

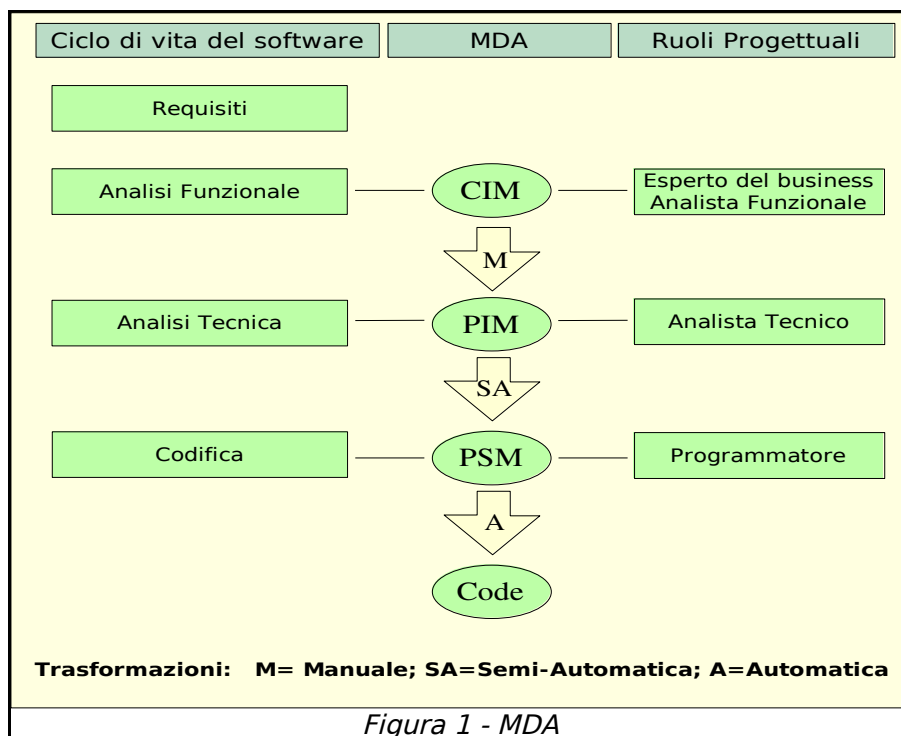
1. Un approccio Model Driven Architecture allo sviluppo di Processi di Business in un contesto SOA

In uno scenario di mercato in cui le tecnologie ed i requisiti funzionali dei sistemi software cambiano continuamente, le aziende sono alla ricerca di soluzioni per esporre in modo chiaro i loro processi di business verso utenti interni ed esterni alla loro struttura. OMG, che molto sta facendo per standardizzare e mettere a punto modelli e processi per migliorare il lavoro degli sviluppatori di sistemi, propone il Model Driven Architecture (MDA) per affrontare in modo moderno il ciclo di sviluppo del software. Non è un singolo strumento, uno standard oppure un processo, ma è piuttosto un insieme di specifiche, regole e standard che abilitano alla produzione automatica di applicazioni, utilizzando i modelli come punto di partenza. UML non è più solo un blue-print, ma modella elementi eseguibili costitutivi di un sistema software. Nel corso di questo articolo si presenta una possibile e concreta interpretazione dell'approccio MDA per lo sviluppo di servizi di business con un'architettura SOA, un connubio di meta-tecnologie che produce come effetto secondario quello di avere maggiore qualità ed efficienza nella gestione dell'intero ciclo di sviluppo del software.

1.1 Parte I - Impostazione Architeturale

Model Driven Architecture. Il livello di astrazione delle piattaforme con cui si opera oggi permette di sviluppare sistemi software in cui si tratta ad esempio con Java EJB, con interfacce CORBA, con Message Oriented Middleware, è possibile ad esempio collegarsi a sorgenti di dati distribuite sulla rete utilizzando JDBC oppure Entity EJB, scambiare dati in formato XML, senza la necessità di accedere a singoli file con strutture dati proprietarie. Ma nonostante ciò, il coesistere di tecnologie eterogenee, l'integrazione di sistemi legacy, la diversa forma di produzione e rappresentazione dei dati, obbligano a disperdere un'enorme quantità di energia lontano da quello che è il vero DNA dei sistemi informativi, il modello funzionale. Una via per dominare la complessità dello sviluppo del software e per applicare le tecnologie di modellazione come fulcro dell'intero processo, è il Model Driven Architecture (MDA) proposto da OMG. In una definizione molto sintetica e pragmatica MDA fornisce gli standard ed i processi con cui affrontare i passi che da un modello di sistema conducono direttamente alla codifica dello stesso, in modo indipendente dal linguaggio, dal middleware e dai prodotti di mercato.

OMG - Object Management Group - è un'organizzazione senza fini di lucro, con aziende affiliate in tutto il mondo, che si è costituita con lo scopo di stabilire delle regole per gestire e regolamentare la complessità nell'IT, diminuire i costi e soprattutto facilitare l'interoperabilità tra i sistemi (il primo risultato di OMG in questo senso è stata la creazione di CORBA). OMG mira a guidare l'evoluzione delle tecniche di sviluppo ed integrazione definendo standard fondamentali, come è per UML, XMI, CWM e CORBA, raccogliendo esperienze e linee guida da tutte le aziende che contribuiscono a tale evoluzione, e mantenendo la posizione più neutra possibile rispetto ai tool vendor.

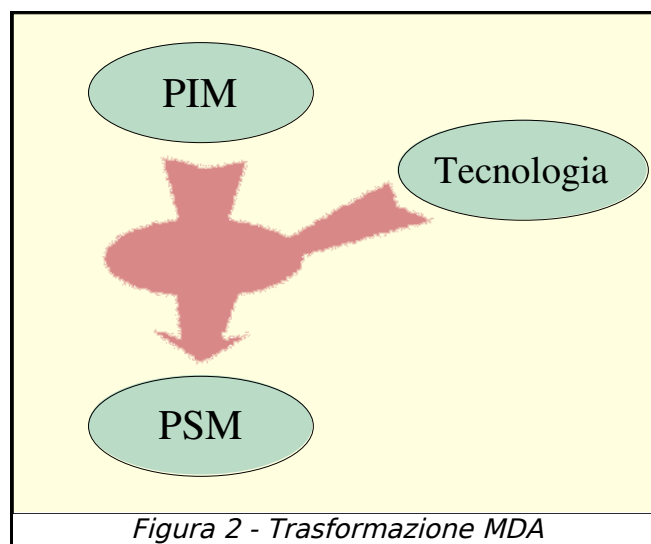


Nella figura 1 è sintetizzato il rapporto che esiste tra il ciclo di sviluppo del software, MDA ed i ruoli progettuali coinvolti nella realizzazione dei sistemi. E' possibile immaginare i modelli MDA come in una pila, al vertice della quale si rappresenta il "business", mentre la piattaforma tecnologia ne è alla base. Gli esperti del dominio di business - l'area finanziaria, le vendite, il marketing, la produzione - lavorano in alto, nello spazio della modellazione, dove hanno la possibilità di formalizzare le caratteristiche di ogni processo.

Tra le possibili scelte alternative, UML dalla versione 2.0, è un linguaggio adatto ad essere utilizzato per modellare i processi di business e dispone di meccanismi sufficientemente flessibili per adattarsi alle caratteristiche specifiche dei diversi domini funzionali. UML è ormai molto diffuso e conosciuto, e sul mercato si trovano diversi validi strumenti che supportano le attività di modellazione, mettendo a disposizione dell'utente sofisticate interfacce grafiche, per rendere più rapida la formalizzazione dei modelli e la loro comprensione, e per supportare le attività di collaborazione del team degli esperti di dominio.

Il primo prodotto del ciclo di sviluppo è, nella terminologia MDA, il modello CIM (Computational Independent Model) che rappresenta i singoli processi di business e le loro interdipendenze funzionali, senza fare riferimento ad alcun tipo di tecnica implementativa. Il ruolo del CIM in MDA è quello di unificare il lavoro degli esperti del business e degli analisti funzionali, i primi conoscono a fondo i processi ed i requisiti, mentre gli ultimi traducono tali indicazioni in un modello di partenza per impostare il disegno e la costruzione della soluzione software. Il CIM è definito anche *modello di dominio* ed utilizza nel suo dettaglio il vocabolario utilizzato dagli esperti del business.

Per fare un esempio si può immaginare la modellazione del processo di business relativo alla vendita di un biglietto aereo da parte di un'agenzia di viaggi, in questo scenario il modello CIM tratta di ruoli come "cliente" e "commesso di agenzia", specifica che dopo aver raccolto le esigenze del cliente, il commesso deve consultare il sistema di pubblicazione delle offerte speciali per sottoporle all'attenzione del cliente stesso, descrive i passi da completare per arrivare ad individuare il biglietto giusto, descrive l'utilizzo dei sistemi remoti di prenotazione e dei sistemi locali di contabilità, per emettere e consegnare il documento di prenotazione alla fine del processo. Nel modello CIM di questo esempio non si descrive il business utilizzando concetti come metodo, operazione, parametro, attributo, classe; può essere anche tralasciato il dettaglio che specifica se le offerte speciali devono essere consultate sul video di un terminale oppure su di un report cartaceo stampato quotidianamente. Da questo primo passo di modellazione, incipit di tutto il processo, è possibile procedere specificando in modo più dettagliato il sistema nel modello PIM (Platform Independent Model). Questo secondo prodotto, a cura dell'analista tecnico, è il completamento del CIM ed è la rappresentazione computabile del sistema, ma da un punto di vista indipendente dalla piattaforma; in questo modello ad esempio non si descrive qual è la tecnologia con la quale un servizio viene esposto. Nel PIM sono definite tutte le funzionalità di business ed i comportamenti dell'applicazione che dovrà essere realizzata, in modo indipendente dal dettaglio dell'implementazione. Per riprendere l'esempio dell'agenzia di viaggi, il modello PIM deve procedere a scomporre l'intero sistema in componenti software e stabilisce come questi devono interagire, come il componente che controlla il sistema remoto di prenotazione comunica al sistema locale di contabilizzazione il valore totale del biglietto da fatturare al cliente, specifica se i servizi sono state-less o state-full, ma senza fare riferimento al fatto che il componente di prenotazione possa essere implementato come un Session EJB, che attraverso un TP monitor accede ad una procedura in Cobol. Man mano che si procede verso la base della pila di modelli suggerita nella figura 1, ci si avvicina alla parte infrastrutturale, dove MDA finalizza l'aspetto chiave di tutto il processo, la trasformazione del modello PIM in un modello specifico per una piattaforma tecnologica: il PSM (Platform Specific Model).

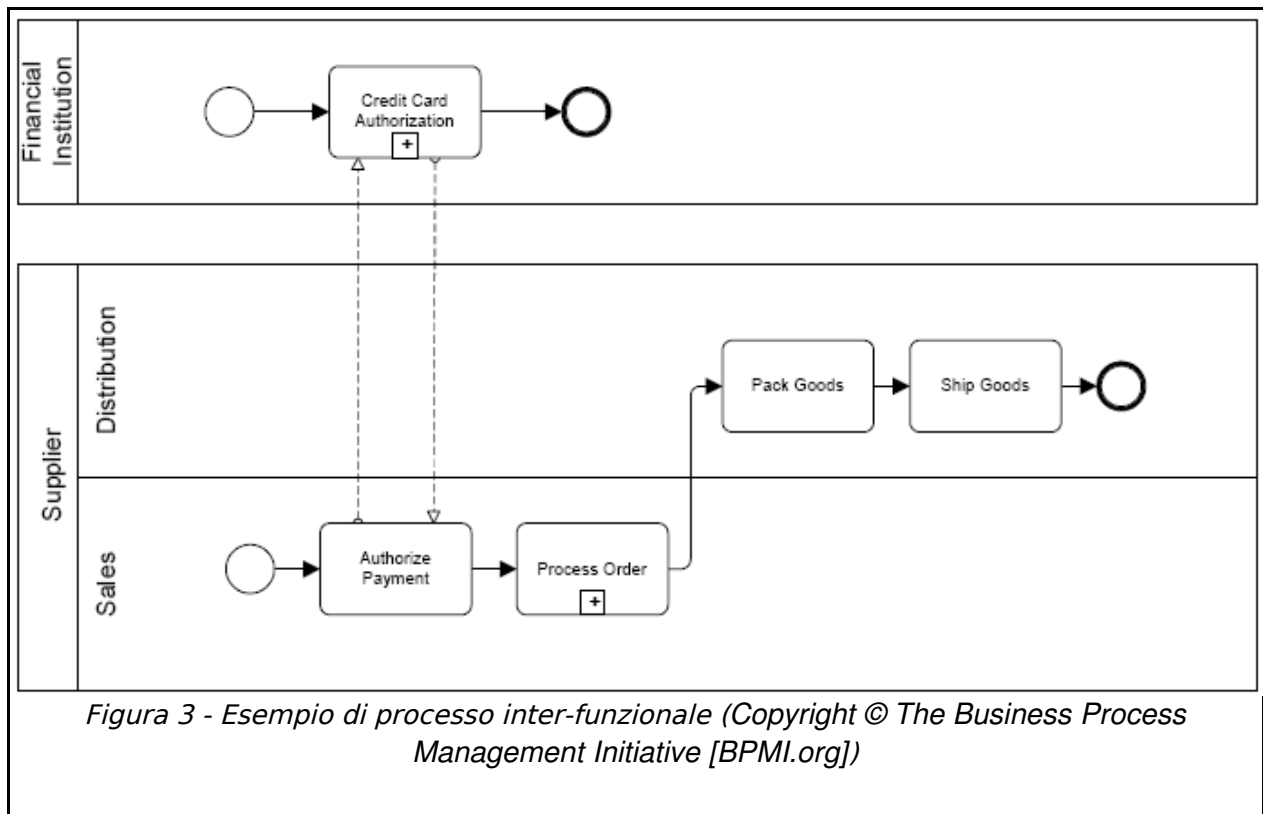


La figura 2 indica che il processo di trasformazione è guidato da istruzioni che necessariamente sono legate ad un'infrastruttura tecnologica, il modello PSM prodotto è un modello computabile, o meglio a questo punto è il *modello eseguibile* su una specifica piattaforma target per il sistema. Il modello PSM è a tutti gli effetti l'applicazione finale, ma è formalizzato ancora attraverso un linguaggio di modellazione (UML) invece che in uno specifico linguaggio di programmazione. E' possibile quindi procedere a generare automaticamente da PSM per .NET, J2EE, CORBA o WebServices un'intera e completa applicazione.

Per riprendere l'esempio relativo all'agenzia di viaggi si potrebbe ipotizzare di tradurre automaticamente il modello PSM in Java EJB, per distribuire i componenti così ottenuti su application server J2EE. Ad esempio il componente locale che esegue le chiamate al sistema contabile potrebbe essere trasformato in un Session EJB che chiamando i metodi remoti del Session EJB del sistema di prenotazione, recupera i dati da utilizzare nell'emissione della fattura al cliente.

Il passaggio intermedio attraverso il modello PSM non è indispensabile, ma dà l'opportunità agli architetti e agli specialisti della piattaforma tecnologica di aggiungere altre specifiche trasformazioni per rispondere ai requisiti extra-funzionali come la scalabilità e la sicurezza delle applicazioni. *Alla fine del processo MDA, tutto ciò che ad alto livello era stato codificato nel modello concettuale del sistema, diventa un'applicazione pronta per essere distribuita ed utilizzata su una piattaforma specifica.* Per approfondire la conoscenza di MDA, si rimanda alla consultazione del sito di OMG (www.omg.org/mda).

Modellazione dei Processi di Business. I tipi di processi che si svolgono nel contesto di un'azienda sono molteplici, si possono individuare ad esempio i Processi Cliente, quelli cioè che producono un risultato che è oggetto della relazione con i clienti dell'impresa, oppure i Processi Amministrativi, che governano attività vitali per la gestione. In una prima analisi i processi si possono classificare in Processi Funzionali, che sono tipicamente specialistici e che si sviluppano principalmente all'interno di una delle aree o divisioni aziendali, e in contrapposizione ai primi, in Processi Inter-funzionali, che sono invece trasversali rispetto all'organizzazione in settori aziendali. Un esempio di processo inter-funzionale è modellato nella figura 3, dove le diverse aree contribuiscono al completamento di un processo che si snoda partendo da un input della divisione Vendite fino ad interessare la divisione Distribuzione; con il protrarsi del processo l'azienda vede allungarsi i tempi, aumentare i costi e necessariamente mette alla prova la sua capacità di governance. Nell'ambito della Business Process Management Initiative (BPMI), organismo confluito in OMG - www.bpmi.org - si stanno definendo standard che possano supportare l'intero ciclo di vita del Business Process Management, il disegno, la distribuzione, la manutenzione, l'ottimizzazione e molto altro ancora. In questo contesto è stato definito il Business Process Modeling Notation (BPMN) - www.bpmn.org, si tratta di una notazione grafica in grado di rappresentare tutti i passi di un processo di business, per dare alle organizzazioni l'abilità di comunicare queste procedure in modo standard e comprensibile.



Anche se BPMI ha rilasciato per il momento la versione 1.0 , già in questo momento il mercato è in grado di fornire strumenti utili alla modellazione dei processi con BPMN, e ancora più importante è che il sottostante meta-modello è compatibile con lo standard MOF di OMG: il meta linguaggio con cui è stato scritto UML. BPMN è quindi la notazione più adeguata oggi da utilizzare nella modellazione CIM dei processi di business, è un formalismo che può essere condiviso sia dagli esperti del business che dagli analisti funzionali. Per riprendere l'esempio relativo all'agenzia di viaggi, la parte di modello CIM relativa al processo di acquisto del biglietto aereo potrebbe essere trascritto in notazione BPMN dagli analisti funzionali, con il risultato di ottenere nel contempo un modello leggibile e formale, adatto ad essere implementato quasi direttamente e a divenire quindi un modello PIM. Si potrebbe obiettare che il BPM non è una novità nel panorama del software engineering, certamente questo è vero, progetti sono stati fatti già nei primi anni 90, ma in questo caso si parla di modelli computabili che sono direttamente eseguibili e che rappresentano l'attuale implementazione del sistema, non vi sono passaggi intermedi.

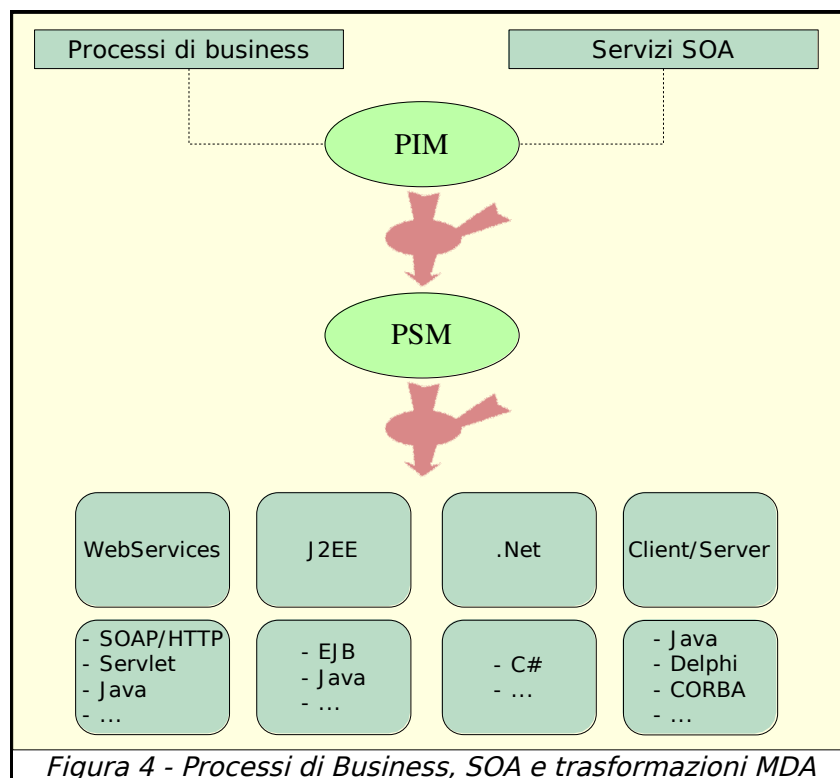
Ultimo ma non meno importante è il fatto che sia per la parte di notazione, (BPMN) rappresentazione (Business Process Definition Metamodel) ed esecuzione (BP Execution Language, XML Process Definition Language¹) si tratta di standard aperti che garantiscono il riuso e la scalabilità.

1 Definito dalla Workflow Management Coalition

1.2 Parte II - Sviluppo e Caso d'Uso

Processi di Business, MDA e SOA. Il termine SOA fa riferimento ad un'architettura software che definisce l'uso dei servizi come strategia per soddisfare i requisiti di un sistema. In un ambiente SOA, i nodi di una rete sono risorse disponibili come *servizi* o *Service Building Block (SBB)*, che secondo SOA sono idealmente funzioni di business indipendenti, che accettano una o più richieste e restituiscono altrettante risposte secondo uno standard stabilito. I singoli servizi non devono dipendere dallo stato di altri servizi, o dalla tecnologia utilizzata per la loro implementazione. Nella terminologia SOA, *provider* è la funzione che implementa un servizio in risposta ad una richiesta da parte di un *consumer*, ed il meccanismo dinamico con cui il consumer stabilisce una relazione con i servizi del provider si definisce *binding*. SOA prevede inoltre che i servizi disponibili possano essere elencati in un *registry*, al fine di descriverne le funzionalità e l'indirizzo necessario per il binding, su tale registry si basa il servizio di *discovery* che SOA garantisce per consentire la scoperta e l'utilizzo dei servizi stessi. In una tale architettura, la realizzazione di un sistema che gestisce processi di business è fondata sul meccanismo di organizzazione dei servizi in precise sequenze, sulla possibilità di invocare il medesimo servizio più volte da processi diversi, sulla logica necessaria per controllare e comporre le sequenze di servizi così ottenute: in terminologia SOA si definisce *orchestration*. Si può sintetizzare il lavoro di impostazione del progetto dei servizi di business su architettura SOA identificando i passi fondamentali da compiere:

- Mapping dei servizi legacy
- Definizione dei nuovi servizi
- Integrazione dei servizi nei processi



La fase di mappatura delle applicazioni legacy consiste nella definizione di Service Building Block in grado esporre le funzioni proprie dei sistemi legacy come servizi SOA, ed in alcuni casi potrebbe essere richiesta l'integrazione di diversi sotto-sistemi legacy in un unico servizio. La definizione dei nuovi processi e l'integrazione dei servizi nei processi dovrebbero invece seguire il ciclo di sviluppo MDA, fino ad arrivare ad un modello PSM da trasformare automaticamente per il deploy su una piattaforma Java, .Net o altro. Riprendendo l'esempio dell'agenzia di viaggi sarebbe necessario creare, tra gli altri, un servizio per fornire l'elenco aggiornato delle offerte speciali da proporre al cliente, un servizio per le funzioni di consultazione dei posti viaggio disponibili, uno per le prenotazioni ed uno per le cancellazioni. Il modello di business imporrebbe poi di definire un processo in cui il servizio di prenotazione sarebbe invocato per ottenere il valore dell'acquisto, subito prima di chiamare il servizio contabile per l'emissione della fattura. Inoltre l'applicazione di orchestrazione dei servizi dovrebbe prevedere anche l'interazione con l'utente e quindi un'interfaccia grafica come parte integrante del sistema che potrebbe essere generata automaticamente con trasformazioni MDA a partire dai modelli.

Nella figura 4 è rappresentato, in modo molto semplificato, il processo di trasformazione MDA, mettendo in evidenza come nel processo stesso si parta sempre dai requisiti e dalle funzionalità di un sistema per arrivare ad un modello PSM, che è poi possibile trasformare automaticamente in una o più piattaforme, di cui sono stati riportati qui alcuni esempi.

Un altro aspetto importante di questo approccio è che la modellazione dell'intero sistema è fatta in modalità top-down, partendo cioè dai processi di alto livello per procedere poi via via ad individuare componenti di più basso livello fino ad incontrare le interfacce dei servizi, comprendendo anche quelli che incapsulano ed integrano le applicazioni legacy. Il processo top-down evita ai progettisti l'errore di costruire le interfacce dei servizi partendo dalla struttura dei servizi stessi. Si potrebbe sfruttare ad esempio una funzione comunemente disponibile negli ambienti di sviluppo software, l'automatismo di generazione delle interfacce WSDL per integrare applicazioni in un contesto SOA basato su WebServices (WSDL wizard), ma tale generazione spesso lega indissolubilmente l'interfaccia del servizio alla struttura interna dei dati che scambia, e questo *legame* diventa in seguito un potenziale ostacolo per l'interoperabilità di servizi che risiedono su ambienti tra loro eterogenei. E' preferibile che il disegno delle interfacce sia guidato dal modello di business senza la contaminazione di aspetti tecnologici, il cui scopo è coordinare e armonizzare l'esecuzione di tutti i servizi di base, indipendentemente dalla piattaforma su cui saranno realizzati.

Coopservice e Pant@: un progetto MDA e SOA. Coopservice è un'azienda italiana con base a Reggio Emilia, che opera nel settore del Facility Management: ha oltre 10.000 dipendenti, €370 milioni di fatturato annuo, €50 milioni in beni consolidati. Nell'ambito del Facility Management, Coopservice opera con servizi di pulizia, di smaltimento dei rifiuti e nei servizi ambientali, nella manutenzione, nella gestione degli edifici e dei beni immobiliari in genere, nella sicurezza e nel catering. Per riuscire a fornire i suoi servizi integrati, Coopservice si avvale di una rete di partner costituita da altre piccole e medie imprese italiane.

Questo significa governare un servizio completo di gestione dei beni e dei servizi, che spesso comprende servizi di base come la pulizia, la manutenzione degli immobili, la sicurezza e il servizio mensa, fino all'amministrazione dei magazzini. In media, circa venti altre piccole e medie imprese sono impegnate nella fornitura di un così completo pacchetto di servizi. Il tipico progetto di Facility Management di Coopservice dura un arco di tempo pari a 3-4 anni, con un valore economico medio pari a circa €80 milioni.

Coopservice ha la necessità di coordinare le proprie attività con clienti e fornitori, e intende in futuro gestire l'intera catena delle forniture con un unico sistema di classe enterprise che possa presentare due diversi livelli di funzionalità, un sistema personalizzato di Enterprise Resource Planning (ERP) che possa supportare i processi interni ed esterni, ed una architettura di collaborazione Business to Business (B2B) che possa soddisfare l'esigenza di collegarla alle altre imprese coinvolte nei servizi. Il progetto che oggi sta nascendo in Coopservice prende il nome di Pant@. I modelli di business ed i processi collegati assumono per questo tipo di azienda un'importanza strategica, la possibilità di non disperderli in una pletera di sistemi isolati è fondamentale per eliminare la gestione superflua dei documenti cartacei, per pianificare ed erogare servizi in collaborazione con i partner, per ricevere i rapporti di attività e per implementare un sistema di governance centralizzato.

I dati tecnici del progetto rivelano un panorama complesso da governare: problematiche tipiche dell' Enterprise Application Integration (EAI), 45 sistemi legacy da integrare, tre diverse tecnologie di base da conciliare (IBM AS/400, .Net, Java EJB) 40 sedi in Italia e 4 in Europa, con attività dei partner da coordinare utilizzando il telefono e il fax. Il progetto Pant@ è partito con l'obiettivo di realizzare un sistema integrato basato su SOA, che implementa EJB e WebServices, con lo scopo di supportare una rete dinamica per la gestione del B2B, che nel lungo termine veda la dismissione di gran parte dei sistemi legacy.

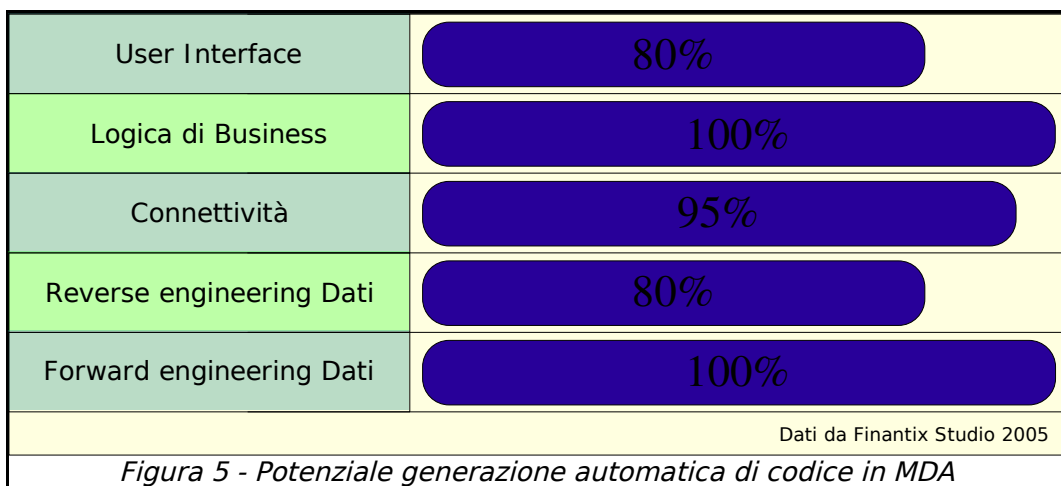
I tempi dedicati all'analisi funzionale del progetto Pant@ saranno circa il 20% più lunghi se paragonati a quelli del classico software engineering, con lo scopo di produrre modelli PIM automaticamente trasformabili, ma con un risparmio di circa l'80% nella codifica dell'applicazione per le specifiche piattaforme. Sui 16,5 anni uomo valutati per lo sviluppo dell'intero sistema seguendo l'approccio classico, si arriverà ad impiegarne solo 7,5 seguendo il processo MDA ed utilizzando i tool di sviluppo automatico, con un risparmio che si aggira intorno al 53% del tempo di sviluppo globale di sviluppo tradizionale previsto. Il tempo di rilascio potrà essere ridotto del 28%, con un obiettivo teorico di riduzione che può arrivare al 40%. Tradotto in termini economici significa un risparmio totale di circa €150.000 solo per la parte di sviluppo, non comprendendo la parte più consistente della manutenzione funzionale che viene sensibilmente diminuita. Nella fase di manutenzione ed evoluzione tutti i meccanismi di trasformazione perfezionati potranno essere riapplicati una volta fatti i necessari interventi sui modelli concettuali dei processi di business: il tutto a run-time, con tempi di update di alcune ore contro circa una settimana con l'approccio classico evitando, nel caso peggiore codifica, test e deploy. L'approccio MDA per realizzare un sistema a componenti nel contesto di un'architettura SOA ha trovato quindi nel progetto Pant@ una sua applicazione pratica, convincendo da subito il management aziendale, dimostrando la sua sostenibilità ed efficacia.

Il progetto Pant@ è stato scelto come uno dei MDA case history che verrà pubblicato in un libro di prossima pubblicazione edito da OMG Press.

Conclusioni.

Come l'architettura MDA si propone di alzare il livello su cui basare un processo di sviluppo del software, ponendo al centro dell'attenzione i modelli e non il codice delle applicazioni, anche SOA opera una proposta nella stessa direzione, e consente agli analisti di manipolare Service Building Blocks, senza affrontare la complessità di procedure di più basso livello, e supporta interfacce definite in modo indipendente dalla piattaforma su cui SOA si implementa. L'approccio proposto si presenta quindi ideale per sfruttare le caratteristiche complementari di MDA e SOA, al fine di ottenere efficienza ed una riduzione dei costi nello sviluppo di sistemi di orchestrazione di processi (si veda la figura 5).

Il principio guida nella realizzazione dei servizi di business è il business stesso, non la tecnologia sottesa, non i sistemi multi-canale con cui le informazioni e gli eventi possono essere notificati, non le innovative soluzioni hardware e software che possono essere utilizzate per il delivery. L'approccio MDA restituisce ai "modelli" l'importanza che essi devono avere nel processo di sviluppo, perché i modelli forniscono una rappresentazione ad alto livello del sistema e la loro manipolazione diretta garantisce il miglior adattamento possibile del software al cambiamento dei processi di business, che come già evidenziato è uno dei requisiti più pressanti da soddisfare.



E' da sottolineare ancora una volta l'importanza che ricopre la standardizzazione a supporto di queste architetture e di questi processi di sviluppo, e quindi nel caso concreto l'importanza del ruolo di OMG. Nel passato, in più momenti storici relativi all'evoluzione dell'ingegneria del software, sono stati proposti nel mercato strumenti e ambienti integrati di lavoro che supportavano processi completi per lo sviluppo di sistemi software, e molte di queste proposte erano ben congegnate e obiettivamente efficaci nel raggiungere gli obiettivi che si proponevano. Tra gli altri, il motivo per cui questi prodotti non hanno riscontro nell'attuale mercato è da ricercare nel fatto che essi proponevano standard di riferimento proprietari, senza la possibilità alcuna di interoperare con ambienti diversi e con standard alternativi o complementari: se non altro era un forzò nel quale l'utente era lasciato da solo.

Nonostante la strada sia stata aperta e sia già percorribile, alla comunità degli “sviluppatori di modelli” servono strumenti ancora più completi ed efficaci, ambienti di sviluppo integrati estesi a tutte le fase del cicli di vita, soluzioni per il testing e il debugging al livello dei modelli PIM sempre più evoluti, trasformazioni più complete e potenti per garantire una ulteriore capacità di gestione della complessità dei progetti enterprise.

Con il diffondersi degli approcci MDA nel futuro ipotizziamo ed auspichiamo la nascita di un mercato di modelli PIM da utilizzare come plugs-in in un sistema MDA, che darebbe un ulteriore impulso al riuso e alla valorizzazione delle qualità di questo approccio.

Soluta.net. Soluta.net è costituita da un team di professionisti dell'IT che hanno esperienza, a livello mondiale, di Sviluppo Basato sui Componenti e di Architetture Enterprise sin dal 1997.

- Il team di Soluta.net è impegnato nella realizzazione del progetto Pant@, commissionato da Coopservice (www.coopservice.it). Soluta.net è ideatrice e promotrice del progetto open source JunoMDA. (junomda.sourceforge.net/), un'implementazione che segue il pattern MDA per realizzare applicazioni Web Based partendo da un modello PIM, progetto che inizia a raccogliere interesse concreto di sviluppo a livello internazionale.
- Soluta.net ricopre il ruolo di Chief Architect nel progetto Digital Business Ecosystem (<http://www.digitalecosystem.org>) finanziato dalla Commissione Europea (€14Mil) del 6FP, uno dei più sofisticati ed avanzati sistemi sugli ecosistemi digitali, completamente basato su MDA. Il DBE è stato invitato al “World Summit of the Information Society” in Tunisia il 15-18 Novembre (<http://www.itu.int/wsis/>).
- Soluta.net parteciperà al prossimo OMG Technical Meeting di Burlingame, California, USA, dal 5 al 9 Dicembre 2005 (www.omg.org/registration/registration-info.htm). Il Dr. Ferronato terrà uno speech su “MDA and Agility in the development of ERP Facility Management project” in cui presenterà il caso di Coopservice, discutendo gli obiettivi, le caratteristiche tecniche e i benefit dell'applicazione di MDA in un progetto ERP-EAI-WebServices.
- Soluta.net è stata invitata come speaker al secondo workshop "SOA, MDA and WebServices: Integrating the Enterprise, and Beyond" a Washington, DC, dal 27 11 30 Marzo 2006.

Paolo Miotti, si è laureato in Scienze dell'Informazione presso l'Università Statale di Milano, attualmente ricopre la posizione di Chief Consultant in Soluta.net. E' stato coinvolto per oltre 15 anni nel disegno e nello sviluppo di innovativi progetti di sistemi distribuiti. Durante la sua carriera ha applicato OOT, CBD e modellazione UML. Ha tracciato le linee guida per portare allo “stato dell'arte” i progetti di molte tra le prime 100 imprese italiane in diversi settori tra cui il bancario, l'assicurativo, il finanziario e l'immobiliare.

Pierfranco Ferronato, Chief Architect di Soluta.net, ha oltre 15 anni di esperienza in ogni aspetto relativo allo sviluppo di sistemi distribuiti ed è riconosciuto a livello internazionale come esperto in large-scale architecture ed object-oriented/component development. Il Dr. Ferronato ha condotto dal punto vista tecnico e architeturale diversi progetti europei utilizzando avanzate tecnologie legate ad Internet, sviluppo basato sui componenti, MDA, WebServices e tecnologie wireless, in diversi domini di business tra cui telecomunicazioni, farmaceutico, sanità, CRM, ERP, EAI e turismo.

BIBLIOGRAFIA

- *Milano, "Secondo Forum AICA sulle Tecniche Agili per il Software Process Improvement", 15 Novembre 2005, "MDA e Agilità nello sviluppo del software" (www.soluta.net/events.php)*
- *Athens, MDA QSP Seminar, 19 Febbraio 2005, "MDA supporting a two face Architecture (ERP and B2B)" (www.soluta.net/docs/20050411ferronatoOMG_MDA_QSP.pdf)*
- *"MDA Guide Version 1.0.1", Jishnu Mukerji, Joaquin Miller (www.omg.org/docs/omg/03-06-01.pdf)*
- *"Business Process Modeling Notation (BPMN)", Versione 1.0, 3 Maggio 2004 (www.bpmn.org)*

- end of document -