

1. MDA

1.1 Scopo

L'approccio Model Driven Architecture (MDA) intende far sì che le applicazioni software vengano generate attraverso un processo di trasformazione di modelli.

MDA, tramite formalismi e standard aperti non proprietari, consente che i modelli funzionali, scritti in UML da analisti di business, senza alcuna competenza sulla piattaforma di sviluppo ed esecuzione, possano creare un'applicazione completa, in tutto il ciclo di vita del software. Ciò evitando di preoccuparsi di quale particolare piattaforma tecnologica realizzerà questi requisiti.

inoltre l'applicazione è specificata e implementata in maniera indipendente dalla piattaforma. I modelli funzionali non sono più mescolati con le specifiche tecnologiche: non è più necessario coinvolgere un programmatore Java per aggiungere ad esempio un piano tariffario, basterà cambiare un modello senza dover immergersi in linee di codice.

1.2 Approccio

MDA è un approccio, un processo e dei formalismi che, incrociando il modello funzionale, con il modello dell'architettura supporta la generazione di un'intera applicazione, o meglio, tutto quello che è stato specificato nel modello stesso dell'architettura. Di conseguenza, le due viste funzionale e strutturale sono indipendenti: se si cambia la funzionalità non è necessario coinvolgere nessuno sviluppatore, se si cambia l'architettura informatica, non si debbono conoscere le funzionalità sottese.

Il modello funzionale viene incrociato con lo specifico modello dell'architettura da un motore di generazione che crea l'applicazione con i file e le strutture che sono necessari per la distribuzione e l'esecuzione.

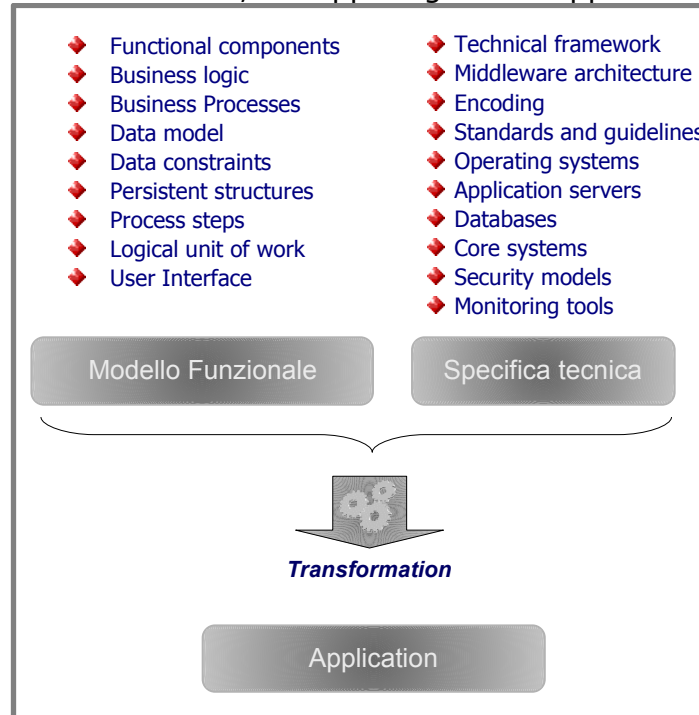
1.3 PIM & PSM

MDA identifica l'esistenza di due tipi di modelli: indipendente dalla piattaforma (Platform Independent Model, PIM) e specifico per la piattaforma (Platform Specific Model, PSM). Sia PIM che PSM sono descritti nel linguaggio UML, che è oggi l'unico linguaggio per metamodelli usato dai produttori di strumenti di modellazione, permettendo così di usare un qualunque tool di modellazione e per questo di salvaguardare gli investimenti in formazione e strumenti

1.4 Elementi del Tool e Skills

Una soluzione MDA è generalmente costituita da tre elementi:

1. un ambiente visuale per creare modelli, simile ad un modellatore grafico UML;
2. uno o più modelli dell'architettura, spesso chiamati "Cartridge", che descrivono la tecnologia implementativa attraverso uno stile architetturale;
3. un generatore che prende in input i modelli prodotti dai due strumenti precedentemente elencati, li mappa e genera l'applicazione.



Relativamente alle risorse coinvolte, il programmatore MDA crea i modelli indipendenti dalla piattaforma con un editor UML, mentre il programmatore di piattaforma scrive i Cartridge.

1.5 Vantaggi

Uno degli obiettivi di MDA è quello di separare i requisiti funzionali e la loro modellazione analitica dalla tecnologia. Il modello funzionale, scevro da aspetti tecnologici, sta divenendo il vero capitale dell'azienda, assumendo un'importanza rilevante anche in prospettiva di interoperabilità tra applicazioni. Da questo obiettivo, derivano i seguenti vantaggi:

- Riutilizzo del modello funzionale, grazie all'indipendenza dai modelli tecnologici della piattaforma;
- Riutilizzo del modello tecnologico, grazie all'indipendenza dai modelli funzionali;
- Disaccoppiamento forte tra funzionalità e tecnologia (separation of concern): i due elementi sono fortemente disaccoppiati ed indipendenti;
- Lunga durata dell'applicazione sviluppata: la tecnologia o l'uso che ne viene fatto può cambiare o essere migliorato, ma una volta che un requisito

funzionale è stato specificato ed implementato, questo non viene influenzato;

- Migliore qualità del prodotto: le scelte tecnologiche sono ripetibili in modo trasparente;
- Elevata produttività, esperti di dominio (bancario, farmaceutili, aeronautico, sistemi embedded...) non debbono essere anche esperti di piattaforma (C, C++, SOAP, Java...);
- Upgrading e portabilità: se nasce una nuova piattaforma durante il ciclo di vita dell'applicazione è possibile definire una diversa regola di trasformazione e ri-generare l'applicazione su questa nuova piattaforma;
- La possibilità di effettuare test funzionali PRIMA che l'applicazione stessa venga generata, in questo modo di evidenziano in anticipo i bug;

1.6 Applicazioni nell'Industria e nella Ricerca Europea

Crescente l'adozione di tool MDA per lo sviluppo di applicazioni e l'integrazione di sistemi software in diversi settori dell'Industria, dal Finance (Wells Fargo), ai Trasporti (Austrian Railways, Lockheed Martin Aeronautics). Di rilievo anche l'attenzione della Ricerca Europea verso l'interoperabilità delle applicazioni sviluppate attraverso l'uso di MDA (in particolare il progetto europeo Digital Business Ecosystem),

2. Bibliografia e riferimenti

OMG MDA official web site <http://www.omg.org/mda>

OMG "MDA Guide", <http://www.omg.org/docs/omg/03-06-01.pdf>

Model Driven Architecture: Applying MDA to Enterprise Computing, by David S. Frankel, Wiley, ISBN 0471319201

MDA Explained: Model Driven Architecture - Practice & Promise Warmer, Kleppe & Bast Addison-Wesley, ISBN 032119442X

Naked Objects Architecture Series Richard Pawson, Robert Matthews & Dan Haywood www.theserverside.com, Mar - Apr 2004