcanologia. CINECA ha collaborato nell'attività di sviluppo dei modelli numerici e nell'integrazione e visualizzazione dei dati. Tutta l'attività computazionale si è svolta sul sistema IBM SP Power 5 di CINECA.

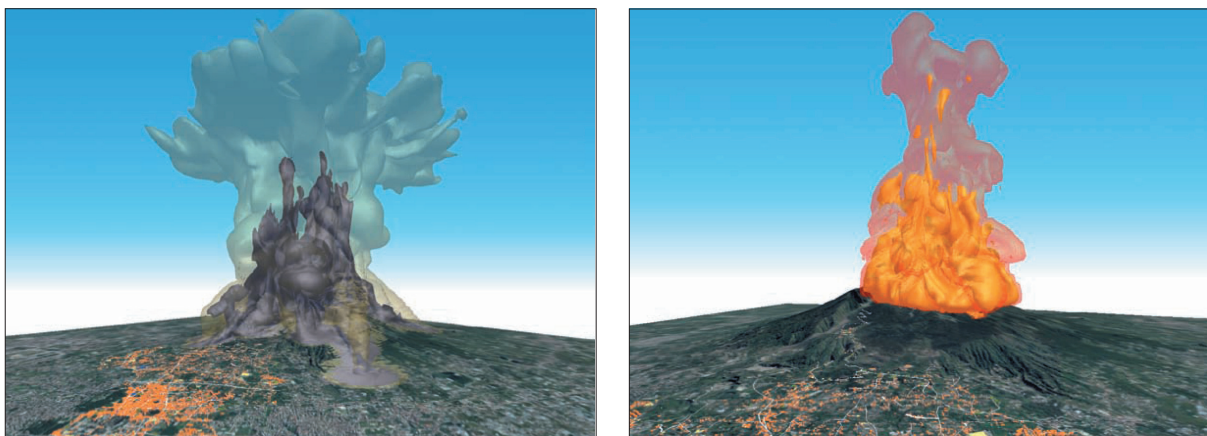


FIGURA 4

Il Vesuvio pochi minuti dopo l'inizio dell'eruzione virtuale; due diverse gradazioni di rosso evidenziano le isosuperfici della temperatura, a 100 (superficie esterna) e 350 (superficie interna) gradi Celsius. Le isosuperfici grigie rappresentano due diversi livelli di concentrazione del materiale eruttivo

ner, micro-arrays⁴ ecc. producono con facilità enormi quantità di dati, indispensabili per la formulazione di modelli più rigorosi e complessi.

Le simulazioni numeriche, che si appoggiano su modelli complessi e dati di input consistenti, a loro volta producono moli di dati altrettanto grandi che spesso richiedono una post-elaborazione al fine di poter recuperare tutta l'informazione utile, e che vanno conservati per successive elaborazioni.

La produzione di dati sta subendo una grande accelerazione e il volume di dati prodotti cresce sia come dimensioni che in termini di complessità: si stima che il volume di dati prodotti a livello mondiale raddoppi ogni anno. Occorre quindi studiare nuove metodologie per organizzare, manipolare ed analizzare i dati, nonché strumenti per garantirne l'accesso in modo efficiente, ed ottimizzare così il processo di produzione della conoscenza.

In molti campi della ricerca si assiste quindi a una forte spinta verso la creazione di infrastrutture capaci di garantire la massima diffusione e fruibilità delle informazioni. Per esempio, la comunità internazionale astrofisica da alcuni anni lavora alla creazione del *Virtual Observatory* (VO), definito come: “*an enabling and coordinating entity to foster the development of tools, protocols, and collaborations necessary to realize the full scientific potential of astronomical databases in the coming decade*” [7].

I sistemi di supercalcolo non possono quindi limitarsi ad offrire solo potenza di calcolo, ma diventano vere e proprie infrastrutture in grado di fornire una serie di servizi complementari per la gestione e la fruizione efficace dei dati prodotti, limitando la movimentazione degli stessi su reti spesso troppo lente o inaffidabili.

Inoltre è sempre più importante tenere se-

⁴ I micro-arrays, o matrici ad alta densità, sono una tecnica che, sfruttando le caratteristiche peculiari della doppia elica del DNA, ha aperto di fatto la strada alla possibilità di analizzare i profili di espressione genica di un intero organismo. In generale, un esperimento di analisi dei profili di espressione fornisce come risultato una matrice di dati in cui le righe rappresentano i geni monitorati e le colonne corrispondono alle diverse condizioni sperimentali, quali punti temporali, condizioni fisiologiche, tessuti. Ogni elemento della matrice rappresenta quindi il livello di espressione di un particolare gene in uno specifico stato fisiologico.

La gestione e l'interpretazione dei dati generati dalle matrici ad alta densità rappresenta un aspetto fondamentale di questa tecnologia. Infatti, queste matrici diventano sempre più grandi e richiedono tecniche di analisi statistiche avanzate, quali il data mining, che impegnano risorse di calcolo e di memorizzazione importanti. Nel caso dei profili di espressione genica, le tecniche di data mining rappresentano un utile strumento per identificare ed isolare particolari pattern di espressione che di fatto rappresentano delle vere e proprie impronte digitali genetiche di un determinato stato fisiologico [6].

parata l'attività di produzione dei dati da quella più vasta di analisi. Occorre quindi strutturare i dati prodotti attraverso metadati efficaci in grado di descrivere i dati stessi, secondo metodologie standard, in modo da poterli accedere, analizzare e visualizzare attraverso strumenti software appropriati, ma al tempo stesso poterli condividere efficientemente nell'ambito di discipline scientifiche diverse.

4. ARCHITETTURE PER IL SUPERCALCOLO

L'introduzione reale dei sistemi di supercalcolo avvenne alla fine degli anni '70, quando i primi sistemi vettoriali divennero un supporto concreto ed efficace alla comunità scientifica per realizzare simulazioni numeriche importanti. Da allora l'evoluzione di tali sistemi è stata rapida e impetuosa sia in termini di potenza che di innovazione architetture e il loro utilizzo si è radicato in differenti contesti applicativi.

Le tappe di questa evoluzione architetture hanno riguardato dapprima i sistemi vettoriali mono-processore (come per esempio, il Cray 1, e il CDC Cyber 203) poi, a partire dai primi anni '80, i sistemi vettoriali multi-processore a memoria condivisa. Si trattava di sistemi SMP (*Symmetric Multi Processor*) dove il singolo processore aveva funzionalità vettoriali, come, per esempio, i sistemi Cray della serie X-MP e Y-MP, i sistemi NEC della serie SX e, successivamente, i sistemi Cray C90 e i sistemi IBM della serie 3090 dotati di "vector facilities". Nella seconda metà degli anni '80 comparvero i primi sistemi paralleli a memoria distribuita, come per esempio, l'Intel *iPSC/1* che utilizzava microprocessori Intel 80826 interconnessi da una rete *Ethernet* secondo una topologia ipercubo o la *Connection Machine CM2* della *Thinking Machine Corporation* [8].

A partire dai primi anni '90 i sistemi *Massively Parallel Processors* (MPP) si sono imposti sul mercato HPC. La CM5, il Cray T3D e il Cray T3E sono gli esempi di maggior successo.

Da allora si è consolidata la presenza di sistemi HPC costituiti da *cluster* di nodi SMP basati su processori RISC, o microprocessori *off the shelf* X86-32 e X86-64, intercon-

nessi da reti veloci. Queste architetture, seguendo la rapida evoluzione tecnologica dei microprocessori oggi caratterizzano la maggior parte dei sistemi HPC sul mercato. La distinzione tra sistemi convenzionali e supercomputer diviene molto più sfumata. Un sistema convenzionale, opportunamente "clusterizzato", può diventare di fatto un supercomputer. Nella attuale classifica della Top 500⁵ il numero di sistemi basato su processori *commodity* rappresenta ormai oltre l'80% del totale.

Le ragioni della rapida affermazione dei cluster nel mondo HPC sono da ricercarsi nella disponibilità di componenti *commodity* di prestazioni elevate e a costi contenuti. Questo ha fatto sì che a parità di potenza di picco un cluster potesse costare fino a 10 volte meno di un supercalcolatore tradizionale. Una tale differenza rende più difficile giustificare la scelta di supercalcolatori tradizionali, anche a fronte della maggior robustezza ed affidabilità di questi ultimi.

4.1. Classificazione delle architetture parallele

La maggior parte dei sistemi paralleli attuali ricade nella classe MIMD (*Multiple Instruction stream Multiple Data stream*) secondo la classica tassonomia di Flynn [9], che distingue i sistemi paralleli in base al flusso delle istruzioni e a quello dei dati interno alla architettura stessa. La classe MIMD racchiude sistemi paralleli costituiti da un insieme di processori indipendenti, ciascuno dei quali è governato da una propria unità di controllo.

Una modalità forse più utile di classificare le architetture parallele è in base al loro modello di memoria: sistemi a memoria condivisa e sistemi a memoria distribuita, come schematizzato nella figura 5. Nei sistemi a memoria condivisa i processori condividono la memoria centrale e vedono un singolo spa-

⁵ La classifica top 500 (www.top500.org) fornisce la graduatoria dei 500 calcolatori più potenti a livello mondiale. La misura viene fatta in base alle prestazioni raggiunte nella soluzione di un sistema lineare denso $Ax = b$, ricorrendo alla libreria LINPACK (www.netlib.org). Tale classifica viene pubblicata due volte all'anno a partire dal 1993.

zio degli indirizzi. I sistemi a memoria condivisa si distinguono a loro volta in sistemi SMP (*Symmetric Multi Processors*), detti anche sistemi UMA (*Uniform Memory Access*), dove l'accesso alla memoria è uniforme per ogni processore, e sistemi NUMA (*Non-Uniform Memory Access*) per i quali il tempo di accesso alla memoria dipende dalla localizzazione locale o remota dei dati.

Nei sistemi a memoria distribuita, i processori sono connessi tramite una rete di interconnessione, ogni processore può indirizzare direttamente solo la propria memoria locale e occorre un protocollo a scambio di messaggi per scambiare informazioni tra un processore e l'altro. Tali sistemi sono detti sistemi NORMA (*No Remote Memory Access*). Ormai da oltre un decennio, i sistemi a memoria distribuita non interconnettono più il singolo processore ma nodi, costituiti da sistemi SMP: i nodi, spesso costituiti da processori *off-the-shelf* e memoria DRAM

commodity, sono interconnessi da reti più o meno performanti, anch'esse spesso *commodity* [10].

A questi modelli architetturali corrispondono altrettanti modelli di programmazione parallela: il *modello a memoria condivisa*, che sfrutta la condivisione della memoria per lo scambio delle informazioni tra i diversi thread che operano sui processori del sistema parallelo; il *modello a memoria locale*, dove lo scambio di informazioni e la sincronizzazione tra i processi viene gestita per mezzo del paradigma *message passing*. In entrambi i casi la programmazione avviene per mezzo di linguaggi ad alto livello di tipo procedurale o ad oggetti (C, C++, Fortran) con il supporto di API specifiche per la gestione del parallelismo: OpenMP⁶ per il modello a memoria condivisa e MPI⁷ per quello a memoria locale [2, 11].

4.2. Trend dei sistemi HPC

Se si considera l'andamento della classifica Top500 dal 1993, anno della sua prima pubblicazione, ad oggi, si traggono utili informazioni circa la rapida evoluzione dei sistemi per il supercalcolo. Nel 1993, il supercomputer al primo posto raggiungeva 59,7 Gflop/s mentre quello al 500° posto raggiungeva i 422 Mflop/s; nel 2007 quello al primo posto raggiunge i 478 Tflop/s contro i 5,9 Tflop/s di quello all'ultimo posto, con un incremento in termini di prestazioni di circa un fattore 8.000 per il sistema al primo posto e di oltre 13.000 per quello in fondo alla classifica.

Dal 2005 entrano nella graduatoria solo sistemi con una potenza superiore al Tflop/s e nel 2007 il sistema al 500° posto ha una potenza di 5,9 Tflop/s. Questo sistema nella lista precedente (giugno 2007) sarebbe stato al 255° posto e ben 245 sistemi della lista precedente risultano troppo poco potenti per avere ancora un posto nella lista attuale.

La figura 6 mostra l'andamento negli anni della potenza effettiva, rispettivamente del sistema al 1° posto e di quello al 500° della Top500. Nel grafico è riportata anche la potenza effettiva, negli anni, del sistema italiano più potente: solo sistemi installati in

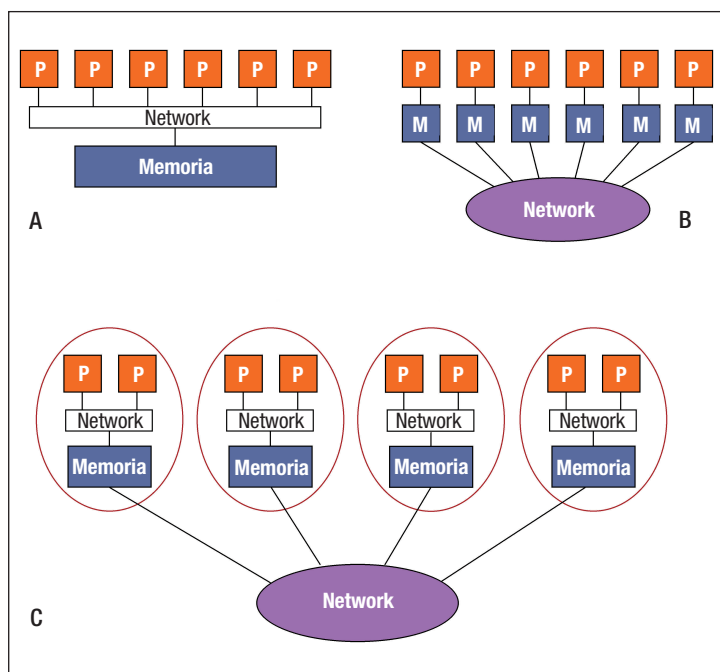


FIGURA 5

Schema architetturale dei sistemi paralleli.

A) Architettura a memoria condivisa: tutti i processori indirizzano l'intera memoria (singolo spazio degli indirizzi). **B)** Architettura a memoria distribuita: ogni processore indirizza solo la propria memoria locale; per scambiare dati tra i processori occorre adottare un paradigma a scambio di messaggi.

C) Architettura a memoria distribuita costituita da nodi SMP: all'interno del nodo si può adottare un paradigma a memoria condivisa mentre per comunicare all'esterno del nodo occorre un paradigma a scambio di messaggi

⁶ <http://www.openmp.org>

⁷ <http://www.mcs.anl.gov/mpi/>
<http://www.mpi-forum.org/>

CINECA hanno guadagnato questa posizione nella Top 500.

Nella lista di novembre 2007, su 500 sistemi, ben 354 usano processori Intel (70,8%), 78 sistemi (15,6%) usano processori della famiglia AMD Opteron e 61 sistemi (12,2%) usano processori IBM Power.

I sistemi cluster rappresentano l'architettura più comune nella Top500 con una presenza di ben 406 sistemi. Per quanto riguarda le reti di interconnessione la tecnologia Infiniband⁸ accresce la sua presenza con una presenza di 121 sistemi, mentre Gigabit Ethernet è ancora la tecnologia di interconnessione più utilizzata (270 sistemi).

Gli Stati Uniti continuano ad essere i leader di questa classifica con 284 sistemi installati, contro i 149 sistemi dell'Europa. Ma un nuovo trend geografico si impone in modo significativo in questi ultimi anni: la crescita dei si-

stemi nel continente Asiatico che vede oggi 50 sistemi, di cui 20 sono in Giappone, 10 in Cina e 9 in India.

Tra le nazioni europee, il primo posto è ricoperto dall'Inghilterra con ben 48 sistemi installati, seguita dalla Germania con 31 sistemi. L'Italia è presente con soli 6 sistemi.

Il primo di questi è al 48° posto ed è installato in CINECA. Si tratta di un sistema cluster linux BladeCenter Xeon dual core (5120 core a 3.0 GHz) con una memoria RAM complessiva di 20 Tbyte e una rete di interconnessione full bisection Infiniband 4X.

4.3. Evoluzione della tecnologia dei microprocessori

Lo sviluppo dei microprocessori mostra alcune linee di tendenza pressoché indipendenti dal singolo costruttore e dal disegno stesso del processore che valgono sia per processori



FIGURA 6

Andamento (in scala logaritmica) delle prestazioni dei sistemi nella lista Top 500 dall'inizio della pubblicazione della classifica (Giugno 1993) ad oggi, rispettivamente del sistema al 1° posto (linea azzurra), del sistema al 500° posto (linea arancio) e del sistema più potente installato in Italia (linea verde). Tutti i sistemi italiani sono stati installati in CINECA

⁸ www.infinibandta.org



CISC (*Complex Instruction Set Computer*), RISC (*Reduced Instruction Set Computer*), EPIC (*Explicitly Parallel Instruction Computing*) e anche per processori *custom* (vettoriali) [10].

La prima e più importante tendenza è quella relativa al mantenimento del processo di miniaturizzazione basato su tecnologia al silicio per i prossimi 5-6 anni. In particolare i processori si avviano a una nuova generazione basata sul processo tecnologico con scala litografica a 65 nm, contro i 90 nm dei processori attuali. È già prevista la costruzione di linee di produzione con processi tecnologici a 45 nm e 32 nm. Questo consentirà ai produttori di rimanere sulla curva delineata dalla legge di Moore che prevede un raddoppio della densità dei circuiti ogni 18-24 mesi, che è esattamente l'incremento che si ha nel passaggio da una scala di integrazione alla successiva.

Per quel che riguarda le frequenze di clock non ci dovremmo invece aspettare incrementi spettacolari come quelli a cui abbiamo assistito negli scorsi anni, questo soprattutto per i problemi connessi al calore sviluppato dalle correnti parassite che, con la tecnologia attuale, scalano col quadrato della frequenza. Per questo i produttori stanno investendo molto nella ricerca di materiali ad alta costante dielettrica (k) per diminuire le correnti parassite. Come conseguenza della impossibilità di elevare il clock oltre un certo livello i produttori, per aumentare la potenza allo stesso passo dell'integrazione (raddoppio ogni 18-24 mesi) si sono orientati verso la produzione di microprocessori *multi-core*, ovvero contenenti più CPU. Questa è ormai una tendenza consolidata, che al momento ha determinato la comparsa di microprocessori *dual-core* e *quad-core*, e che nel prossimo futuro porterà ad avere sul mercato microprocessori con un numero sempre più elevato di core [10].

Questo fatto combinato con l'altrettanto consolidata tendenza ad avere sul mercato sistemi SMP con 2, 4, 8 microprocessori, farà crescere nei sistemi di prossima generazione il numero di CPU per nodo, con conseguente stress del sottosistema di memoria, che risulterà sempre più critico per le prestazioni del sistema globale.

Risulta quindi naturale che vi sia anche una tendenza generale a dotare i sistemi di memoria cache di terzo livello (L3) di dimensioni

generose e condivisa tra i processori SMP. Mentre la cache di secondo livello (L2) sarà condivisa a livello di microprocessore e quella di primo livello (L1) a livello di singolo core. L'organizzazione gerarchica della memoria sarà quindi sempre più marcata e saranno presenti almeno 5 livelli: Registri, L1, L2, L3, RAM. La gestione degli accessi alla memoria dovrà essere efficiente affinché le applicazioni riescano a sfruttare appieno la potenza del processore.

È opportuno sottolineare come la linea di processori RISC Power di IBM abbia anticipato tutte queste tendenze, essendo stata la prima ad introdurre processori multi-core e cache di livello 3.

Infine è ormai definitivo il passaggio di tutte le architetture a microprocessori all'indirizzamento a 64 bit, che consente di aumentare la quantità di RAM indirizzabile dal singolo processore.

La tendenza ad aumentare il numero di core all'interno dei microprocessori, è documentata dalla disponibilità di un processore Intel Xeon e di un processore AMD Opteron *quad-core* per singolo chip, entrambi prodotti con un processo tecnologico a 65 nm. Osserviamo che la disponibilità di processori *quad-core*, compatibili con i *dual-core*, fa sì che a parità di volume fisico occupato, si possa raddoppiare la potenza di elaborazione di un cluster pre-esistente, semplicemente sostituendo i processori.

Altre novità per i sistemi ad alte prestazioni sono l'evoluzione del processore Power IBM e il Cell processor.

Il Power 6 IBM si presenta come la naturale evoluzione del Power 5 e sarà presentato inizialmente solo come dual core, e con un clock molto elevato (4.7 GHz). Introdurre solo un dual core, in questo caso può essere un vantaggio perché permetterà di avere un miglior rapporto potenza/banda di memoria [12]. Poco si sa ancora del Power 7 successore del Power 6. Potrebbe uscire tra un paio di anni e potrebbe essere *pin compatible* con altri processori già presenti sul mercato in modo da permettere l'integrazione di sistemi disomogenei.

Il Cell processor IBM (derivato dal processore della PlayStation di terza generazione) presenta un disegno architetturale innovativo (*Syner-*

gic Processor)⁹. In breve, l'unità di elaborazione è costituita da un processore Power e da 8 co-processor specializzati per operazioni floating-point, integrati in una architettura SIMD in cui tutti i co-processor eseguono la stessa istruzione su dati diversi. Tale architettura si presta al calcolo tecnico scientifico, e sulle operazioni a singola precisione il processore può dimostrare una potenza prossima ai 250 Gflop/s, mentre sulla doppia precisione la stima è di 25 Gflop/s. Sebbene molto promettente, questo tipo di processore potrà, nel breve periodo, raggiungere prestazioni non adeguate alle sue potenzialità perché richiede una sostanziale revisione delle applicazioni e del supporto dei compilatori.

4.4. Evoluzione delle reti di interconnessione

Le reti di interconnessione dei sistemi di calcolo ad alte prestazioni devono essere all'altezza dei processori che collegano, e quindi un calcolatore parallelo deve essere corredato da una rete di interconnessione adeguata in termini di *bandwidth* (banda passante) e *latenza* [10].

Al giorno d'oggi esistono varie reti di interconnessione utilizzate per i sistemi HPC, che possiamo dividere in reti proprietarie, e quindi di uso esclusivo del fornitore dei sistemi e spesso integrate con il sistema stesso, e in reti standard *off-the-shelf*, che i fornitori integrano nelle loro soluzioni spesso utilizzando componenti di fornitori terzi.

Alcuni esempi di reti di interconnessione proprietarie sono:

□ l'*High Performance Switch* (HPS) - Federation adottato da IBM nei sistemi della serie SP Power. HPS è uno *switch flat* che collega i processori direttamente, senza passare attraverso hardware addizionale come l'interconnessione PCI;

□ la rete toroidale 3D dei sistemi paralleli Cray XT4, composta da router proprietari denominati *SeaStar2*¹⁰. I processori AMD Opteron di XT4 sono collegati ai SeaStar2, tramite HyperTransport a 6.4 GB/s e, in prospettiva, HyperTransport 3.0;

□ le reti del sistema Blue Gene/L: Il sistema è composto da nodi biprocessore (PowerPC 440

e coprocessore matematico) interconnessi da 3 reti di comunicazione adibite al calcolo, con caratteristiche e velocità differenti: rete toroidale tridimensionale (per le comunicazioni punto-punto), rete collettiva ad albero binario (supporto hardware per le operazioni collettive) e una rete barrier di pura sincronizzazione. Da tempo sono presenti sul mercato reti di interconnessione standard *off-the-shelf* come Myrinet, Qsnet, SCI Dolphin, ma recentemente grande aspettativa è riposta sulle tecnologie Infiniband¹¹ e 10 Gbit Ethernet (GbE), tanto che le case produttrici di tali reti offrono prodotti basati su queste tecnologie.

In futuro sarà probabilmente definito un protocollo per la 40-GbE o per la 100-GbE, ma il dato vero è che queste reti difficilmente si presteranno nell'immediato futuro come reti di interconnessione dei cluster per il calcolo ad alte prestazioni: il loro obiettivo primario è quello di sostituire la GbE in ambito enterprise nelle situazioni in cui questa non risulti sufficiente.

InfiniBand (IB) è stata creata come standard "open" per il supporto ad una architettura di I/O ad alte prestazioni. Lo standard nasce quindi orientato all'I/O, ma l'interconnessione oggi giorno è largamente utilizzata come rete di connessione tra nodi di calcolo per la realizzazione di sistemi paralleli. Il vantaggio di essere "open" risiede nel fatto che è possibile utilizzare in modo cooperativo hardware e software di produttori diversi all'interno della stessa rete, così come avviene per altri dispositivi (Ethernet, SCSI, Fiber Channel ecc.).

Lo standard IB corrente supporta le bandwidth 1X (2.5 Gbit/s), 4X (10 Gbit/s) e 12X (30 Gbit/s). Inoltre lo standard prevede anche trasmissioni DDR e QDR che permettono di raddoppiare e quadruplicare tali valori. Quindi lo standard prevede di fatto reti di trasmissione in grado di trasmettere fino a 120 Gbit/s.

Quanto previsto dallo standard è correntemente implementato principalmente da 4 fornitori: Mellanox, Cisco, Silver Storm e Voltaire. Le implementazioni si differenziano per vari fattori quali: bandwidth supportata, numero di porte dello switch, tipo di interconnessione (rame o fibra ottica), tipo e caratteristiche dell'Host Channel Adapter.

⁹ <http://www.research.ibm.com/cell/>

¹⁰ <http://www.cray.com/>

¹¹ <http://www.infinibandta.org/>

Per esempio, lo switch monolitico full bisectional Cisco, permette il collegamento di 288 porte 4X DDR, e una full bisection bandwidth di 11.52 Tbit/s. Con sistemi di questo tipo è possibile realizzare cluster composti da migliaia di nodi di calcolo, la cui dimensione è limitata principalmente dalla massima lunghezza dei cavi (ovvero dal costo e dalla stabilità, visto che nel caso di collegamenti in fibra ottica, non ci sono limitazioni importanti relativamente alla lunghezza dei cavi).

4.5. Evoluzione delle architetture per il supercalcolo

L'affermazione delle architetture *cluster* basate prevalentemente su componenti quali processori e rete *off the shelf* è sempre più radicata in ambito HPC. Tali sistemi saranno caratterizzati da prestazioni sempre più elevate, in funzione della disponibilità di sistemi multi-core che caratterizzeranno i processori dell'immediato futuro.

Accanto a queste architetture, emergono anche architetture di supercalcolo *special purpose*, e architetture eterogenee che indirizzano meglio il *capability computing*. Il *capability computing*, contrapposto al *capacity computing*, indirizza quei problemi *challenge* che richiedono architetture potenti e ben bilanciate, in termini di potenza del processore, capacità di memoria e ampiezza della banda di interconnessione.

Un esempio di architettura *special purpose* di successo è il sistema IBM BluGene/P¹² (BG/P) successore di BluGene/L inizialmente progettato per lo studio del problema del folding delle proteine, poi commercializzato anche in altri ambiti applicativi. La caratteristica principale di questo sistema è quella di aver spinto il parallelismo interno all'estremo, ben oltre quello ottenibile con i cluster tradizionali, contenendo contemporaneamente il consumo energetico

e l'occupazione fisica dello spazio. A differenza dei cluster i nodi di BG/P non ospitano un sistema operativo "completo" e sono collegati da più reti dedicate a diverse funzioni, tutto per aumentare il sincronismo a livello applicativo. I principali punti deboli di BG/P sono sia la scarsa disponibilità di memoria sul singolo nodo che la difficoltà di adattare le applicazioni alle caratteristiche della macchina.

BG/P, grazie all'utilizzo del processore PowerPC 450 (quad-core), raggiungerà il Tflop/s di picco, rispetto ai 360 Gflop/s di BG/L. L'evoluzione è rappresentata da BG/Q che, nel 2010, potrebbe adottare architettura Power o Cell e raggiungere i 10 Tflop/s.

Un'altra architettura di rilievo per il supercalcolo è quella offerta da Cray con i sistemi XT5, in grado di tenere assieme tecnologia di microprocessore general purpose (processore AMD Opteron quad-core) e tecnologia di rete dedicata.

Il Cray XT5 è il punto attuale di una *roadmap* Cray più ampia e tesa a produrre sistemi in grado di raggiungere il Pflop/s sostenuto su applicazioni reali, prima del 2010. Il primo passo in questa direzione è la piattaforma selezionata dal U.S. Department of Energy's (DOE) per l'*Oak Ridge National Laboratory* (ORNL) che, basata sull'evoluzione XT5, raggiungerà il Pflop/s di picco entro il 2009. Sarà un sistema con 23936 processori AMD quad-core a 2.8 GHz (95744 cores).

Cray, con il progetto Cascade¹³, sta progettando una nuova generazione di tecnologia HPC che realizza il modello di *adaptive supercomputing*. Tale modello fa perno attorno al concetto di architettura adattiva che integra in un unico sistema multi-architetturale tecnologie di elaborazione scalari, vettoriali, multithreading e di computing riconfigurabili (FPGA¹⁴). Il progetto Cascade prevede anche

¹² <http://www-03.ibm.com/systems/deepcomputing/bluegene/>

¹³ <http://www.cascade.cray.com>

¹⁴ Un FPGA (*Field Programmable Gate Array*) è un circuito integrato *general-purpose* che può essere riprogrammato, anche dopo essere stato inserito in un sistema. La programmazione consiste nel caricare un programma di configurazione chiamato *bitstream* in un'apposita memoria statica. Come il codice binario per un processore, il *bitstream* è prodotto da strumenti di compilazione che traducono le astrazioni di alto livello definite dal progettista.

Un FPGA si presenta come un *array bidimensionale* e configurabile di risorse utili ad implementare una vasta gamma di funzioni aritmetiche e logiche quali ricerca, ordinamento, elaborazione di segnali, manipolazione di immagini, crittografia, correzione degli errori, codifica/decodifica, generazione di numeri casuali ecc.. Un FPGA può essere utilizzato come co-processore per accelerare le applicazioni o parti di esse.

lo sviluppo di un adeguato corredo software che spazia dai sistemi operativi ai compilatori agli ambienti di sviluppo. L'obiettivo ambizioso è quello di realizzare supercomputer che si possono adattare alle applicazioni, anziché forzare le applicazioni ad adattarsi al supercomputer.

Cascade è la risposta Cray all'HPCS *program* lanciato dal DARPA¹⁵ per finanziare lo sviluppo di una nuova generazione di supercomputer di classe Pflop/s, facili da programmare e in grado di gestire un'ampia gamma di applicazioni (general purpose).

Solo Cray e IBM hanno raggiunto l'ultima fase del programma HPCS e nel novembre 2006 sono state finanziate rispettivamente con 250 e 245 milioni di dollari per arrivare a produrre sistemi Pflop/s con tali caratteristiche, entro il 2010.

5. VERSO NUOVE SFIDE

Come si è visto, la disponibilità di sistemi paralleli multi-core con un elevato numero di processori è realtà e ci si aspetta che il numero di core per chip raddoppi ad ogni generazione di processori. Questi sistemi pongono sfide importanti per il programmatore. Il *multi-core* non è una nuova versione di sistema SMP e i *core* non si possono considerare del tutto indipendenti ai fini della programmazione infatti i *core* condividono sul chip risorse in modo diverso da come fanno i processori SMP tradizionali. Questa situazione sarà ancora più complessa in un prossimo futuro quando altre componenti non standard verranno integrate a livello architetturale, come tipi di core differenti e acceleratori hardware quali GPU¹⁶ e FPGA. Bisognerà quindi introdurre tecniche innovative per gestire il problema dell'accesso ottimale ai dati, nelle diverse gerarchie di memoria che i sistemi *multi-core* inevitabilmente adotteranno.

Occorrerà introdurre nuove tecniche di pro-

grammazione per scalare su numeri elevati di processori senza perdere di efficienza negli algoritmi. Uno degli aspetti critici nell'utilizzo di risorse computazionali avanzate è la scalabilità che caratterizza gli algoritmi numerici. Questo aspetto riguarda non solo la scalabilità in termini di tempo di esecuzione in funzione del numero di processori ma anche l'interdipendenza tra il tempo di esecuzione, le richieste di memoria e la dimensione computazionale del problema.

Per almeno due decenni i programmatori in ambito HPC si erano abituati al fatto che ad ogni nuova generazione di microprocessori corrispondeva un aumento delle prestazioni del codice pre-esistente anche senza pesanti ottimizzazioni. Ora invece siamo di fronte a una nuova sfida che vede il programmatore stimolato a fare un salto innovativo nello studio di nuovi algoritmi e di metodologie capaci di sfruttare adeguatamente le nuove architetture parallele per risolvere problemi non pensabili anche solo un po' di anni fa, nei campi applicativi più diversi.

Di seguito sottolineiamo alcuni aspetti che occorre affrontare in questo processo innovativo:

□ È importante semplificare il processo di sviluppo di applicazioni in grado di raggiungere alte prestazioni sulle nuove architetture, e sviluppare nuovi approcci e modelli di programmazione sufficientemente flessibili e adattivi per le nuove architetture. In questo senso si muove il progetto DARPA per la realizzazione del Petaflop/s computer entro il 2010, che prevede anche un'attività di ricerca e sviluppo per la realizzazione di un linguaggio parallelo per le scienze computazionali. Cray, IBM e SUN in collaborazione con partner accademici stanno rispondendo a questa richiesta con lo sviluppo di 3 nuovi linguaggi paralleli, chiamati rispettivamente Chapel [13], X10¹⁷ e Fortress¹⁸. I 3 linguaggi, di tipo *object-oriented*, si basano tutti sul modello

¹⁵ <http://www.highproductivity.org>

¹⁶ Graphical Processing Unit. Il modello G80 prodotto da NVIDIA ha 128 processing units ed è in grado di raggiungere 500 GFlop/s.

¹⁷ <http://www.research.ibm.com/x10/>

¹⁸ <http://research.sun.com/projects/plrg/faq/index.html>

di programmazione parallela dinamico con *Partitioned Global Address Space* (PGAS)¹⁹ derivato da *Co-Array Fortran* (CAF)²⁰ e *Unified Parallel C* (UPC)²¹. È prevedibile che questi linguaggi si fondano in uno unico o, per lo meno, che si possa definire un supporto di basso livello comune.

□ Anche se è presto per dire se l'introduzione di questi linguaggi avrà un impatto significativo e stabile sulla comunità scientifica, occorre predisporre un'adeguata attività di formazione e supporto che vada verso l'introduzione di metodologie *object oriented* e più scalabili di MPI se si vogliono realizzare applicazioni parallele per i supercomputer del prossimo futuro, caratterizzate da un'alta scalabilità. Uno strumento importante in questa direzione è rappresentato dal paradigma Charm++²², una libreria parallela *object oriented* costruita sopra C++ e portabile su piattaforme eterogenee.

□ Per ottenere un alto livello di performance sarà poi necessario cambiare la struttura interna delle applicazioni e sperimentare nuovi algoritmi che tengano conto di architetture specifiche e scalabili. Solo un numero limitato di metodologie numeriche sono fondamentali in ambito scientifico e ingegneristico: algebra lineare su matrici dense e sparse, metodi spettrali, metodi N-body, griglie strutturate e non strutturate, metodi Monte-Carlo. È importante predisporre metodologie scientifiche ottimizzate e che contemplino la scalabilità di questi metodi su sistemi con migliaia di processori.

□ Infine va ricordato l'aspetto della *fault tolerance*: il *mean-time-to-failure* (MTTF) di sistemi con un numero molto elevato di processori diventa per forza di cose molto più basso rispetto ai sistemi HPC più tradizionali, diventa pertanto indispensabile pensare ad applicazioni in grado di gestire elementi di fault tolerance in modo semplice.

6. UNO SGUARDO ALLO SCENARIO EUROPEO

L'HPC nella sua accezione più avanzata (*capability computing*) si basa su sistemi caratterizzati da un elevato numero di processori, grande capacità di memoria e un'alta banda di interconnessione con una bassa latenza. La disponibilità di sistemi innovativi di questa classe è indispensabile per accrescere le competitività nelle diverse aree della scienza e della ricerca in Europa, tant'è che l'ESFRI²³, il Forum Strategico sulle Infrastrutture di Ricerca in Europa, recentemente, ha identificato l'HPC come una priorità strategica per l'Europa [14].

L'Unione europea ha fatto proprio questa priorità e nell'ambito del VII programma quadro (2007-2012) prevede di supportare l'installazione di alcuni sistemi HPC di classe Pflop/s (Tier 0). Tali sistemi verranno poi integrati con i sistemi HPC presenti nelle singole Nazioni; questi sistemi, che ci si aspetta essere dell'ordine delle centinaia di Tflop/s, costituiscono il cosiddetto Tier 1 e a loro volta saranno integrati con sistemi HPC locali o regionali, meno potenti (Tier 2), secondo un modello a piramide che definisce un preciso eco-sistema HPC a supporto della comunità scientifica europea. Il progetto PRACE²⁴ (*Partnership for Advanced Computing in Europe*) finanziato dalla Unione europea ha l'obiettivo di studiare la creazione di tale sistema HPC pan-europeo e individuare alcuni centri di classe Pflop/s. Questi sistemi rappresenteranno il cosiddetto Tier 0 dell'eco-sistema HPC europeo.

Invece il progetto DEISA²⁵ (*Distributed European Infrastructure for Supercomputing Applications*), sempre finanziato dalla Comunità europea, ha l'obiettivo di consolidare il Tier 1 tramite l'integrazione degli 11 maggiori centri HPC a livello europeo.

¹⁹ <http://www.cs.berkeley.edu/~kamil/papers/pascoo7.pdf>

²⁰ <http://www.hipersoft.rice.edu/caf/>

²¹ <http://upc.lbl.gov/>

²² <http://charm.cs.uiuc.edu/>

²³ <http://cordis.europa.eu/esfri/>

²⁴ <http://www.csc.fi/english/pages/prace>

²⁵ <http://www.deisa.org>

DEISA gestisce un'iniziativa di *Extreme Computing* che consiste nell'erogare importanti quantità di risorse computazionali e di supporto specialistico alle comunità scientifiche dei Paesi che partecipano al progetto. Tali risorse consentono di realizzare ricerche di punta con un grosso impatto innovativo ed eccellenza scientifica e difficili da realizzare senza l'infrastruttura DEISA.

Infine va sottolineato HPC-Europa²⁶, un altro progetto europeo che supporta l'accesso dei ricercatori a sei tra le maggiori infrastrutture di supercalcolo a livello europeo, con l'obiettivo di fornire in modo integrato il supporto ai ricercatori coinvolti nelle attività computazionali che necessitano di importanti servizi HPC. Il servizio viene erogato a largo spettro sia in termini di accesso alle infrastrutture HPC europee, ma anche in termini di fruizione di ambienti computazionali avanzati per consentire ai ricercatori europei di rimanere competitivi con gli altri gruppi di ricerca a livello mondiale.

È fondamentale che anche l'Italia come sistema Paese possa giocare un ruolo importante in questo contesto, predisponendo un ambiente di supercalcolo in linea con quelli che le diverse nazioni europee stanno predisponendo a supporto delle loro comunità scientifiche. Sistemi HPC con potenze dell'ordine delle centinaia di Tflop/s per sistema sono già disponibili (o lo diverranno nel corso del 2008) in Francia, Spagna, Inghilterra, Germania, Norvegia, Svezia e Finlandia. Solo l'integrazione di sistemi a questo livello di potenza, e che evolvono nel tempo, potrà contribuire in modo decisivo a innalzare il livello competitivo degli scienziati europei.

7. CONCLUSIONI

L'evoluzione della ricerca scientifica comporta l'utilizzo di strumenti computazionali complessi che si avvalgono di sistemi di elaborazione avanzati ed in rapida evoluzione. Dispositivi per la memorizzazione, l'archiviazione e l'analisi di grosse mole di dati e strumenti innovativi per la visualizzazione dei ri-

sultati sono il necessario complemento ai sistemi per il calcolo ad alte prestazioni se si vogliono affrontare in modo adeguato i problemi di frontiera posti oggi dalla scienza, ma anche dalla vita di tutti i giorni.

Il miglioramento delle prestazioni dei sistemi di supercalcolo si ottiene ormai con l'integrazione di più processori sullo stesso chip. Già oggi i cluster HPC impiegano componenti *quad-core* e la presenza di sistemi *multi-core* (con decine di processori per chip) diverrà sempre più concreta nell'immediato futuro. Inoltre, una nuova classe di architetture ad elevate prestazioni è in fase di studio con l'obiettivo di aumentarne l'efficienza e la scalabilità. Tali architetture sfruttano il computing eterogeneo, integrando processori con funzionalità differenti come FPGAs, *Processing Unit Grafiche* (GPU), acceleratori SIMD e quant'altro, per minimizzare il tempo di esecuzione. I programmatori HPC e i ricercatori computazionali sono quindi chiamati a un salto innovativo che comporta l'utilizzo di nuovi algoritmi, paradigmi di programmazione parallela efficienti, librerie e strumenti di supporto per lo sfruttamento ottimale di architetture *multi-core* ed eterogenee altamente parallele.

Già oggi sono fruibili supercomputer con una potenza del centinaio di Tflop/s e, entro il 2010, la disponibilità di sistemi capaci di raggiungere il Pflop/s sarà reale.

Le maggiori potenze industriali nel mondo, Stati Uniti, Estremo Oriente, Europa, stanno concentrando i loro sforzi tecnologici ed economici per garantire al loro sistema di ricerca e sviluppo la disponibilità di infrastrutture per il calcolo in grado di reggere il confronto competitivo.

Anche a livello italiano è di fondamentale importanza, da parte del sistema Paese, sostenere i ricercatori nei loro bisogni computazionali avanzati, sia in termini di infrastrutture HPC, in linea con quelle degli altri Paesi europei, che di supporto e di competenze. Solo in questo modo si potranno affrontare e risolvere le sfide scientifiche del nostro tempo, per le quali esistono in Italia capacità e potenzialità di indubbio valore, e generare nuova conoscenza in grado di attraversare i confini disciplinari tradizionali.

²⁶ <http://www.hpc-europa.org>

Bibliografia

- [1] Pople John A.: *Quantum Chemical Models, Nobel Lecture*. December 8, 1998.
- [2] Grama A., Gupta A., Karypis G., Kumar V.: *An Introduction to Parallel Computing*. Design and Analysis of Algorithms, 2nd edition, Addison-Wesley, 2003.
- [3] Dongarra J., Fostr I., Fox G., Gropp W., Kennedy K., Torczon L., White A.: *Sourcebook of Parallel Computing*. Morgan Kaufmann Publishers, 2003.
- [4] Neri A., Esposti Ongaro T., Menconi G., De' Michieli Vitturi M., Cavazzoni C., Erbacci G., Baxter P.J.: *4D simulation of explosive eruption dynamics at Vesuvius*. Geophys. Res. Lett., 34, L04309, doi:10.1029/2006GL028597, 2007.
- [5] Esposti Ongaro T., Neri A., Cavazzoni C., Erbacci G., Salvetti M.V.: Parallel multiphase flow code for the 3D simulation of explosive volcanic eruptions. *Parallel Computing*, Vol. 33, August 2007, p. 541-560.
- [6] Pevzner P.A.: *Computational Molecular Biology. An algorithmic approach*. The MIT press, 2000.
- [7] *The US National Virtual Observatory White Paper*. In Virtual Observatories of the Future, ASP Conference Proceedings, Vol. 225. Edited by Brunner R.J., Djorgovski S.G., Szalay A.S.. San Francisco: Astronomical Society of the Pacific, ISBN: 1-58381-057-9, 2001.
- [8] Hoffmann E.: Evoluzione e prospettive nell'High Performance Computing. *Mondo Digitale*, n. 1, marzo 2003, p. 51-65.
- [9] Flynn M.J.: Some computer organizations and their effectiveness. *IEEE Transaction on Computers*, Vol. C-21, n. 9, 1972.
- [10] Hennessy J., Patterson D.: *Computer architecture: A quantitative approach*. 4th edition, Morgan Kauffman, 2007.
- [11] Quinn M.J.: *Parallel Programming in C with MPI and OpenMP*. Mc Graw Hill, 2004.
- [12] Mehta H. (Guest Editor): IBM POWER 6 Microprocessor Technology. *IBM Journal of Research and Development*, Vol. 51, November, 2007.
- [13] Chamberlain B.L., Callahan D., Zima H.P.: Parallel Programmability and the Chapel Language. *International Journal of High Performance Computing Applications*, Vol. 21, n. 3, 291-312, DOI: 10.1177/1094342007078442, 2007.
- [14] *ESFRI European Roadmap for Research Infrastructures*. Report 2006; European Community, Luxembourg, 2006, ISBN 92-79-02694-1.

GIOVANNI ERBACCI si è laureato in Informatica presso l'Università di Pisa, dal 1999 coordina il gruppo Supercalcolo del CINECA e partecipa a diversi progetti europei nel settore dell'HPC. E' professore a contratto di architetture e programmazione parallela presso l'Università di Ferrara. I suoi interessi principali riguardano le architetture parallele, lo studio e la realizzazione di algoritmi paralleli efficienti, nonché la valutazione delle prestazioni dei sistemi e dei programmi paralleli. Giovanni Erbacci è autore o co-autore di oltre 40 articoli pubblicati su riviste e atti di convegni, ed è membro dell'ACM.
E-mail: g.erbacci@cenea.it



EVOLUZIONE DEI SISTEMI OPERATIVI

I^a Parte: dai sistemi batch a Linux

Paolo Ancilotti
Maurelio Boari

3.2

Nel ripercorrere le motivazioni che hanno portato all'evoluzione delle funzioni di un sistema operativo viene evidenziato come tale evoluzione sia stata fortemente influenzata dai progressi di altre branche dell'intera disciplina informatica e al tempo stesso ne abbia a sua volta influenzato il progresso. L'articolo è suddiviso in due parti: la prima dedicata all'evoluzione del concetto stesso di sistema operativo con riferimento, soprattutto, ai sistemi *general purpose*, dai primi sistemi *batch* agli attuali sistemi *multithreading*; la seconda verrà dedicata all'evoluzione di specifiche categorie di sistemi.

1. INTRODUZIONE

La conoscenza del modo in cui una disciplina si è sviluppata, delle motivazioni che stanno alla base delle innovazioni che ha prodotto e delle tecnologie da questa derivate, hanno spesso un valore che trascende il puro e semplice interesse culturale in quanto aiutano a comprendere meglio il perché di certe scelte e ad orientare gli stessi sviluppi futuri sia scientifici che tecnologici. Se ciò è vero per discipline che hanno alle spalle un percorso evolutivo molto lungo, spesso di secoli, le precedenti motivazioni possono sembrare prive di fondamento per una disciplina come l'informatica che ha soltanto pochi decenni di vita. In realtà ciò è vero, ma è altrettanto vero che la rapidità con cui è avvenuta l'evoluzione di questa scienza, con cui sono state introdotte nuove tecnologie e con cui tali tecnologie hanno pervaso tutte le altre discipline, non ha eguali in nessun altro ambito scientifico e tecnologico. È proprio da questa frenetica evoluzione, dalla rapidità con cui nuove tecnologie vengono introdotte ed altre abbandonate, che

deriva l'opportunità anche, in questo campo, di ripercorrere il cammino evolutivo, cercando di capire e giustificare perché certe scelte siano state privilegiate nei confronti di altre.

Nel caso poi dello specifico tema dei Sistemi Operativi, l'importanza di conoscere il loro percorso evolutivo è ulteriormente avvalorata dal fatto che le funzioni che vengono svolte da questo importante componente di un sistema di elaborazione sono molto varie, spesso anche molto diverse da un sistema all'altro, al punto che negli stessi libri dedicati a questo argomento [3, 20, 21, 22] il sistema operativo viene definito in vari modi che, come ad esempio quelli riportati di seguito, non sempre sono coincidenti:

□ il sistema operativo è un programma che agisce come intermediario fra l'utente di un calcolatore e le componenti hardware del calcolatore stesso.

□ Il sistema operativo è quel componente di un sistema di elaborazione destinato a fornire l'ambiente nel quale vengono eseguiti i programmi degli utenti del sistema.

❑ Il sistema operativo è il gestore delle risorse hardware e software di un sistema di elaborazione; alloca tali risorse ai programmi e ai relativi utenti, secondo le loro necessità.

❑ Il sistema operativo è quel componente del sistema di elaborazione che ha il compito di creare una macchina astratta più semplice da usare rispetto alla macchina fisica.

Ciascuna delle precedenti definizioni, e altre ancora presenti in molti libri dedicati a questo tema, è sicuramente corretta ma, in generale, da sola non è sufficiente a identificare l'intero insieme delle funzioni che, di volta in volta, possono trovarsi in un sistema operativo, limitandosi a mettere in evidenza soltanto alcuni aspetti che, in certi casi possono essere quelli più importanti, ma senza che ciò sia necessariamente vero per tutti i sistemi.

L'obiettivo di questo articolo è proprio quello di ripercorrere le motivazioni che hanno portato, da un lato alla necessità di dotare ogni sistema di elaborazione di un proprio sistema operativo e dall'altro all'evoluzione delle diverse funzioni che, di volta in volta, gli sono state aggiunte a seconda dei diversi ambiti applicativi per i quali il sistema di elaborazione viene prioritariamente progettato.

Come avremo modo di evidenziare nel corso dell'articolo, non è facile isolare l'evoluzione dei sistemi operativi da quella delle altre branche dell'intera disciplina informatica essendo l'una parte integrante dell'altra. Le motivazioni e le scelte secondo le quali sono stati sviluppati i sistemi operativi hanno fortemente risentito dei progressi di altre branche dell'intera disciplina e al tempo stesso ne hanno a loro volta influenzato l'evoluzione. Il caso più evidente è quello relativo alla reciproca influenza fra l'evoluzione dei sistemi operativi e quella delle architetture dei sistemi di elaborazione. Per esempio, l'introduzione del meccanismo di interruzione automatica può essere considerata la pietra miliare nello sviluppo della multiprogrammazione che è alla base di tutti i moderni sistemi operativi. Ma da quel punto in poi molte sono state le connessioni tra architettura e sistema operativo. Tanto per citarne alcune delle più significative, ricordiamo l'introduzione dei meccanismi per la gestione della memoria (MMU - *Memory Management Unit*) quali quelli della paginazione e della segmentazio-

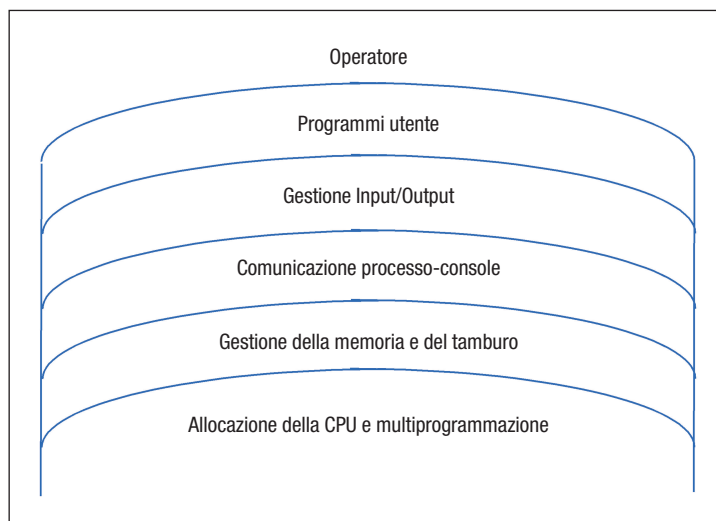


FIGURA 1

Organizzazione gerarchica di un sistema operativo: i sei livelli del sistema operativo THE

ne; i meccanismi architetturali per la protezione, come il doppio stato di esecuzione del processore; i meccanismi hardware per la mutua esclusione e la realizzazione degli *spin semaphore* ed altri ancora. Anche i settori relativi ai linguaggi di programmazione e all'ingegneria del software hanno risentito dell'evoluzione dei sistemi operativi. Per esempio, l'intera branca dei linguaggi concorrenti è una diretta conseguenza dell'introduzione del meccanismo della multiprogrammazione e, per quanto riguarda le metodologie di progetto del software è indubbia l'influenza che il progetto del sistema operativo THE (Figura 1), [9] ha avuto nello sviluppo delle metodologie di strutturazione di un sistema software a livelli gerarchici. Infine, con riferimento alle basi di dati, è notevole l'influenza del lavoro di Gray [11] nello sviluppo delle tecniche per garantire l'atomicità delle transazioni.

2. L'AVVENTO DEI PRIMI CALCOLATORI (1945-1955: ASSENZA DI SISTEMI OPERATIVI)

Il periodo che intercorre tra la fine della II Guerra Mondiale e la metà degli anni '50 è caratterizzato dalla nascita dei primi calcolatori elettronici, inizialmente programmabili dall'esterno tramite connessioni su un pannello di controllo e successivamente mediante programma memorizzato. I primi calcola-



tori a programma memorizzato sono del 1948 (il Mark I presso l'*università di Manchester*) e del 1949 (l'EDSAC progettato da M. Wilkes presso l'*università di Cambridge*).

I calcolatori di questa prima generazione, completamente realizzati mediante valvole termoioniche, erano estremamente costosi, poco affidabili e di difficile utilizzabilità essendo privi di qualunque supporto software. La stessa programmazione era effettuata inizialmente in linguaggio macchina e i programmi caricati manualmente in memoria tramite interruttori presenti sulla consolle del sistema. Successivamente, fu introdotto il linguaggio assembleativo e solo nel 1954 J. Backus annunciò il primo compilatore FORTRAN per il calcolatore IBM 704. Le applicazioni cui erano dedicati erano quelle relative alla risoluzione di problemi di tipo scientifico e tecnologico.

A causa della difficoltà di utilizzazione di questi sistemi, il loro uso era praticamente limitato ai soli specialisti per cui una stessa persona era contemporaneamente progettista, operatore di macchina, programmatore e utente. Era l'unico responsabile della scrittura di un programma applicativo, ivi incluse le particolari istruzioni destinate a controllare il funzionamento dei dispositivi d'ingresso/uscita (I/O), oltre che di caricare in memoria il programma e di controllarne l'esecuzione tramite la consolle. I tipici dispositivi di I/O erano costituiti da macchine elettromeccaniche estremamente lente rispetto alla velocità di esecuzione delle istruzioni da parte della CPU (lettori di schede perforate, perforatori di schede, stampanti) e da nastri magnetici come supporto di memoria massa.

Chiaramente veniva eseguito un solo lavoro (*job*) alla volta. L'utente-programmatore-operatore, aveva il sistema completo a sua esclusiva disposizione per tutto il tempo in cui rimaneva seduto alla consolle.

Con la realizzazione dei primi traduttori, sia di linguaggi assembleativi che di linguaggi ad alto livello (FORTRAN), la tipica esecuzione di un *job* consisteva nel caricare in memoria lo specifico programma traduttore (residente ad esempio su nastro magnetico) ed eseguirlo per tradurre il programma applicativo (tipicamente disponibile su un pacco di schede precedentemente preparato perforando su

ogni scheda una diversa istruzione del programma). La traduzione forniva il programma tradotto in linguaggio macchina direttamente su nastro magnetico, oppure perforandolo su un diverso pacco di schede. A quel punto il programma tradotto poteva essere caricato ed eseguito.

Questa modalità operativa era caratterizzata dal grosso inconveniente di limitare fortemente sia l'efficienza di uso delle costose risorse di macchina sia la produttività del sistema (*throughput*). Infatti, a causa delle numerose operazioni manuali, l'efficienza di uso della CPU aveva valori estremamente bassi. Contemporaneamente il *throughput* non superava il valore di uno o due *job* l'ora. D'altro canto, il fatto che l'utente fosse anche programmatore ed operatore e cioè avesse l'intera macchina a sua disposizione e potesse quindi monitorare l'esecuzione dei programmi direttamente dalla consolle, consentiva allo stesso di eseguire facilmente la messa a punto di un programma potendo effettuare il *debugging* direttamente da consolle, per esempio, potendo bloccare l'esecuzione del programma, modificare manualmente i contenuti di alcune locazioni di memoria o di alcuni registri e rilanciare subito l'esecuzione.

3. NASCITA DEI PRIMI SISTEMI BATCH (1955-1959)

Nella seconda metà degli anni '50 cominciarono ad uscire i primi sistemi realizzati mediante transistor, molto più affidabili dei precedenti a valvole. Erano ancora sistemi fondamentalmente dedicati a elaborazioni di tipo tecnico/scientifico, notevolmente costosi sia in termini di acquisto che di gestione e manutenzione. Tali calcolatori, noti come *mainframes*, erano disponibili quasi esclusivamente presso i centri di calcolo universitari o di banche e grandi imprese ed erano programmati prevalentemente in FORTRAN o in linguaggio assembler.

La disponibilità di linguaggi ad alto livello rese possibile l'accesso al calcolo elettronico anche a ricercatori e tecnici non strettamente informatici ai quali però non poteva essere più lasciato libero accesso al sistema di elaborazione che era riservato, vice-

versa, a particolari tecnici specializzati, gli operatori di macchina. L'utente-programmatore, non più anche operatore, portava al centro calcolo il proprio pacco di schede contenente il programma applicativo da tradurre, caricare ed eseguire, lasciandolo a disposizione degli operatori. Alla fine dell'esecuzione di un singolo *job*, l'operatore prelevava dalla stampante il tabulato con i risultati della sua esecuzione, lasciava il tabulato a disposizione dell'utente del *job* appena terminato e prelevava un pacco di schede relativo al *job* di un altro utente al fine di sottoporlo al sistema.

Pur essendo le operazioni manuali svolte da un tecnico specializzato, queste continuavano a limitare drasticamente l'efficienza di uso del sistema e la sua produttività. È per limitare l'intervento dell'operatore che nacque l'esigenza di dotare la macchina di un programma di sistema, il primo vero e proprio sistema operativo, spesso identificato col termine di *monitor*, a cui demandare il compito di ridurre al minimo l'intervento manuale dell'operatore. Per ottenere questo risultato, per prima cosa i programmi componenti il sistema di programmazione (traduttori dei linguaggi ad alto livello, assemblatore ecc.) furono resi disponibili su nastri magnetici. Quindi, per automatizzare le singole fasi relative all'esecuzione di un *job*, fu necessario definire un semplice linguaggio di controllo, detto anche *Job Control Language* (JCL) (l'antenato dei moderni *shell*). Utilizzando tale linguaggio l'utente preparava il pacco di schede relativo al proprio *job*, in modo tale che, oltre alle schede su cui erano perforati sia il programma sorgente che i dati da elaborare, fossero presenti anche le cosiddette schede di controllo (nelle quali erano perforati i comandi del JCL) facilmente riconoscibili contenendo, nelle prime colonne della scheda, dei caratteri particolari come, ad esempio \$. Mediante queste schede l'utente identificava i programmi da eseguire come singoli passi del proprio *job*. Il compito di caricare in memoria i programmi da eseguire non era più demandato all'operatore, ma a un programma (*monitor*) il quale era residente in una porzione della memoria ad esso riservata. Questo era un semplice programma ciclico

(*cyclic executor*) che si limitava a leggere una scheda e, se questa era riconosciuta come scheda di controllo, ne veniva interpretato il contenuto, mediante un semplice interprete del JCL, prelevando dal nastro magnetico il programma eseguibile richiesto mediante quella scheda e caricandolo in memoria. Terminato il trasferimento in memoria del programma richiesto, mediante un'istruzione di salto veniva trasferito il controllo della CPU alla prima istruzione del programma caricato. Al termine della sua esecuzione il programma così attivato, invece di eseguire l'istruzione di *halt* che avrebbe fermato la macchina, restituiva il controllo al monitor, ancora tramite un salto, e questo continuava con l'esame della successiva scheda di controllo e così via fino alla scheda di controllo che rappresentava la fine dell'esecuzione dell'intero *job*.

Per ridurre ulteriormente l'intervento dell'operatore, fu anche automatizzata la transizione da un *job* al successivo caricando sul lettore di schede invece di un solo pacco, corrispondente al *job* di un singolo utente, un insieme di pacchi di schede, corrispondenti ad altrettanti *job* di utenti diversi che l'operatore aveva raccolto precedentemente (da ciò deriva il nome di *sistemi batch* con cui questi primi sistemi operativi furono conosciuti). In questo modo il monitor poteva passare dall'esecuzione di un *job* a quella del *job* successivo senza soluzione di continuità. L'intervento manuale dell'operatore avveniva solo alla fine dell'esecuzione di un *batch di job*. A quel punto veniva inserito sul lettore un nuovo *batch*, collezionato durante l'esecuzione del *batch* precedente, e la macchina veniva di nuovo riattivata manualmente.

Il primo sistema operativo (GM OS) funzionante con questo criterio fu realizzato dalla *General Motors Research Laboratories* per il loro calcolatore IBM 701 nel 1956 [12]. Successivamente IBM realizzò il sistema operativo FMS (*Fortran Monitor System* [17]) per il proprio calcolatore IBM 709 da cui ebbe origine nel 1959 il *mainframe* IBM 7090 completamente transistorizzato e su cui continuò ad operare lo stesso sistema operativo FMS.

L'introduzione di questi primi sistemi operativi funzionanti con la tecnica *batch* garantirono un drastico aumento dell'efficienza di

uso delle risorse dei sistemi di elaborazione. Alcuni problemi restavano comunque ancora aperti e altri furono introdotti a causa dalle modalità operative proprie dei sistemi *batch*:

a. il primo inconveniente, di natura tecnologica, riguardava l'efficienza. Pur essendo migliorata, questa era comunque ancora molto bassa (dell'ordine di poche unità percentuali). Il principale motivo di questa inefficienza era dovuto alla notevole differenza fra la velocità della CPU e quella, molto più lenta, dei dispositivi periferici (soprattutto lettore di schede e stampante) e al fatto che, non essendo ancora disponibile il meccanismo hardware delle interruzioni, tutti i trasferimenti di I/O avvenivano ancora con la tecnica del "controllo di programma". Per cui, quando la CPU attivava un trasferimento di dati (per leggere una scheda o stampare i risultati di un programma) doveva restare in attesa della fine del trasferimento, e questi tempi di attesa erano di alcuni ordini di grandezza superiori a quelli con cui la CPU era in grado di eseguire un intero programma.

b. Il secondo inconveniente, legato alla tecnica *batch*, riguardava la scarsa flessibilità nella scelta dell'ordine con cui i *job* venivano eseguiti che corrispondeva allo stesso ordine con cui comparivano i relativi pacchi di schede sul lettore. Poteva quindi accadere che un utente che aveva presentato un programma molto semplice, e quindi eseguibile molto rapidamente, dovesse aspettare un tempo molto lungo prima di ricevere i propri risultati (*turnaround time*) e ciò solo per il fatto che il pacco di schede del proprio programma era stato posizionato sul lettore di seguito a quello di un utente che aveva richiesto un tempo di esecuzione molto lungo. In altri termini l'ordine con cui i vari *job* venivano eseguiti (*scheduling*) era preordinato dall'operatore senza che il sistema operativo potesse intervenire per modificarlo.

c. Il terzo inconveniente, anch'esso legato alla tecnica *batch*, riguardava il tempo estremamente lungo che era richiesto per completare la "messa a punto" di un programma applicativo. Come già indicato precedentemente, quando il programmatore era anche operatore, poteva quindi beneficiare del fatto che tutto sistema era a sua completa disposizione e quindi poteva interagire direttamente

con la consolle della macchina. Aver allontanato il programmatore dalla macchina influì in maniera estremamente negativa sul processo di scrittura, compilazione, *testing* e *debugging* di un programma. L'utente iniziava tale processo sottoponendo il suo pacco di schede all'operatore. Dopo un tempo che poteva essere dell'ordine di alcune ore, se non di giorni, riceveva il tabulato da cui poteva risultare che il programma non era stato eseguito per alcuni errori sintattici, al limite anche molto banali. Iniziava quindi tutta una serie di tentativi per ottenere un programma sintatticamente corretto e quindi eseguibile. A quel punto iniziava la fase di *testing* per verificare la presenza di errori semantici. Per ogni errore individuato era necessario correggerlo e ricominciare con la compilazione.

4. NASCITA DELLA MULTIPROGRAMMAZIONE: I SISTEMI BATCH MULTIPROGRAMMATI (1960-1970)

Agli inizi degli anni '60 la IBM, per garantire una maggiore efficienza nell'uso delle costose risorse di calcolo dei propri *mainframe* (7090 e 7094) decise di eliminare l'inconveniente a) visto nel precedente paragrafo, obbligando ad utilizzare per tali macchine solo dispositivi di I/O che consentissero elevate velocità di trasferimento dati come i nastri magnetici. Per trasferire un *batch* di *job* da lettore di schede a nastro magnetico veniva usata, fuori linea, una macchina satellite poco costosa come la IBM 1401. Analogamente per trasferire i risultati delle elaborazioni di tutti i *job* di un *batch* dal nastro su cui erano stati scritti alla stampante, veniva di nuovo utilizzata fuori linea la 1401. In questo modo mentre il *mainframe* eseguiva un *batch*, operando sia in ingresso che in uscita soltanto con nastri magnetici, contemporaneamente venivano trasferiti da schede a nastro i *job* del *batch* successivo e da nastro alla stampante i risultati del *batch* precedente.

Il sistema operativo IBSYS della IBM funzionava con tale criterio riuscendo ad ottenere efficienze di uso della CPU del *mainframe* dell'ordine del 90%.

Con l'introduzione delle memorie di massa ad accesso casuale come i dischi magnetici e

tenendo conto che i calcolatori della classe 7090/7094 erano dotati di canali di I/O, veri e propri processori *special purpose* che operavano in maniera asincrona con la CPU, fu possibile ottenere gli stessi risultati dello schema precedente senza utilizzare il calcolatore satellite. Infatti, utilizzando il disco come un *buffer*, mentre il calcolatore, mediante il canale di input, leggeva i *job* di un *batch* trasferendoli su disco, lo stesso poteva eseguire i *job* del *batch* precedentemente trasferito su disco, prelevando i dati e inserendo i risultati sempre su disco. E, contemporaneamente mediante il canale di output, poteva trasferire alla stampante i risultati, presenti anch'essi su disco e relativi al *batch* ancora precedente. Essendo inoltre i *job* da eseguire residenti su disco, era possibile scegliere l'ordine con cui eseguirli non essendo più vincolati dalla sequenzialità con cui gli stessi comparivano sul nastro magnetico, eliminando quindi anche il precedente inconveniente b). Questo tipo di sistemi *batch* fu noto col nome di "*Spooling*" (*Simultaneous Peripheral Operation On Line* [2]).

Nei primi anni '60, con l'introduzione dei meccanismi hardware delle interruzioni automatiche e del DMA (*Direct Memory Access*) fu possibile estendere ulteriormente le funzionalità di un sistema operativo abilitandolo a caricare in memoria principale più programmi contemporaneamente in modo tale che, quando l'esecuzione di uno di essi non poteva proseguire, per esempio in seguito all'attivazione di un trasferimento di I/O, fosse possibile salvare i suoi risultati parziali e passare in esecuzione un altro programma presente in memoria. Soltanto all'arrivo dell'interruzione relativa alla fine del precedente trasferimento, l'esecuzione del primo programma poteva riprendere dal punto in cui era stata precedentemente interrotta. Questa tecnica, nota col nome di *multiprogrammazione*, consentiva di ridurre al minimo il tempo di inattività della CPU e, al tempo stesso, garantire l'utilizzazione contemporanea di tutte le risorse del sistema (CPU, dispositivi periferici, memoria), aumentando ancora l'efficienza globale di uso delle risorse e diminuendo il *turnaround time*. Come contropartita, la complessità del sistema operativo aumentò drasticamente. Dovendo

infatti gestire l'esecuzione contemporanea di più programmi, fu necessario delegare al sistema operativo il compito di implementare le strategie di allocazione di tutte le risorse di calcolo che, se rese disponibili per l'esecuzione di un programma, non potevano essere rese disponibili anche per altri. Analogamente, al fine di evitare che l'esecuzione di un programma erroneo producesse interferenze nelle esecuzioni di altri programmi contemporaneamente presenti in memoria, fu necessario implementare ben precise strategie di protezione anche con l'ausilio di meccanismi hardware all'uopo concepiti. La conseguenza di ciò fu un aumento delle dimensioni e quindi dell'occupazione di memoria del sistema operativo e un maggior tempo di CPU richiesto per l'esecuzione delle sue funzioni, tempo sottratto all'esecuzione dei programmi applicativi (*overhead* di sistema). I vantaggi della multiprogrammazione furono comunque tali da rendere del tutto trascurabili questi problemi, tenendo anche conto dei continui progressi sul versante delle architetture.

Nei primi sistemi multiprogrammati la risorsa che limitò maggiormente le prestazioni del sistema fu la memoria. Infatti, la necessità della contemporanea presenza in memoria dei programmi fra i quali poteva essere commutato l'uso della CPU, limitava il numero dei programmi stessi in base alle loro dimensioni e alle dimensioni della memoria. Poiché in un sistema multiprogrammato la CPU resta inutilizzata quando tutti i programmi presenti in memoria sono in attesa della terminazione di un trasferimento di I/O da essi attivato, è ovvio che questo rischio è tanto minore quanto maggiore è il numero di tali programmi.

Uno dei primi sistemi operativi *batch* multiprogrammati fu il sistema OS/360 realizzato dalla IBM per i propri calcolatori della serie System/360 [13].

Fino alla metà degli anni '60 le case costruttrici di computer producevano due diverse tipologie di macchine del tutto incompatibili tra loro: i *mainframe*, macchine di grosse dimensioni, fondamentalmente orientate al calcolo tecnico/scientifico, e macchine di più piccole dimensioni orientate al calcolo di tipo commerciale. Nel 1964 l'IBM annun-

ciò la produzione di una serie di macchine di terza generazione (utilizzando la tecnica dei circuiti integrati), la serie System/360. Ad essa appartenevano macchine di dimensioni diverse, con prezzi e prestazioni diverse, ma tutte software-compatibili tra loro e concepite per soddisfare le esigenze di qualunque tipo di utenza sia scientifica che commerciale.

Lo sviluppo del sistema operativo OS/360 fu concepito in modo tale che tutte le macchine della serie dovessero impiegare varianti, tra loro compatibili, dello stesso sistema operativo. Naturalmente le esigenze delle varie macchine della serie erano diverse tra loro, a seconda che dovessero lavorare in ambiente scientifico o commerciale, che fossero dotate di molte o poche periferiche. Il risultato fu un sistema operativo molto complesso, di enormi dimensioni (milioni di righe di codice, forse il più grande e complesso programma mai sviluppato) scritto completamente in assembler da moltissimi programmatori. Oltre ad impegnare molta memoria e tempo di CPU, il sistema operativo si caratterizzò fin dall'inizio per il rilevante numero di errori in esso contenuti. Col passare degli anni furono rilasciate nuove versioni per aggiungere caratteristiche nuove e per correggere gli errori. Tuttavia la correzione di un errore spesso ne provocava un altro in qualche parte remota del sistema. Un interessante rapporto sull'esperienza con OS/360 è descritto da *Fred Brooks*, uno dei progettisti del sistema, che scrisse in seguito un libro famoso nel settore dell'ingegneria del software [5].

L'esperienza vissuta servì a rendere sempre più evidente la necessità di adottare tecniche per una maggiore strutturazione nella realizzazione di sistemi software di grandi dimensioni e di significativo grado di complessità come nel caso dei sistemi operativi. È dell'ottobre 1968 la prima conferenza di *software engineering*, tenuta a *Garmisch* in Germania, nell'ambito della quale emerse chiara la necessità di fare riferimento, nello sviluppo di software, a metodi formali e metodologie di programmazione strutturata per far fronte alla inerente complessità di sistemi software di grandi dimensioni [18]. Sempre del 1968 è un fa-

moso articolo di *Dijkstra* sulla programmazione strutturata [10].

Un approccio che fu ritenuto valido per affrontare la complessità dello sviluppo di un sistema operativo fu quello di fare riferimento a tecniche di modularizzazione in modo tale da suddividere il sistema in componenti (moduli) ciascuno destinato a fornire una delle sue funzionalità. Secondo questo criterio, ogni modulo deve essere caratterizzato da una ben precisa interfaccia che specifichi la funzionalità offerta dal modulo ed il modo di utilizzarla, e da un corpo contenente la sua implementazione, non visibile all'esterno del modulo.

Nonostante gli indubbi vantaggi indotti dall'uso di tecniche di modularità la complessità dei sistemi operativi andava progressivamente crescendo, richiedendo quindi ulteriori paradigmi di progetto. Fra questi ha svolto un ruolo particolarmente importante la tecnica di organizzazione del software a *livelli gerarchici di astrazione*. Senza entrare nel merito di queste tecniche, peraltro ben note nel campo della programmazione (*top-down, bottom-up*), uno degli aspetti principali della tecnica di organizzazione gerarchica consiste nel ridurre il numero di possibili interconnessioni fra i moduli di un sistema, semplificando quindi sia la fase di progetto che quella di verifica. I moduli che costituiscono il sistema operativo vengono organizzati gerarchicamente in diversi livelli di astrazione in modo tale che i moduli realizzati ad un certo livello utilizzano esclusivamente le funzionalità offerte dai moduli di livello più basso e forniscono le loro funzionalità ai moduli di livello più alto. Questo tipo di organizzazione è oggi presente, in misura più o meno completa, in tutti i moderni sistemi operativi. Va ricordato che il primo sistema operativo realizzato secondo questo modello strutturale fu il sistema THE, un sistema batch realizzato alla fine degli anni '60 presso la *Technische Hogeschool di Eindhoven* da *Edsger Dijkstra* e dai suoi studenti [9]. L'importanza di tale sistema, che rimase un caso di studio, fu proprio nella possibilità offerta di sperimentare in che modo poteva trovare la sua realizzazione il modello a livelli gerarchici calato nella realtà di un semplice sistema operativo.

5. I PRIMI SISTEMI TIME-SHARING (1960-1970)

Lo sviluppo dei sistemi batch multiprogrammati consentì di raggiungere alti livelli di efficienza e di produttività (*throughput rate*), e ciò decretò il loro successo in tutti i casi nei quali non era necessario prevedere interazioni con l'utente; quindi si imposero come sistemi ideali per far girare sia programmi che richiedevano lunghe elaborazioni, come nel caso di applicazioni tecnico/scientifiche, sia programmi che richiedevano l'elaborazione di grosse moli di dati come nel caso di applicazioni commerciali. Restava comunque aperto il problema legato allo sviluppo di tali programmi. Come abbiamo già messo in evidenza nel terzo paragrafo (vedi inconveniente c), l'aver negato al programmatore l'accesso alla macchina implicò una crescita molto marcata del *turnaround time* di un *job* e quindi la corrispondente crescita del tempo necessario per la messa a punto di un programma [23].

È per risolvere questo problema che agli inizi degli anni '60 nacque una nuova linea di ricerca tendente a sviluppare una nuova categoria di sistemi operativi che consentissero ad un utente di tornare a diretto contatto con la macchina, abilitandolo ad operare in maniera conversazionale col sistema stesso. Per questo motivo fu necessario interfacciare la macchina con un diverso tipo di unità periferiche, in particolare di terminali provvisti di una tastiera, con la quale inviare al sistema i comandi che lo stesso avrebbe dovuto eseguire, e di un monitor su cui avrebbero dovuto comparire i risultati dell'esecuzione dei comandi stessi. È del 1960 la nascita del primo computer commerciale provvisto di monitor e tastiera, il PDP-1 della *Digital Equipment Corporation*.

Utilizzando il terminale l'utente poteva quindi inviare al sistema i comandi battendoli direttamente sulla tastiera invece che perforarli su schede. Il sistema operativo, interpretando il comando digitato dall'utente, caricava in memoria ed eseguiva il programma destinato a svolgere il compito da lui richiesto. Alla fine dell'esecuzione del programma, i risultati venivano visualizzati sul video sul quale poi compariva un carattere di *prompt* per indicare all'utente la disponibilità del si-

stema a ricevere il prossimo comando. Questo tipo di comportamento implicava che l'utente operasse col sistema all'interno di *sessioni di lavoro* che iniziavano quando l'utente si sedeva al terminale. Specificando il proprio codice identificativo (*user-ID*) e la propria parola d'ordine (*password*), si collegava al sistema operativo (*login*) iniziando una sessione di lavoro (invio di un comando al sistema – esecuzione da parte del sistema del compito richiesto – invio del comando successivo – ecc.) fino a quando, eseguendo la procedura di chiusura (*logout*) non decideva di terminare la sessione di lavoro.

Inoltre, per evitare che il sistema fosse monopolizzato da un solo utente alla volta, avendo disponibili più terminali connessi alla macchina e sfruttando il meccanismo della multiprogrammazione, era possibile far accedere più utenti contemporaneamente. In questo modo il sistema poteva eseguire, in multiprogrammazione, i programmi richiesti dai singoli utenti tramite i comandi da loro digitati sulle rispettive tastiere.

Essendo il comportamento del sistema di tipo conversazionale, durante una sessione di lavoro la frazione di tempo che la CPU dedicava all'esecuzione dei programmi richiesti da un utente era sicuramente trascurabile nei confronti del tempo che l'utente stesso dedicava a battere i propri comandi sul terminale. Per cui il sistema operativo poteva sfruttare questi tempi "morti" relativi a un utente per far avanzare i programmi richiesti da altri utenti. Il risultato era quello di dare ad ogni utente l'illusione che la macchina fosse a sua completa disposizione (*macchina virtuale*).

Ciò richiese però anche una diversa filosofia di funzionamento del sistema. L'obiettivo prioritario di questo tipo di sistemi non era più quello di raggiungere la massima efficienza di uso delle risorse di macchina ma bensì quello di ridurre il tempo di risposta cioè il tempo che l'utente avrebbe dovuto attendere per avere sul video i risultati dell'esecuzione del programma richiesto mediante un comando. Inoltre, non era solo importante minimizzare il tempo di risposta di un programma ma, ancora più importante era rendere questo tempo proporzionale alla complessità del programma. Infatti, un uten-

Multics

Il sistema operativo *Multics* fu sviluppato nella seconda metà degli anni 60 presso il *Massachusetts Institute of Technology* (MIT) in collaborazione con i *Bell Telephone Laboratories* e il *Computer Department* della General Electric che fornì il calcolatore sul quale sviluppare *Multics* (versione modificata del GE 635, denominata GE 645). Il sistema fu pensato come una *computer utility* in grado di venire incontro ad una vasta comunità di utenti. In modo suggestivo il sistema fu presentato come un servizio, alla stessa stregua dell'acqua o dell'elettricità a disposizione di una vasta comunità di utenti (nella sua concezione originale il sistema avrebbe dovuto poter servire le necessità di calcolo dell'intera città di Boston). Il servizio quindi doveva avere caratteristiche di alta affidabilità, in pratica funzionare 7 giorni su 7, 24 h al giorno. Gli utenti avrebbero potuto accedere sia da terminali remoti che centralizzati. *Multics* si rivelò un progetto troppo ambizioso per il suo tempo: la sua trasformazione in un prodotto industriale risultò molto complicata anche perché i *Bell Laboratories* abbandonarono il progetto e la GE cessò l'attività nel settore computer. Il MIT continuò tuttavia a lavorare sul progetto e lo condusse a termine. Il sistema fu venduto dalla Honeywell che aveva acquisito il ramo calcolatori della GE ed installato in 80 grandi aziende e università nel mondo.

te è tendenzialmente disponibile ad attendere al terminale il completamento di un programma molto complesso per un tempo maggiore rispetto a quanto sia disponibile ad attendere il completamento di un programma banale. I principi di funzionamento tipici dei sistemi *batch* non erano adatti a fornire questo tipo di comportamento. In tali sistemi, infatti, quando entrava in esecuzione un programma che richiedeva lunghe elaborazioni (*CPU-bound*) senza trasferimenti di I/O, questo poteva mantenere il controllo della CPU per molto tempo senza consentire l'esecuzione di altri programmi, neppure di quelli molto semplici che, viceversa, avrebbero dovuto terminare il più rapidamente possibile. Per ovviare a questo problema fu deciso che la CPU, di volta in volta, fosse dedicata all'esecuzione di un programma per un intervallo di tempo dell'ordine di qualche decina di millisecondi (*time slice*), trascorso il quale il programma, se non fosse terminato o non si fosse sospeso in attesa di un trasferimento di I/O, doveva comunque essere interrotto e la CPU essere dedicata all'esecuzione di un altro programma e così via in maniera circolare prima che la CPU fosse di nuovo dedicata al programma interrotto. È da questo tipo di comportamento che nacque il nome di *time-sharing*, o a divisione di tempo, con cui furono identificati i sistemi operativi di questa nuova categoria.

I primi studi su questa tipologia di sistemi

operativi furono svolti da *John McCarthy*, *Robert Fano* e *Fernando Corbato* al *Department of Electrical Engineering* del MIT. Tali studi condussero alla realizzazione, nel 1961, del primo sistema *time-sharing*, il sistema *CTSS – Compatible Time-Sharing* [7]. Questo sistema utilizzava un calcolatore della classe IBM 7090 ritenuto, per quei tempi, molto potente (IBM 7094, con 32 K di memoria, di cui 5 k occupate dal monitor, parole a 36 bit, tamburo veloce come memoria secondaria). Gestiva fino a 32 utenti che avevano la possibilità, utilizzando un insieme di comandi del sistema operativo, di operare sul sistema in modo interattivo tramite terminali compilando e mettendo in esecuzione i propri programmi e operando sui propri file.

Come prosecuzione dell'esperienza, sotto molti aspetti positiva, del *CTSS*, nel 1964 presso il MIT fu finanziato il Progetto *MAC (Machine Aided Cognition)*, che avrebbe dovuto condurre al successivo passo in avanti nello sviluppo dei sistemi *Time-Sharing* con la progettazione del sistema *Multics* [6] in collaborazione con i *Bell Telephone Laboratories* ed il *Computer Department* della General Electric (riquadro). Il *Multics* introdusse una serie di innovazioni di grande importanza che troveremo anche nei recenti sistemi.

Con l'introduzione dei sistemi *time-sharing* aumentò molto la complessità del sistema operativo rispetto ai precedenti sistemi di tipo *batch*. Infatti, a causa della *multiutenza* fu necessario affrontare nuovi problemi di sicurezza e di protezione delle informazioni, controllando gli accessi al sistema (gestione degli *account*) e implementando meccanismi di protezione sia del sistema che dei dati e dei programmi dei singoli utenti. Inoltre si complicò notevolmente il problema relativo alla gestione della memoria. Infatti, per non limitare il numero degli utenti contemporaneamente attivi fu necessario garantire elevati livelli di multiprogrammazione e cioè di programmi contemporaneamente in esecuzione. Fu quindi necessario svincolare il numero di questi programmi dalle dimensioni della memoria fisica centrale che li avrebbe dovuti contenere. Nacque quindi la necessità di implementare strategie di allocazione dinamica della memoria e di realizzazione del concetto di memoria virtuale [8]. Il *Multics* fu il primo

sistema nel quale molte di queste nuove tecniche furono sperimentate.

I progettisti di *Multics* furono i primi ad introdurre il concetto di processo. Per la prima volta il comportamento dei programmi in esecuzione veniva regolamentato facendo loro corrispondere processi dotati di uno stato (pronto, in esecuzione, bloccato) cui venivano assegnate da parte del sistema operativo le risorse sulla base di determinate politiche. Il concetto di processo è stato poi adottato da tutti i sistemi operativi ed è alla base delle metodologie e tecniche di programmazione concorrente [3].

Le tecniche di comunicazione tra processi utilizzate in *Unix* ed in altri sistemi operativi attuali derivano da quelle introdotte in *Multics*. L'aspetto forse più interessante introdotto da *Multics* è quello che riguarda la protezione con l'adozione di un'architettura basata su più livelli (*ring architecture*) che è una generalizzazione di quella classica basata sui due livelli (*user e supervisor*). I livelli di protezione sono organizzati in modo gerarchico in una struttura ad anelli numerati da 0 a 7. In generale indicando con R_i e R_j due generici anelli, se $i < j$ significa che R_i ha più diritti di R_j . I primi livelli sono quelli utilizzati dal sistema operativo (R_0 è il livello del nucleo), gli altri dai programmi di utente. Il tipo di soluzione adottato è stato ripreso ed è presente in molte architetture di moderni microprocessori.

È importante notare che, praticamente, nello stesso periodo e nello stesso luogo, la IBM sviluppò a sua volta il primo sistema operativo *time-sharing* per le proprie macchine. In particolare, nel 1964, presso il centro scientifico IBM di Cambridge, fu sviluppato il sistema CP/CMS per un particolare modello del sistema IBM/360 [16]. Questo modello, il 67, fu l'unico di tutta la serie 360 a disporre di meccanismi hardware utili per la realizzazione di sistemi *time-sharing*, primo fra tutti il meccanismo della paginazione a due livelli necessario per implementare memorie virtuali di grandi dimensioni.

Il CP/CMS è stato il primo sistema operativo ad implementare l'architettura a macchine virtuali che, successivamente, verrà utilizzata in altri settori applicativi. CP/CMS possedeva infatti due componenti principali: il CP

(*Control Program*) era un sistema a macchine virtuali il quale forniva ad ogni singolo utente un ambiente assimilabile ad un *mainframe* personale; il CMS (*Conversational Monitor System*) era un semplice sistema operativo monoutente progettato per essere eseguito, principalmente, in ambiente CP. Ad ogni utente, quindi, di CP/CMS è fornita la propria macchina virtuale su cui eseguire CMS. Sulla macchina virtuale, però, l'utente poteva caricare un altro sistema operativo al posto del CMS. Ciò risultò estremamente utile come ambiente nel quale sviluppare nuovi sistemi operativi.

Il CP/CMS è stato il predecessore di tutta la famiglia di sistemi operativi VM (*Virtual Machine*) successivamente utilizzati in tutte le macchine della serie IBM/370 [23].

6. I SISTEMI GENERAL PURPOSE (1970-1980)

Dalla fine degli anni '60 e per tutto il decennio 70-80, l'evoluzione dei sistemi operativi continua sull'onda del successo riscattato dai sistemi *time-sharing* e favorita anche dalla realizzazione di architetture dei calcolatori sempre più efficienti e sofisticate, dalla nascita dei primi minicomputer realizzati utilizzando circuiti integrati e dallo sviluppo dei protocolli di comunicazione tra calcolatori (TCP/IP). In particolare, nascono i primi sistemi operativi dedicati ad applicazioni *real-time*. Vengono inoltre sviluppati sistemi operativi *general purpose* dove gli utenti possono far girare programmi con la filosofia tipica di sistemi *batch* e, al tempo stesso, eseguire altri programmi in *time-sharing* e specificare al sistema anche la necessità di eseguire programmi con caratteristiche *real-time*.

Nel 1979 esce il primo minicomputer a 16 bit della serie PDP-11 della DEC, serie che avrà un enorme successo sia per la semplicità con cui poteva essere programmato sia per il costo notevolmente inferiore ai grossi *mainframe*.

Nel 1973 la Digital fornisce per la serie PDP-11 un sistema operativo molto semplice dedicato sia allo sviluppo di programmi che a semplici applicazioni *real-time* (*data acquisition*): il sistema RT-11 (Real-Time-11). Si trattava di un sistema operativo monoutente che non forniva il *multitasking* sebbene potesse es-

sere configurato per fornire supporto ad un singolo processo in “foreground” (ad alta priorità) contemporaneamente ad un singolo processo in “background” (bassa priorità). Nel 1974, sempre per la famiglia di calcolatori PDP-11, esce il sistema RSX-11, un sistema multiprogrammato ancora adatto ad applicazioni *real-time*, un sistema che avrà notevoli influenze sul progetto di successivi sistemi, ivi incluso Windows NT.

Nel 1977 la DEC fornisce una nuova serie di sistemi destinati a sostituire le macchine della serie PDP-11: i sistemi della serie VAX-11 per i quali viene realizzato il sistema operativo VMS (*Virtual Memory System* [24]) che recepisce molte caratteristiche dal sistema RSX-11 e che avrà un notevole successo, in particolare in ambienti scientifici, dove è rimasto in uso anche in anni recenti operando anche sulle macchine della serie Alpha e apprezzato in particolare per la sua flessibilità e sicurezza. VMS è un sistema operativo multiutente, multiprogrammato concepito per essere utilizzato come sistema *time-sharing*, particolarmente adatto per applicazioni basate su transazioni, ma abilitato ad eseguire anche programmi in *batch* e, specificando opportunamente le relative priorità, anche in tempo reale. È stato utilizzato per l'architettura di rete DECnet, sviluppata da Digital, oltre che per l'implementazione dei protocolli TCP/IP.

Dopo la vicenda OS/360 anche IBM decise di sviluppare un sistema operativo di tipo *general purpose* e nel 1974 rilasciò la prima versione del sistema MVS (*Multiple Virtual Storage*) destinato ad operare sulle macchine della serie 370 e, successivamente, 390 (in quel caso noto anche col nome di OS/390). MVS, in quanto successore di OS/360 era adatto ad eseguire *job* di tipo *batch* ma, mentre uno di questi girava in background, era possibile operare anche in maniera interattiva mediante l'opzione *time-sharing*. Era un sistema in grado di fornire il supporto anche a configurazioni multielaboratore e garantiva l'interconnessione di più macchine mediate i protocolli SNA (*System Network Architecture*), protocolli proprietari, e successivamente anche mediante i protocolli TCP/IP.

Nel 1969 Brinch Hansen sviluppò il sistema RC 4000 secondo un modello strutturale che di-

venterà noto come “modello a scambio di messaggi” o “modello a memoria locale” duale rispetto a quello utilizzato per lo sviluppo del THE basato sul “modello a memoria comune”. RC 4000 è il primo sistema che introduce il concetto di *microkernel*. Anche in questo sistema un utente può scegliere di aprire una sessione *multitasking* con *preemption*, un altro può scegliere la modalità *single-user mode* per operare in *batch*. Anche lo *scheduling real-time* è previsto utilizzando la possibilità di inviare messaggi al processo *timer*.

7. LO SVILUPPO DI UNIX

La nascita di *Unix* si deve in gran parte agli ostacoli incontrati nel trasformare il progetto *Multics* in un prodotto industriale (riquadro). L'obiettivo degli sviluppatori fu quello di realizzare un sistema operativo che pur mantenendo molte delle caratteristiche di *Multics*, ne semplificasse la struttura, riducendone sensibilmente le dimensioni e la complessità.

Mentre la prima versione era stata scritta in assembler, le versioni successive furono scritte in un linguaggio ad alto livello ideato da Ken Thompson che fu chiamato B (basato

Nascita di UNIX

La nascita di *Unix* si deve in gran parte agli ostacoli incontrati nel trasformare il progetto *Multics* in un prodotto industriale. I *Bell Labs* che avevano partecipato fin dall'inizio al progetto, ritenendo troppo complessa l'operazione, la abbandonarono nel 1969. Alcuni ricercatori dei *Labs* decisero nonostante tutto di continuare lo sviluppo del progetto. Furono in particolare Ken Thompson e Dennis Ritchie a non arrendersi: fu grazie ai loro sforzi che, su una macchina non più utilizzata all'interno dei *Labs*, un PDP-7, nacque la versione finale di *Unics* (in seguito *Unix*). Nome che stava a sottolineare la voluta semplicità del progetto rispetto alla mal gestita complessità di *Multics*. Visto l'interesse desto tra i ricercatori dei *Labs*, *Unix* fu trasportato su un calcolatore più potente della Digital, il PD11/20 e più tardi sul PDP11/45 ed infine sul PDP11/70.

Nel 1974 Ritchie e Thompson pubblicarono un articolo su *Unix* [19], per il quale i due autori ricevettero più avanti il prestigioso premio Turing dell'ACM, che spinse molte Università a chiedere ai *Bell Labs* una copia del sistema operativo. Il che avvenne, a condizioni particolarmente vantaggiose essendo fatto divieto dalle leggi federali americane alla AT&T (proprietaria dei *Bell Labs*) di vendere prodotti per computer.

sul linguaggio BCPL) con l'obiettivo di aumentarne la portabilità. La versione definitiva fu scritta in un nuovo linguaggio progettato da Ritchie, chiamato C con il risultato, anche, di dimostrare quanto fosse vantaggioso utilizzare un linguaggio di alto livello per scrivere, se non tutto, almeno la gran parte del codice di un sistema operativo.

La versione 6 divenne la prima disponibile fuori dai Bell Labs nel 1976. La versione 7, successiva, uscì nel 1978 cominciò ad essere usata, oltre che nelle Università, anche nelle aziende industriali.

Una realizzazione importante di *Unix* fu quella sviluppata presso l'Università di Berkeley in California, conosciuta come *Berkeley Software Distribution Unix* (BSD *Unix*), di cui furono prodotte diverse versioni fino alla più nota, la 4.3BSD sviluppata per il calcolatore VAX [15]. Questa versione apportò significativi arricchimenti alle versioni precedenti, tra cui l'utilizzo della memoria virtuale e dell'impaginazione ed in particolare l'utilizzo delle socket come strumento per la comunicazione tra processi in rete tramite TCP/IP.

Negli anni '80 la grande popolarità di *Unix* ed il fatto che fosse disponibile il codice sorgente, inevitabilmente determinò il proliferare di svariate realizzazioni, spesso con notevoli differenze tra loro che hanno ostacolato una piena portabilità delle applicazioni. In particolare, le due famiglie di sistemi *Unix* maggiormente affermatesi sono state *Unix System V* [4], prodotto dai laboratori dell'AT&T¹ e BSD *Unix*.

Nel 1988, grazie all'intervento dell'IEEE Standard Board, fu definito lo standard POSIX (*Portable Operating System Unix*) [14] che definisce un'interfaccia (API) tra sistema operativo e applicazioni. Le applicazioni che utilizzano i servizi del sistema operativo attraverso l'interfaccia POSIX hanno ottime caratteristiche di portabilità verso altri sistemi operativi conformi allo standard. Lo standard

POSIX fu riconosciuto dall'*International Standard Organization* (ISO) e le due versioni *Unix System V* e BSD (versione 4.3), pur mantenendo alcune differenze tra loro, sono state sviluppate entrambe in maniera conforme allo standard POSIX².

Alcune aziende si crearono la loro versione di *Unix*, come *Solaris* della SUN (basata su *System V*)³ e AIX di IBM (basato sia su *System V* sia su BSD). Anche Microsoft si interessò per un certo numero di anni ad *Unix* creando il sistema operativo XENIX.

La derivazione di *Unix* da *Multics* fece sì che fin dalle sue prime versioni fossero presenti nel sistema operativo molte delle caratteristiche di *Multics*. Innanzi tutto il concetto di processo, la multiprogrammazione e la gestione della multiutenza secondo le tecniche del *time-sharing*, l'uso del file per la rappresentazione di ogni risorsa di sistema, l'organizzazione gerarchica del file system. Il sistema di protezione fu volutamente semplificato con l'introduzione delle liste di controllo degli accessi per regolare l'accesso degli utenti ai file.

Le versioni successive portarono ad una crescita delle funzionalità di *Unix*, in particolare la memoria virtuale, le modalità di interazione tra i processi con l'utilizzo dei segnali per la loro sincronizzazione e delle *pipe* e *socket* per la comunicazione rispettivamente in ambiente a memoria comune e in sistemi distribuiti. Per facilitare l'interazione dell'utente, all'originario linguaggio di comandi fu affiancato un sistema di interfacce grafiche che fanno uso di finestre, icone etc. La più nota tra queste è CDE (*Common Desktop Environment*) frutto di un lavoro di standardizzazione portato avanti dalle più importanti aziende produttrici di *Unix*, che presenta caratteristiche analoghe a quelle dell'interfaccia grafica del sistema operativo Windows NT.

Al di là di questi aspetti, che oggi sono propri di tutti i moderni sistemi operativi, *Unix*

¹ Nel 1984 il governo degli Stati Uniti permise la suddivisione della AT&T; nacque così una consociata che trattava computer che mise sul mercato UNIX System III, che fu infine rimpiazzato da UNIX System V.

² Un altro gruppo di sviluppatori di UNIX (IBM, DEC, HP e molti altri) preoccupati dal fatto che AT&T avesse il controllo di UNIX formarono la OSF (*Open Software Foundation*) con lo scopo di produrre una propria versione di UNIX senza vincoli di proprietà. Nel tempo la OSF lentamente scomparve a favore delle versioni AT&T e BSD.

³ Solaris, il cui primo nome era SunOS, fu creato da Bill Joy che fondò nel 1982 la Sun Microsystems.

ricopre nella storia dei sistemi operativi una posizione di grande importanza. Innanzitutto, per i motivi detti, è il primo sistema operativo non proprietario di cui fu reso disponibile liberamente il codice. Il fatto che fin dalle sue prime versioni, *Unix* venne scritto utilizzando, per la maggior parte del progetto, il linguaggio C (circa il 60%), ne facilitò la diffusione presso Università ed aziende. Infatti, oltre ai vantaggi relativi ad una sua più agevole comprensione, il sistema risultava dotato di una buona portabilità, intesa come capacità di fornire lo stesso ambiente di esecuzione e di sviluppo su architetture diverse. Questa caratteristica, decisamente inusuale per l'epoca in cui *Unix* è nato, è stata la chiave di volta della capillare diffusione che ha raggiunto ai tempi nostri.

Di pari passo anche il linguaggio C, nato come formalismo per lo sviluppo di *Unix*, si è diffuso in maniera tale da divenire uno standard di fatto tra i linguaggi di programmazione imperativi rimpiazzando, per esempio, gradualmente il linguaggio *Pascal* come linguaggio prescelto nei corsi di programmazione nelle università (anche se, a parere degli autori, si possono nutrire fondati dubbi sulla sua validità a fini didattici).

8. LINUX

Visto il successo di *Unix*, furono sviluppati molti altri sistemi operativi basati su di esso. Nel 1985, presso l'*Università di Carnegie Mellon*, fu sviluppato il progetto Mach destinato a realizzare un sistema operativo (il sistema Mach [1]) con lo scopo di fornire un ambiente di supporto alla ricerca sui sistemi operativi. Tale sistema forniva un'interfaccia (API) simile a quella di *Unix* ed era in grado di girare su un'ampia varietà di calcolatori commerciali, fornendo anche il supporto ad architetture multielaboratore. Ma la caratteristica più importante di questo sistema è stata quella di essere strutturato secondo il modello a *microkernel* di cui è stato sicuramente uno dei primi esempi e forse il più famoso. L'idea che sta alla base del *microkernel* è quella di fornire minime funzionalità nel *kernel* per renderlo efficiente ed affidabile (in generale le funzioni del *microkernel* coincidono con la gestione dei processi, delle loro interazioni e

con la gestione delle interruzioni), mentre funzioni come la gestione della memoria ed il file system operano come processi utente.

Pur essendo ormai terminata l'esperienza di ricerca di Mach, questa ha influenzato molti sistemi commerciali, fra questi Mac-OS, il sistema operativo dei calcolatori Macintosh.

Nel 1987, Andrew Tanenbaum dell'*Università di Vrije di Amsterdam* sviluppò MINIX [22], una versione ridotta di *Unix* realizzata per insegnare le basi dei sistemi operativi, funzionalmente molto simile a *Unix* e conforme a POSIX. Va ricordato che MINIX fu anch'esso uno dei primi sistemi operativi tipo-*Unix* che utilizzò una struttura a *microkernel*.

Il desiderio di scrivere una versione professionale gratuita (contrapposta a quella didattica) di MINIX fece sì che uno studente dell'Università di Helsinki, Linus Torvalds, nel 1991 sviluppasse una nuova versione del *kernel* di MINIX per il processore 80386, la prima vera CPU a 32 bit dell'Intel (riquadro p. 61).

I sorgenti della prima versione di *Linux* (il codice totale era di 9300 righe di C e 950 righe di assembler) furono resi accessibili gratuitamente attraverso Internet [25].

Questa possibilità invogliò molti programmatori ad analizzare, modificare ed estendere le varie componenti del sistema operativo. Questo fenomeno provocò un velocissimo progresso nel completamento e nel miglioramento del sistema, che in breve tempo, da un prototipo "giocattolo" di sistema operativo divenne un sistema operativo completo, moderno, robusto ed efficace che aveva come suo punto di riferimento *Unix*, la sua struttura, le sue chiamate di sistema, le librerie, gli algoritmi (più dell'80% delle chiamate di sistema *Linux* sono una copia esatta delle corrispondenti chiamate di sistema in POSIX, BSD o System V), ma che nel tempo è stato modificato ed esteso acquisendo alcune caratteristiche che oggi lo differenziano rispetto al sistema da cui ha tratto ispirazione.

Una delle differenze più significative riguarda la gestione dei processi; in *Linux* sono presenti e gestiti a livello di *kernel* i *threads* (processi leggeri) che rappresentano quindi l'unità di *scheduling* e che consentano rispetto ad una gestione basata solo sui *processi pesanti* di ottenere migliori prestazioni in termini di efficienza e sfruttamento del parallelismo.

Nascita e sviluppo di Linux

Linux nasce nel 1991 in seguito al lavoro di uno studente finlandese, Linus Torvalds, che, con lo scopo di studiare, sperimentare e, al tempo stesso, divertirsi, riprogettò il *kernel* del sistema operativo MINIX, realizzato da Tanenbaum per scopi didattici, cercando di sfruttare al meglio le caratteristiche delle architetture basate sui microprocessori Intel i386, all'epoca le più diffuse e, secondo Linus, non efficientemente utilizzate da MINIX che era strutturato secondo il modello a *microkernel*.

Con questo obiettivo il nuovo *kernel* fu organizzato secondo il modello monolitico, tipico di *Unix*, sicuramente meno flessibile rispetto a un'organizzazione a *microkernel*, ma certamente più efficiente.

La prima versione di *Linux*, oltre al nuovo *kernel* completamente riprogrammato, continuava ad utilizzare, per semplicità, la restante parte del sistema operativo MINIX da cui il nuovo progetto era originato. Successivamente però, anche per svincolarsi dai soli scopi didattici a cui MINIX era orientato, Torvalds sostituì quella parte di MINIX con il software relativo al progetto GNU (progetto lanciato nel 1984 dalla *Free Software Foundation* per sviluppare un sistema operativo *Unix-compatibile* completo e che fosse reso disponibile come software libero).

Per questo motivo, Torvalds decise inoltre di rendere disponibile tutto il software prodotto con la licenza GPL (*General Public License*) della GNU dando quindi vita a un progetto *open source* aperto a contributi esterni, cosa che, come è noto, ha fortemente contribuito al successo di *Linux*.

La nascita di *Linux*, originariamente come alternativo a MINIX, dette vita a una lunga diatriba su internet fra Torvalds e Tanenbaum. Quest'ultimo, nel 1992, probabilmente infastidito dal crescente interesse verso *Linux*, scrisse un messaggio dal titolo estremamente chiaro: "*Linux is obsolete*", criticando aspramente l'architettura basata su *kernel* monolitico sicuramente meno innovativa di quella di MINIX. A tale messaggio rispose Torvalds cercando di mettere in evidenza i vantaggi del nuovo sistema in termini di efficienza. Il dibattito è poi continuato nel tempo come si può verificare su vari siti internet (vedi per esempio, http://www.dina.dk/~abraham/Linus_vs_Tanenbaum.html).

Linux supporta inoltre molte funzionalità avanzate quali l'elaborazione multipla parallela (SMP, *Symmetric Multiprocessing*), l'accesso a memoria non uniforme (NUMA, *Non Uniform Memory Access*) ed il supporto ad un ampio spettro di architetture hardware. La complessità raggiunta dall'attuale prodotto rende difficile, ed in alcune situazioni infaticabile, la pratica adottata negli anni passati di scaricare personalmente il software da Internet e provvedere direttamente alla sua installazione. Sono nate, per ovviare a questo problema, delle distribuzioni che includono il *kernel*, le applicazioni ed interfacce utente, insieme ad altri strumenti ed accessori. Attualmente si contano più di 300 tipi di distribuzione. Tra questi possiamo citare *Red Hat*, *SuSE (Novell)*, *Debian*, *Mandriva Linux*, *Slackware*.

Bibliografia

- [1] Accetta M., Baron R., Bolosky W., Golub D.B., Rashid R., Tevanian A.S., Young M.: *Mach: a New Kernel Foundation for Unix Development*. Proceedings of the Summer 1986 USENIX Conference, June 1986.
- [2] Ancilotti P., Boari M., Ciampolini A., Lipari G.: *Sistemi Operativi*. McGraw-Hill, 2008, 2ª edizione.
- [3] Ancilotti P., Boari M.: *Programmazione concorrente e distribuita*. McGraw-Hill, 2007.
- [4] Bach M.J.: *The Design of the Unix Operating System*. Prentice-Hall, Englewood Cliffs, N.Y., 1987.
- [5] Brooks Jr. F.P.: *The Mythical Man-Month: Essays in Software Engineering*. Reading, Mass: Addison Wesley, 1975.
- [6] Corbato F.J., Vyssotsky, *Introduction and overview of the Multics system*. Proceedings of the AFIPS Fall Joint Computer Conference, 1962.
- [7] Crisman P.A. et al.: *The Compatible Timesharing System*. Cambridge, Mass: MIT Press, 1964.
- [8] Denning P.J.: *Virtual Memory*. ACM Computing Surveys, Settembre 1970.
- [9] Dijkstra E.W.: The structure of the THE Multiprogramming System. *Communications of the ACM*, Vol. 11, n. 5, May 1968.
- [10] Dijkstra E.W.: Go To Statement Considered Harmful. *Communications of the ACM*, Vol. 11, 1968, p. 147-148.
- [11] Gray J.N.: Notes on Database Operating Systems. *Lectures Notes in Computer Science*, Vol. 60, Operating Systems – An Advanced Course, Springer Verlag, 1978.
- [12] Grosch H.R.J.: The Way It Was in 1957. *Datamation*, September 1977, p. 121-132.
- [13] IBM: *IBM System/360 Operating System – Introduction*. IBM System Reference Library, file Number S360-20, Order number GC28-6534-3.
- [14] Inst of Elect & Electronic Eng: *IEEE Standard Portable Operating System Interface for Computer Environments 1003.1*. Revised edition (September 1988).

- [15] Leffler S.J., McKusick M.K., Karels M.J., Quarterman J.S.: *The Design and Implementation of the 4.3BSD UNIX Operating System*. Addison Wesley, November, 1989.
- [17] Meyer R. A., Seawright L. H.: A virtual machine time-sharing system. *IBM Systems Journal*, Vol. 9, n. 3, September 1970, p. 199-218.
- [17] Mooney J.: *Early Operating Systems*. West Virginia University, www.csee.wvu.edu/~jdm/classes/cs258/OScat/early.html, January 27, 2004.
- [18] Naur P., Randell B. (eds.): *Software Engineering*. NATO Scientific Affairs Division, Brussels. Report on a Conference Sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 October, 1998, 1969.
- [19] Ritchie D.M., Thompson K.: The Unix Time-Sharing System. *Communications of the ACM*, Vol. 17, n. 7, July 1974.
- [20] Silberschatz A., Galvin P.B., Gagne G.: *Operating System Concepts*. Seventh Edition, John Wiley & Sons, December 2004.
- [21] Stallings W.: *Operating Systems: Internals and Design Principles*. Fifth edition, Prentice Hall, 2005.
- [22] Tanenbaum A.S, Woodhull A.S.: *Operating System: design and implementation*. 3 ed. Upper Saddle River, NJ: Prentice Hall, 2006.
- [23] Weizer N.: A History of Operating Systems. *Datamation*, January 1981, p. 119-126.
- [24] <http://h18000.www1.hp.com/info/SP2501/SP2501PF.PDF>
- [25] <http://www.linux.org/info/index.html>

PAOLO ANCILOTTI è professore ordinario di Sistemi Operativi presso la Scuola Superiore Sant'Anna di Pisa di cui è stato Direttore. È autore di numerosi articoli scientifici ed alcuni libri. Ha interessi di ricerca nel settore dei sistemi operativi, dei sistemi real-time e della programmazione concorrente.
E-mail: p.ancilotti@sssup.it

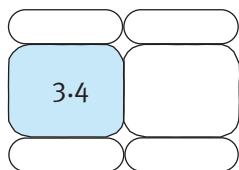
MAURELIO BOARI è professore ordinario di calcolatori elettronici presso la Facoltà di Ingegneria dell'Università di Bologna. È autore di numerosi articoli scientifici ed alcuni libri. Ha interessi di ricerca nel settore dei sistemi distribuiti, linguaggi di programmazione e sistemi operativi. È attualmente delegato del Rettore per l'informatica e le reti di Ateneo.
E-mail: maurelio.boari@unibo.it



RETE IP E INTERNET

UN QUADRO EVOLUTIVO

Danilo Ciscato



Questo articolo, partendo dall'evoluzione storica, traccia un sintetico quadro introduttivo relativo alle evoluzioni del paradigma Internet verso tutte le forme di comunicazione multimediale. Lo sviluppo delle reti IP è la base fondamentale di questa rivoluzione in cui saremo proiettati e che porterà all'utilizzo di uno strumento di comunicazione di massa in grado di consentire lo scambio di informazioni di qualunque genere e di coniugare sia le comunicazioni in tempo reale che quelle in tempo differito.

1. INTRODUZIONE

Nell'ultimo decennio, lo sviluppo delle reti di comunicazione ha raggiunto traguardi tecnologici e di diffusione di massa tali da determinare un profondo cambiamento nello scenario dei media, grazie anche alla crescita dei servizi Internet, che nel panorama dei *mass-media* (giornali, cinema, radio-televisione ecc.) hanno assunto un ruolo di crescente rilievo. In questo cambiamento ha dato un contributo fondamentale l'evoluzione dell'*Information and Communication Technology* e, in particolare, hanno giocato un ruolo di rilievo i nuovi paradigmi della rete IP.

L'evoluzione cui abbiamo assistito sinora è tuttavia solo l'inizio di un percorso di innovazione, che ha già portato a profondi cambiamenti nel modo in cui comunichiamo, ma che avrà conseguenze ancora più rilevanti sugli individui, sulle aziende e di conseguenza sull'intera società.

Il presente articolo si propone di dare un quadro introduttivo delle evoluzioni più recenti delle reti IP e dei servizi Internet che ca-

ratterizzeranno il futuro del mondo della comunicazione.

2. EVOLUZIONE STORICA: DAI SISTEMI DI TELECOMUNICAZIONI ANALOGICI ALLA RETE IP

Per comprendere la portata del cambiamento bisogna partire dalla storia delle comunicazioni.

Le reti di telecomunicazioni esistono ormai da moltissimo tempo: prima il telegrafo (1835) e poi il telefono (1876) hanno abilitato il concetto di comunicazione umana a distanza, che ha avuto indubbiamente conseguenze enormi sullo sviluppo economico e sociale.

Dopo l'avvento dei primi elaboratori elettronici si è fatta strada anche il concetto di trasmissione ed elaborazione dei dati a distanza, arrivando così alla creazione di reti dedicate alla comunicazione tra macchine e non più soltanto tra esseri umani.

Inizialmente si è trattato di fornire accesso ad elaboratori centrali (*mainframe*) tramite terminali remoti non intelligenti e successivamente,

con lo sviluppo dell'informatica distribuita e dei *Personal Computer*, si è passati alla realizzazione di vere e proprie Reti di Computer.

Parallelamente lo sviluppo delle tecnologie digitali ha portato alla realizzazione di reti a larga banda, cioè ad alta capacità trasmissiva, protette assai efficacemente da tutte le anomalie di trasmissione ed in grado di trasportare indifferentemente qualunque tipo di informazione codificabile in *bit* (dati, immagini, voce, video ecc.), grazie anche all'introduzione di tecniche di "compressione" dei segnali.

Le reti di computer erano state impiegate da decenni, ma per molto tempo sono state caratterizzate dall'impiego di tecnologie proprietarie, sviluppate dai diversi produttori e quindi non interoperabili tra loro. Queste caratteristiche ne limitavano l'impiego ad applicazioni specifiche, di tipo professionale o scientifico, ma sempre confinato ad ambiti ristretti.

L'affermazione del *protocollo IP* e del *paradigma di rete aperta*, in grado di gestire tutti i servizi Internet, ha consentito per la prima volta la realizzazione di una rete globale, in grado di mettere in comunicazione i sistemi di qualunque produttore, indipendentemente dall'*hardware* e dal *software* utilizzati e a costi contenuti. Lo sviluppo di questa infrastruttura di rete any-to-any, in grado cioè di far comunicare un computer collegato in rete con qualunque altro, ha abilitato l'affermazione di un nuovo modello di accesso alle informazioni, basato sul concetto di ipertesto.

Grazie a un *linguaggio standard* (HTML) [1] per la formattazione dei contenuti è stata resa possibile la "navigazione" tra le informazioni presenti su tutti gli elaboratori collegati in rete, attraverso il *web browser*, un programma disponibile su tutti i computer. È nato così il concetto di Web, che ha portato negli ultimi vent'anni ad una crescita esponenziale della rete Internet.

I fattori di successo di Internet sono stati fondamentalmente due:

□ una infrastruttura di **rete aperta**, che ha potuto espandersi semplicemente attraverso l'interconnessione di reti basate sul protocollo IP realizzate da soggetti diversi, senza una regia centrale (come tradizionalmente avveniva nelle reti di telecomunicazioni possedute dagli operatori);

□ una modalità di condivisione e accesso alle informazioni semplice e standardizzata: il **Web**.

Questi due fattori hanno dato origine a una **reazione a catena**:

□ la rete aperta ha reso possibile il Web;

□ l'utilità e la facilità d'uso del Web ha determinato l'espansione della rete (semplice da realizzare, in quanto aperta e senza una regia centrale);

□ una rete più vasta e capillare ha reso ancora più utile e potente il Web.

In pratica lo sviluppo della rete è incentivato dalla quantità di informazioni accessibili online, che a loro volta aumentano all'espandersi della rete.

Guardando le cose da un altro punto di vista si può osservare che questo primo modello di Web prevedeva una persona con un computer dotato di un *Web-browser*, con il quale poteva accedere ai contenuti presenti su tutti i *Web-server* collegati alla rete. Perciò, sebbene Internet fosse tecnicamente una rete nata per collegare tra loro dei computer attraverso il protocollo IP, l'applicazione prevalente era l'accesso da parte delle persone a informazioni predisposte sui *server*.

In questa fase aziende e istituzioni hanno realizzato siti rivolti ai propri interlocutori, inizialmente per illustrare prodotti e servizi, e successivamente per far svolgere autonomamente transazioni. Nascevano così i concetti di *e-commerce* e di *e-Government*. Questo modello è noto come **Web 1.0**.

Si tratta certamente di un'applicazione molto importante della rete IP, che continuerà ad essere impiegata, ma fu subito chiaro che le potenzialità insite nella rete Internet avrebbero permesso di allargarne significativamente le modalità di utilizzo.

3. LO SVILUPPO DEL WEB 2.0

La grande flessibilità ed apertura del protocollo IP ha consentito la sperimentazione e lo sviluppo di nuove applicazioni da parte di chiunque avesse accesso alla rete. L'innovazione poteva partire dal basso, da parte di chiunque avesse un'idea da sperimentare e condividere, senza la necessità di dover aspettare progetti calati dall'alto. Come conseguenza si è rapidamente verificata l'evol-



zione nelle tipologie di contenuti presenti in rete e nelle relative tecniche di presentazione, passando da semplici ipertesti a contenuti grafici e multimediali.

Le innovazioni concettualmente più importanti sono state due:

1. l'utilizzo della rete IP per *comunicazioni in tempo reale*, in particolare il trasporto della voce (VoIP);

2. la *predisposizione di nuovi contenuti* da parte degli utilizzatori e, più in generale, l'affermazione del paradigma di Internet come strumento di comunicazione tra persone.

Le persone quindi non sono più semplicemente utilizzatori passivi di siti web predisposti da altri soggetti, ma diventano protagonisti attivi.

Internet è stata così definitivamente consacrata al ruolo di **new media** e si è affermato un nuovo modello di utilizzo della rete, il cosiddetto Web 2.0, caratterizzato da blog, social networks, wikipedia, upload di video, P2P (Peer to Peer) ecc. [2].

È importante osservare che in veste di *new media* Internet coniuga per la prima volta modalità di comunicazioni ben diverse tra loro, quali:

- da *uno a molti*, tipica dei mezzi di comunicazione di massa come la stampa, la radio e la televisione;

- da *uno a uno*, tipica della telefonia; ma coniuga anche la possibilità di comunicare sia in *tempo reale* che in *tempo differito* (cioè tra utenti che non sono simultaneamente collegati).

Uno strumento di comunicazione di massa in grado di combinare tutte queste caratteristiche non era mai stato creato prima.

4. IMPATTO SULL'INFRASTRUTTURA DI RETE: LA NEXT GENERATION NETWORK

L'evoluzione di Internet ha determinato quindi un uso estensivo della rete IP, che ne costituisce l'infrastruttura di base, ben più ampio di quanto immaginato inizialmente.

La limitazione incontrata per prima, all'epoca del Web 1.0 è stata quella legata alla *larghezza di banda disponibile*. Con la diffusione di contenuti grafici e multimediali si è reso necessario sviluppare nuove tecniche di trasmissione

(ADSL) per sfruttare al meglio, con prestazioni del tutto inimmaginabili fino a pochi anni fa, l'unica infrastruttura fisica di accesso all'utente disponibile capillarmente e cioè il doppino in rame della tradizionale rete telefonica.

Tuttavia è ormai definitivamente affermata la consapevolezza che sia necessario creare una rete di nuova generazione, progettata per far fronte alle esigenze di sviluppo delle nuove forme di comunicazione, senza le limitazioni strutturali e architetture della rete telefonica, che era stata realizzata con obiettivi ben diversi. Questa rete di nuova generazione, nota come NGN (*Next Generation Network*), dovrà avere caratteristiche tali da poter supportare efficacemente il Web 2.0 e gli sviluppi futuri di Internet.

Spesso parlando di NGN e più specificatamente del suo segmento di accesso NGAN, per consentire la larga banda fino all'utente, si fa riferimento ad un impiego esteso delle reti ottiche anche nell'area di accesso o all'impiego di tecnologie VDSL per un doppino di utente limitato alla sola terminazione. Il solo incremento della capacità trasmissiva, per quanto indispensabile, non risolve però tutte le problematiche legate alle nuove modalità di utilizzo e alla coesistenza di diverse classi di traffico.

La **simmetria** è un attributo spesso trascurato della NGAN. Nel Web 1.0 il flusso di informazioni era asimmetrico: il traffico dal web server al web browser era di gran lunga superiore a quello in senso inverso. L'ADSL fu perciò progettato per fornire prestazioni in *download* molto superiori che in *upload* (la A sta, infatti, per Asymmetrical).

Tuttavia nel Web 2.0 le persone producono dei contenuti e le prestazioni di *upload* assumono importanza, e quindi una vera rete NGAN dovrà essere simmetrica.

A questo proposito occorre osservare che lo sviluppo di servizi IPTV, vale a dire la trasmissione di canali televisivi sulla rete IP, può essere fuorviante per la pianificazione dell'architettura di rete. Finora, infatti, i servizi IPTV si limitano ad effettuare la diffusione di canali televisivi *broadcast* o di contenuti video in modalità *streaming on-demand*, mantenendo quindi un profilo di traffico fortemente asimmetrico. Usare una rete IP per trasmettere contenuti televisivi o cinematografici già disponibili per altre vie non è certo un servi-

zio avveniristico e non può essere considerato il punto d'arrivo di una NGN. I contenuti autoprodotti e l'interattività porteranno senza dubbio all'evoluzione dei contenuti video e quindi dei profili di traffico.

La NGN deve pertanto essere una rete simmetrica a larga banda, ma per quanto le risorse trasmissive siano elevate non saranno mai illimitate. Per poter dare garanzie sulle prestazioni agli utilizzatori è *necessario aumentare l'intelligenza della rete*, rendendola in grado di distinguere e differenziare varie tipologie di traffico, in altre parole di gestire la cosiddetta Qualità del Servizio (riquadro).

Le aspettative, infatti, sulla qualità dei servizi ottenibili su Internet sono molto cresciute nel tempo. I primi collegamenti telefonici realizzati su Internet attraverso il VoIP (Voice over IP) avevano una qualità ed affidabilità inferiori a quelle della rete telefonica tradizionale, ma molte persone erano disposte ad accettarne le prestazioni comunque, perché si trattava di un servizio gratuito. Oggi sono offerti molti servizi commerciali basati sul VoIP, che gradualmente sostituiscono quelli tradizionali ed è perciò naturale che le aspettative di qualità e di affidabilità siano notevolmente cresciute.

Qualità del Servizio (QoS): tecnologia e considerazioni di business

Con il termine generico di **QoS** (*Quality of Service*) ci si riferisce ad una serie di funzioni intelligenti di smistamento del traffico all'interno della rete. La QoS, in particolare:

- gestisce la *priorità dei diversi flussi di traffico*, conciliando l'obiettivo di dare garanzie prestazionali a certe classi di traffico con quello di massimizzare il fattore di utilizzo dell'infrastruttura;
- *controlla il flusso* di pacchetti trasmessi in rete dagli utilizzatori;
- *regola la trasmissione di traffico a bassa priorità*, quando necessario, per evitare la congestione della rete e per garantire il rispetto di predeterminati livelli di servizio al traffico ad alta priorità.

Per la QoS, semplificando, possono essere presi in considerazione tre approcci di base:

- **Best effort**: questa modalità significa semplicemente l'assenza di qualunque meccanismo di QoS. La rete cerca di trasportare il traffico al meglio, senza garanzie prestazionali o di consegna di alcun tipo. In caso di congestioni e sovraccarichi momentanei il traffico in eccesso è scartato senza criteri di scelta particolari.

- **Differentiated Services** (*DiffServ*) - a volte chiamato "*Soft QoS*": i pacchetti di dati sono suddivisi in classi e contrassegnati. Con l'approccio DiffServ la rete offre servizi differenziati trattando alcune classi di traffico meglio di altre [4].

- **Reserved Services** (o *Integrated Services*, *IntServ*) - come lo stesso nome chiarisce, in questo caso sono riservate specifiche risorse di rete per determinati flussi di traffico. Questo approccio, chiamato anche "*Hard QoS*", garantisce livelli di servizio predeterminati a specifici flussi di traffico [5].

I tre approcci non sono alternativi, ma complementari, e hanno caratteristiche, applicabilità e scalabilità diverse. Tuttavia nella realtà articolata e globale di Internet, le problematiche tecniche ed architetturali per la gestione del QoS si mescolano con le considerazioni di business e organizzative, rendendo assai complesso definire regole precise in materia.

Infatti, nel momento in cui si vuole andare oltre il modello best-effort e definire diverse classi di traffico occorre decidere a chi appartiene la responsabilità di associare i pacchetti alle diverse classi e accettare l'idea che chi gestisce la rete non tratterà tutti i flussi allo stesso modo. È comprensibile che gli operatori che gestiscono le reti vedano la QoS come un *abilitatore per vendere servizi con qualità differenziata*, cioè per associare ai livelli prestazionali più elevati o garantiti ad un prezzo maggiore. Viceversa gli operatori che attualmente offrono contenuti sul Web senza disporre di una propria infrastruttura di rete (i cosiddetti operatori *Over The Top*) vedono la QoS come un rischio per il proprio modello di business, in quanto per mantenere livelli prestazionali adeguati potrebbero vedersi costretti a condividere parte dei profitti con gli operatori di rete oppure dovrebbero rassegnarsi a essere relegati nelle classi di servizio più basse, con il rischio di essere penalizzati rispetto ai loro concorrenti che viaggiano su classi privilegiate.

Questo è il tema della cosiddetta **Net Neutrality** e come è facile immaginare è assai caldo e dibattuto. Come in tutti i temi complessi non esiste una posizione corretta ed un'errata, ma esistono posizioni legittimamente diverse, in quanto appartenenti ad aziende che operano con modelli di business diversi e basati su assets complementari (rete e/o contenuti). È semplicistico quindi ridurre la controversia ad una battaglia tra i paladini della libertà e della democrazia della rete ed operatori di telecomunicazioni che vogliono imporre una sorta di contrappasso. In realtà i principali soggetti coinvolti nel dibattito sono aziende, che operano legittimamente secondo una logica di profitto e che si trovano in competizione tra loro.

La **QoS rappresenta, dunque, una caratteristica qualificante di una rete intelligente** di nuova generazione, necessaria per garantire lo sviluppo su larga scala di servizi di qualità e innovativi. Spetterà quindi a chi ha il compito di definire le regole, trovare il punto di equilibrio, che garantisca ovviamente la libertà di informazione e l'innovazione nei servizi, ma che riesca a far coesistere modelli di business assai differenti, caratterizzati ad esempio da entità degli investimenti e da periodi di ammortamento totalmente diversi.

Analogamente i filmati visualizzabili in **streaming** attraverso Internet hanno avuto tipicamente una qualità limitata, non idonea per la visione su uno schermo televisivo, soprattutto oggi, con la crescente diffusione di schermi televisivi di grandi dimensioni e ad alta definizione (HD). I servizi commerciali **IP-TV** (televisione su rete IP) hanno cominciato a diffondersi e anche in questo caso l'aspettativa di chi acquista un film per vederlo su un moderno televisore a grande schermo è oggi diversa da chi guarda gratuitamente un filmato di *YouTube* su un PC.

Si può, quindi, concludere che l'impiego per la voce o per il video di Internet, per quanto in continuo incremento, non hanno ancora raggiunto la diffusione dei servizi telefonici e televisivi tradizionali e sono quindi ancora ben lontani da costituire un'effettiva alternativa ad essi. Affinché il modello Internet possa realmente diventare non soltanto un nuovo media, ma "il media" per antonomasia, in grado di supportare qualunque forma di comunicazione su scala globale; queste limitazioni vanno completamente eliminate e il tema della Qualità del Servizio diventa di fondamentale importanza e tecnicamente ineludibile.

Il supporto della QoS sulla rete di un singolo operatore è ormai una realtà da diversi anni e ciò ha consentito ad esempio di realizzare reti VoIP con caratteristiche qualitative e di affidabilità paragonabili o addirittura superiori alle reti telefoniche tradizionali. Tuttavia se Internet vuole diventare un *media globale* la gestione della QoS non può limitarsi al dominio di un singolo operatore e, nell'evoluzione di Internet, le modalità di interazione tra operatori saranno quindi cruciali, andando oltre il semplice modello di interconnessione *best effort*. Un altro attributo della NGN che andrebbe approfondito è la *sicurezza*, intesa sia in termini di protezione dell'infrastruttura da attacchi, volti a causarne un malfunzionamento, che di garanzia di confidenzialità delle informazioni e prevenzione dei crimini informatici. Si tratta di un tema ampio, che merita un approfondimento specifico, ma non c'è dubbio che per garantire efficacemente la sicurezza sarà necessario introdurre nuove funzioni nella rete.

Le caratteristiche delle NGN sono pertanto al centro della trasformazione di Internet e do-

vranno essere *molto più sofisticate (intelligenti)* per garantire funzioni di intermediazione efficienti tra operatori, sicurezza e riservatezza degli accessi, scalabilità di crescita, ma soprattutto per avere intrinsecamente la capacità di adattarsi velocemente al fenomeno della trasformazione in atto verso il Web 2.0.

5. LA VIRTUALIZZAZIONE DELLE RISORSE INFORMATICHE E LE APPLICAZIONI MACHINE-TO-MACHINE

I cambiamenti architetturali che stanno avvenendo nella rete IP per consentire la realizzazione di una rete NGN si inseriscono in un contesto più ampio di evoluzione dei sistemi informatici.

La diffusione di reti IP molto flessibili e a larga banda, associata all'ampia disponibilità di risorse di CPU e di memoria di massa distribuite, sta portando, infatti, all'affermazione di **nuove architetture elaborative**. L'interconnessione completa di tutte le risorse informatiche, sia essa a livello di azienda, di Paese o addirittura globale, tramite Internet, è ormai una realtà, che porterà inevitabilmente all'affermazione del concetto di **virtualizzazione**.

Il concetto di virtualizzazione, vale a dire di una visione logica delle risorse, svincolata dalle caratteristiche *hardware* dei sistemi, non è nuovo nell'informatica, ma l'interconnessione in rete permette di applicarlo su scala globale, e non più solo all'interno di un singolo elaboratore, come è avvenuto sin dal tempo dei primi sistemi operativi.

Di questa infrastruttura ICT distribuita saranno, infatti, visibili solo i servizi messi a disposizione e non sarà più rilevante sapere quale sia il server dove risiede l'applicazione o in quali unità di memoria siano allocati i dati, tanto meno le caratteristiche tecniche o prestazionali dei singoli elementi. Questa evoluzione è in atto ormai da tempo e continuerà ad affermarsi ulteriormente, producendo cambiamenti profondi in tutto il settore ICT: da chi sviluppa soluzioni applicative a chi sviluppa software di base; da chi sviluppa hardware informatico (server, PC, dischi, PDA e telefonini) a chi sviluppa infrastrutture di rete.

Il cambiamento in atto può essere scomposto in quattro filoni evolutivi: quello dello sviluppo applicativo, dei sistemi di elaborazio-

ne, dei sistemi di memorizzazione e naturalmente delle reti, che sono state il fattore trainante di questa evoluzione, in quanto esse rappresentano il punto di congiunzione tra i vari elementi.

Si è parlato in precedenza dell'evoluzione del Web verso un modello collaborativo, denominato Web 2.0, che sta portando oltre il modello *Web Browser-Web Server* e trasformando Internet nella rete globale per la comunicazione tra persone.

È stata avviata di recente, tuttavia, un'altra evoluzione nella comunicazione, "l'Internet degli oggetti", vale a dire l'utilizzo di Internet per consentire il dialogo *machine-to-machine* tra oggetti "intelligenti", dotati cioè di capacità elaborative e di software.

La rete IP si è rivelata efficacissima nel consentire lo *scambio di pacchetti IP tra diversi computer*, ma nel modello Web 1.0 la maggior parte dei computer collegati in rete sono dei PC dotati di *Web Browser* e guidati da un utente. Ormai esistono molteplici oggetti (telefonini, palmari, sensori, attuatori, *smart tag*, elettrodomestici, prodotti di elettronica di consumo ecc.) che sono tecnicamente in grado di comunicare in rete e con i quali si possono realizzare innumerevoli applicazioni.

Sebbene tutti questi oggetti abbiano un livello più o meno elevato di intelligenza a bordo, è impensabile che si trasformino tutti in PC, anche perché il livello di complessità sarebbe troppo elevato per l'utilizzatore. Perciò, in questo scenario, in cui le risorse informatiche sono virtualizzate e un gran numero di oggetti di ogni tipo si collega in rete, non costituisce più una soluzione adeguata disporre di una rete IP "*stupida*" o meglio "*elementare*", in grado cioè di smistare solo semplici pacchetti in base all'indirizzo che essi contengono all'interno. Occorre introdurre una maggiore capacità elaborativa distribuita nella rete, *trasformarla in un'infrastruttura in grado di erogare servizi più evoluti* a livello applicativo.

La rete deve quindi senza dubbio diventare più "intelligente", anche se l'evoluzione architettuale del protocollo IP è ovviamente un tema complesso e articolato, che dovrà necessariamente partire dalle limitazioni emerse sinora, ma anche conciliarle con nuove modalità di utilizzo (esempio, reti P2P) [3].

6. LA "INTELLIGENT INFORMATION NETWORK"

Che cosa significa rendere la rete più intelligente? E quali sono i servizi che possono risiedere nella rete?

Da un punto di vista tecnico si tratta di aumentare la capacità elaborativa e le prestazioni dei dispositivi utilizzati per realizzare la rete, come *router* e *switch*, apparati costituiti da una combinazione di *hardware* e *software*, che devono svolgere funzioni molto sofisticate e con prestazioni molto superiori a quelle di un generico computer.

Naturalmente esistono alcune funzioni che ha senso trasferire nella rete, altre che dovranno necessariamente essere svolte dal livello applicativo: si tratta quindi di una ridistribuzione (Figura 1). La rete è la piattaforma ideale per una classe di servizi generalizzati (vale a dire di uso comune in gran parte delle applicazioni), basati sulla conoscenza del livello fisico (informazioni di topologia e modalità di collegamento) oppure che richiedono l'assistenza di hardware specializzato per ottenere prestazioni adeguate. La rete, infatti, è l'unica componente del sistema ICT che tocca tutte le altre componenti, tutti i servizi che è possibile svolgere nella rete sono, quindi, automaticamente disponibili in modo coerente a tutte le applicazioni.

Come esempio può essere citato il controllo degli accessi e l'autenticazione, la gestione della presenza e della mobilità, la **virtualizzazione delle risorse**, la conversione di formato in tempo reale (per esempio, il "transrating"¹ e il "transcoding"² di flussi multimediali per adattarli ai terminali), la comunicazione unificata. L'intelligenza della rete non sostituisce quella delle applicazioni, ma la affianca, per realizzare un sistema più in-

¹ Il termine *transrating* si riferisce a un processo teso a variare la velocità di cifra (*bit rate*) di un flusso senza eseguire un processo di elaborazione completo e, in particolare, senza cambiamenti del formato.

² Nel caso del *transcoding* un flusso è decodificato e ricodificato a un bit rate inferiore, spesso convertendo il flusso numerico in un nuovo formato. In questo modo (utilizzando particolari algoritmi) è possibile passare da un bitstream a un altro senza decodificare completamente il segnale.

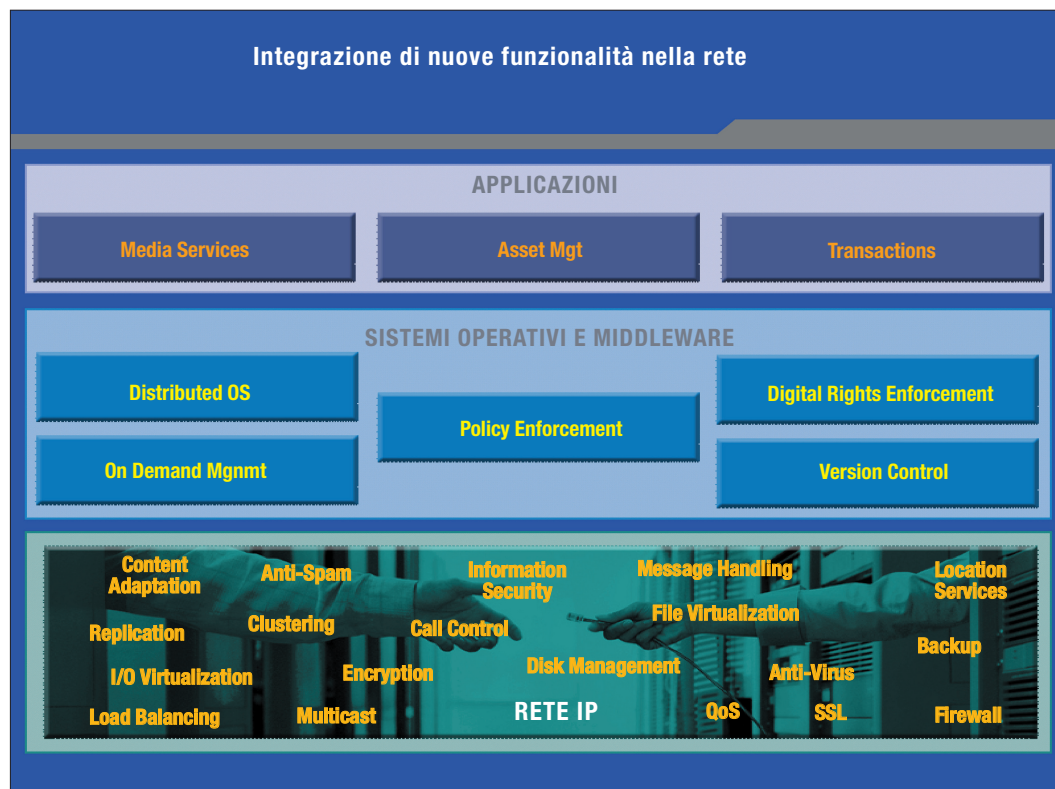


FIGURA 1

In un mondo sempre più interconnesso è in atto un'evoluzione nella distribuzione dei servizi tra l'ambito applicativo, l'ambito del middleware/ sistema operativo e l'ambito della rete. Alcuni servizi tendono a migrare all'interno della rete, che permette una maggiore scalabilità e standardizzazione. Tuttavia nascono nuove esigenze ed altri servizi negli altri due ambiti, che non rischiano certo di svuotarsi

telligente nel suo complesso. Per usare una metafora si possono paragonare le applicazioni al cervello e la rete al sistema nervoso, e cioè al sistema che gestisce autonomamente una serie di funzioni concettualmente semplici, ma vitali, coadiuvando il cervello. Alcuni specifici servizi che una rete intelligente può mettere a disposizione sono:

a. Backup: la scelta di quali particolari dati richiedano un *backup* è di pertinenza delle applicazioni, ma la replicazione dei dati può essere gestita dalla rete, che può anche contestualmente virtualizzare le risorse di *memorizzazione* impiegate; Si ottiene così una notevole semplificazione, perchè si svincola la necessità logica di backup dalla gestione della infrastruttura fisica.

b. Identity Management: consiste nell'*enforcement* a livello di rete, quindi in modo omogeneo e standardizzato, delle *policy* di sicurezza per l'accesso alle risorse. La sicurezza non deve più essere gestita nelle singole ap-

plicazioni o per le varie modalità di accesso (per esempio, per rete fissa o *wireless*). La funzione che autorizza l'accesso alle applicazioni viene virtualizzata in rete, vale a dire implementata in modo distribuito su tutti gli apparati di rete. Le autorizzazioni si definiscono e si gestiscono una volta sola centralmente, eliminando la necessità di configurare separatamente le policy nelle varie applicazioni e negli apparati di sicurezza. Questo semplifica enormemente la gestione, ma accresce anche sensibilmente il livello di sicurezza.

c. Location-Based Services: la rete conosce il punto di accesso e le modalità di collegamento di ciascun sistema collegato. Per le reti cablate ovviamente si arriva ad identificare il cavo cui è fisicamente collegato. Con reti *wireless* (esempio, WiFi) si può non solo conoscere l'area nella quale si trova (cioè la cella), ma anche la sua posizione, ricorrendo a sofisticati algoritmi di triangolazione. Questa possibilità può essere estesa anche ad oggetti o per-

sone senza computer, ricorrendo a sensori *wireless* o *tag* RFID. Il servizio offerto dalla rete consiste nella generazione di eventi (nel senso informatico del termine) sulla base di queste informazioni, per abilitarne un uso più efficace da parte delle applicazioni.

d. Caching: per inviare copie degli stessi dati (esempio, pagine HTML, immagini o video) a più destinatari si può sfruttare la capacità della rete di immagazzinarli in una memoria temporanea e trasmetterli efficientemente, replicandoli solo ove necessario (in base alla topologia), senza doverli ritrasmettere ogni volta; in questo modo si riesce a ridurre drasticamente il tempo di attesa degli utenti remoti, a contenere il carico di lavoro dell'elaboratore centralizzato e ad usare più efficientemente le risorse trasmissive, perché si evita di dovere ritrasmettere i dati ogni volta da un punto centrale.

e. Governance Risk Compliance (GRC): l'inserimento di tecnologia in rete in questo caso rappresenta un ausilio per facilitare la gestione integrata di *governance*, *risk management* e *compliance* per i sistemi informatici. Per esempio la rete può ispezionare in tempo reale i messaggi scambiati dalle applicazioni e generare degli eventi quando si verificano determinate condizioni, come il mancato rispetto di un SLA (*Service Level Agreement*) sui tempi di risposta. Oppure può provvedere alla distribuzione controllata di contenuti confidenziali o protetti, come pure vigilare sulla presenza di minacce e di attacchi realizzati attraverso la rete, anche a livello applicativo.

La rete IP è la piattaforma naturale per un'altra classe di servizi, quelli che sfruttano il fatto che essa è l'unico componente del sistema IT in cui tutto converge in un flusso di dati dal formato standardizzato. La rete quindi non si limiterebbe più a smistare i pacchetti IP, ma grazie alla *deep packet inspection* permetterebbe di ricostruire e comprendere i messaggi a livello applicativo, diventando quindi una piattaforma per lo **scambio di informazioni tra entità applicative**, non più solo un elemento di connettività in grado di trasportare pacchetti IP in base all'indirizzo numerico di destinazione. Smistare in modo più intelligente e sicuro messaggi in base al loro contenuto informativo significa entrare maggiormente a livello applicativo (nel senso OSI del termine)

nei protocolli di comunicazione. Si possono così aumentare i servizi che la rete rende *ubiquamente* disponibili, semplificando il sistema nel suo complesso ed evitando duplicazioni funzionali su ogni terminale. È opinione diffusa, ad esempio, che l'impiego di Personal Computer tradizionali, dotati di un ambiente software ormai molto articolato e complesso, sarà sempre meno conveniente quando l'uso prevalente sarà l'accesso a servizi web. Ancora più improbabile è che questa complessità venga riprodotta su ogni oggetto che si desidera collegare in rete. Molti vorrebbero utilizzare servizi web sul telefono cellulare, ma certamente nessuno vorrebbe trovarsi costretto a dotarlo di un *firewall* e antivirus da tenere aggiornati, come si fa con i Personal Computer. Stiamo quindi andando verso un modello architetturale nel quale il concetto di applicazione da installare sul computer si evolve nel concetto di applicazione disponibile in rete.

L'evoluzione delle architetture distribuite e la diffusione del Web stanno, infatti, comportando un trasferimento massiccio di applicazioni su Internet. Dopo la crescita iniziale, l'obiettivo fondamentale diventa quello dell'integrazione, sia tra nuove architetture e applicazioni pre-esistenti, sia tra soluzioni diverse sviluppate con piattaforme Web incompatibili. In questo scenario, un ruolo fondamentale è svolto dal linguaggio XML e dai *Web Services*, che si propongono come **"collante applicativo universale"**, svolgendo un ruolo simile a quello svolto negli scorsi decenni dalle basi di dati relazionali e dal linguaggio SQL nell'integrazione dei dati aziendali. Si tratta dell'evoluzione denominata **Application Oriented Networking (AON)**, nella quale la rete smista messaggi applicativi di cui comprende il formato. Per esempio, i lettori RFID, in qualità di oggetti collegati in rete, generano un volume enorme di eventi, che possono essere più o meno rilevanti per le varie applicazioni. Invece di trasmettere tutti gli eventi a tutte le applicazioni o peggio inviarli a una sola applicazione che li replichi poi verso altre applicazioni, si può fare in modo che la rete, sfruttando la conoscenza della topologia e replicando il messaggio solo ove necessario, possa inviare un certo tipo di evento solo a quelle applicazioni che si sono registrate per riceverlo. Un altro esempio è costituito

dalla transcodifica: in questo caso se un terminale richiede un contenuto video che è stato memorizzato in un formato non compatibile con le possibilità di visualizzazione in esso previste, la rete potrebbe effettuare automaticamente una conversione, in modo trasparente per l'utilizzatore.

7. CONCLUSIONI

Le reti IP stanno evolvendo verso una maggiore intelligenza, per arrivare alla realizzazione di infrastrutture di rete di nuova generazione (NGN) che vanno intese non come una modalità di rendere più veloce l'Internet che già conosciamo, ma soprattutto come la realizzazione di una piattaforma per servizi innovativi ed in particolare:

□ **collaborazione** delle persone, mediante il trasporto di voce e video in tempo reale, come naturale evoluzione della telefonia;

□ **condivisione di contenuti**, generati dagli utenti o prodotti professionalmente, con possibilità di fruizione on-demand e attraverso numerosi dispositivi, fissi e mobili,

che rappresenta la naturale evoluzione dei sistemi per l'informazione e l'intrattenimento di massa;

□ **connessione in rete di qualunque dispositivo intelligente** che costituirà un punto di svolta in una Società sempre più permeata dalle applicazioni dell'*Information and Communication Technology*.

Bibliografia

- [1] Berners-Lee T., Connolly D.: *Hypertext Markup Language - 2.0*. IETF RFC1866, November 1995.
- [2] O'Reilly Tim: *What Is Web 2.0 - Design Patterns and Business Models for the Next Generation of Software*, 9/30/2005 - <http://oreillyn.net.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-2.0.html>
- [3] Decina M., Giacomazzi P.: Il futuro del protocollo Internet. *Mondo Digitale*, n. 2, Giugno 2007.
- [4] Blake S., Black D., Carlson M., Davies E., Wang Z., Weiss W.: *An Architecture for Differentiated Services*. IETF RFC2475, December 1998.
- [5] Braden R., Clark D., Shenker S.: *Integrated Services in the Internet Architecture: An Overview*. IETF RFC1633, July 1994.

DANILO CISCATO Laureato con lode in Ingegneria Elettronica al Politecnico di Milano nel 1985. Ha trascorso nove anni in IBM, ricoprendo vari ruoli tecnici, commerciali e manageriali, sia negli USA che in Italia. È entrato in Cisco nel luglio 1994, nella fase di startup della Società in Italia, in qualità di consulente tecnologico nello staff europeo. Ha operato a stretto contatto con i maggiori clienti italiani nella realizzazione dei più importanti progetti di rete e ricoperto ruoli manageriali di crescente importanza. Attualmente è Direttore Business Development e Marketing ed è responsabile di tutte le attività relative allo sviluppo del business, con particolare riferimento all'introduzione sul mercato di tecnologie innovative, che hanno da sempre costituito il motore di crescita della Cisco.

E-mail: dciscato@cisco.com

ARCHITETTURE INFORMATICHE L'ORIENTAMENTO AI SERVIZI

Le architetture a servizi costituiscono il paradigma emergente per la progettazione e implementazione di applicazioni di rete. I servizi sono strumenti indipendenti dalla piattaforma, che possono essere descritti, pubblicati e composti ottenendo reti di applicazioni distribuite. Questo lavoro si propone di analizzare sia le sfide tecnologiche poste a chi progetta applicazioni distribuite con il paradigma orientato ai servizi, che il problema della definizione di nuovi modelli computazionali che favoriscono la realizzazione di strumenti software innovativi superando i limiti delle soluzioni tecnologiche oggi disponibili.

1. INTRODUZIONE

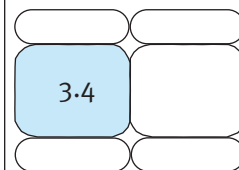
Il crescente sviluppo delle reti di telecomunicazioni e delle applicazioni informatiche di rete ha portato l'industria IT a confrontarsi con nuove possibilità e nuove sfide. Da un lato la grande diffusione della rete ha permesso lo sviluppo di applicazioni distribuite basate su protocolli "aperti" per fornire servizi agli utenti (si pensi ai portali web per avere un esempio significativo), dall'altro è aumentata la complessità delle applicazioni e la necessità di integrare sistemi informativi e piattaforme eterogenee. In particolare, le aziende in un mercato globalizzato sentono sempre più spesso la necessità di rendere accessibili i loro servizi **globalmente**. Questa esigenza comporta la necessità di avere qualcosa in più di un *front-end* di rete (il portale web) visto come l'interfaccia di rete del sistema informatico aziendale. Se un'azienda vuole essere un **attore attivo** del mercato globale, deve necessariamente integrarsi con l'esterno, con le applicazioni di partner, fornitori e clienti. Il problema di come integrare in modo efficace servizi e applicazioni eterogenee non è solo un

problema di natura tecnologica, ma necessita di un approccio interdisciplinare accompagnato da una forte azione di formazione di nuove competenze. Nel contesto di questo articolo non intendiamo affrontare le problematiche legislative, gli aspetti di organizzazione aziendali (modelli di business innovativo, la strutturazione dei processi ecc.), di formazione, messe in essere dalla globalizzazione dell'economia. Il nostro punto di vista sarà prettamente scientifico e tecnologico. In particolare, intendiamo evidenziare i limiti e le potenzialità delle soluzioni informatiche basate sulla nozione di servizio.

Le *Service Oriented Architecture* (SOA) [9, 14] sono state introdotte per affrontare le sfide delineate in precedenza. Il paradigma SOA suggerisce una metodologia di progettazione in cui applicazioni complesse sono realizzate attraverso l'interazione di entità chiamate *servizi*. I servizi sono componenti computazionali autonomi, indipendenti dalla piattaforma, che possono essere descritti, pubblicati, individuati, composti ottenendo reti di applicazioni distribuite in grado di



Massimo Bartoletti
Vincenzo Ciancia
Gian Luigi Ferrari
Roberto Guanciale
Daniele Strollo
Roberto Zunino



operare all'interno di una singola organizzazione e attraversarne i confini. L'applicazione così ottenuta diviene a sua volta un nuovo servizio disponibile sulla rete. È importante notare che:

□ fornitori non vendono più software da installare su sistemi proprietari del cliente, ma forniscono un servizio;

□ l'integrazione di più servizi è trasparente al cliente, che vede la collaborazione di più applicazioni come un unico servizio. Questo permette sia al cliente che al fornitore di poter cambiare i loro rispettivi fornitori, indipendentemente da come questi abbiano sviluppato le loro funzionalità.

Nell'ambito di un'architettura SOA è quindi possibile modificare, in maniera relativamente semplice, le modalità di interazione tra i servizi, oppure la combinazione nella quale i servizi vengono utilizzati all'interno di un processo di business. Allo stesso modo risulta più agevole aggiungere nuovi servizi e modificare le soluzioni software per rispondere alle specifiche esigenze di business. In altri termini, il processo di business specifico di un'azienda non è più vincolato da una particolare piattaforma o da un'applicazione ma può essere considerato come un componente di un processo più ampio e quindi può essere riutilizzato o modificato. Concettualmente, il paradigma SOA può essere descritto come il paradigma computazionale in cui i servizi interagiscono mediante un meccanismo di descrizione autonomo, il contratto del servizio, indipendente dalla piattaforma sottostante e dalle tecnologie di sviluppo. Pertanto, i servizi in esecuzione su una piattaforma possono anche utilizzare servizi in esecuzione su altre facilitando quindi la riusabilità.

I servizi web (*web services*), sono probabilmente la tecnologia maggiormente sviluppata per la progettazione e realizzazione di applicazioni SOA. In letteratura, esistono definizioni differenti di cosa sia un servizio web. La nozione di servizio web definita dal *World Wide Web Consortium* (W3C) evidenzia bene gli aspetti caratterizzanti dei servizi web. "A *Web service is a software system designed to support interoperable machine-to-machine interaction over a network. It has an interface described in a machine-processable format (specifically WSDL). Other systems interact*

with the Web service in a manner prescribed by its description using SOAP-messages, typically conveyed using HTTP with an XML serialization in conjunction with other Web-related standards" (cfr <http://www.w3.org/TR/ws-gloss/>).

La definizione riportata in precedenza evidenzia bene le caratteristiche essenziali di un paradigma computazionale basato sui servizi web. Le operazioni dei servizi (descrizione, pubblicazione, ricerca e l'invocazione) sono basate sullo scambio di messaggi codificati in XML. XML è uno standard di rappresentazione dei dati indipendente dalle diverse piattaforme applicative. Il protocollo SOAP (*Simple Object Access Protocol*) definisce i meccanismi di interazione tra servizi. WSDL (*Web Service Description Language*) è il linguaggio di descrizione delle operazioni del servizio e delle sue modalità di interazione. Infine, i servizi web utilizzano i protocolli standard di internet quali HTTP (*Hyper Text Transfer Protocol*), SMTP (*Simple Mail Transfer Protocol*), e MIME (*Multipurpose Internet Mail Extension*). Una volta definiti e resi disponibili i servizi si avverte la necessità di disporre di un registro dei servizi disponibili e di un protocollo di ricerca dei servizi all'interno della propria rete. Il protocollo UDDI (*Universal Description, Discovery and Integration*) è la risposta standard a questa esigenza. Per una presentazione dettagliata della struttura e delle caratteristiche dei servizi web e per un'introduzione agile e ragionata alle problematiche dei servizi si rimanda alla bibliografia [1, 15].

Uno degli aspetti innovativi delle architetture SOA e dei servizi web in particolare riguarda la composizione e il coordinamento dei servizi. La definizione delle politiche di coordinamento dei servizi avviene attualmente in maniera pragmatica, risolvendo gli aspetti cruciali del coordinamento caso per caso. In questo modo, si garantisce l'interoperabilità della soluzione ma alcune proprietà significative come il rispetto dei vincoli di sicurezza o l'utilizzo efficiente delle risorse non sono necessariamente assicurate. A seconda del modello utilizzato per realizzare le politiche di coordinamento si parla di **orchestrazione** o di **coreografia**. La differenza sostanziale tra queste due modelli di coordinazione consiste nella visione che i servizi hanno l'uno del-

l'altro. Nell'orchestrazione l'attenzione è incentrata su ogni singolo partecipante mentre, nella coreografia, ad ogni partecipante è offerta una visione parziale o addirittura globale del sistema. La babele di proposte di standard per orchestrazione e coreografia di servizi sviluppati negli ultimi anni (WSCI, BPML, WSFL, BPEL4WS, BTP, WS-CDL) testimonia la necessità di linguaggi per il coordinamento dei servizi. Rimanendo sul versante tecnologico, la *Service Component Architecture* (SCA) [8] è molto probabilmente l'esempio maggiormente significativo di modello di programmazione per servizi che pone una particolare enfasi sugli aspetti di definizione e realizzazione delle politiche di coordinamento dei servizi nel progetto di soluzioni business (gli *assembly* dei servizi nella terminologia SCA).

Realizzare politiche di coordinamento dei servizi non è solamente un problema tecnologico. Nella letteratura scientifica sono presenti diverse proposte di natura fondamentale che cercano di risolvere alcuni aspetti essenziali del problema. Altrettanto forte è l'interesse del mondo accademico e della ricerca di base alle problematiche relative alla definizione di linguaggi per il coordinamento dei servizi [13, 14, 20]. Le diverse proposte forniscono strumenti matematici per esprimere e certificare in modo formale proprietà generali sul comportamento di politiche di coordinamento di servizi. Un esempio significativo di politica di coordinamento riguarda le "transazioni di lunga durata" (*long running transactions*). Le transazioni di lunga durata sono un meccanismo fondamentale per aumentare la robustezza delle applicazioni di business realizzate tramite servizi. Le transazioni *long running* perdono la proprietà di serializzabilità e possiedono una nozione meno restrittiva di atomicità garantita da *rollback ad-hoc* chiamati "compensazioni" [11]. Un altro aspetto importante riguarda la definizione di politiche di sicurezza. I meccanismi forniti dagli standard di WS-security non sono sufficienti a garantire la sicurezza di una soluzione business, anche nel caso in cui i protocolli adoperati siano sicuri. La sicurezza di un sistema progettato con un'architettura a servizi può essere violata a causa di comporta-

menti inattesi e indesiderati che si rilevano solamente coordinando i servizi.

La definizione di modelli formali di certificazione di proprietà di politiche di coordinamento per architetture a servizi è essenziale per comprendere e garantire che una specifica soluzione di business soddisfi i requisiti richiesti e, in caso contrario, per comprendere quando e perché si è verificato un problema. Più in generale la capacità di analizzare e certificare proprietà delle politiche di coordinamento dei servizi consente di progettare soluzioni software robuste in grado di prevedere ed evitare i problemi prima che si verifichino. Chi progetta un sistema di norma conosce un servizio solamente tramite le informazioni presenti nella descrizione WSDL del servizio stesso. È difficile comprendere quale servizio ha causato un problema e perché. La definizione di meccanismi di certificazione delle proprietà di coordinamento costituisce, a nostro parere, il valore aggiunto delle soluzioni SOA basate su servizi web.

Questo articolo si pone l'obiettivo di presentare e analizzare alcune delle problematiche tecnologiche e scientifiche poste dalla necessità di definire compiutamente politiche per il coordinamento di servizi. Il primo aspetto che prenderemo in esame in questo nostro percorso riguarda le problematiche relative alla progettazione di servizi con transazioni long-running. La parte finale dell'articolo sarà dedicata alla presentazione di una metodologia che si propone di aiutare il designer nella fase di progettazione degli orchestratori di servizi in modo tale che l'applicazione nella sua interezza rispetti tutti i contratti di sicurezza posti in essere. Il nostro obiettivo è quello di illustrare l'idea (provocatoria) che la sinergia tra lo sviluppo tecnologico e la ricerca di base possa favorire la progettazione e lo sviluppo di soluzioni innovative per le aziende che operano nel mercato globale. In particolare, intendiamo mostrare come l'introduzione di metodologie basate su tecniche e strumenti di certificazione di proprietà favorisca la realizzazione di nuovi strumenti software a valore aggiunto in una prospettiva di *internetworked enterprise* superando i limiti delle soluzioni software oggi disponibili.

2. LE POLITICHE DI COORDINAMENTO DEI SERVIZI

La programmazione di applicazioni SOA richiede di affrontare nuove problematiche [18, 19]. La composizione di servizi e la descrizione del loro flusso di esecuzione non può essere arbitraria. Le politiche che regolano l'esecuzione di servizi sono spesso indicate con il termine generico di politiche di *coordinazione*. A seconda del modello utilizzato per implementare la coordinazione, si parla di *orchestrazione* o di *coreografia*. La differenza sostanziale tra i due modelli consiste nella visione che i servizi hanno l'uno dell'altro. Nell'orchestrazione l'attenzione è incentrata su ogni singolo partecipante mentre, nella coreografia, ad ogni partecipante è offerta una visione parziale o addirittura globale del sistema. Nel seguito verranno meglio evidenziate queste differenze cercando di enfatizzare quelli che sono i punti di forza e gli svantaggi dei due modelli.

Il termine orchestrazione si rifà alla metafora di un'orchestra in cui i singoli musicisti hanno conoscenza della propria attività da svolgere e attendono dal direttore di orchestra (*orchestrator*) le direttive prima di eseguire la prossima attività. Il maggiore svantaggio che questo modello comporta riguarda la centralizzazione del punto di controllo da parte dell'orchestratore. Seppure esistano soluzioni che prevedono la distribuzione del flusso di controllo tra diversi orchestratori resta di fatto che le informazioni di controllo siano gestite solamente da orchestratori. Il vantaggio che ne deriva da questo modello riguarda il totale isolamento dei servizi stessi, i quali non sanno in quale flusso di lavoro sono coinvolti e si limitano a svolgere le proprie attività comportandosi sempre allo stesso modo indipendentemente dal contesto di esecuzione.

Il linguaggio per la specifica di orchestrazione di servizi web più diffuso attualmente, risulta essere il *Business Process Execution Language* (BPEL) [6,12]. BPEL definisce dei costrutti ad alto livello (ad esempio condizionale, cicli, join di flussi paralleli) che vengono utilizzati dall'orchestratore per controllo di flusso. BPEL è un linguaggio eseguibile in quanto, una volta date le specifiche del *workflow* all'engine di orchestrazio-

ne, questo si occupa di eseguire le opportune attività.

Nella metafora della coreografia, ogni partecipante, in funzione delle azioni intraprese dagli altri partecipanti (non necessariamente da tutti) decide quale sia l'operazione da intraprendere. Di solito questo paradigma di coordinamento viene descritto dalla metafora dei ballerini "*Dancers dance following a global scenario without a single point of control*". Il maggior vantaggio di questa strategia risiede nella totale distribuzione delle informazioni di *workflow* ai componenti. Tuttavia, si deve evidenziare il fatto che i servizi coinvolti in una coreografia sono ben consapevoli del contesto in cui si trovano ad agire e, pertanto, risultano meno isolati rispetto a servizi utilizzati nell'orchestrazione. I servizi devono essere progettati preventivamente di essere utilizzati all'interno di coreografia per cui devono esportare alcune funzionalità di supporto per poter gestire lo stato del flusso. Solitamente talune strategie che implementano coreografia, impongono l'utilizzo di servizi con stato (servizi *statefull*) al fine di mantenere consistenti i dati relativi alla gestione del flusso.

Uno standard per la descrizione di coreografie è WS-CDL (*Web Service Choreography Description Language*) [16]. La specifica WS-CDL prevede la stipula un contratto che contiene la definizione globale di vincoli necessari a garantire il normale svolgimento di tutte le attività. Questa visione globale viene successivamente proiettata nei singoli partner i quali devono garantire di assumere un comportamento conforme con quanto loro richiesto.

3. LA CERTIFICAZIONE DEI SERVIZI

I linguaggi attualmente usati per descrivere le operazioni che i servizi web offrono agli utenti sono basati su XML e, pertanto, puramente sintattici, cioè non specificano alcuna proprietà comportamentale associata all'invocazione del servizio. Tipicamente è la documentazione semi-formale del servizio a spiegare quali sono gli effetti che ci si aspetta quando si utilizza una determinata funzionalità.

Il problema di superare i vincoli sintattici della suite dei protocolli dei servizi web è stato affrontato facendo ricorso a metodi differenti.

Una delle soluzioni presenti in letteratura è quella del *Semantic Web* [7]. L'idea di base di questa soluzione consiste nel formalizzare relazioni fra simboli (le *ontologie*) in modo da guidare l'attività di ricerca di un servizio. Se si descrive tramite una ontologia che "regione" è allo stesso tempo un sottoinsieme di "stato" e un sovrainsieme di "città", un cliente che cerca servizi web legati ad aziende della Lombardia può avere come risposta non solo quelli che identificano esattamente, la regione, ma anche altri che coprono quella più ampia dell'Italia, o quella più ristretta di Milano. Sono in fase di sviluppo alcuni standard per specificare ontologie e servizi. Tra queste proposte citiamo WSMML (*Web Service Modeling Language*), OWL (*Ontology Web Language*), WSMO (*Web Service Modeling Ontology*).

Una differente linea di ricerca affronta questo problema mediante l'uso di metodi formali per la certificazione e verifica di proprietà [20]. Quando si parla di verifica di componenti software, ci si riferisce all'uso di un meccanismo che consenta di dimostrare matematicamente l'aderenza del componente a un insieme di requisiti, tipicamente espressi usando un linguaggio formale, derivato dalla logica matematica. Nel contesto di questo lavoro considereremo le tecniche di verifica di *model-checking*. Le tecniche di model-checking permettono di verificare proprietà di un modello, sia esso specificato a tempo di progettazione, oppure estrapolato dal codice stesso del programma dopo la fase di implementazione. Gli strumenti automatici che sono in grado di eseguire il processo di verifica sono chiamati *model-checker*. Se applicato a modelli creati durante la fase di progettazione, il *model-checking* consente di evitare di propagare eventuali errori alle fasi successive dello sviluppo, a differenza di testing che invece opera sul prodotto finito. Inoltre, il model checking ha un ruolo importante anche all'interno di una metodologia *agile*, dato che è possibile integrare il testing con esecuzioni dello strumento di verifica, e segnalare immediatamente allo sviluppatore eventuali problemi non catturati dai casi di test. Una delle caratteristiche che rendono interessante l'uso del model-checking nella progettazione software è che, nel caso in cui un requisito venga violato, il model-checker restituisce un percorso di esecuzione che rappresenta tutti gli stati attraversati dal sistema

fino al fallimento. Questo rende più facile evitare gli errori di progettazione, che influenzano a volte in maniera determinante l'implementazione, e se scoperti troppo tardi possono costringere a costosi *re-factoring*.

Le tecniche di model-checking si applicano in modo naturale a problemi in cui ciò che conta non è quali dati passino nel sistema, ma piuttosto il **flusso di controllo**, o il **flusso dei dati**. Per estrapolare il flusso di controllo o dei dati di un programma o di un progetto software, si possono usare tecniche raffinate come l'astrazione, che consiste in una esecuzione "simbolica" del programma stesso, che genera via via un modello non completo, ma sufficiente a fare una verifica di molte proprietà interessanti. Un esempio significativo è fornito da Java Pathfinder. Java Pathfinder è un ambiente di sviluppo per programmi Java sviluppato dalla NASA in cui sono disponibili un meccanismo di astrazione e un *model-checking* che analizza bytecode Java. Java Pathfinder consente di verificare automaticamente proprietà di assenza di deadlock e trovare eccezioni non gestite, generando una traccia dell'esecuzione del software che porta alla condizione di errore.

Storicamente, il model-checking è stato proposto in seguito ai lavori pionieristici di E. M. Clarke, E. A. Emerson nel 1981, e di J. P. Queille, J. Sifakis nel 1982. Clarke, Emerson e Sifakis sono stati recentemente insigniti del *Turing Award* per il 2008 per i risultati ottenuti nel campo del *model-checking*. Il *Turing Award* è il principale riconoscimento scientifico assegnato a ricercatori per il loro contributo allo sviluppo dell'informatica. Nella direzione del *model-checking* si stanno muovendo anche le grandi imprese fornitrici di software e servizi.

4. LE TRANSAZIONI DI LUNGA DURATA

Un aspetto importante e innovativo delle politiche di coordinamento dei servizi riguarda gli aspetti transazionali. Spesso l'esecuzione di una politica di coordinamento richiede numerose interazioni con l'utente o con processi intermedi che vanno anche al di fuori dell'applicazione. Si pensi per esempio ad un servizio di vendita online che accetta pagamenti tramite servizi che effettuano bonifico

bancario. In questa situazione, il pagamento potrebbe essere notificato alcuni giorni dopo che l'attività di prenotazione dell'acquisto del bene ha iniziato il suo ciclo. Per questi motivi, le tecniche tradizionali dei sistemi distribuiti non sono sufficienti a garantire la correttezza di comportamenti transazionali. Infatti non è possibile riutilizzare le tecniche tradizionale per la gestione delle transazioni (transazioni ACID) dato che i tempi di esecuzione delle operazioni coinvolte sono molto lunghi ed è impensabile attuare meccanismi tradizionali di *locking*, *rollback* e isolamento. Le *Transazioni Long Running*, transazioni di lunga durata, sono lo strumento principale che è stato adottato per rispondere alle richieste di atomicità di servizi. Il meccanismo computazionale adottato per operare agilmente con transazioni di lunga durata è la compensazione (*compensation*). La nozione di compensazione è stata introdotta nei processi SAGAS [11]. Le compensazioni sono quelle attività che permettono di programmare quelle azioni da eseguire per ripristinare lo stato dell'esecuzione parziale di un processo di business. Intuitivamente, le compensazioni possono essere viste come un meccanismo di gestione delle eccezioni programmabile. Si consideri, per esempio, il seguente processo transazionale:

$P := [A_1 \ll B_1]; [A_2 \ll B_2]; [A_3 \ll B_3]$

Il processo transazionale P è costituito dalla composizione sequenziale di tre attività (A1, A2 e A3). Ad ogni attività è associata una compensazione specifica. Per comprendere il comportamento del processo, supponiamo che l'attività A1 completi con successo la pro-

pria esecuzione, mentre l'esecuzione dell'attività A2 generi una situazione di fallimento. In seguito alla segnalazione del fallimento di A2, viene mandato in esecuzione la compensazione B1 (quella associata all'attività A1 terminata con successo) per ripristinare lo stato dell'esecuzione del processo. In altri termini, il fallimento dell'attività ha comportato il fallimento dell'intero processo transazionale. Si noti che la compensazione B2 non deve essere mandata in esecuzione dato che A2 non ha completato la propria esecuzione con successo. Similmente, nel caso in cui A1 e A2 abbiano completato con successo l'esecuzione il processo transazionale termina con successo solamente nel caso in cui A3 abbia successo. In caso di fallimento di A3 saranno mandate in esecuzione le compensazioni B2 e B3. Infine, le compensazioni B3, B2 e B1 saranno eseguite in quest'ordine nel caso in cui P sia un sottoprocesso di un processo transazionale e che P abbia ricevuto una notifica di fallimento dopo aver concluso con successo la propria esecuzione.

Questo semplice esempio bene evidenzia l'idea che le compensazioni sono quelle azioni che devono essere eseguite localmente per ripristinare la consistenza dello stato locale del servizio nel caso di notifica di fallimento. Il diagramma illustrato nella figura 1 descrive la specifica ad alto livello della componente transazionale di un servizio che ha il compito di modificare in parallelo lo stato di un certo numero di oggetti distinti tra loro. Si suppone che gli oggetti siano accessibili dal servizio e che siano memorizzati in un sistema *legacy* che garantisce il *roll-back* perfetto tramite le azioni standard di *Commit* e *Rollback*.

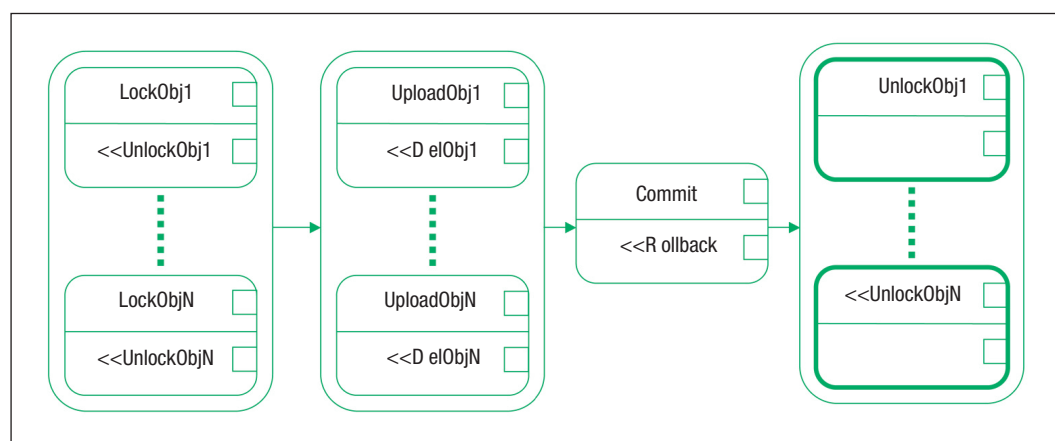


FIGURA 1
Transazioni di lunga
durata in Sagas

La maggior parte dei modelli di *workflow* presenti in letteratura adotta l'idea di compensazione per la corretta gestione delle transazioni di lunga durata. Le diverse proposte differiscono tra loro per la presenza di compensazioni aventi caratteristiche differenti (per esempio compensazioni gerarchiche). Si rimanda alla nota [17] per un'analisi dettagliata delle diverse proposte.

Per comprendere come affrontare il problema della progettazione transazioni *long running*, consideriamo un caso di studio piccolo ma significativo. Consideriamo il caso di un veicolo equipaggiato con un sistema di diagnostica digitale. Il sistema controlla periodicamente lo stato dell'automobile, e quando si presentano situazioni di guasto (surriscaldamento del motore, liquidi sotto livello di soglia ecc.) il sistema attiva automaticamente un servizio di emergenza. Il servizio di emergenza mediante una interazione con l'utente determina e seleziona il servizio necessario per la gestione della situazione di emergenza. Per semplicità, supponiamo che il sistema di emergenza possa selezionare e richiedere un servizio di carro-attrezzi e un servizio di officina per la riparazione del guasto rilevato. Supponiamo, inoltre, che possa venire

richiesto opzionalmente un servizio di macchina a noleggio.

La figura 2 illustra il design del *workflow* del caso di studio descritto in precedenza. La rappresentazione grafica del workflow utilizza la notazione standard denominata BPMN (*Business Process Modeling Notation*). Per maggiori dettagli su BPMN si consiglia di fare riferimento al sito <http://www.bpmn.org/>.

Nel *workflow* di figura 2, la sottotransazione che gestisce l'operazione di noleggio dell'autoveicolo è isolata dal resto del processo. Questo comporta che il fallimento della sua esecuzione non determina automaticamente il fallimento del workflow nella sua interezza.

5. UNA METODOLOGIA PER L'ORCHESTRAZIONE SICURA DI SERVIZI

La sicurezza dei sistemi software è una questione cruciale in un mondo sempre più permeato dall'Information Technology. Anche le applicazioni basate sui servizi web presentano forti requisiti di sicurezza visto che operano nel contesto distribuito e insicuro di Internet. Autorevoli enti, come il CERT della *Carnegie Mellon University*, riportano un numero sempre crescente di segnalazioni relative

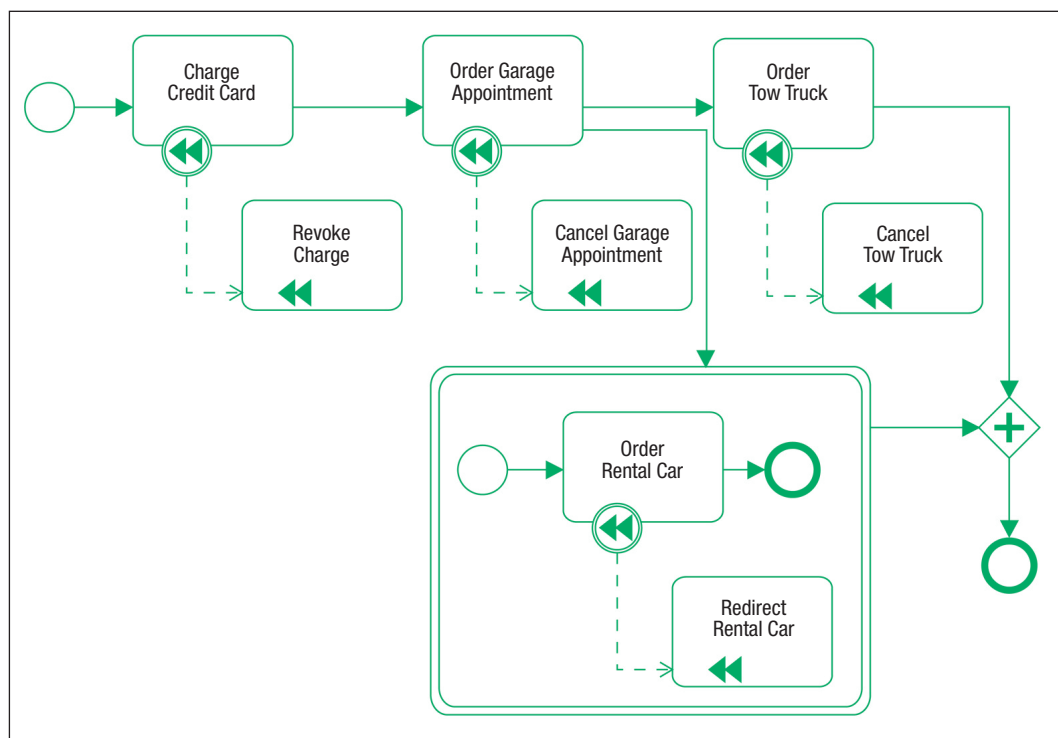


FIGURA 2
Un processo di business con compensazione nella notazione BPMN

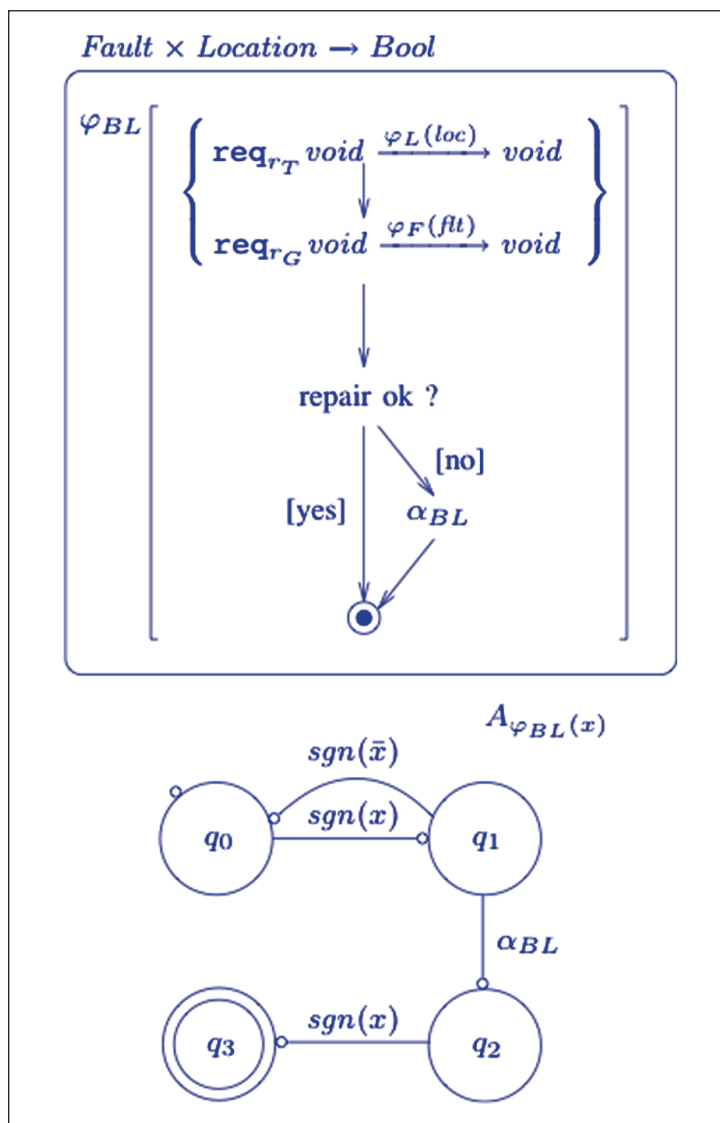


FIGURA 3
Car-emergency
service

alla vulnerabilità dei sistemi informatici, spesso dovute a problemi nella progettazione e realizzazione del software. La soluzione tipica è quella di riparare tali “falle” rendendone più difficile l'utilizzo da parte degli attaccanti, ma questo approccio non va certamente alla radice del problema. Un approccio complementare è, invece, quello di modellare e verificare i requisiti di sicurezza fin dalla specifica di un sistema software, al fine di ridurre al minimo le vulnerabilità sul prodotto finale. L'adozione di tecniche formali può quindi giocare un ruolo molto importante per individuare possibili problemi di sicurezza fin dalle prime fasi di progettazione per capirne a fondo le cause e per rimuoverli prima che altre scelte progettuali costringano a

“inventare” rimedi a posteriori, non sempre possibili e soddisfacenti. I meccanismi forniti dagli standard di WS-security non sono sufficienti a garantire la sicurezza di una soluzione business, anche nel caso che i protocolli adoperati siano sicuri. Il problema principale è come orchestrare servizi, garantendo di non abortire l'esecuzione dell'orchestrazione a causa di azioni che tentano di violare la sicurezza. I servizi che soddisfano localmente la proprietà richiesta non sono sempre buoni candidati, dato che il loro comportamento può altresì invalidare la sicurezza dell'intera orchestrazione.

Per comprendere come affrontare il problema della sicurezza, consideriamo una piccola estensione del caso di studio presentato in precedenza. In particolare assumiamo che la selezione di un servizio possa tenere conto delle preferenze del conducente del veicolo. Per semplicità, supponiamo che il sistema di emergenza possa selezionare e richiedere un servizio di carro-attrezzi e un servizio di officina per la riparazione del guasto rilevato. Il nostro obiettivo è quello di illustrare le principali caratteristiche di una metodologia di supporto progettazione della orchestrazione dei servizi. Per fare questo utilizzeremo una notazione grafica a la UML (diagrammi dell'attività, e diagrammi dei casi di studio).

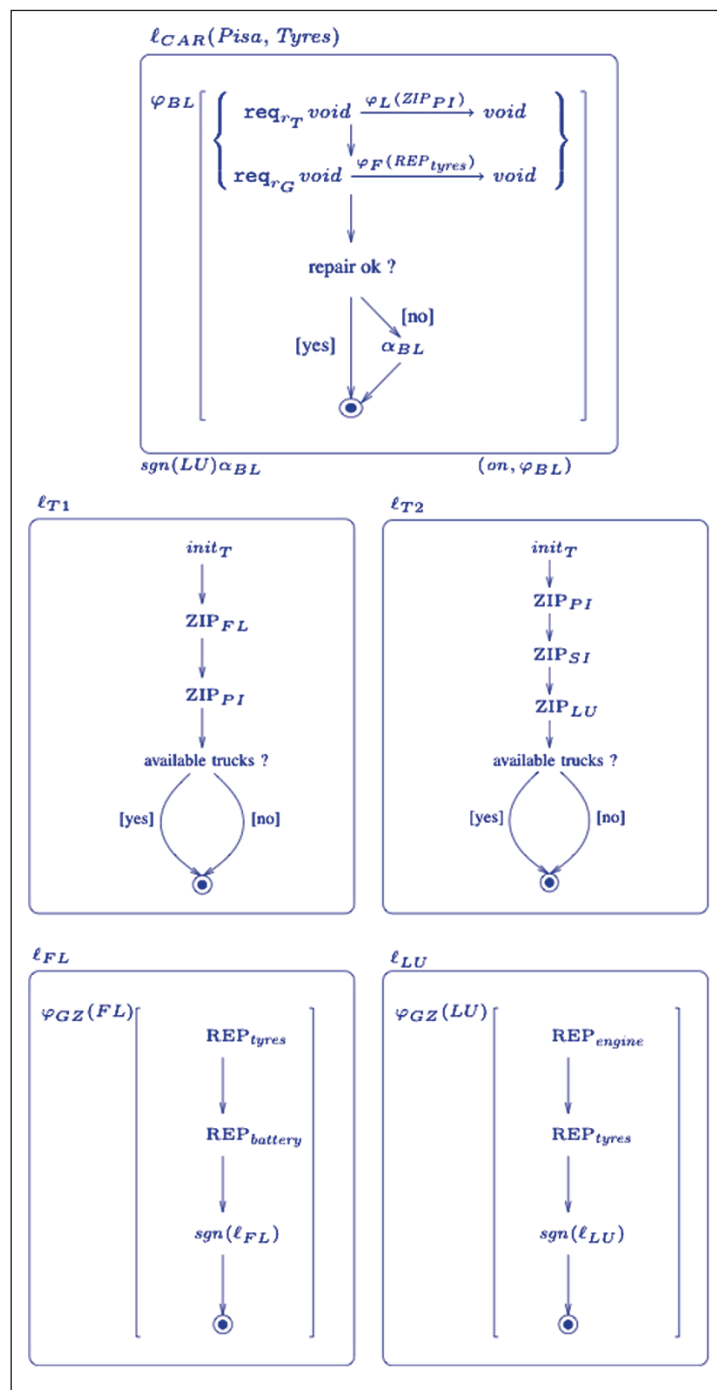
La metodologia per la definizione dell'orchestrazione sicura dei servizi è stata definita in [2, 3, 4] e sfrutta una tecnica di *model-checking* che a partire dai contratto di servizio determina tutte le possibili orchestrazioni che soddisfano i contratti in opera. Tecnicamente, i contratti di servizi sono una estensione dei documenti WSDL e sono formalmente definiti mediante un'opportuno sistema di tipo.

Per introdurre in modo semplice la notazione, presentiamo il servizio di *car-emergency*, ovvero il servizio invocato dal sistema di diagnostica del veicolo per la gestione delle situazioni di emergenza. Il diagramma nella figura 3 descrive la specifica del comportamento del servizio di *car-emergency*. Il servizio è definito mediante una forma particolare di diagramma delle attività in cui sono descritti l'interfaccia del servizio (la parte superiore del diagramma) ed il flusso

del suo comportamento. Un aspetto importante della metodologia è che devono essere definite anche le politiche che vincolano il comportamento del servizio. In particolare, la politica φ_{BL} indica che il servizio memorizza, localmente, le officine che non hanno soddisfatto l'utente. L'idea è che al termine della riparazione il servizio di officina deve firmare digitalmente un certificato di qualità. Se l'utente non è soddisfatto della qualità del servizio offerto inserisce l'officina nella lista nera (black list - BL) delle officine. Le officine inserite nella black-list non saranno più considerate per la gestione di emergenze. La politica di qualità è definita dall'automa a stati finiti nella parte inferiore della figura. Analogamente al servizio di emergenza, tutti i servizi presenti possono specificare la loro politica di qualità. Per esempio, il servizio di officina assume la presenza di una politica di raggiungibilità da parte del carro attrezzi. La politica φ_{GZ} controlla che il carro attrezzi sia effettivamente in grado di raggiungere l'officina.

La parte innovativa del diagramma delle attività riguarda l'introduzione del blocco di orchestrazione, ovvero quella parte della sequenza del flusso di esecuzione compresa tra le parentesi grafe. In questo caso, il blocco permette di specificare la richiesta di orchestrare due chiamate di servizio, una per il carro attrezzi e l'altra per l'officina di riparazione. La caratteristica originale della metodologia è che l'orchestrazione è definita mediante un meccanismo di chiamata dei servizi per contratto. La chiamata per contratto prevede che la selezione dei servizi sia basata sulle proprietà di cui essi godono, piuttosto che sull'identità del produttore del servizio, come accade normalmente. Il blocco di orchestrazione ha il compito di individuare una orchestrazione valida per entrambe l'invocazioni di servizio. Infatti non ha significato alcuno prenotare il carro attrezzi quando l'officina non è disponibile. L'orchestrazione viene individuata mediante l'invocazione di uno strumento di model checking che determina la lista delle orchestrazioni valide.

La figura 4 descrive un diagramma di caso di studio (*Use Case Diagram*) in cui sono presenti un cliente e quattro servizi (due officine e due servizi di carro attrezzi). Ipotizziamo che l'automobile abbia una emergenza ai pneumatici nella zona di Pisa (ZIP_{PI}) e che l'officina in Lucca sia stata inserita nella lista nera (sgn_{LU}) in precedenza. Inoltre, un servizio di carro attrezzi opera nelle provincie di Pisa, e Firenze, mentre l'altro servizio opera nelle provincie di Pisa, Siena e Lucca. Infine, entrambe le officine offrono il servi-



ne e due servizi di carro attrezzi). Ipotizziamo che l'automobile abbia una emergenza ai pneumatici nella zona di Pisa (ZIP_{PI}) e che l'officina in Lucca sia stata inserita nella lista nera (sgn_{LU}) in precedenza. Inoltre, un servizio di carro attrezzi opera nelle provincie di Pisa, e Firenze, mentre l'altro servizio opera nelle provincie di Pisa, Siena e Lucca. Infine, entrambe le officine offrono il servi-

FIGURA 4
Diagramma caso di studio

zio di riparazione e sostituzione dei pneumatici. Si può facilmente notare che l'orchestrazione che invoca il servizio di carro attrezzi T2 e l'officina che opera a Lucca, viola la politica definite dall'utente. L'orchestrazione che invoca il servizio di carro attrezzi T1 e il servizio di officina di Firenze rispetta tutti i contratti posti in opera.

La metodologia descritta in precedenza è supportata da un *middleware* di programmazione denominato JSCL (*Java Signal Core Layer*) [10]. JSCL è equipaggiato con un ambiente di sviluppo grafico (tecnicamente un Plug-in di Eclipse) che permette di generare automaticamente il codice della orchestrazione a partire dalla descrizione del flusso delle attività. Infine, JSCL fornisce un supporto automatico alla progettazione ed implementazione di transazioni long-running [5].

6. CONCLUSIONI

In questo lavoro abbiamo cercato di individuare alcune delle sfide poste a chi progetta applicazioni distribuite con il paradigma orientato ai servizi. Il nostro obiettivo è stato quello di illustrare l'idea che la giusta miscela tra innovazione tecnologica e ricerca fondamentale di base favorisca la realizzazione di strumenti software innovativi superando i limiti delle soluzioni tecnologiche oggi disponibili.

Le problematiche descritte in questo articolo sono state affrontate in progetti di ricerca sia in ambito nazionale che europeo. In Italia è attualmente in corso il progetto di ricerca di base FIRB denominato TOCAI.IT, *Tecnologie Orientate alla Conoscenza per Aggregazioni di Imprese in Internet* (<http://www.dis.uniroma1.it/~tocai/index.php>). In particolare nel contesto di questo progetto sono studiate e sperimentati i servizi di base forniti dal middleware JSCL per la progettazione di comportamenti transazionali per servizi. Nel contesto del 6th – (FP6) è attivo il progetto SENSORIA, *Software Engineering for Service-Oriented Overlay Computers* (<http://www.sensoria-ist.eu/>) che si pone l'obiettivo di definire un insieme di metodologie innovative per la progettazione e sviluppo di architetture software orientate ai servizi. La specifica e l'implementazione di JSCL, la metodologia

per l'orchestrazione sicura di componenti sono stati realizzati all'interno di questo progetto.

Bibliografia

- [1] Alonso G., Casati F., Kuno H., Machiraju V.: *Web Services: Concepts, Architecture and Applications*. Springer Verlag 2004.
- [2] Bartoletti M., Degano P., Ferrari G.: *Types and Effects for Secure Service Orchestration*. Proc. 19th IEEE Computer Security Foundations Workshop (CSFW'06), IEEE Press, 2006.
- [3] Bartoletti M., Degano P., Ferrari G., Zunino R.: *Secure Service Orchestration*. Foundations of Security Analysis and Design IV, FOSAD 2006/2007 Tutorial Lectures, Lecture Notes in Computer Science 4677, Springer 2007.
- [4] Bartoletti M., Degano P., Ferrari G., Zunino R.: *Semantics-based design for Secure Web Services*. *IEEE Transactions on Software Engineering*, TSE, Vol. 34, n. 1, 2008.
- [5] Bruni R., Ferrari G., Melgratti H., Montanari U., Strollo D., Tuosto E.: *From Theory to Practice in Transactional Composition of Web Services*. Formal Techniques for Computer Systems and Business Processes, Lecture Notes in Computer Science 3670, Springer 2005.
- [6] *Business Process Execution Language (BPEL)*. Disponibile on line all'indirizzo <http://www-128.ibm.com/developerworks/library/specification/ws-bpel/>
- [7] Cardoso J., Sheth A.: *Amit. Semantic Web Services, Processes and Applications*. Springer 2006.
- [8] Chappel D.: *Introducing SCA*. Disponibile on line all'indirizzo (http://www.davidchappell.com/articles/Introducing_SCA.pdf), July 2007.
- [9] Erl T.: *SOA Principles of Service Design*. The Prentice Hall Service-Oriented Computing Series, Prentice Hall 2007
- [10] Ferrari G., Guanciale R., Strollo D.: *JSCL: A Middleware for Service Coordination*. 26th IFIP WG 6.1, Formal Techniques for Networked and Distributed Systems – FORTE 2006, Lecture Notes in Computer Science 4229, Springer 2007.
- [11] Garcia-Molina H., Salem K.: *Sagas*. International Conference on Management of Data, ACM SIGMOD, ACM Press, 1987.
- [12] Juric M., Mathew B., Sarang P.: *Business Process Execution Language for Web Services*. PACKT Publishing, 2006.
- [13] Kitchin J., Cook W.R., Misra J.: *A Language for Task Orchestration and Its Semantic Properties*. CONCUR 2006, Lecture Notes in Computer Science, 4137 Springer 2006.

- [14] Papazoglou M., Georgakopoulos D.: Special issue on service oriented computing. *Communications of the ACM*, Vol. 46, n. 10, 2003.
- [15] Pernici B., Plebani P.: Introduzione ragionata al mondo dei Web Service. *Mondo Digitale*, marzo 2004, AICA.
- [16] Ross-Talbot S., Fletcher T.: *Web Services Choreography Description Language*. Primer, W3C Working Draft, 2006 (<http://www.w3.org/TR/2006/WD-ws-cdl-10-primer-20060619/>)
- [17] Sheth A., Worah D.: *Transactions in Transactional Workflows*. Advanced Transaction Models and Architectures, Kluwer, 1997.
- [18] Stal M.: Web services: Beyond component-based computing. *Communications of the ACM*, Vol. 55, n. 10, 2002.
- [19] Vogels W.: Web services are not distributed objects. *IEEE Internet Computing*, Vol. 7, n. 6, 2003.
- [20] Wirsing M., et al: *Semantic-based development of service-oriented systems*. 26th IFIP WG 6.1, Formal Techniques for Networked and Distributed Systems – FORTE 2006, Lecture Notes in Computer Science 4229, Springer 2007.

MASSIMO BARTOLETTI ha ottenuto il Dottorato di Ricerca in Informatica, Scuola Galilei, Università di Pisa. Titolare di un assegno di ricerca presso il Dipartimento di Informatica Università di Pisa. I suoi interessi riguardano le problematiche relative alla specifica e verifica di proprietà di sicurezza in sistemi distribuiti.
E-mail: bartolet@di.unipi.it

VINCENZA CIANCIA sta terminando il Dottorato di Ricerca in Informatica, Scuola Galilei, Università di Pisa. I suoi interessi riguardano l'utilizzo di metodi formali nella certificazione di programmi.
E-mail: ciancia@di.unipi.it

GIAN LUIGI FERRARI professore associato presso il Dipartimento di Informatica Università di Pisa. I suoi interessi più recenti riguardano i metodi e gli strumenti per la progettazione e la realizzazione di architetture software orientate ai servizi. È autore di molte pubblicazioni internazionali, tra cui oltre cento articoli su riviste e atti di convegni.
E-mail: g.ferrari@di.unipi.it

ROBERTO GUANCIALE sta terminando il Dottorato di Ricerca in Scienze e Tecnologie dell'Informatica, Institutes for Advances Studies, IMT, Lucca. I suoi interessi riguardano il progetto e la realizzazione di architetture software orientate ai servizi.
E-mail: roberto.guanciale@imtlucca.it.

DANIELE STROLLO sta terminando il Dottorato di Ricerca in Scienze e Tecnologie dell'Informatica, Institutes for Advances Studies, IMT, Lucca. I suoi interessi riguardano il progetto e la realizzazione di architetture software orientate ai servizi.
E-mail: daniele.strollo@imtlucca.it.

ROBERTO ZUNINO Ricercatore Universitario presso il Dipartimento di Informatica e Telecomunicazioni, Università di Trento. I suoi interessi riguardano le problematiche relative alla specifica e verifica di proprietà di sicurezza in sistemi distribuiti.
E-mail: zunino@di.unipi.it



ICT E INNOVAZIONE D'IMPRESA

Casi di successo

Rubrica a cura di

Roberto Bellini, Chiara Francalanci

La rubrica *ICT e Innovazione d'Impresa* vuole promuovere la diffusione di una maggiore sensibilità sul contributo che le tecnologie ICT possono fornire a livello di innovazione di prodotto, di innovazione di processo e di innovazione di management. La rubrica è dedicata all'analisi e all'approfondimento sistematico di singoli casi in cui l'innovazione ICT ha avuto un ruolo critico rispetto al successo nel business, se si tratta di un'impresa, o al miglioramento radicale del livello di servizio e di diffusione di servizi, se si tratta di una organizzazione pubblica.

Caso Y-Tech

Roberto Bellini

1. INTRODUZIONE

Y-TECH srl nasce nel dicembre 2005 a seguito di un *management buy-out* da parte di un senior team che opera sul mercato dall'anno 2000 come divisione Web&CRM di aziende del settore marketing e tecnologico.

Dopo due anni spesi a consolidare il modello di business con primarie aziende note per la capacità di attuare soluzioni innovative di marketing e, contestualmente, a completare il consolida-

mento finanziario della neo impresa, nata da una situazione di crisi del gruppo in cui operava come divisione, Y-Tech è oggi pronta a sviluppare con maggiore vigore la propria strategia.

2. IL MERCATO DI RIFERIMENTO IN CUI OPERA Y-TECH

La maggior parte dei progetti e dei servizi è acquistato da imprese private e da pubbliche amministrazioni di grandi dimensioni che debbono implementare strategie di promozione e *retention* nei confronti dei propri consumatori, ieri raggiungibili per via telefonica e oggi via web. Y-Tech opera quindi secondo uno schema B2B2C: i suoi servizi sono acquistati dall'area Marketing&Vendita per supportare azioni di marketing diretto e commerciale nei confronti degli utenti di servizi pubblici dell'organizzazione cliente, come indicato nello schema rappresentato nella figura 1.

Y-TECH si posiziona quindi come fornitore della funzione di marketing, ma accompagna la sua proposta con la competenza tecnologica dei suoi tecnici di talento, capaci di realizzare con successo progetti sul "*bleeding-edge*" tecnologico per aziende leader in Italia.

I clienti di Y-TECH sono imprese ed enti della Pubblica Amministrazione a cui viene riconosciuta una sofisticata capacità di marketing co-

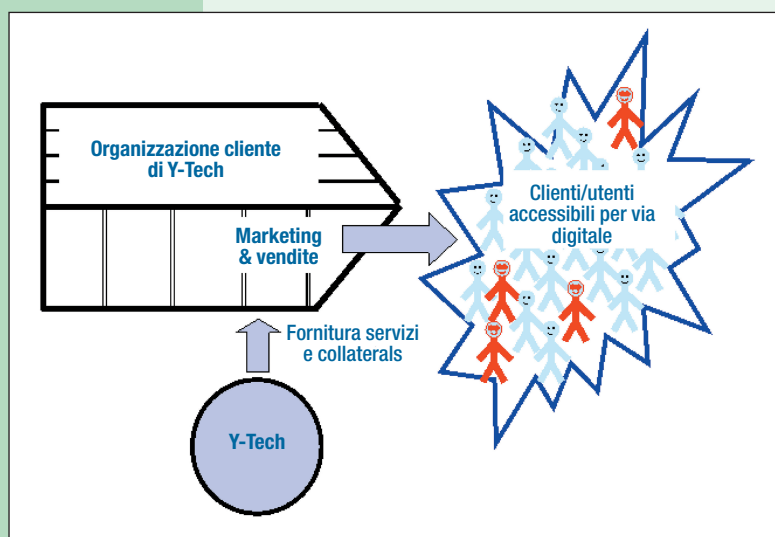


FIGURA 1

Schema operativo B2B2C impiegato da Y-tech

me per esempio: il Gruppo COIN, CNP Capitalia Vita (ex FinecoVita / FinecoLife), Gruppo Editoriale l'Espresso, A.Manzoni&C., RCS Media-Group, A.C. Milan, Cerved, Tiscali, Wind, Gruppo Ventaglio, Regione Campania ecc..

Data la natura dei progetti e servizi, il processo di "fidelizzazione" è strutturalmente collegato alla capacità di portare a buon fine i progetti e di erogare servizi di qualità costante nel tempo.

Dal punto di vista della proposta commerciale, i servizi offerti da Y-Tech si collocano nella tipologia delle prestazioni personalizzate e integrate nella "macchina operativa" del cliente; questo determina la continuità di relazione e la nascita di forti sinergie operative con il cliente stesso. È quindi naturale che la scelta commerciale sia strutturalmente orientata al *cross-selling* e alla fidelizzazione (estensione della durata dei servizi), anche in forza di un atteggiamento di "*partnership collaborativa*" che sempre più viene richiesto dagli interlocutori.

Il processo di vendita è strettamente correlato alla qualità e all'elevata specializzazione del know-how e dei servizi offerti dal senior team, che è l'elemento chiave nel processo di mantenimento ed estensione sul mercato.

La limitazione dello sforzo di acquisizione pura è sostanzialmente collegata alla scelta di mantenere il focus sulla realizzazione di progetti e sull'automazione di processi altamente personalizzati e non "*pacchettizzabili*", evitando la competizione sul mercato dei software

tools (cms, crm packages, hosting ecc.), sui quali non si valutano margini economici e competitivi sufficienti, e delle commesse gestite dalle grandi *consultancy agencies*, verso le quali si compete.

Un'ulteriore linea di sviluppo è basata sul networking di partnership con aziende fornitrici di servizi collaterali di alta specializzazione, come la *customer satisfaction* e la gestione di comunità sociali su piattaforme web 2.0.

3. L'OFFERTA

Y-TECH offre le proprie competenze ed esperienze alle aziende che necessitano di sviluppare programmi e progetti WEB & CRM nell'ambito delle strategie di Marketing Diretto; il posizionamento è basato sulla competenza in Marketing e Tecnologia e su un modello d'offerta che privilegia la proposta di servizi "*chiavi-in-mano*", nei quali il cliente (*marketing & communication manager*) può trovare la soluzione completa alle proprie esigenze operative.

Come illustrato nella figura 2, le tecnologie di CRM, di *Campaign Management* e *Web & Multimedia* vengono proposte ai responsabili marketing delle organizzazioni private e pubbliche che operano nei confronti di grandi masse di consumatori e cittadini a cui spesso i prodotti servizi dell'organizzazione cliente arrivano per canali indiretti; le tecnologie vengono utilizzate per supportare la Strategia di Marketing, la conoscenza approfondita del

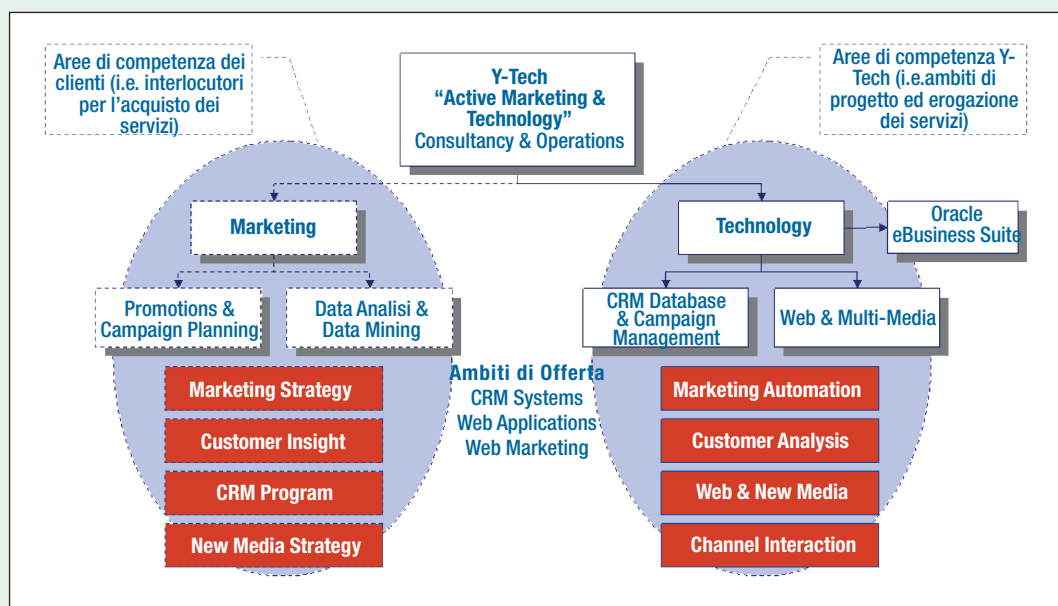


FIGURA 2
Iter di vendita

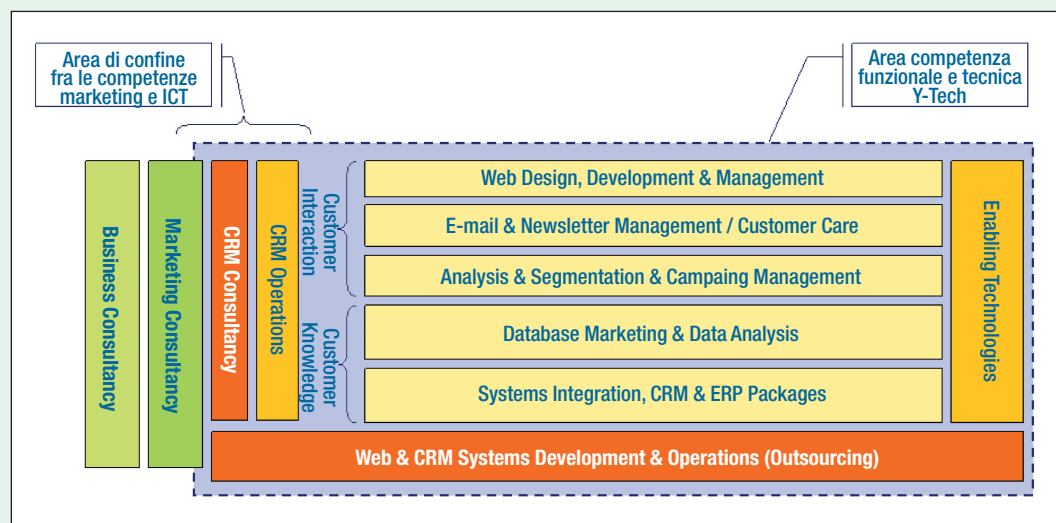


FIGURA 3

cliente finale (customer insight), il programma di gestione delle relazioni e la strategia sui nuovi media e canali di comunicazione.

Y-TECH si propone, nei confronti dei propri clienti, come un partner in grado di affiancarsi al committente nella progettazione e implementazione di programmi di marketing multi-canale; alla capacità progettuale in ambito sia marketing sia tecnologico, viene affiancata quella di implementazione e gestione continuativa dei programmi operativi (*full outsourcing*). L'offerta è arricchita da una competenza fortemente specializzata su moduli verticali della Oracle eBusiness Suite.

In particolare, la capacità di Y-Tech è quella di saper impostare, realizzare e gestire servizi a base digitale con un elevato livello di ottimizzazione tra le funzioni di marketing e il sistema tecnologico.

Il Direct Marketing e le sue naturali evoluzioni verso il CRM e il Web Marketing sono fondati su due pilastri della tecnologia: il database e la capacità di collegare i dati ai canali di relazione col cliente finale in processi che coinvolgono grandi numeri (target "consumer").

Y-TECH assume il ruolo di partner "facilitatore" nella messa in opera e gestione di sistemi di marketing evoluto, nei quali la componente tecnologica è costituita dall'*hardware* e dal *software* applicativo (o il *middleware* dove possibile) e le metodologie di marketing costituiscono, insieme ai dati, il content del sistema, che insieme generano il valore aggiunto nei processi di marketing del cliente.

Lo schema di figura 3 illustra come le tecnolo-

gie abilitano, secondo Y-Tech, sia le operazioni che l'attività di supporto nella gestione della relazione con il cliente.

4. IL MODELLO DI BUSINESS

Sulla base delle esperienze maturate, dal punto di vista "strategico" il modello di offerta si è consolidato secondo uno schema che prevede "il tentativo di" trasformare ogni opportunità di progetto (sviluppo *one-shot* di un sistema tecnologico) in un servizio da erogare in continuità (*outsourcing operativo*).

Questo schema si basa su due assunti principali:

□ il primo è la natura del mercato di riferimento, costituito da marketing & communication managers di medie e grandi aziende che intendono investire in processi di marketing operativo, i quali necessitano, per loro stessa natura, di solide infrastrutture tecnologiche;

□ il secondo è la capacità di proporre, a valle delle soluzioni "primaria" al problema (la realizzazione del sistema) anche la soluzione al problema "sottostante", ovvero la messa in opera e la continuità di erogazione del corrispondente servizio (per esempio, un sistema di database marketing).

Il modello, se correttamente attuato, permette di transitare in continuità da un bilancio "project oriented", con costi di produzione proporzionali allo sforzo richiesto, a un bilancio "service oriented", nel quale i costi di gestione possono essere ammortizzati in economie di scala, anche con progetti di dimensioni limitate.

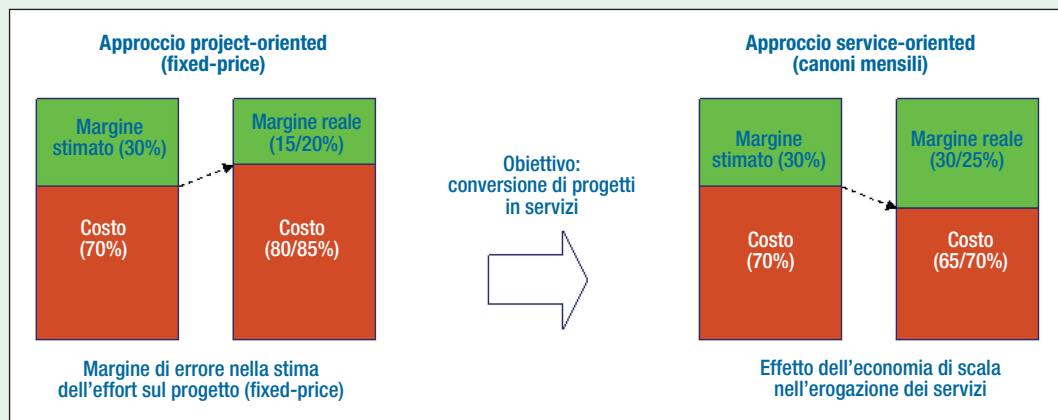


FIGURA 4
L'approccio project-oriented e quello service-oriented

Questo “flusso” alimenta la *pipeline* dei servizi, che costituiscono la base del sistema di business di Y-Tech, a partire dalla *pipeline* dei progetti, che a sua volta viene alimentata dall'evoluzione o estensione dei servizi erogati (*up-selling* e *cross-selling* sui servizi).

La figura 4 illustra il modello di business centrato sulla capacità di acquisizione e sviluppo di progetti personalizzati (approccio *project oriented*, in cui il margine di progetto tende a slittare sempre da una stima a priori del 30% ad una consuntivazione a posteriori del 15-20%) e successivamente sulla trasformazione in commessa di servizio a base digitale (approccio *service oriented*, in cui il margine del servizio tende sempre ad incrementare da una stima a priori del 30% a una consuntivazione a posteriori del 30-35%).

Il modello di offerta può evolvere secondo diverse direzioni, ma con alcuni punti di riferimento che nella esperienza Y-Tech sono particolarmente rilevanti.

□ Primo elemento: il mantenimento e rafforzamento di una catena del valore capace di porre il senior management a contatto col cliente finale, nella possibilità di fornire “soluzioni” complete (anche complesse) grazie ad una catena del valore che giunge a risolvere i problemi **tecnologici** collegati alla realizzazione dei sistemi necessari per la messa in opera del servizio.

□ Secondo elemento: la capacità di cogliere l'esigenza (l'opportunità) di servizio nel rapporto con l'interlocutore finale, non tecnico, del quale si deve comprendere il **content** che gli interessa, su cui stimolare la sua attenzione e l'apprezzamento sul valore generato.

□ Terzo elemento: altrettanto importante risul-

ta essere l'approccio proattivo alla partnership collaborativa col cliente, presupposto “imprenditoriale” chiave per la costruzione di un programma evolutivo di sempre maggiore fiducia reciproca e coinvolgimento nella macchina operativa dell'organizzazione cliente.

5. LA PIATTAFORMA TECNOLOGICA

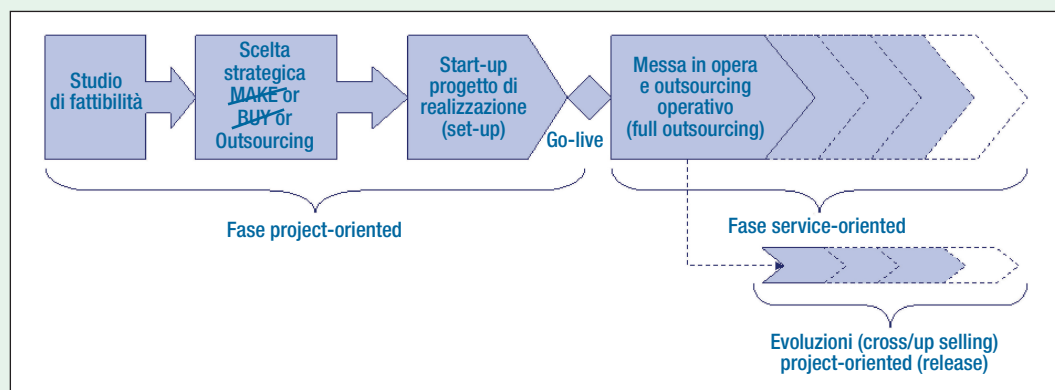
Attraverso consolidate partnership operative Y-Tech, è in grado di offrire servizi di outsourcing tecnologico di alto livello, disponendo di locali e infrastrutture tecnologiche dotati di tutte le caratteristiche di sicurezza, prestazioni e continuità di servizio per garantire la qualità di servizio richiesta dalle grandi aziende. Y-Tech si appoggia, per offrire prestazioni di livello qualitativo adeguato, a primarie società ICT da cui acquista a sua volta in outsourcing, capacità di elaborazione e di erogazione dei servizi digitali progettati e installati per conto del cliente: i rapporti consolidati riguardano in particolare IT Net (gruppo Wind Telecomunicazioni) e Atos Origin.

6. STRUTTURA OPERATIVA E RISULTATO ECONOMICO

La struttura operativa si articola in due componenti fondamentali: da una parte la struttura diretta interna composta da 9 professional che si appoggiano ad un network di partnership operative con piccole società specializzate e alcuni free lance senior consultants, dall'altra la struttura tecnologica esterna, costituita da un paio di provider di servizi a base digitale, in grado di soddisfare tutte le esigenze di continuità, sicurezza e gestione degli im-

FIGURA 5

Esempio - Start-up progetto di messa in opera di un sistema di CRM analitico



pegni contrattuali assunti da Y-Tech nei confronti dei propri clienti; la qualità e stabilità del network è fattore chiave nel determinare la qualità dei servizi e dei progetti.

A tutti gli effetti Y-Tech si colloca come un **service integrator** che opera come outsourcer di 1° livello nei confronti dei propri clienti, e a sua volta si avvale di outsourcer di 2° livello, sofisticati sul piano della gestione della tecnologia e specializzati sul piano delle applicazioni più avanzate.

Dal punto di vista del risultato economico, su un fatturato di circa 1,3 milioni di euro nel 2007, la struttura operativa assorbe il 65% del valore generato dalla clientela fra costi diretti e costi generali, e utilizza il 22% circa del fatturato per pagare il 2° livello di outsourcing, costituito dalle infrastrutture tecnologiche di Web e CRM, generando un margine lordo di circa il 13% (Figura 5).

6. CONCLUSIONI

Y-Tech è un'azienda di consulenza e servizi specializzati per il Marketing diretto e la gestione dei clienti finali di aziende di grandi dimensioni, per le quali opera come *outsourcer* capace di offrire soluzioni complete, dal progetto personalizzato all'erogazione di servizi a base digitale che si arricchiscono nel tempo.

Ha adottato un modello di business che trasforma i progetti di marketing sviluppati per i suoi clienti in servizi da erogare su base continuativa, con cui alimentare una propria *pipe line* comune per l'insieme dei servizi offerti a diversi clienti. Con l'approccio indicato ottiene i seguenti risultati:

□ il cliente committente diventa molto fidelizzato;

□ ogni nuovo progetto migliora l'ottimizzazione dei costi di gestione e l'esperienza cumulata a favore di ogni singolo committente;

□ nel caso in cui il committente ritenga di non avere più interesse al servizio, Y-Tech può valutare la convenienza a rilevare il business in proprio, con la possibilità di farsi anche finanziare il *take over* da parte del committente, vincolato a scegliere vie di uscita graduali e morbide per salvaguardare il rapporto con la propria clientela finale.

Il caso fornisce infine tre indicazioni interessanti sulla importanza e il ruolo dei servizi in outsourcing, che riassumiamo di seguito:

1. il modello di business da sviluppare è molto importante: si parte da progetti personalizzati e di alta complessità e successivamente, una volta completato il progetto, si passa ad una gestione della erogazione dei servizi a base digitale, con miglioramenti significativi del margine;

2. quanto più l'outsourcing va a toccare i processi primari della macchina operativa del cliente, come in questo caso il processo di Marketing e di Comunicazione, tanto più è necessario sviluppare competenze ibride, non solo tecnologiche ma anche specifiche sul settore di business del cliente e sulla funzione di marketing, tali da essere riconosciute credibili ed efficaci da parte dei clienti di marketing;

3. i servizi in outsourcing sofisticati si erogano sia nei paesi sviluppati che nei paesi emergenti; nei paesi emergenti in particolare può essere opportuno acquisire in outsourcing servizi poveri, mentre è sicuramente vero che servizi complessi siano acquistati in outsourcing solo là dove esiste una profonda e riconosciuta competenza di business, oltre che tecnologica.

INTERVISTA

La storia di Marco Banfi, managing partner di Y-Tech

La storia professionale di Marco Banfi, managing partner di Y-Tech, costituisce un interessante paradigma di trasformazione da professional a imprenditore.

Subito dopo il conseguimento della laurea in Scienze dell'Informazione presso la Statale di Milano, Banfi inizia a lavorare nel CED Borsa, oggi SIA, per occuparsi di sviluppo nell'ambito mainframe e database, ai tempi notoriamente considerati molto complessi. Imbocca decisamente la strada dell'esperienza di marketing, ampliando e consolidando la sua professionalità negli anni successivi attraverso due tappe:

□ in Ogilvy One Consulting per un paio di anni a fine degli anni '90, si focalizza sugli aspetti tecnici del business basato sulla gestione in outsourcing del Marketing Data Base per conto terzi; in questa azienda sviluppa i primi esempi di applicazione dei principi del Marketing Relazionale in Italia;

□ nei successivi cinque anni comincia la sua esperienza imprenditoriale, partecipando a due iniziative:

- nella prima, contribuisce alla costituzione di una nuova società di consulenza, Yoda SpA, in cui va a ricoprire il ruolo di responsabile tecnico;
- nella seconda, contribuisce con i partner di Yoda alla fusione con un'altra azienda di consulenza; nel giro di poco tempo emerge che il matrimonio non sta in piedi, ma il team tecnico ex Yoda mantiene comunque dritta la barra continuando a servire i clienti.

A fine 2005, il senior team fa un management buy-out per costituire Y-Tech, di cui Banfi diventa managing partner.

L'esperienza fatta attraverso le ultime due iniziative imprenditoriali, permettono al senior team Y-Tech, di completare e arricchire il modello di business già sperimentato: fornire in outsourcing servizi tecnologici a supporto del Marketing Operativo di grandi organizzazioni con centinaia di migliaia/milioni di clienti-utenti, e in particolare:

□ la gestione del *back-end* applicativo di grandi Customer Data Base a supporto del *Customer Relationship Management* di grandi masse di utenti e profilazioni;

□ l'integrazione di queste gestioni attraverso la costruzione di comunità sul web per clienti come Wind, RCS Web, Atlantec ecc.;

□ la gestione del *Front-End*, per la relazione diretta on line con il cliente consumer, stimolato attraverso giochi, programmi di *loyalty*, servizi di vario interesse aziendale.

Y-Tech è accreditata oggi di una significativa capacità concreta di costruzione, lancio e gestione di iniziative di Web Marketing e di Social Networking specializzati nelle varie tipologie di funzioni di marketing, come per esempio la gestione delle relazioni e la *customer satisfaction*.

ROBERTO BELLINI è docente di Marketing e Gestione della Relazione con il cliente nell'ambito del MIP - Politecnico di Milano, con una focalizzazione sulla innovazione nelle reti di imprese. Presiede la Sezione AICA di Milano ed è responsabile per Aica del Cantiere dei Mestieri ICT.
E-mail: roberto.bellini@polimi.it

CHIARA FRANCALANCI è professore associato di Sistemi Informativi al Politecnico di Milano. Ha scritto numerosi articoli sulla progettazione e sul valore economico delle tecnologie informatiche, svolto attività di ricerca e consulenza nel settore finanziario e manifatturiero sia in Italia sia presso la Harvard Business School ed è editor del Journal of Information Technology.
E-mail: francala@elet.polimi.it



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.

Diritto all'informazione e nuove tecnologie: dalle testate on-line ai blog

Antonio Piva, David D'Agostini

1. INTRODUZIONE

Tra i diritti fondamentali riconosciuti nei paesi democratici, il diritto di manifestare il proprio pensiero è quello che meglio simboleggia lo Stato liberale, rappresentando una delle conquiste più significative del secolo scorso.

Già nella dichiarazione dei diritti dell'uomo e del cittadino nella Francia post rivoluzionaria del 1789 si affermava che la libera manifestazione delle opinioni è uno dei diritti più preziosi dell'uomo e, di conseguenza, che ogni cittadino può parlare, scrivere, stampare liberamente, salvo a rispondere dell'abuso di questa libertà; il celebre primo emendamento della costituzione statunitense impedisce al Congresso di limitare la libertà di parola o di stampa; mentre nel 1948 l'Assemblea generale delle Nazioni Unite nella dichiarazione universale dei diritti dell'uomo stabiliva che ogni individuo avesse diritto alla libertà di opinione e di espressione incluso il diritto di cercare, ricevere e diffondere informazioni e idee attraverso ogni mezzo e senza riguardo a frontiere.

Nello stesso anno entrava in vigore la costituzione italiana la quale all'art. 21 riconosce a tutti il diritto di manifestare liberamente il proprio pensiero con la parola, lo scritto e ogni altro mezzo di diffusione, aggiungendo che la stampa non può essere soggetta ad autorizzazioni o censure e che il sequestro è ammesso soltanto per atto motivato dell'autorità giudiziaria nel caso di delitti, per i quali la legge sulla stampa espressamente lo autorizzi.

In buona sostanza viene sancito con la massima ampiezza, salvo il limite generale del buon costume, il diritto alla libera manifestazione del proprio pensiero con ogni mezzo di diffusione, quale principio fondamentale dell'ordinamento democratico; inoltre la carta costituzionale fornisce un'ampia e dettagliata garanzia alla libertà di stampa, rivolta a limitare le interferenze dei pubblici poteri e a scongiurare il rischio di un ritorno a forme illiberali di controllo preventivo sulle pubblicazioni.

Nei decenni successivi la Corte di Cassazione ha più volte precisato che il diritto di stampa (cioè la libertà di diffondere il proprio pensiero attraverso la stampa) è legittimo quando concorrano le seguenti tre condizioni:

1. utilità sociale dell'informazione;
2. verità (anche soltanto putativa) dei fatti esposti;
3. forma "civile" della esposizione dei fatti e della loro valutazione: cioè non eccedente rispetto allo scopo informativo da conseguire, improntata a serena obiettività almeno nel senso di escludere il preconcetto intento denigratorio e, comunque, in ogni caso rispettosa di quel minimo di dignità cui ha sempre diritto anche la più riprovevole delle persone, sì da non essere mai consentita l'offesa triviale o irridente i più umani sentimenti.

La verità dei fatti, cui il giornalista ha il preciso dovere di attenersi, non è rispettata quando, pur essendo veri i singoli fatti riferiti, siano, dolosamente o anche soltanto colposamente, taciuti altri fatti, tanto strettamente ricollegabili



ai primi da mutarne completamente il significato; pertanto anche la verità incompleta viene equiparata alla notizia falsa.

La forma della critica non è civile, non soltanto quando è eccedente rispetto allo scopo informativo da conseguire o difetta di serenità e di obiettività o, comunque, calpesta quel minimo di dignità cui ogni persona ha sempre diritto, ma anche quando non è improntata a leale chiarezza.

2. DALLA STAMPA AI GIORNALI ON LINE

Per garantire i diritti sopra elencati, venne approvata la legge 8 febbraio 1948 n. 47 (detta legge sulla stampa), che comprendeva tutte le riproduzioni tipografiche o comunque ottenute con mezzi meccanici o fisico-chimici, in qualsiasi modo destinate alla pubblicazione.

Quotidiani, periodici e agenzie di stampa hanno l'onere di indicare alcuni elementi quali il luogo e la data della pubblicazione, il nome e il domicilio dello stampatore, il nome del proprietario e del direttore (o vice direttore) responsabile, che dev'essere iscritto nell'albo dei giornalisti; inoltre nessun giornale o periodico può essere pubblicato se non sia stato registrato presso la cancelleria del tribunale, nella cui circoscrizione la pubblicazione deve effettuarsi.

La legge 47 punisce la diffamazione commessa col mezzo della stampa, la diffusione di immagini raccapriccianti e impressionanti e le pubblicazioni che "corrompono" gli adolescenti e i fanciulli. Obbliga i direttori alla rettifica delle notizie inesatte e alla pubblicazione delle sentenze dei tribunali a tutela dei diritti dei cittadini.

Evidentemente il legislatore del '48 pensava a una stampa esclusivamente cartacea e solo con la legge 6 agosto 1990 n. 223 (detta "legge Mammi") è stato esteso alle emittenti televisive e radiofoniche l'obbligo di registrazione delle rispettive testate giornalistiche e quello della rettifica.

In assenza di una disposizione di legge specifica, alla fine degli anni '90 la registrazione dei giornali telematici era frutto di un'interpretazione dei giudici¹.

Solamente con la legge 7 marzo 2001 n. 62, relativa ai contributi e ai sussidi all'editoria, si è

ricompreso nella definizione di "prodotto editoriale" anche quello realizzato su supporto informatico destinato alla pubblicazione o alla diffusione di informazioni presso il pubblico con ogni mezzo, anche elettronico.

L'approvazione di tale normativa, anche per via della sua formulazione ambigua, diede luogo a vibranti proteste di coloro che pensarono a un tentativo di imbavagliare la rete equiparando i siti internet a dei giornali telematici.

In realtà l'obbligo della registrazione presso i Tribunali e l'assoggettamento alla disciplina sulla stampa riguarda solo i siti che *diffondono informazioni presso il pubblico* legate all'attualità e sono contraddistinti da una periodicità regolare e da una testata *costituente elemento identificativo del prodotto* [riquadro 1]. Il tribunale è quello nella cui circoscrizione la testata on-line ha la redazione, mentre lo stampatore è il provider, che *"concede l'accesso alla rete, nonché lo spazio nel proprio server per la pubblicazione dei servizi informativi realizzati dal fornitore di informazioni"*².

Tali giornali on line sono tenuti a nominare un direttore responsabile il cui compito primario è quello di impedire che *siano commessi delitti con il mezzo della stampa* e di far rispettare le norme deontologiche della professione giornalistica (compreso il Codice di deontologia sulla privacy).

Ulteriore onere da considerare è l'iscrizione al *Registro degli Operatori di Comunicazione* (ROC), tenuto dall'Autorità per le garanzie nelle comunicazioni (AGCOM), a cui sono tenute le imprese editrici.

Ruquadro 1

Le testate giornalistiche elettroniche da registrare sono quelle quotidiane, settimanali, bisettimanali, quindicinali, mensili, bimestrali o semestrali caratterizzate (secondo l'insegnamento costante della Cassazione):

a. dalla raccolta, dal commento e dall'elaborazione critica di notizie (attuali) destinate a formare oggetto di comunicazione interpersonale;

b. dalla tempestività di informazione diretta a sollecitare i cittadini a prendere conoscenza e coscienza di tematiche meritevoli, per la loro novità, della dovuta attenzione e considerazione.

¹ Si segnala in particolare l'ordinanza del presidente del Tribunale di Roma del 6 novembre 1997 per la testata InterLex.

² Così si esprime il Tribunale di Cuneo con provvedimento del 23 giugno 1997.

Naturalmente si applicano alle testate *on line* anche le norme penali relative alla stampa tradizionale, tra le quali si ricorda per importanza e frequenza l'art. 57 c.p.; quest'ultimo punisce il direttore o il vice-direttore responsabile che omette di esercitare sul contenuto del periodico da lui diretto il controllo necessario a impedire che col mezzo della pubblicazione siano commessi reati. Si pensi al caso di un articolo dal contenuto diffamatorio: in tale eventualità oltre al giornalista che ha redatto il pezzo, sarà punito anche il direttore del periodico (si veda in proposito l'art. 596 bis c.p. "diffamazione col mezzo della stampa"). Al di fuori dei casi dei giornali telematici, i siti internet non sono equiparabili alle riviste e, pertanto, non risultano sottoposti alla normativa sulla stampa.

Questo comporta non solo l'assenza degli oneri di registrazione, ma anche l'inapplicabilità delle menzionate sanzioni penali previste nella disciplina speciale: in osservanza a un basilare principio giuridico secondo il quale è vietata l'interpretazione analogica *in malam partem* (cioè a sfavore della persona), non si può assimilare il sito internet a uno stampato e, conseguentemente, non si può equiparare il titolare di un sito al direttore di un giornale.

Interessante leggere la motivazione³ espressa da un giudice in ordine alla sostanziale differenza tra la carta e i bit: "il concetto di riproduzione, che costituisce il fulcro della definizione di stampato, presuppone una distinzione fisicamente percepibile tra l'oggetto da riprodurre e le sue riproduzioni, essendo poi indifferente il procedimento fisico-chimico mediante il quale la riproduzione viene posta in essere. Il testo pubblicato su sito internet non può invece essere considerato una riproduzione. Il relativo *file*, invero, si trova in unico originale sul sito stesso, e può essere consultato dall'utente mediante l'accesso al sito. La riproduzione del *file*, del tutto eventuale, viene posta in essere solo in seguito dallo stesso utente il quale, se lo desidera, può provvedere a stampare il *file* scaricato. Non può quindi ritenersi che il titolare del sito internet sia responsabile di tali riproduzioni, in quanto del tutto eventuali e poste in essere dagli stessi

utenti. Come è già stato sopra argomentato, i *files* pubblicati su internet non sono riproduzioni, ma documenti informatici originali⁴".

Se così non fosse, chi gestisce un sito internet dovrebbe essere punito penalmente per non aver controllato quanto scritto o fatto da altri nel suo sito.

Sarà, invece, punibile per diffamazione aggravata ai sensi dell'art. 595 comma 3 c.p. chi offenda l'altrui reputazione comunicando con più persone attraverso internet (siti web, ma anche newsgroup, mailing list, chat ecc.) [riquadro 2]. A proposito di diffamazione *on line*, si ricorda che è un reato d'evento (non fisico, ma psicologico) e non di mera condotta, il reato si consuma non al momento della diffusione del messaggio lesivo da parte del soggetto agente, ossia l'upload sul server (smtp, nntp, http ecc.), bensì in quello successivo della percezione dello stesso da parte dei terzi, vale a dire il download sul computer dell'utente.

In virtù del cosiddetto principio dell'ubiquità (che consente al giudice italiano di procedere nel caso in cui si sia verificato nel territorio nazionale anche solo parte della condotta o dell'evento), sussiste la giurisdizione italiana qualora nel nostro paese sia ubicato il computer da cui sono stati immessi i dati oppure il provider o anche la persona diffamata.

Per quanto attiene alla competenza territoriale, in base all'art. 8 c.p.p. è competente il giudice del luogo in cui il reato è stato commesso: il luogo in cui si è verificato l'evento offensivo andrebbe individuato laddove sia stata letta la notizia offensiva per la prima volta, ma ciò risulta di difficile, se non di impossibile, determinazione.

Supplirebbero, quindi, i criteri stabiliti dall'articolo successivo: la competenza si radica nel luogo in cui è avvenuta parte dell'azione criminosa (ossia l'upload), o in subordine, dove l'imputato abbia la residenza (o la dimora, o il domicilio).

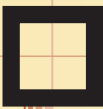
Si segnalano, per completezza, anche alcuni

Ruquadro 1: Fonti normative

- art. 21 Costituzione
- legge 8 febbraio 1948 n. 47
- legge 6 agosto 1990 n. 223
- legge 7 marzo 2001 n. 62
- artt. 57, 595 e 596 bis c.p.

³ Sentenza del Giudice per le indagini preliminari presso il Tribunale di Aosta n. 22/2002 del 15 dicembre 2002.

⁴ Si richiama, a tal proposito, quanto evidenziato in merito al documento informatico nella rubrica "ICT e diritto" del numero 04/2005 di questa rivista.



casi nei quali un sito internet (non registrato come testata giornalistica) è stato ritenuto prodotto editoriale ai sensi della legge n. 62/01, con applicazione della disciplina penale prevista dalla legge 47/48⁵. Il Giudice nella fattispecie, ha ordinato il sequestro preventivo del sito incriminato presso l'internet provider, ma si ritiene che avrebbe potuto assumere il medesimo provvedimento anche senza invocare la normativa sulla stampa.

3. I BLOG

Negli ultimi anni è letteralmente esploso il fenomeno "blog", termine coniato dalla crasi tra i vocaboli "web" e "log", che tradotto letteralmente significa "traccia in rete" e indica un sito internet nel quale sono pubblicate notizie, informazioni e opinioni, visualizzate in ordine cronologico inverso. Per tali caratteristiche si può considerare un blog quale un ibrido tra un sito personale e un forum di discussione che rappresenta una forma di libera espressione del pensiero [riquadro 3]. La semplicità nella pubblicazione dei contenuti su un blog (non è richiesta la conoscenza del linguaggio HTML) ha contribuito a diffondere tale strumento di comunicazione e informazione non

Riquadro 3: Tipologie di blog (Fonte: wikipedia)

- blog personale
- blog di attualità
- blog tematico
- blog directory
- photoblog
- blog vetrina
- blog politico
- urban blog
- watch blog
- m-blog
- vlog o video blog
- Audio Blog
- nanopublishing
- moblog
- multiblogging
- blognovel o blog novel o blog fiction

solo tra chi lo utilizza come diario personale, ma anche tra chi ne fa un uso professionale (giornalisti, politici, uomini pubblici in generale).

Non vi è dubbio che anche il blog costituisca un mezzo di diffusione attraverso il quale ogni individuo possa veicolare le proprie idee e, pertanto, sia garantito dal sopra citato art. 21 della costituzione repubblicana; d'altro canto si è posta la problematica della responsabilità di quanto pubblicato nei blog stessi.

A far discutere il "mondo dei blog", suscitando vibranti proteste per la lesione del principio secondo il quale la responsabilità penale è personale, è stato ancora una volta un provvedimento giudiziario⁶.

La vicenda prende le mosse dal caso, tutt'altro che infrequente, di alcuni scritti offensivi pubblicati su un blog; il giudice da un lato ravvisa la responsabilità penale per diffamazione aggravata in capo all'autore delle frasi offensive (e fin qui *nulla quaestio*), dall'altro affronta la questione della responsabilità penale del titolare del blog. Ad avviso del tribunale, il soggetto che gestisce un blog e può quindi cancellare i messaggi inviati da altri avrebbe la medesima posizione del direttore di una testata giornalistica stampata: il blogger *"ha infatti il totale controllo di quanto viene postato e, per l'effetto, allo stesso modo di un direttore responsabile, ha il dovere di eliminare quelli offensivi"*. La conseguenza di tale ragionamento è che anch'egli risponde ai sensi dell'articolo 596 bis c.p. per la diffamazione compiuta da terzi sul suo blog!

Il Tribunale, poi, per valutare se le affermazioni sul blog siano diffamatorie richiama i menzionati canoni giurisprudenziali costituiti da interesse pubblico alla conoscenza, verità del fatto e correttezza del linguaggio e nel caso di specie conclude per la condanna del blogger.

Tale sentenza risulta sintomatica della tendenza, sempre più frequente nel settore delle nuove tecnologie, di tutelare gli interessi dei terzi ponendo gli operatori della rete.

Vale la pena rammentare in proposito che la direttiva 2000/31/CE⁷, come regola, non impone ai prestatori di servizi della società dell'informazio-

⁵ Si pensi all'ordinanza del Giudice per le Indagini preliminari presso il Tribunale di Latina del 7 giugno 2001 con la quale è stato disposto il sequestro di un sito che vilipendeva la religione cattolica.

⁶ Sentenza del Tribunale Aosta n. 553 del 26 maggio 2006.

⁷ La direttiva è stata attuata in Italia con il decreto legislativo 9 aprile 2003 n. 70.

ne un obbligo generale di sorveglianza sulle informazioni trasmesse o memorizzate, né un obbligo di ricercare attivamente fatti o circostanze che indichino la presenza di attività illecite⁸.

Al massimo può essere stabilito (come ha fatto il legislatore italiano) il dovere di informare senza indugio la pubblica autorità competente di presunte attività o informazioni illecite compiute da parte degli utenti, nonché di comunicare alle autorità competenti, a loro richiesta, informazioni che consentano l'identificazione degli utenti stessi.

In buona sostanza, non vi è un onere di controllo preventivo, ma di intervento e di collaborazione successivi.

A questi principi si ritiene debba essere improntata anche la disciplina relativa ai blog, rispetto ai quali non pare verisimile, né condivisibile l'idea di filtrare le comunicazioni o di bloccare o cancellare automaticamente quelle in cui ricorrano determinate parole chiave di natura offensiva.

4. CONCLUSIONI

Recentemente ha fatto gridare alla censura la proposta di legge approvata dal Governo nel Consiglio dei Ministri del 12 ottobre 2007. Il

provvedimento, intitolato *“Nuova disciplina dell’editoria e delega al Governo per l’emanazione di un testo unico sul riordino della legislazione nel settore editoriale”* obbliga a iscriversi al Registro degli Operatori di Comunicazione, in uno speciale registro custodito dall’Autorità per le Comunicazioni, chiunque faccia attività editoriale; in quest’ultima rientrebbe -secondo la definizione normativa- *“qualsiasi prodotto contraddistinto da finalità di informazione, di formazione, di divulgazione, di intrattenimento, che sia destinato alla pubblicazione, quali che siano la forma nella quale esso è realizzato e il mezzo con il quale esso viene diffuso”*, ivi compresi i blog che pertanto dovrebbero essere registrati.

A seguito della polemica sul contenuto di tali disposizioni, il ministro delle comunicazioni Gentiloni ha ammesso la svista, promettendone la correzione; considerata la fine anticipata della legislatura, non resta di attendere il prossimo Governo, auspicando un intervento risolutore.

Appare evidente, infatti, che fino a quando il Parlamento non si pronuncerà in maniera chiara ed esaustiva sulla definizione di prodotto editoriale, la questione rischia di venir decisa in maniera difforme a seconda delle interpretazioni dei singoli tribunali, salvo attendere una pronuncia esegetica della Corte di Cassazione, con evidenti rischi per la libertà d’opinione e d’espressione sancita dalla Costituzione.

⁸ A questa regola generale fanno eccezione i casi di semplice trasporto (mere conduit) delle informazioni, di memorizzazione temporanea (caching) e di “hosting”, che in questa sede non rilevano.

ANTONIO PIVA, laureato in Scienze dell’Informazione, *Vice Presidente dell’ALSI* (Associazione Nazionale Laureati in Scienze dell’Informazione ed Informatica) e Presidente della commissione di informatica giuridica. Docente a contratto di *diritto dell’ICT e qualità* all’Università di Udine. Consulente sistemi informatici e Governo Elettronico nella PA locale, valutatore di sistemi di qualità ISO9000 ed ispettore AICA.
E-mail: antonio@piva.mobi

DAVID D’AGOSTINI avvocato, master in informatica giuridica e diritto delle nuove tecnologie, collabora all’attività di ricerca scientifica dell’Università degli studi di Udine e ha fondato l’associazione *“Centro Innovazione & Diritto”*. È componente della Commissione Informatica dei Consigli dell’Ordine del Triveneto, responsabile dell’area *“Diritto & informatica”* della rivista *“Il foro friulano”*, membro dell’organo di Audit Interno di Autovie Venete SpA.
E-mail: studio@avvocatodagostini.it

LA RADIOFONIA NELL'ERA DIGITALE¹

Dopo una breve sintesi sull'evoluzione della fruizione radiofonica da parte degli utenti, l'articolo passa in rassegna i nuovi standard dei sistemi per la diffusione della radiofonia digitale che potranno interessare il nostro Paese. Questi nuovi sistemi prenderanno progressivamente il posto delle Onde Medie e della Modulazione di Frequenza nella banda UHF e sono caratterizzati da un'alta efficienza di occupazione spettrale e da una forte resistenza a tutti i disturbi di propagazione, prestazione particolarmente importante per l'ascolto, in continuo aumento, nella condizione di mobilità.

1. PREMESSA

La grande rivoluzione che le tecnologie digitali hanno portato in tanti settori non poteva non toccare anche la filiera radiofonica. L'area della produzione radiofonica è ormai digitalizzata da alcuni anni, mentre l'area della diffusione, almeno in Italia, è ancora sostanzialmente analogica. Nel seguito, sono brevemente descritte le principali nuove soluzioni proposte per la diffusione e si cerca di valutare quali fra esse siano le più adatte al nostro Paese².

Non va dimenticato che la convergenza dei *media* e dei terminali tende a favorire un'aggregazione di segmenti di mercato oggi differenziati (radio, TV, telefonia fissa e mobile ecc.) e conseguentemente una concentrazione nelle mani

dei *provider* più forti. Tale prospettiva può certamente costituire una minaccia per la tutela del pluralismo e del localismo e vanno pertanto adottate quelle soluzioni di diffusione che garantiscano i *provider* più deboli, senza peraltro frenare lo sviluppo tecnologico e di mercato. Prima di esaminare i nuovi sistemi di diffusione, si ritiene utile premettere un breve commento alle evoluzioni che hanno subito le tradizionali modalità di fruizione della comunicazione radiofonica nel più generale contesto del cammino che la Società sta percorrendo sulla strada sempre più pervasiva della multimedialità. Tale quadro è particolarmente utile per meglio inquadrare le specifiche caratteristiche che devono avere i moderni sistemi di diffusione.

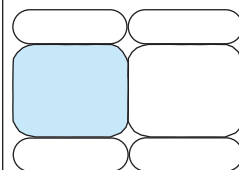
2. LO SCENARIO DEL MERCATO DI FRUIZIONE PER I SISTEMI RADIOFONICI

In sintesi, queste sono le principali considerazioni che si possono fare relativamente a tale tematica:

□ l'ascolto **stanziale** (casalingo) rimarrà una



Salvatore Morello
Guido Vannucchi



¹ Il presente articolo riprende ed aggiorna un Position Paper pubblicato nell'ottobre del 2005 dalla AICT (Associazione per la Tecnologia dell'informazione e delle Comunicazioni), struttura dell'AEIT (Federazione Italiana di Elettrotecnica, Elettronica, Automazione, Informatica e Telecomunicazioni).

² Si pensi, per esempio, alla situazione di affollamento dello spettro delle frequenze radiofoniche in Italia.

realità di notevole rilevanza, anche se con una duplice ed in un certo senso contraddittoria tendenza:

- ascolto di massima qualità (in modalità multicanale) in un unico punto concentrato della casa (spesso coincidente con la posizione del ricevitore TV);
- ascolto esteso ai vari ambienti all'interno della casa, risolvibile con reti LAN domestiche, *wired* o *wireless*, oggi molto economiche e realizzabili con buona qualità di ascolto per merito delle eccellenti prestazioni degli altoparlanti anche di piccola dimensione;
- l'ascolto in **mobilità** (**autoradio o dispositivi "handheld"**³) diventerà sempre più richiesto e sarà importante scegliere le soluzioni in grado di assicurare una buona ricezione dei segnali radiofonici su una percentuale molto alta del territorio;
- l'ascolto in **nomadicità** (tipico di chi usa il PC portatile collegato a reti WiFi in alberghi, stazioni, aeroporti ecc.) tenderà anch'esso ad aumentare sensibilmente;
- **nuove offerte di programmi**, in particolare tematici o locali, saranno rese possibili dal deciso miglioramento dell'efficienza spettrale conseguente all'impiego delle tecnologie digitali;
- il mix dell'utenza radiofonica, sempre più ricco di **giovani**, propenderà progressivamente verso forme di innovazione radicali, con nuovi paradigmi di servizio associati alle modalità della fruizione radiofonica, ciò che determinerà nel tempo lo scostamento dal concetto di "**radio classica**" ed in particolare tenderà a sviluppare:
 - integrazione del segnale audio con altri *media* (dati ed immagini);
 - interattività attraverso opportuni canali di ritorno;
 - modelli di *comunicazione* innovativi, *free* o *pay*, quali in particolare il multicast (da uno a molti) e l'*unicast* (da uno ad uno, ossia *on demand*), che supereranno il classico *broadcast* (da uno a tutti);
 - fruizione, oltre che in *real time*, anche in *shifted time* tra cui, in misura sempre più esplosiva, il servizio *PODcast* (*Personal-Option-Digital-cast*).

³ Con tale denominazione ci si intende riferire a tutte le classi di terminali portatili (cellulari, radio portatili, PDA ecc.).

3. LE TECNOLOGIE PER I SISTEMI DI DIFFUSIONE DI RADIOFONIA DIGITALE

Per seguire meglio il richiamo nel successivo paragrafo sulle caratteristiche tecniche dei vari standard proposti, è utile ricordare che la diffusione in forma digitale dei segnali radiofonici avviene attraverso i passaggi classici delle trasmissioni digitali e più precisamente:

a. Codifica di sorgente: è il processo attraverso cui il segnale audio analogico viene prima convertito in una sequenza numerica (digitalizzato) e quindi "**compresso**" secondo tecniche di riduzione di ridondanza che, preservando sostanzialmente la qualità soggettiva, riescono a ridurre sostanzialmente il *bit-rate* del segnale e ne consentono la trasmissione agli utenti con una sensibile efficienza spettrale, molto superiore a quella ottenibile in analogico (riquadro a p. 41, per qualche maggior dettaglio sulle tipologie di codifiche di sorgente nel caso dei segnali radiofonici).

b. Codifica di canale: è l'operazione con la quale il segnale digitale, dopo la codifica di sorgente, viene ulteriormente elaborato (con un moderato aumento del *bit-rate*) per difenderlo al meglio dalle possibilità di errore introdotte dal particolare canale di trasmissione impiegato. La parte più interessante dell'elaborazione consiste nell'introdurre i cosiddetti **codici a correzione di errore** (per eliminare cioè gli errori nel riconoscimento dei *bit*), sistemi rilevanti in ogni caso, ma indispensabili per i mezzi trasmissivi radio resi critici dai problemi di propagazione quali, in particolare, riflessioni, interferenze, evanescenze (*fading*) di tipo aleatorio e, per le frequenze alte, le attenuazioni da pioggia.

c. Modulazione: è il "blocco circuitale" della catena trasmissiva in cui il segnale, dopo le due precedenti elaborazioni, viene "**impresso**" su una "**portante**", rendendone possibile la trasmissione e/o la diffusione a grande distanza. Le metodologie di modulazione per segnali digitali sono scelte sulla base delle particolari caratteristiche del mezzo trasmissivo impiegato (radio terrestre, satellite, cavo, fibra), della specifica banda di frequenza adottata e delle relative caratteristiche di propagazione.

Un metodo di modulazione particolarmente efficace (in combinazione con i codici correttori) per combattere le anomalie di propagazione sopra citate (e perciò adottato nella dif-

La codifica di sorgente

La trasformazione di un segnale audio analogico, prodotto da un trasduttore elettroacustico, in un segnale **digitale** è un processo delicato e fondamentale al fine di garantire una qualità **soggettiva** di alta fedeltà.

Si agisce su parametri *analitici*, quali:

- la modulazione del segnale per trasformarlo in impulsi-**campione** per la quale, per i CD, si usa la *Pulse Code Modulation (PCM)* ad 1,4 Mbit/s;
- la **frequenza di campionamento** che, tipicamente, sempre per i CD, è di **44.100 kHz**;
- la **quantizzazione** di un campione che nei CD, ha ormai lo standard di riferimento con un valore di **16 bit per campione**;
- Il ritardo di elaborazione (**processing delay**) che rappresenta la somma dei tempi impiegati da *encoder* e *decoder* per eseguire le rispettive operazioni di codifica (compressione) e decodifica (decompressione). Un ritardo di elaborazione superiore a qualche decina di millisecondi (ms) può creare problemi per la diffusione della trasmissioni in diretta; e inoltre si agisce su parametri *percettivi*, che tengono conto cioè delle complesse caratteristiche fisiche del sistema di percezione auditivo umano. In questo caso, si può eliminare dal segnale una certa quantità d'informazione, sapendo che essa sarebbe ininfluente ai fini della sensazione auditiva finale, sfruttando per esempio:
 - la **non linearità** della risposta del nostro orecchio medio ed interno;
 - il fenomeno del **mascheramento**, che "nasconde" un suono di basso livello, quando è presente un altro di livello elevato. Tra l'altro, questo processo è dipendente anche dalle frequenze dei due suoni e si ha anche se i due suoni non sono contemporanei, ma comunque molto ravvicinati nel tempo.

Poiché la banda necessaria a trasmettere o memorizzare un flusso di dati corrispondente ad un segnale audio della durata di 1 s (**bit rate**) è un elemento essenziale dal punto di vista del costo, negli ultimi decenni i maggiori studi del settore sono stati dedicati alla messa a punto di modelli di compressione sempre più elaborati, che qui ci si limita ad elencare:

- **MUSICAM: MPEG 1, Layer 2**, che è il primo ad aver avuto successo, con fattore di compressione di **7,3**;
- **MP3: MPEG2, Layer 3**, noto in particolare per l'uso diffusissimo in internet e nei moderni sistemi di *storage* (vedi *i-Pod*) con fattore di compressione di circa **11**;
- **AAC (Advanced Audio Coding): MPEG2 e 4 - AAC** con fattore di compressione di **14,7**;
- **AAC+** che con un processamento più evoluto dell'AAC raggiunge un fattore di compressione di **22**;
- **AAC+2**: ancora più avanzato, con fattore di compressione di quasi **30**.

fusione radio-televisiva terrestre) è la tecnica **multicarrier COFDM (Coded Orthogonal Frequency Division Multiplexing)**, in cui il flusso digitale dei *bit* da trasmettere viene distribuito, nell'ambito dello spettro a disposizione, su un numero elevatissimo di frequenze portanti. Il risultato di tale metodologia - che nella sua implementazione pratica è realizzata esclusivamente attraverso tecniche di **digital processing** del flusso nativo dei *bit* - è una trasmissione estremamente robusta, in grado di resistere a molti tipi di perturbazioni. Gli specifici **vantaggi tecnologici conseguenti alla digitalizzazione** si possono così riassumere:

- **alto sfruttamento dello spettro**, con capacità di moltiplicare da quattro a sei volte il numero di programmi trasmessi nelle classiche bande analogiche della radiofonia (OC, AM, FM) e di utilizzare ad alta efficienza nuove bande di cui si parlerà nel seguito;
- **ottima qualità del servizio** all'utente, anche in situazioni di propagazione perturbata;
- capacità di **offrire servizi multimediali complementari** al segnale audio (dati, immagini, video lenti ecc.), ciò che ha giustificato per alcune applicazioni il termine di "**visual radio**";

□ **riduzione della potenza elettrica** necessaria alla diffusione a parità di area di servizio, con conseguente diminuzione delle emissioni elettromagnetiche.

Con tali vantaggi la transizione al digitale si presenta anche come elemento propulsore dell'economia dei *media*, non solo per lo sviluppo di nuovi apparati, ma per tutti i contenuti e servizi innovativi che si potranno creare.

4. GLI STANDARD DI DIFFUSIONE PROPOSTI PER LA RADIOFONIA DIGITALE

In questa sede, per brevità, si prenderanno in esame solo i sistemi che possono interessare la realtà italiana.

4.1. Sistemi terrestri

a. I sistemi T-DAB (Terrestrial -Digital Audio Broadcasting) e T-DMB (Terrestrial-Digital Multimedia Broadcasting)

Il **T-DAB (Terrestrial-Digital Audio Broadcasting)** è un sistema il cui studio è iniziato alla fine degli anni '80 ed è stato il primo standard di radiofonia digitale.

La denominazione originale di questo stan-

dard e dei successivi non introduceva il prefisso "T" che è stato introdotto di recente, per differenziarlo dalle soluzioni per satelliti che permettono la lettera "S".

La codifica di sorgente adottata è il *MUSICAM* (riquadro a p. 41), codifica di prima generazione che richiede 190 kbit/s per programma stereofonico.

La modulazione adottata è il sistema *multi-carrier COFDM* già richiamato. Le caratteristiche della modulazione consentono la realizzazione di reti SFN⁴ per una copertura nazionale o regionale.

Il sistema è caratterizzato dall'impiego di bande non appartenenti al normale spettro in uso per la radiofonia AM e FM, quali la **Banda VHF-III** (cioè la banda 174-240 MHz storicamente allocata per servizi televisivi VHF) e la porzione 1452-1469 MHz, della **Banda L** (storicamente per servizi punto-punto).

Le bande di frequenza assegnate vengono poi ripartite in "**canali DAB**" di circa 1.5 MHz. (nel caso VHF un canale televisivo da 7 MHz è in grado di portare 4 canali DAB). Ciascun canale DAB ospita un "**multiplex**" di 6 programmi radiofonici (detto anche "**blocco**") con qualità "CD like". Lo standard permette l'integrazione di servizi "dati" a banda stretta associati al programma audio. Lo standard è nato con l'obiettivo di ricoprire essenzialmente la ricezione "*outdoor*", ossia in mobilità, ed offre ottime prestazioni anche per velocità elevate di movimento contro l'effetto *Doppler* (ricezione garantita fino a 200 km/h). Per la ricezione "*indoor*", in assenza di antenne sulla sommità degli edifici, è necessario un progetto di rete a maggiore densità di trasmettitori e/o con potenze incrementate, per via delle ulteriori attenuazioni all'interno di un edificio. Analoga situazione si presenta anche per gli altri sistemi candidati per la radiofonia digitale (DMB e DVB-H) esaminati nel seguito.

Il T-DAB, pur adottando una modulazione di concezione molto valida, mostra tutti gli anni passati dalla sua ideazione. Offre, infatti, uno sfruttamento dello spettro non ottimale

sia per l'uso di una codifica di sorgente di prima generazione che richiede, per trasportare un programma, un *bit-rate* più che doppio delle codifiche attuali, sia per le tecnologie impiegate nella codifica di canale e nella modulazione che non permettono le flessibilità offerte dai sistemi più moderni⁵.

Per le limitazioni appena dette, tra il 2002 ed il 2003 nasce lo **standard T-DMB** (*Terrestrial-Digital Media Broadcasting*) per migliorare le caratteristiche del DAB e con l'obiettivo di estenderne le prestazioni anche alla diffusione di *Mobile TV*. Il sistema DMB è totalmente ispirato al DAB ed è compatibile con quest'ultimo in quanto adotta **stesse allocazioni spettrali ed identico tipo di modulazione**. La **codifica di sorgente** del segnale radiofonico impiega invece tecniche di compressione audio di **seconda generazione** (*MPEG4-AAC*)⁶ ed una **codifica di canale** più robusta. Aggiunge inoltre la possibilità di una codifica video a standard MPEG4. Se si desidera invece mantenere la compatibilità radiofonica con i terminali DAB esistenti, si perdono i vantaggi delle nuove codifiche audio e si sfrutta esclusivamente la potenzialità del sistema per la trasmissione di *Mobile TV*.

Per quanto detto, un sistema DMB può essere realizzato, ove sia preesistente o predisposta una rete DAB, con investimenti minimi e rappresenta pertanto la sua naturale evoluzione. Tralasciando i problemi di compatibilità, il DMB è certamente uno dei sistemi maggiormente candidato a fornire servizi radiofonici digitali di qualità su frequenze terrestri.

Per questo si parla oggi di **sistemi DAB/DMB** perchè per la radiofonia la prevalenza del primo o del secondo standard dipende dalla scelta strategica effettuata sulla base del numero dei ricevitori DAB preesistenti nel Paese. Questo standard ha avuto la prima implementazione completa in Corea con considerevoli investimenti. Sperimentazioni e prime installazioni sono in corso in Germania e in Italia ed altri Paesi europei.

⁴ **SFN = Single Frequency Network:** rete, per copertura nazionale o regionale, costituita da trasmettitori diffusi nel territorio funzionanti in modo sincrono alla stessa identica frequenza, senza problemi di interferenza. Tale soluzione è impensabile con le tecniche analogiche.

⁵ Il sistema, nel frattempo, non ha raggiunto un vero decollo commerciale in nessun Paese ad eccezione di UK e Corea.

⁶ *MPEG4-AAC = Moving Picture Experts Group-Layer 4-Advanced Audio Coding.*

b. Il sistema DVB-H (Digital Video Broadcasting- Handheld)

È uno standard europeo aperto del consorzio DVB (*Digital Video Broadcasting*), finalizzato nel Novembre 2004, **orientato alla diffusione di Mobile-TV**, in cui il termine “*Handheld*” indica che si tratta di un “componente” della famiglia di standard DVB studiato per trasmissioni verso ricevitori mobili alimentati a batteria (segnatamente cellulari, palmari e PC portatili). Il sistema cura, in particolare, la robustezza della codifica di canale per la ricezione da terminali mobili (il cui guadagno di antenna può anche essere minore dell’unità) ed il basso consumo del ricevitore al fine di una massima durata delle batterie.

Il DVB-H integra le funzionalità delle reti DVB-T (ossia l’ormai diffuso sistema televisivo digitale terrestre) cercando di migliorarne le prestazioni, dal momento che questo standard non è stato concepito per la ricezione TV in piena mobilità.

Il DVB-H, oltre servizi di mobile-TV, per le sue prestazioni a carattere multimediale, è **perfettamente adatto alla trasmissione broadcast di programmi radiofonici e dati**, mantenendo un alto grado di compatibilità con i servizi della rete DVB-T. Il basso consumo del terminale ricevente è ottenuto attraverso la cosiddetta tecnica del “*timeslicing*” che consiste nel tenere acceso il terminale solo quando riceve i dati di interesse (*burst*) del programma TV desiderato. Per i restanti *burst* di altri programmi il ricevitore rimane spento, consentendo un notevole risparmio di energia (fino a circa il 90% del consumo).

Il sistema non presenta limitazioni sulle codifiche di sorgente. In particolare, per la radiofonia, sono previste le codifiche audio-video di seconda generazione (*MPEG-AAC*) che realizzano la massima efficienza spettrale. Come modulazione, il DVB-H adotta, come il DVB-T, una modulazione *multicarrier COFDM* e può coesistere con il DVB-T nell’ambito dello stesso multiplex di programmi, come normalmente avviene. All’interno della propria “porzione di multiplex”, il sistema DVB-H migliora la codifica di canale aggiungendo un **ulteriore codice correttore**. Ciò rende la codifica più robusta nelle condizioni di mobilità con le antenne a stilo o integrate dei terminali portatili, in quanto il sistema DVB-T a cui si appoggia è

nato con l’obiettivo di una ricezione indoor tramite le classiche antenne televisive Yagi.

Lo standard DVB-H è stato promosso da costruttori di apparati e da operatori di rete come sistema di **diffusione per qualunque tipo di programma multimediale**.

c. Il sistema DRM (Digital Radio Mondiale)

Il sistema **DRM** è uno standard universale studiato espressamente per la diffusione sulle bande di frequenza inferiori ai 30 MHz delle **Onde Lunghe (LW), Medie (MW) e Corte (SW)** ed è in grado di trasportare un programma a qualità CD con il corredo dei relativi dati informativi. La codifica di sorgente adottata è, al solito, la cosiddetta “*MPEG4-AAC*” di seconda generazione, seguita da codifiche di canale molto potenti ed il sistema di modulazione è ancora una volta il COFDM. Particolari combinazioni di codifica di sorgente ed “*enhancers*” permettono di variare il *bit-rate* audio da 8 a 48 kbit/s, in funzione della qualità/ricevibilità del segnale trasmesso. Possono essere anche impiegate codifiche avanzate solo per la voce: la codifica CELP (*Coded Excited Linear Prediction*) per audio di alta qualità senza contenuti musicali e la codifica HVXC (*Harmonic Vector Excitation*) per il parlato che ha un *bit-rate* particolarmente basso.

Lo standard DRM consente anche di diffondere contenuti testuali e grafici.

Sono in corso gli studi per estendere l’applicabilità dello standard a bande di frequenza superiori ai 30 MHz e fino a 120 MHz, in modo tale da poterlo **utilizzare anche nelle classiche bande FM** della radiofonia (**DRM+**).

Il DRM non prevede il trasporto di multiplex, ma di **singoli programmi**, soluzione questa molto apprezzata dalle piccole emittenti locali (mono-programma) che preferiscano mantenere una loro immagine autonoma e non desiderano essere costrette a consorziarsi con altre emittenti per realizzare un multiplex di programmi come previsto dai precedenti standard. Dal punto di vista tecnico, il vantaggio di trasmettere un programma singolo invece di un multiplex, si paga in termini di prestazioni a radiofrequenza: infatti i sistemi a banda stretta non beneficiano della “diversità di frequenza” e risentono maggiormente delle evanescenze del segnale (dovute agli echi) che risultano molto selettive in frequenza.

La larghezza di banda proposta per ogni singo-

lo canale di questo sistema varia tra i 7 e i 10 kHz, con possibilità di raggruppare più canali.

4.2. Sistemi via satellite

a. I sistemi radiofonici da satelliti

È in corso di sviluppo un sistema satellitare puramente per radiofonia digitale spinto dalla Società Worldspace per la copertura dell'Europa e dell'Africa operante nella banda 1480-1492 MHz (banda L) che adotta lo **standard ESDR** (*ETSI Standard for Digital Radio*) di cui sono scarse le informazioni e che sembra orientato a ricevitori proprietari. Il sistema prevede un'**architettura ibrida di rete terrestre-satellitare**, fondata sull'utilizzo della medesima interfaccia radio, che potrebbe consentire in molti casi l'ottimizzazione tecnico-economica della distribuzione punto-multi-punto perché le frequenze impiegate per i relativi *gap-filler* terrestri siano rigorosamente le stesse di quelle usate nel satellite. Nell'attuale configurazione di questo progetto, la copertura prevista per l'Europa consiste in 6 fasci "linguistici", che permettono la copertura integrale dei maggiori Paesi del continente, grazie anche alla componente ausiliaria di ripetitori terrestri che permettono di raggiungere i terminali in area urbana ed *indoor*.

b. Il sistema DVB-SH (*Digital Video Broadcasting - Satellite for Handheld receivers*)

In aggiunta alla soluzione sopra menzionata che nasce per radiofonia da satellite è in fase avanzata di studio, per impiego in banda 2170-2200 (banda S), il sistema di diffusione DVB-SH che è, in realtà, una nuova generazione del sistema DVB-H (aperto pertanto ad impieghi anche in altre bande oltre la S) con maggiore efficienza spettrale e, basato anche esso, su un'architettura satellite - *gap filler* terrestri. Tale sistema potrà svolgere **un ruolo importante per completare in tempi brevi la copertura DVB-H terrestre, anche in zone a scarsa densità abitativa** dove i limiti di natura economica impedirebbero verosimilmente uno sviluppo a breve della copertura terrestre. Un sistema di tale natura è ovviamente in grado di trasmettere la radiofonia, ma è maggiormente concepito per generiche applicazioni multimediali e più in particolare per Mobile-TV.

In generale le soluzioni satellitari possono consentire la fornitura di servizi diffusivi su

vaste aree territoriali con investimenti contenuti ed immediatezza nella realizzazione della copertura. Per quanto riguarda la copertura di popolazione concentrata soprattutto nelle città, una soluzione satellitare richiede assolutamente di essere integrata, come già detto, con piccoli trasmettitori terrestri (*gap fillers*) tenuto conto delle barriere al debole segnale satellitare costituito dagli edifici ed altri ostacoli (sottopassi, ponti ecc.).

Per le ragioni anzidette, le soluzioni radiofoniche satellitari vanno seguite con una certa attenzione anche se non è ancora chiaro quale possa essere il peso dei *gap-filler* rispetto ad una completa copertura da sistemi terrestri, con pericoli di raddoppio. Non c'è dubbio che, in ogni caso, i sistemi satellitari potranno giocare un ruolo complementare soprattutto per una vasta copertura del territorio.

4.3. Nuove evoluzioni e quadro riassuntivo

La tabella 1 riporta le principali caratteristiche e l'allocatione in frequenza delle soluzioni per diffusione radiofonica che, in linea di principio, possono interessare il territorio italiano.

Si sta completando la standardizzazione di **due nuove proposte**, che sono anche conseguenza dei conflitti tra costruttori e tra enti di standardizzazione, ma che si possono far ricadere per le loro prestazioni nei sistemi della tabella 1.

La prima proposta è il sistema **DAB-IP** caratterizzato dall'introduzione della codifica televisiva per Mobile-Tv (come il DMB) e che agli effetti dell'occupazione spettrale è pertanto equivalente al DMB, da cui si distingue solo per il più moderno uso del protocollo IP. La seconda proposta (in via di standardizzazione) è il sistema **DAB+**, esclusivamente radiofonico, con prestazioni superiori a quelle del DMB nel caso di impiego esclusivo per radiofonia.

Va notato che l'uso del termine DAB in questi due sistemi confonde (forse ad arte) le idee perché essi sono molto più vicini al DMB e come tali possono essere considerati e verranno considerati nel seguito: un loro eventuale successo o meno dipenderà solo dalla diffusione e politica dei relativi ricevitori. Ciò che invece è certo è che questi ultimi due sistemi decretano **definitivamente la morte del DAB propriamente detto** e dei corrispondenti ricevitori per quei Paesi, come l'Italia, che non hanno una storia pregressa. Non a

	T-DAB	T-DMB/DAB+	DVB-H & DVB-SH	DRM
Codifica Sorgente: <i>Audio</i>	MUSICAM	AAC <i>efficienza doppia</i>	AAC <i>efficienza doppia</i>	AAC+ <i>efficienza tripla</i>
<i>Video</i>	Non Prevista	MPEG-4	MPEG-4	Non Prevista
Canalizzazione per un Multiplo da:	1,5 MHz (4 blocchi per canale TV) 6 programmi stereo con qualità CD	1,5 MHz (4 blocchi per canale TV) 10 programmi stereo per DMB e 20 programmi con DAB+, con qualità CD	8 MHz 120 programmi stereo con qualità CD	1 Canale OM/AM/FM Singolo canale
Gamma Frequenza terrestre	• Banda III (200 MHz); • Banda L (1,5 GHz)	• Banda III (200 MHz); • Banda L (1,5 GHz)	• Banda III- IV (500-800 MHz); • Banda L (1,5 GHz)	• LW, MW, SW; • Banda II (FM: 88-108 MHz)
Satellite	• Banda L (1,5 GHz)	• Banda L (1,5 GHz)	• Banda S (2.2 GHz)	-----

caso il WorldDAB forum, che ha in carico gli standard della famiglia DAB/DMB, ha assunto recentemente il nome di WorldDMB.

5. CONSIDERAZIONI E CONFRONTI TRA I SISTEMI RADIOFONICI DIGITALI PER UN UTILIZZO IN ITALIA

Per tutto quanto detto in precedenza, per la diffusione terrestre, ci si può limitare all'esame dei soli tre sistemi **DMB/DAB+**, **DVB-H/DVB-SH** e **DRM**, almeno per le soluzioni di diffusione terrestre, avendo associato le soluzioni molto simili che possono coesistere. La prima importante osservazione da fare è che tali sistemi possono essere considerati non in concorrenza tra loro, ma **complementari** e per applicazioni di fascia diversa.

Il **DMB/DAB+** potrebbe diventare il **sistema principe con alta qualità (qualità CD o superiore) dei grandi e medi broadcaster radiofonici**, in analogia al ruolo svolto dai sistemi FM nel momento in cui si affiancarono ai programmi radiofonici trasmessi in Onde Lunghe, Medie e Corte. Inoltre le possibilità del DMB di trasmettere immagini in movimento (da cui la denominazione visual-radio) consentirebbe l'impiego di servizi interessanti, per esempio per il traffico, nel caso in cui si volessero trasmettere immagini in movimento anziché immagini fisse. La prestazione per la trasmissione contemporanea di radiofonia e video e la maggior protezione richiesta, limita peraltro il massimo di capacità nell'uso puramente radiofonico, ciò che ha in-

dotto all'introduzione del DAB+ al fine di ottenere il massimo sfruttamento spettrale e quindi il massimo di economicità. Sarebbe invece una politica sbagliata, usare in Italia tale sistema per trasmissioni di *Mobile-TV* (come avviene in Corea) in quanto per una tale applicazione l'Europa si sta tutta orientando all'impiego del DVB-H, per sua natura molto legato al sistema televisivo DMB-T ed all'uso delle relative gamme di frequenza. Inoltre, non è corretto andare ad occupare bande destinate alla radiofonia con trasmissioni di tipo televisivo.

Il **DRM** ha il grande vantaggio di operare la trasformazione in digitale del singolo canale radiofonico e tale tecnica può essere impiegata nelle Onde Corte, Medie ed in FM. La specifica tecnica per il DRM in banda FM non è tuttavia ancora pronta e così pure i ricevitori devono raggiungere uno stato di maturazione e di più vasto consumo di massa, risultando ancora costosi ed ingombranti.

Proiettandosi negli anni a venire il sistema DRM potrà rappresentare, in tecnica digitale, il corrispondente dell'Onda Media nella storia dell'analogico, mentre il DMB rappresenterebbe, come già detto, l'equivalente dell'introduzione della *Modulazione di Frequenza* (FM). Peraltro ambedue i sistemi sarebbero di qualità superiore ai parallelismi citati in quanto il DRM può avere la qualità dell'attuale FM, per di più senza interferenze, mentre il DMB avrebbe qualità notevolmente superiore.

Il DRM è pertanto un **sistema adatto ai piccoli gestori radiofonici con stazioni individuali**

TABELLA 1

Quadro riassuntivo dei vari sistemi di radiofonia digitale di interesse italiano

e come qualità è in linea con le teche disponibili dai piccoli gestori radiofonici.

Il processo di trasformazione sarà tuttavia lento e dovrà avvenire progressivamente con la possibilità di coesistenza con gli attuali canali analogici. In particolare tale trasformazione riuscirà a decollare in modo sostanziale solo dopo la migrazione delle trasmissioni radiofoniche al DMB. Il sistema **DVB-H/DVB-SH** ha tutte le potenzialità per trasmettere, oltre il *Mobile-TV*, anche radiofonia di alta qualità e con efficienza spettrale simile a quella del DMB. È tuttavia ormai accertato che il tipo di utilizzazione si concentrerà in Europa sul *Mobile-TV*. Ciò non esclude la possibilità di utilizzarlo con successo negli stessi cellulari per la televisione mobile anche per radiofonia, ma in ogni caso il suo uso si differenzierà da quello del DMB, orientandosi più a considerare i **canali musicali come "Servizi della Società dell'Informazione"** con servizi *premium* (eventualmente *"on demand"* ed a pagamento), ascol-

tabili tramite ricevitori integrati nei cellulari. È infine importante ricordare, come interessante possibilità applicativa, i **sistemi satellitari** che hanno il vantaggio di una rapida copertura del territorio ma che tuttavia nelle aree cittadine ed in presenza di edifici e altri ostacoli, troverebbero gravi difficoltà di funzionamento, non si sa bene quanto facilmente e compiutamente risolvibili attraverso le reti di *gap filler* terrestri operanti alle stesse frequenze. Sembra pertanto opportuno concepirli più in un'ottica di **integrazione delle reti terrestri**. Occorre inoltre fare attenzione che tali sistemi non vengano offerti in modalità non compatibile con la filiera principale DMB, riducendo le possibilità di efficienza spettrale e operando con numerosi *gap-filler* un raddoppio delle reti terrestri a diffusione DMB/DAB+.

Un'ultima considerazione: nel passaggio alla radio digitale, non ha senso parlare di quanti programmi attuali possono trovare allocazione in un multiplex quanto piuttosto è opportu-

Valutazioni economiche

Quali gli investimenti da affrontare e i costi dei ricevitori per una moderna rete radiofonica digitale?

Mentre in passato l'investimento per una rete radiofonica comportava un impegno economico di una certa consistenza oggi la presenza d'infrastrutture già esistenti a cui appoggiarsi (siti radiofonici e televisivi e rete trasmissiva a grande distanza) nonché la continua diminuzione dei prezzi per gli apparati elettronici anche professionali, rende le corrispondenti cifre abbastanza contenute. Solo allo scopo di dare un grossolano ordine di grandezza degli **investimenti sorgenti** (escluso cioè i siti, le torri, i gruppi di continuità e la rete trasmissiva digitale in cui la capacità impiegata è trascurabile rispetto alle necessità della rete televisiva) si può innanzitutto valutare il costo di una rete trasmittente VHF a copertura nazionale in SFN (*Single Frequency Network*) per realizzare la diffusione di un blocco di un multiplex radiofonico da 1,5 MHz.

In tale banda si è in grado di ospitare 10 programmi radiofonici in tecnica DMB (con possibilità di associare ad ogni programma radio anche trasmissioni di servizio in video lento, il cosiddetto *visual radio*) e 20 programmi puramente radiofonici nel caso DAB+. In un multiplex possono coesistere i due tipi di standard poiché i ricevitori più moderni prevedono ormai il bi-standard.

Il calcolo verrà effettuato per una copertura in banda VHF, dal momento che la grande capacità in programmi dei moderni standard radiofonici DMB e DAB+, associata all'evoluzione in atto verso lo *switch-off* della rete televisiva analogica, permetterà man mano di avere disponibili, almeno per le reti a copertura nazionale, gli slot di banda VHF necessari per la radiofonia. Il più costoso impiego della banda L, per il maggior numero di trasmettitori necessari (da 2 a 2,5 volte), sarà confinato ad alcune programmazioni a copertura regionale o provinciale che potrebbero soffrire di interferenze se realizzate in banda VHF.

Nel caso di una rete nazionale, la codifica di sorgente dei singoli programmi e l'apparato di moltiplicazione degli stessi è effettuato a livello nazionale e pertanto l'investimento relativo (circa 20 k€ per ciascun codec di programma e 30 k€ per il multiplex) incide marginalmente sul costo complessivo della rete che è sostanzialmente legato ai trasmettitori per la diffusione.

Ciascun trasmettitore (inclusivo dei modulatori) di varia potenza, corrisponde ad un costo di investimento da 50 a 150 k€ a seconda delle necessità di potenza richiesta nella pianificazione dell'impianto.

Se, in prima battuta, si intende coprire in *"outdoor"* una percentuale della popolazione pari all'80% associata, per la ricezione *"indoor"*, ad una copertura probabilistica all'interno della casa (parametro denominato *"location probability"*) attorno all'85% dei locali, il numero medio di impianti trasmettenti necessario in banda VHF risulta attorno ai 70 di cui il 30% di bassa potenza (200 W) e la restante parte di alta potenza (2 kW). Con i parametri sopra richiamati, il costo complessivo d'investimento di tale rete risulta pertanto attorno agli 8,5 M€ che può arrotondarsi a 10 M€ se si ha in alcune zone necessità di *"gap filler"*. Se si tiene conto di un costo di installazione pari al 30% e delle parti centrali il costo di investimento complessivo per un multiplex a diffusione nazionale si aggira pertanto attorno ai 14 M€.

Per quanto riguarda i **ricevitori**, i prezzi tendono ormai ad essere particolarmente contenuti: un ricevitore fisso DMB semi professionale si aggira attorno ai 250 €, mentre un ricevitore portatile DMB/DAB+ ormai è sotto i 100 € e i prezzi sono ancora destinati a scendere quando aumenteranno i volumi.

no ragionare in termini di *bit-rate* e valore aggiunto rispetto al servizio attuale. In altre parole è importante capire cosa un gestore avrà necessità di dare per creare negli utenti un adeguato **“appeal”**. Ciò è una naturale conseguenza di come possa essere proficuamente **sfruttato il “Dividendo Digitale”** attraverso l'introduzione di servizi innovativi in grado di attirare l'interesse e quindi di fare mercato.

La risposta a questa domanda consentirà di capire se effettivamente si potrà costruire un'offerta interessante e quanti nuovi programmi (per esempio, di servizio) serviranno. Solo in tal modo si sarà in grado di **creare una vera pianificazione della transizione**.

Per concludere, infine, occorre tener presente che le considerazioni di tipo economico, essenziali nelle valutazioni concrete, sono al momento di difficile sintesi, perché non sono attualmente disponibili sufficienti dati certi. Tuttavia, si può trovare un minimo di elementi di riferimento nell'apposito riquadro.

Acronimi

T-DAB: Terrestrial-Digital Audio Broadcasting
T-DMB: Terrestrial-Digital Multimedia Broadcasting
DAB-IP: Digital Audio Broadcasting - Internet Protocol
DAB+: Digital Audio Broadcasting Plus o Version 2
DRM: Digital Radio Mondiale (standard europeo per OC, OM ed OL)
DRM+: Digital Radio Mondiale (standard allo studio per FM)
DVB-H: Digital Video Broadcasting- Handheld
DVB-SH: Digital Video Broadcasting- Satellite Handheld (standard di 2a generazione DVB-H)
ESDR: ETSI Standard for Digital Radio (standard per radiofonia digitale satellitare)

Bibliografia

- [1] Approvazione del programma per lo sviluppo in Italia della radiodiffusione sonora in tecnica di-

gitale. Decreto 14 novembre 2001 del Ministero delle Comunicazioni, *Gazzetta Ufficiale*, n. 291, 15 dicembre 2001.

- [2] DAB: ETSI ETS 300 401 Radio broadcasting systems; Digital Audio Broadcasting (DAB) to mobile, portable and fixed receivers (recepita dall'ITU-R nella Raccomandazione BS.1114).
 REC. ITU-R BS.1114-5 Systems for terrestrial digital sound broadcasting to vehicular, portable and fixed receivers in the frequency range 30-3 000 MHz.
- [3] DMB: ETSI TS 102 428 Digital Audio Broadcasting (DAB); DMB video service; User Application Specification.
 S-DMB: ITU-R BO 1130-4 'System E' Systems for digital satellite broadcasting to vehicular, portable and fixed receivers in the bands allocated to BSS (sound) in the frequency range 1400-2700 MHz.
- [4] DVB-H: ETSI EN 302 304 (DVB); Transmission System for Handheld Terminals (DVB-H).
- [5] DRM: ETSI ES 201 980 Digital Radio Mondiale (DRM) System Specification.
 IEC PAS 62272-1 Digital Radio Mondiale (DRM) - Part 1: System specification (published in 2002).
 Rec. ITU-R BS.1514 (04/01) System for digital sound broadcasting in the bands below 30 MHz.
- [6] <http://www.drm.org/system/universalstandard.php>
- [7] Approvazione del regolamento recante la disciplina della fase di avvio delle trasmissioni radiofoniche terrestri in tecnica digitale. Autorità per le Garanzie nelle Comunicazioni, Deliberazione 9 marzo 2005 (Deliberazione n. 149/05/CONS). *Gazzetta Ufficiale*, n.69, 24 marzo 2005.
- [8] Report to the 61st DVB-TM meeting. (Author: David Crawford, Rapporteur) 29.06.2005.
- [9] EBU Tech review: MPEG-4 HE-AAC v2- audio coding for today's digital media world (2006) http://www.ebu.ch/en/technical/trev/trev_305-moser.pdf
- [10] An MPEG4 White Paper: Understanding MPEG-4: Technologies, Advantages, and Markets <http://www.m4if.org/public/documents/vault/MPEG4WhitePaperV2a.zip>

SALVATORE MORELLO, laurea in Ingegneria Elettronica all'Università di Napoli nel 1965. Corsi IRI post-universitari di Management e Business Administration. Vicedirettore ICT RAI fino al 1998, prima Dirigente Impianti e Tecniche di Produzione dell'Audio, radiofonico e televisivo, poi Direttore Telecomunicazioni. Dal 1998, Consulente libero professionista su Reti di Telecomunicazioni aziendali e relativi scenari di business. Docente in Corsi di Formazione e di Aggiornamento aziendali e nella P.A. Autore di pubblicazioni per riviste specialistiche e relatore in convegni pubblici e privati. È stato Vice Presidente dell'AES (Audio Engineering Society) per l'Italia.
 E-mail: salvatore.morello@tiscali.it

GUIDO VANNUCCHI, laurea in Ingegneria Industriale all'Università di Bologna nel 1958, "Master Science EE" alla Stanford University nel 1963, Libera Docenza in Comunicazioni Elettriche nel 1971. Direttore Generale Telettra dal 1983 al 1990, "Senior Consultant" di Olivetti Telemidia, Vice Direttore Generale della RAI dal 1996 al 1998. Attualmente è Docente al Politecnico di Milano di "Architetture per reti e sistemi multi-servizio", membro ordinario del Consiglio Superiore delle Comunicazioni e Consigliere di Amministrazione dell'Agenzia Spaziale italiana. Laurea "ad honorem" in Ingegneria delle Telecomunicazioni, conferita dall'Università di Padova nel 1998, per i contributi scientifici e manageriali nel campo della trasmissione dei segnali.
 E-mail: guido.vannucchi@fastwebnet.it