UNIVERSITÀ DEGLI STUDI DI PADOVA
FACOLTÀ DI INGEGNERIA

Corso di Laurea in INGEGNERIA INFORMATICA

Strumenti di Reportistica per Business Intelligence

Attività lavorativa svolta presso IBM Ireland (2012)

Relatore: PROF. ANDREA PIETRACAPRINA
Dipartimento di Ingegneria dell'Informazione

Laureando: FRANCESCO BENETTI

A.A. 2012 - 2013

Indice

1. Introduzione

Lo scopo di questo documento è quello di introdurre il concetto di Business Intelligence ed i principali strumenti software che ne danno un'implementazione e che vengono oggi usati in diversi contesti aziendali, facendo leva sulle conoscenze acquisite durante la mia esperienza lavorativa nel team di Business Intelligence & Metrics di IBM Ireland.

Qui ho avuto l'occasione di lavorare sia in ruoli tecnici, come ad esempio nelle operazioni di manutenzione dei sistemi di reportistica in uso nel team, sia in ruoli più analitici, come la raccolta di requisiti per la generazione automatica di un insieme di rendiconti bancari.

A partire dal secondo capitolo, il testo cercherà di dare una definizione al termine Business Intelligence, nelle sue due declinazioni principali: tradizionale e operazionale.

Il terzo capitolo mira a descrivere la natura del team di BI&M di Dublino e i ruoli svolti dai suoi componenti, mentre il capitolo successivo offrirà una panoramica sulle varie tipologie di applicativi di Business Intelligence, con un approfondimento su Eclipse BIRT, il principale strumento software usato dal mio team.

Il quinto e sesto capitolo sono dedicati ai casi di studio, ovvero ai resoconti di due attività, significative per la quantità di nozioni apprese, su cui sono stato impegnato durante i primi mesi del mio lavoro in IBM.

1.1 L'Azienda

IBM è oggi una multinazionale di consulenza direzionale, servizi tecnologici ed outsourcing, a sua volta divisa in aree funzionali o subordinate.

- IBM Global Business Services (GBS): analizza il posizionamento competitivo delle aziende e sviluppa strategie di business e programmi di integrazione dei sistemi, erogando servizi di consulenza direzionale, organizzativa e consulenza manageriale.
- IBM Global Technology Services (GTS): ha l'obiettivo di aiutare i clienti ad implementare il disegno di architetture e infrastrutture tecnologiche, in particolare si occupa di studiare sia nuove applicazioni software che di impostare i programmi di manutenzione per applicativi già operativi.

In Irlanda, IBM opera per i maggiori gruppi finanziari ed industriali nazionali, incluse le maggiori banche e linee aeree del paese. Buona parte delle attività è inoltre focalizzata allo sviluppo delle aree di business afferenti al blocco dei paesi scandinavi.

I servizi di progettazione e sviluppo di applicativi gestionali ex-novo sono di norma affidati all'area Technology Services della società, di cui è parte il team di sviluppo per Business Intelligence & Metrics, sede della mia attività lavorativa.

1.2 Lo Stage

Sono stato inizialmente contattato da IBM tramite una agenzia per il lavoro irlandese, a cui avevo sottoposto il mio CV. Dopo un paio di colloqui telefonici ed una verifica delle competenze, a gennaio 2012 sono stato invitato a sottoscrivere un contratto di un anno con l'azienda, grazie al fatto che il team di Business Intelligence & Metrics valutava positivamente la mia esperienza di tre anni in Accenture (un'altra grossa multinazionale di consulenza informatica per le imprese), che mi aveva consentito di ottenere un certo grado di familiarità con i framework Java Spring e Hibernate, e le conoscenze su database non-relazionali come ad esempio MongoDB, acquisite da autodidatta su progetti personali.

Purtroppo, al momento del mio arrivo a Dublino nel team di BI&M mi è stato comunicato che il progetto su cui era previsto che lavorassi come sviluppatore Spring e Hibernate era stato sospeso a tempo indefinito per questioni finanziarie, ma che il team, carente nel suo complesso di competenze tecniche, avrebbe potuto sfruttare le mie capacità in altri ambiti.

Il mio compito all'interno del team dipendeva quindi dalle esigenze del momento, vertendo soprattutto sulle attività di manutenzione dei sistemi e lo sviluppo di semplici applicativi utili al team stesso, come ad esempio lo sviluppo di un tunnel SSH per la connessione a database remoti protetti, la creazione di un sistema di monitoraggio in tempo reale delle connessioni ai server in uso al team, la migrazione del sistema di reportistica interno su nuovo hardware o la raccolta di requisiti tecnici per la stesura di nuova reportistica.

1.3 Il Business Intelligence & Metrics Team Dublin

BI&M Dublin è l'acronimo usato all'interno della società per indicare il team irlandese che si occupa della generazione della reportistica per i clienti di IBM che necessitano di valutazioni complesse che vanno oltre la reportistica standard fornita nativamente dai software in uso dalla componente manageriale delle società clienti.

La reportistica di cui necessita il cliente può essere indirizzata su diverse aree funzionali, le più comuni sono quelle che riguardano le performance dei gruppi di assistenza tecnica (KPI, Key Performance Indicators) quali le percentuali di risoluzione delle problematiche affrontate, la somma dei costi affrontati per risolverle, oppure le analisi sui tempi di risposta dei vari gruppi di assistenza tecnica a fronte di accordi contrattuali, quali i tempi massimi di risoluzione concordati, detti in gergo aziendale SLA, Service Level Agreements.

Il Team è nato inizialmente come gruppo di supporto tecnico alle operazioni del Delivery Center, il call center europeo di assistenza tecnica dei clienti IBM, e negli anni ha mutato la sua conformazione per venire incontro alle esigenze anche di altri grandi clienti IBM che usufruiscono di pacchetti di supporto dedicati e specifici.

1.4 Reportistica standard e applicazioni analitiche dedicate.

La reportistica standard - detta, appunto, 'standard reports' – è quasi sempre parte integrante dei software di CRM (Customer Relationship Management), i quali si occupano di fornire alle compagnie un modello di gestione della clientela ed in grado di organizzare, automatizzare e sincronizzare i processi di business; principalmente attività di vendita, ma anche marketing e supporto tecnico.
Gli standard reports hanno in genere un formato predefinito, sono totalmente parametrici, e nella loro forma più semplice, vengono processati periodicamente e sono quindi disponibili ad intervalli regolari. In sostanza definiscono l'insieme chiave di informazioni necessarie a comprendere cosa succede in una particolare area di business e costituiscono il cuore dell'area di business intelligence dell'applicativo CRM. A seconda del tipo di attività, i report possono ad esempio riguardare la comparazione delle vendite nell'anno in corso rispetto alle previsioni dei rappresentanti, il consumo mensile delle risorse rispetto alle pianificazioni di servizio, o il grado di responsività sui volumi di chiamate dei dipartimenti di vendita.

La reportistica standard è costituita da una serie di componenti tecnologiche: un designer tool per la definizione dei nuovi report, servizi che garantiscano l'archiviazione storica dei report, la loro corretta esecuzione e la protezione dei dati generati.

Le applicazioni analitiche dedicate hanno una complessità maggiore rispetto agli standard reports. Esse si concentrano su uno specifico processo di business, incapsulando l'insieme di conoscenze necessarie per capire come analizzare ed interpretare il dato processo. Possono includere algoritmi complessi e modelli di data-mining. Alcune applicazioni analitiche avanzate consentono inoltre all'utente di apporre modifiche al modello di

acquisizione delle informazioni, ad esempio per far fronte a nuove evidenze rilevate usando l'applicazione stessa [1].
Alcuni esempi classici di applicazioni analitiche possono essere i sistemi per determinare le previsioni sul budget, l'efficacia dell'attività promozionale, il rilevamento delle frodi, e le performance in generale.

La differenza sostanziale tra standard report ed applicazioni analitiche dedicate sta proprio nella loro complessità, in particolare per quanto riguarda il numero di fonti da cui incrociare i dati.

La reportistica standard ad esempio può fornire un'aggregazione dei dati sui tempi di chiusura di un problema segnalato dal cliente (ticket) di un determinato pool, dove per pool si intende un gruppo di tecnici operatori accomunati per area di lavoro (es. Paesi scandinavi, Irlanda e Regno Unito o Europa del sud).

Questo tipo di report può essere facilmente generato all'interno del software di CRM per la gestione dei ticket, sulla base di template preimpostati all'interno del software stesso.

Supponiamo invece che il calcolo dei tempi di chiusura debba evidenziare quanti dei ticket abbiano "sforato" i tempi massimi di chiusura concordati con il cliente, ma sottraendo dal calcolo del tempo di lavorazione :

- I periodi in cui il ticket è stato riassegnato ad un tecnico che sta operando in loco.
- I periodi in cui il ticket è stato posto in stand-by per mancanza di informazioni dettagliate sul problema.
- I minuti e le ore di lavoro al di fuori del normale orario lavorativo.
- I giorni festivi del paese in cui opera il call center.

Il tutto inoltre tenendo conto del fuso orario in cui opera il call center e che avrà orari e date diverse da quelli presenti nel sistema di gestione dei ticket.

E' chiaro che quest'ultima casistica non può essere prevista nei template del software di CRM, perché il concetto stesso di "tempi di chiusura massimi" non è previsto, e anche perché le informazioni sulla messa in stand-by di un ticket potrebbero essere indirettamente contenute su diversi database extra-sistema, come quello che contiene le informazioni sul numero, provenienza, durata e deviazioni delle chiamate fatte dall'operatore del call center, completamente slegato dal sistema CRM di gestione dei ticket.

E' dunque la complessità a far sorgere l'esigenza di creare un applicativo dedicato in grado di estrapolare dati da sistemi diversi e separati tra loro (nel caso dell'esempio, il sistema di gestione centrale dei ticket e i database locali dei software di gestione delle chiamate in entrata ai call center), in modo da ottenere informazioni che sarebbero altrimenti impossibili da aggregare utilizzando i soli standard reports forniti dal CRM.

2. Business Intelligence

Il secondo capitolo mira a dare una definizione del concetto di Business Intelligence come insieme di approcci per la raccolta, l'analisi e la consultazione dei dati operativi di un'azienda, ed indicando quali sono le applicazioni classiche su cui operano gli strumenti software di Business Intelligence. In seguito, verranno discusse due diverse tipologie di business intelligence: la BI tradizionale, che analizza retroattivamente la collezione di dati, e quella operazionale, che dà enfasi al valore dei dati raccolti in tempo reale.

2.1 Una definizione generale

Nel linguaggio comune, con il termine Business Intelligence (BI) si fa riferimento a tutta una serie di applicazioni e tecnologie sviluppate per l'archiviazione, la raccolta e l'analisi delle informazioni utili ad una impresa, tali da agevolarne il consumo ed aiutare i professionisti ad intraprendere azioni vantaggiose per l'azienda.

Oltre alla componente strettamente tecnologica, la BI inquadra in senso più generale anche il fattore umano, ovvero l'abilità dell'utente finale di sfruttare i concetti propri della BI per mettere in prospettiva ed interpretare i dati ottenuti [1].

Nello specifico, le applicazioni di Business Intelligence possono essere usate dagli utenti (tipicamente la componente manageriale di un'azienda) per ottenere informazioni dettagliate su qualunque aspetto che riguarda il rendimento, compreso l'andamento in termini di performance, e di conseguenza apportare i necessari tagli alla spesa o diversificare la propria strategia.

Le principali aree funzionali di cui può occuparsi un'applicazione di Business Intelligence possono includere [2]:

- L'analisi del comportamento del consumatore: ripetitività degli acquisti e trend di spesa.
- La misura delle previsioni di vendita e le performance finanziarie.
- Pianificazione finanziaria e definizione del budget.
- Monitoraggio dell'efficacia delle campagne promozionali.
- Ottimizzazione delle performance operazionali e logistiche.
- L'analisi del rischio di impresa.

Nella business intelligence si ha inoltre la necessità di presentare efficacemente ed intuitivamente la giusta informazione, sottolineando l'importanza della chiarezza, rapidità e facilità con cui tali informazioni sono fornite all'utente, anche attraverso l'utilizzo di aggregazioni di dati, come ad esempio dei pannelli riassuntivi che espongono un sommario delle informazioni salienti, oppure tramite schematizzazioni e grafici .

Uno dei benefici maggiori dati dall'uso di una applicazione di BI è la possibilità di avere accesso ai dati provenienti da fonti multiple e diverse tra loro, riorganizzate in un unico formato.
E' quindi possibile misurare gli obbiettivi ed analizzare i risultati tra diversi dipartimenti e tenere traccia delle azioni degli acquirenti grazie all'estrazione di informazioni chiare provenienti da dati complessi.

Nonostante la domanda di strumenti di BI stia incrementando di anno in anno, queste applicazioni non godono di una fama particolarmente positiva tra molti operatori, i quali considerano questo tipo di applicativi dispendiosi e difficili da usare, anche se sul mercato si stanno affacciando nuove pacchetti software in grado di garantire soluzioni meno personalizzabili ma decisamente più snelle. Nonostante questo, i software di BI non sono

sempre accessibili sul piano economico, e quindi sono usati soprattutto in organizzazioni di dimensioni medio-grande, piuttosto che nelle piccole attività.

Esistono due tipi diversi di Business Intelligence, ma entrambe hanno lo scopo principale di fornire grandi quantità di informazioni in un formato digeribile e facile da consultare. Si tratta della BI tradizionale e della BI operazionale, che verranno discusse qui di seguito.

2.2 Business Intelligence Tradizionale

La BI tradizionale prevede l'uso di grafici e tabelle per tener traccia delle prestazioni di una particolare area operativa in termini di andamento sul periodo trascorso (passato) e previsionale (futuro). Le previsioni per l'organizzazione sul futuro, ovviamente basate sulla rilevanza dei dati raccolti in passato, è fatta usando sia strumenti di reportistica che di analisi.

Un'applicazione comune di questa declinazione tradizionale della Business Intelligence può essere ad esempio quella operata da una compagnia farmaceutica interessata ad identificare i cicli (pattern) d'uso di un determinato farmaco nel corso dell'anno in una particolare area geografica, in modo tale da dedurre le previsioni di vendita dell'anno successivo o pianificare meglio una campagna di marketing.

Ci sono casi in cui, per la natura stessa della realtà in cui viene usata, la BI tradizionale soddisfa a pieno le necessità dell'azienda. In altri casi però, può sorgere il bisogno di osservare dati sul momento presente (real-time data). L'estensione dell'area di copertura dei dati anche al presente richiede l'uso della BI operazionale.

2.3 Business Intelligence Operazionale

In contrapposizione al caso precedente dell'industria farmaceutica, supponiamo invece il caso di un'industria manifatturiera. Questa società potrebbe essere più interessata a conoscere se le spedizioni di merce che sta effettuando siano o meno consegnate in tempo, e come questo fattore stia impattando direttamente sulle sue attività.

In questo frangente, l'azienda ha bisogno di conoscere e valutare in tempo reale i dati provenienti dalle proprie operazioni logistiche, mentre lo storico sui tempi delle spedizioni fatte nelle settimane o mesi precedenti passa in secondo piano.

La BI operazionale quindi ha a che fare con dati operativi e ad un inferiore livello di astrazione, permettendo ai professionisti di avere informazioni aggiornate il più possibile in modo tale da poter intraprendere azioni immediate. Sostanzialmente, se la BI tradizionale ha a che fare con dati relativi ad eventi passati, la BI operazionale usa dati in tempo reale, permettendo agli utenti di vedere, valutare, e fare i necessari aggiustamenti [3].

In conclusione, la semplificazione e la raccolta di dati da fonti diverse permette agli utenti di appoggiarsi su una memoria di informazioni che contribuisce al processo decisionale e alle strategie di risoluzione dei problemi per comprendere in maniera più efficace il comportamento sia dei clienti che del mercato in generale.

3. Compiti del Team di BI&M

Il Team di Business Intelligence & Metrics irlandese è sostanzialmente composto in larga parte da operatori che si occupano della generazione e della customizzazione di reportistica 'dedicata' in uso alla componente manageriale di IBM che supervisiona determinate aree operative delle società cliente. Fondamentalmente, il committente è quasi sempre interno alla società stessa, in quanto l'interpretazione dei dati ottenuti dai report è spesso affidata ad analisti IBM, che operano sia esternamente, presso le società clienti, sia all'interno di alcune aree funzionali di IBM. Ad esempio, una buona percentuale del lavoro prodotto dal team di BI&M è rivolto ai manager che hanno in gestione il centro di supporto tecnico e vendite per le attività di consulenza IT di IBM in Europa (il Delivery Center), che ha sede anch'esso a Dublino.

Una parte degli operatori del Team di BI&M, gli sviluppatori – minoritaria rispetto alla totalità dei soggetti coinvolti nel team – si occupa invece degli aspetti strettamente tecnici, in modo tale da garantire la produttività del resto del gruppo. Gli sviluppatori del Team, compreso il sottoscritto, hanno quindi il compito di configurare o implementare ex-novo degli strumenti che solitamente hanno una doppia funzionalità: da un lato permettono agli altri operatori del team di 'creare' nuovi report dedicati e di configurare le parametrizzazioni del caso, e dall'altro lato permettono all'utente finale la consultazione degli stessi.

Oltre lo sviluppo in senso stretto, gli sviluppatori del Team hanno inoltre il compito di monitorare e garantire la connettività alle risorse esterne, che concretamente significa testare ciclicamente le connessioni a tutti i database remoti che vengono utilizzati per la generazione automatica della reportistica.

Uno degli incarichi più comuni che deve svolgere un operatore del team di BI&M è quello della raccolta requisiti, ruolo che ho avuto modo di provare in un periodo in cui il management aveva deciso di abbassare momentaneamente la priorità dello sviluppo software per far fronte ad un picco di richieste sulla reportistica ordinaria.

Molto spesso, il manager che richiede un determinato report compone la sua richiesta sulla base di dati empirici, senza essere a conoscenza dell'esistenza o meno di standard report già presenti all'interno del sistema stesso, e senza avere un'idea chiara della struttura e della provenienza dei dati richiesti. Chi lavora nel team di BI&M quindi, oltre alla capacità di interpretare le specifiche fornite, deve avere a priori un'idea chiara delle opzioni possibili e dei legami tra le diverse fonti di dati, ed avere una conoscenza approfondita sulla struttura interna dei database dei vari sistemi, qualora accessibili direttamente, per poter confermare la fattibilità della richiesta. In base a queste conoscenze può allora discutere con il committente eventuali modifiche e tempi necessari all'implementazione.

4. Strumenti di estrazione, analisi e reportistica

Per poter comprendere meglio come i concetti di Business Intelligence vengano applicati, a lato pratico, nelle realtà lavorative, è utile esporre quali sono le tipologie di applicativi software oggi in uso per affrontare il tema della BI, soffermandosi su un pacchetto software in particolare, Eclipse BIRT, che costituisce lo strumento principale in uso nel Team di Business Intelligence & Metrics.

4.1 Panoramica sui tool di BI

Nel corso degli anni, la business intelligence ha goduto della graduale comparsa sul mercato di diversi strumenti software, alcuni dei quali sono specificatamente dedicati all'estrazione, all'analisi o alla rappresentazione dei dati.

Il primo e indispensabile strumento a supporto delle operazioni di BI è stato, almeno storicamente, il foglio di calcolo, come ad esempio Microsoft Excel. Oltre a questo fondamentale strumento però, è possibile individuare oggi un ventaglio di prodotti che possono essere classificati come segue.

- *Software di interrogazione e reportistica*: Sono tool generici in grado di estrarre, ordinare, riassumere e presentare i dati selezionati da un insieme di sorgenti preselezionate, tipicamente database relazionali. Spesso assumono la forma di framework da integrare in meccanismi più ampi, come ad esempio un applicativo Java. Alcuni popolari esempi di questa tipologia di software sono Eclipse BIRT, Jasper Reports e Pentaho.

- *OLAP Software (OnLine Analytical Processing):* OLAP è un approccio all'analisi dei dati che sfrutta il concetto di ricerca analitica multidimensionale.

  Il cuore di qualunque sistema OLAP è rappresentato dal suo "cubo multidimensionale" (OLAP Cube, o Ipercube, nel caso di dimensioni maggiori a tre).

  Mentre in un sistema tradizionale che usa un semplice database relazionale le informazioni vengono collezionate per via transazionale e le analisi sui dati vengono effettuate mediante interrogazioni che mirano a filtrare le informazioni ed incrociare i dati tra le diverse tabelle tramite chiavi comuni, i sistemi OLAP sono organizzati per facilitare e velocizzare il lavoro di analisi sui dati aggregati disponendosi, almeno a livello logico, su un piano di lavoro a tre o più dimensioni, dove ogni dimensione corrisponde ad un parametro di analisi dei dati.

  Ogni cubo OLAP è composto da dati tipologici, detti misure, categorizzati per dimensioni. Ad esempio la dimensione "Geografia" può contenere la lista di tutte le nazioni in cui la società opera. Le misure vengono estratte da un insieme di tabelle del database relazionale oppure organizzate in modo tale da formare uno "schema a stella", dove si ha una tabella per ogni dimensione di analisi contenente i valori ammissibili per la data variabile. Ad esempio un Cubo OLAP contenente i dati logistici di un'attività può avere una dimensione contenente la lista dei prodotti, un'altra i tempi di consegna, un'altra la località di consegna, un'altra ancora il tipo di corriere usato. Ogni tabella dello schema a stella corrisponderà ad una diversa dimensione di analisi, e quindi ad una diversa dimensione del cubo OLAP.

  Ad ogni cubo OLAP poi vi è poi l'aggiunta di un ulteriore dimensione, detta dei fatti (fact table), che consiste in una tabella contenente i dati relativi ad un determinato evento, quasi sempre in forma

numerica, e dove ogni colonna corrisponde ad una dimensione. Ad esempio l'evento di "consegna effettuata" consentirà di popolare record nella fact table contenente i valori numerici relativi alle informazioni di consegna descritte precedentemente.

Questa metodologia consente di ottenere performance di calcolo molto superiori rispetto ad un sistema tradizionale e permette di ottenere informazioni sul dato aggregato da prospettive diverse, suddividendo l'approccio in tre fasi distinte: Consolidamento (o roll-up), estrazione (drill-down), separazione e cubatura. Il consolidamento consiste nell'aggregazione dei dati accumulabili sulle diverse 'dimensioni', e il popolamento delle relative tabelle. Ogni dimensione può eventualmente contenere una gerarchia interna, e quindi dar luogo a dei multi-livelli (tramutando di fatto lo schema a stella in uno schema a fiocco di neve, ovvero con ulteriori diramazioni). Nell'esempio precedente sul dato geografico, ogni dimensione potrebbe contenere il livello "stato", il sottolivello "regione" e l'ulteriore sottolivello "città". L'operazione di estrazione (drill-down) è un'operazione che consente agli utenti di navigare attraverso i dettagli, ovvero i vari livelli gerarchici delle dimensioni. Le operazioni di separazione e cubatura sono funzionalità in cui l'utente può separare (slicing) i dati ottenuti da uno specifico insieme di dati e richiedere di consultarli (dicing) da diversi punti di vista.

Alcuni database, basati su un mix di modelli gerarchico e relazionale, implementano nativamente il concetto OLAP e permettono l'esecuzione di analisi complesse e richieste su misura con rapidi tempi di esecuzione. Esempi di questo genere di database sono IBM Cognos, Oracle Essbase e SAP Business Objects.

- *Data Mining Software:* Il Data Mining è un processo che ha come obbiettivo quello di individuare la presenza di determinati schemi (pattern) all'interno di grandi volumi di dati. E' un sistema che trae

spunto da concetti propri dell'intelligenza artificiale, la statistica ed il machine learning. Si tratta quindi di estrarre informazioni da un insieme di dati e trasformarle in una struttura comprensibile e riutilizzabile, sfruttando tecniche che consentono di isolare schemi o gruppi di dati di interesse, come ad esempio gruppi di record legati logicamente tra loro, individuare anomalie o singolarità nel gruppo di dati, o valutare le dipendenze tra i diversi record.

I software di data mining implementano il processo KDD (Knowledge Discovery in Databases), di cui il Data Mining è solamente uno dei passi necessari per affrontare il problema. In effetti, la definizione propria di data mining è descritta solamente come attività di analisi di grandi volumi di dati tale da estrarre pattern interessanti come ad esempio gruppi di record legati logicamente tra loro, record anomali o gruppi di record interdipendenti.
Un processo KDD invece descrive l'intero ciclo di identificazione di sottoinsiemi di dati che siano in qualche modo significativi per l'utente, e tali da permettere l'identificazione di sottogruppi simili all'interno del dataset [6].
Un processo KDD può essere suddiviso in diverse fasi:

1) Identificazione del dominio dei dati rilevante dal punto di vista dell'obbiettivo che la ricerca mira ad ottenere.
2) Pulizia del dato e pre-processing, ovvero la rimozione dei valori di disturbo sulla base di modelli già disponibili e scelta delle strategie da mettere in pratica nel caso di campi mancanti.
3) Data mining, ovvero l'applicazione dell'appropriato algoritmo di data mining in base al caso in oggetto.
4) Validazione ed interpretazione del risultato.

Gli algoritmi di data mining possono essere a sua volta frazionati in diverse classi di utilizzo:

- *Anomaly Detection*: l'identificazione di record inusuali o i cui valori si discostano di molto dal valore mediano, e che possono rappresentare un dato interessante o un errore che potrebbe richiedere ulteriori analisi.

- *Association Analysis*: Ricerca delle relazioni tra le variabili, ad esempio per determinare se due prodotti correlati tra loro vengono acquistati separatamente o meno.

- *Clustering*: consiste nello scoprire gruppi o strutture di dati che sono in qualche modo simili, senza essere a conoscenza a priori del contenuto del dataset.

- *Classification*: riconoscere all'interno del record le caratteristiche proprie di una struttura di dato di cui si è già a conoscenza. Ad esempio l'assegnazione di una valutazione "positivo" o "negativo" di un commento sulla base di caratteristiche semantiche note.

- *Regression*: il tentativo di trovare una funzione (Regression Function) che modelli il dato, con il minimo margine di errore.

- *Summarization*: fornisce automaticamente una rappresentazione compatta e riassuntiva del dataset.

I software di data mining possono avere un ruolo importante sia dell’area di gestione della clientela, come ad esempio consentire di contattare solo la parte della clientela più incline ad essere

interessata ad una particolare offerta, sia nell'ambito più strettamente decisionale, dove le decisioni pregresse possono individuare un modello decisionale per future casistiche similari.
SAS Enterprise Miner, StatSoft Statistica e Oracle Data Mining sono alcuni esempi di prodotti che si occupano di questa area della BI.

- *Business Perfomance Management Software:* Sono sistemi che permettono all'utente di fissare degli obbiettivi, confrontare i dati rilevanti al raggiungimento di tali obbiettivi e talvolta di effettuare degli interventi alla luce delle informazioni ottenute in modo tale da incrementare il rendimento futuro. I processi chiave a cui fa riferimento la Business Performance Management all'interno di una organizzazione sono tipicamente la pianificazione finanziaria, quella operazionale, il consolidamento della reportistica e il monitoraggio degli indicatori di performance (KPI – Key Performance Indicators) legati alla strategia. Questi tool sono spesso inglobati all'interno di sistemi OLAP e di Data Mining, e raramente vengono commercializzati come prodotti stand-alone.

4.2 Eclipse BIRT

All'interno del team di BI&M, lo strumento più usato per la generazione della reportistica dedicata è sicuramente Eclipse BIRT, dove BIRT è appunto l'acronimo per "Business Intelligence and Reporting Tool", che fa parte della classe dei software tradizionali per l'interrogazione e la reportistica.

Sviluppato come progetto open source dalla Eclipse Foundation, è stato ideato per coniugare le funzionalità di BI con le applicazioni web enterprise, in particolare quelle basate sul linguaggio Java.

BIRT ha due componenti principali: Un “Report Designer” all’interno dell’interfaccia IDE (Interactive Development Environment) di Eclipse, che facilita la creazione degli script e la composizione della veste grafica dei report stessi (Fig. 4.1), ed una componente a runtime per generare i report che può essere messa in esecuzione in qualsiasi ambiente Java.

BIRT inoltre mette a disposizione un motore di generazione dei grafici, completamente integrato nel report designer, e che può essere usato per includere grafici nelle applicazioni.

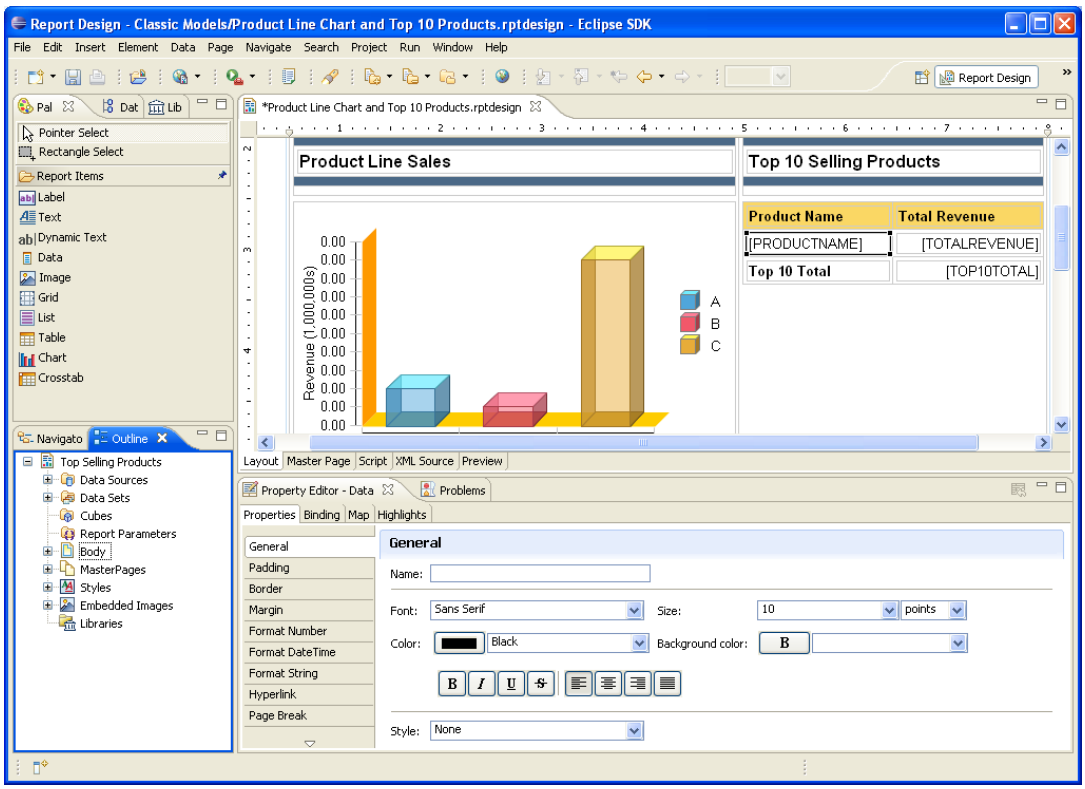

Figura 4.1. *Uno screenshot del report designer all’interno dell’interfaccia Eclipse IDE.*

Il report designer consente di accedere ad una o più fonti di dati (data source), tipicamente database remoti, tramite la specifica di riferimenti, nella forma di stringhe di connessione standard, che utilizzano i driver presenti sul sistema in cui il software andrà in esecuzione [4].

Nel momento in cui il sistema è a conoscenza delle fonti dei dati, è possibile selezionare la lista degli attributi ai quali si vuole accedere, e definire la lista degli attributi previsti nella tabella con i risultati dell'interrogazione.

Un editor consente poi di formulare una query SQL di tipo SELECT per interrogare il data set, i cui campi dovranno corrispondere con quanto specificato precedentemente in fase di impostazione.

Le configurazioni inserite tramite il report designer vengono così processate in tempo reale dal Design Engine che produce un file XML, detto rptDesign XML Design file, contenente tutte le informazioni di configurazione: stringhe di connessione ai data source, layout del report e query SQL compresi.

Quindi, come descritto nella Figura 4.2, dopo che il Design Engine ha generato il file XML di configurazione rptDesign, il passo successivo è la presa in carica dell'operazione da parte del Report Engine, che si occupa dapprima della fase di Generazione, ovvero l'esecuzione fisica delle query SQL e l'aggregazione in memoria dei dati ottenuti, per poi passare alla fase di presentazione che consiste nel generare un file contenente il report finale sulla base delle indicazioni di layout contenute nel documento XML rptDesign. Il risultato può assumere formati diversi, come ad esempio un documento HTML paginato, un file PDF o un documento Excel.

Per operazioni sui dati più complesse, è possibile aggiungere un livello di controllo alle fasi di generazione e presentazione. Ad esempio, in fase di generazione, può sorgere la necessità di modificare i valori uno specifico

attributo con un altro valore calcolato sulla base dei record precedenti, oppure normalizzare i valori di timestamp (data e ora) provenienti da sistemi con fuso orario diverso con l'orario del sistema centrale.

In fase di presentazione invece l'utente potrebbe aver bisogno di eliminare valori nulli, dati non necessari o non congruenti.

Questo è possibile attraverso una caratteristica offerta da Eclipse BIRT, ovvero lo scripting. Attraverso un linguaggio Java-like, spesso detto Javascript ma sostanzialmente diverso nella sintassi dal linguaggio Javascript standard diffuso sul web, è possibile ottenere i riferimenti alle tuple ricavate dall'accesso ai data source ed effettuare operazioni di raffinamento, selezione ed incrocio durante le diverse fasi dell'esecuzione del report [5].

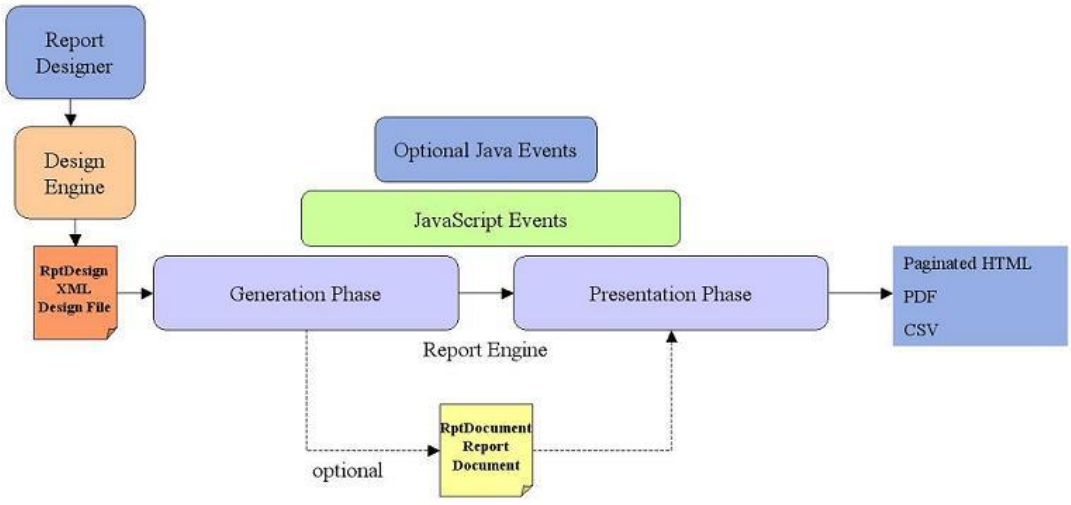

Figura 4.2. *Ciclo di generazione del report in Eclipse BIRT.*

Lo scripting permette di manipolare i dati in fase di generazione e di presentazione ad un elevato livello di granularità, è possibile infatti specificare singoli script che andranno in esecuzione durante le sotto-fasi di questi due passaggi effettuati dal report engine.

La fase di generazione è suddivisa in una serie di passaggi consecutivi corrispondenti ai diversi livelli di impostazione e memorizzazione dei risultati: L'inizializzazione (a sua volta suddivisa nel caso si tratti dell'inizializzazione dell'intero report, della singola tabella, della singola colonna attributo, della singola cella o dell'elemento che sta per essere salvato in memoria), il beforeFactory, con le sottofasi beforeOpen, afterOpen e onFetch, che corrisponde al momento in cui il sistema stabilisce la connessione con il database ed inizia a ricevere i dati grezzi, e l'afterFactory che corrisponde al passaggio finale al termine della ricezione dei dati.

Anche i diversi passaggi della fase di presentazione sono intercettabili tramite la funzionalità di scripting, costituiti da una fase di inizializzazione del documento ancora vuoto, e i momenti in cui i singoli elementi del report (tabelle, righe e celle) vengono effettivamente scritti sul documento di output, compreso l'atto finale (afterRender) in cui il file risultato sta per essere concretamente salvato sul server.

Tra le varie funzionalità di Eclipse BIRT vi è inoltre quella di parametrizzazione del report, in sostanza è possibile personalizzare il risultato definendo dei parametri variabili all'interno della query SQL che saranno richiesti all'utilizzatore ad ogni generazione del report.

Nella loro forma più comune, i risultati dei report sono dati da rappresentazioni tabellari, talvolta con l'ausilio di istogrammi, grafici a linee o a dispersione (Fig. 4.6).

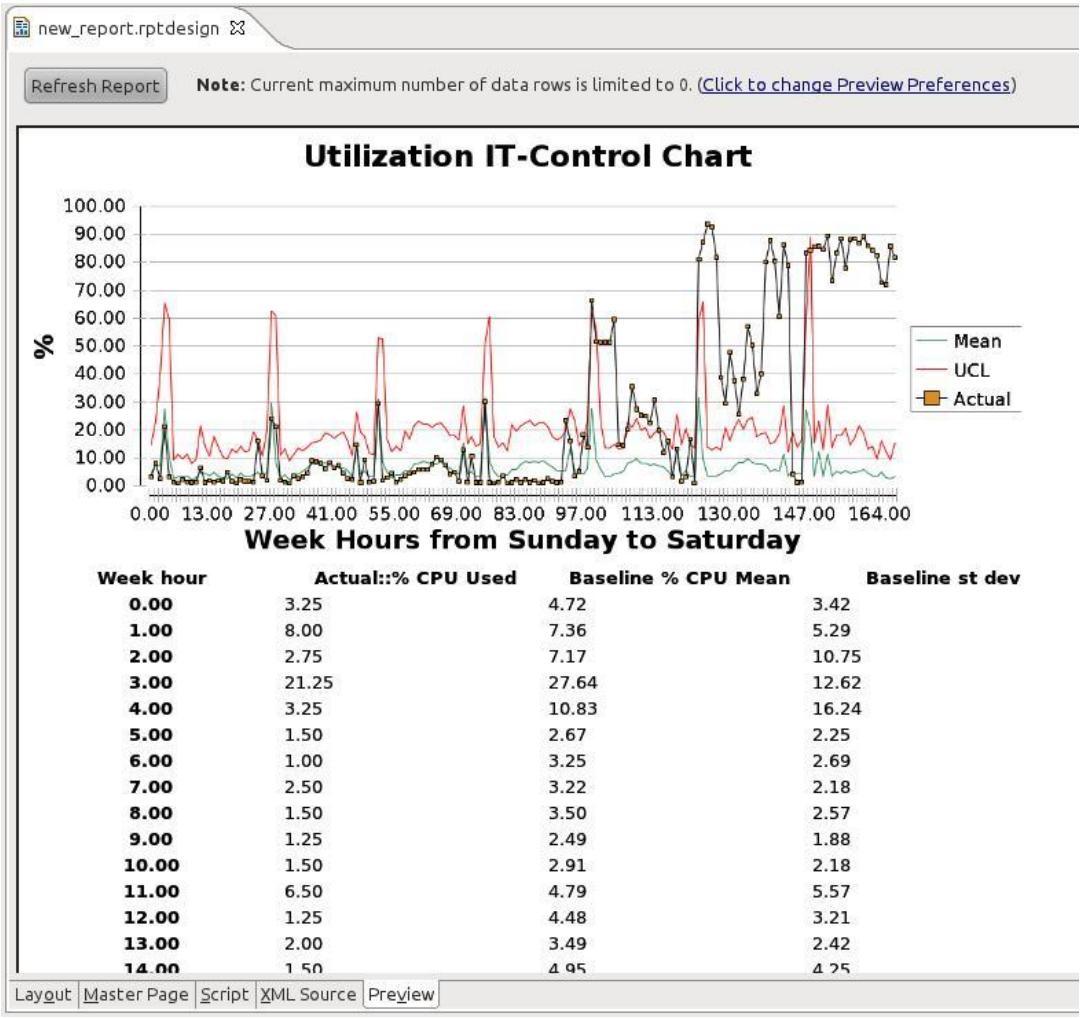

Figura 4.6. *Esempio di report generato utilizzando Eclipse BIRT.*

Casi Studio

Di seguito verranno introdotti due casi di studio come forma esemplificativa del lavoro svolto presso il team di BI&M, con particolare riferimento ai concetti esposti in precedenza.

5. Caso 1: IDC Report
Migrazione del sistema di reportistica interno

5.1. Obbiettivi

IDCReport è il sistema di reportistica ideato dal team di BI&M per far fronte alle richieste dell'IBM Integrated Delivery Center di Dublino, il call center di supporto tecnico IBM in Europa che occupa più di 3000 operatori tra tecnici e divisione commerciale. Il suo compito è perlopiù quello di rendere disponibili ai manager di reparto valutazioni sull'andamento delle operazioni e sulle performance incrociando dati che arrivano dai dettagli registrati dal sistema telefonico dell'help desk, dal sistema di apertura segnalazioni e da quello di gestione dei tecnici in loco.
L'esigenza era quella di migrare l'intero sistema, inizialmente distribuito su una unità centrale contenente lo storico dei report generati, e una serie di unità satellite separate per la generazione dei nuovi report, in un unico server dedicato.

5.2 Architettura del sistema di reportistica esistente

Il sistema IDCReport, nella sua configurazione iniziale, era distribuito su più unità, ovvero server fisici indipendenti, in modo tale da poter suddividere il carico di lavoro: una unità principale, e sei unità satellite.
Tutte e sette le unità erano caratterizzate dalla medesima configurazione hardware, ovvero dei server IBM xSeries 336, dotati di un processore Intel

Xeon da 2.8GHz e 4GB di RAM PC3200 ciascuno. I server erano installati su di un unico rack (il sistema di scaffalatura standard per componenti hardware) all'interno del data center nel Technology Campus, non distante dal mio luogo di lavoro. Per quanto mi riguardava, l'unica via di accesso all'hardware era per via remota usando Secure Shell (SSH), un protocollo di rete che permette di stabilire una sessione di lavoro cifrata sulla macchina remota ed impartire istruzioni tramite riga di comando.

L'unità principale aveva un duplice compito: eseguire un'applicazione web che fungesse da interfaccia utente per la consultazione dei report e servire da unità di memorizzazione permanente dei report generati dalle unità satellite (Fig. 5.1).

Ogni unità satellite per l'appunto aveva il compito di generare dei report predefiniti tramite un applicativo Java in grado di processare automaticamente dei file di configurazione rptDesign, implementati precedentemente, nel BIRT Engine (installato su ognuna delle unità), in modo analogo a quanto descritto nel Cap. 4.2.

Ogni file rptDesign presente sull'unità satellite conteneva le indicazioni per la generazione di un singolo report, la cui esecuzione veniva programmata a scadenze regolari tramite l'uso di cron jobs, ovvero delle indicazioni al sistema operativo per l'esecuzione automatica di processi ad intervalli predefiniti. Nel momento in cui il BIRT engine terminava la generazione del report, il file risultante, generalmente in forma di documento HTML o PDF, veniva automaticamente trasferito, tramite protocollo FTP, in apposite cartelle sull'hard disk dell'unità centrale contenente l'archivio dei report (Fig.5.1).

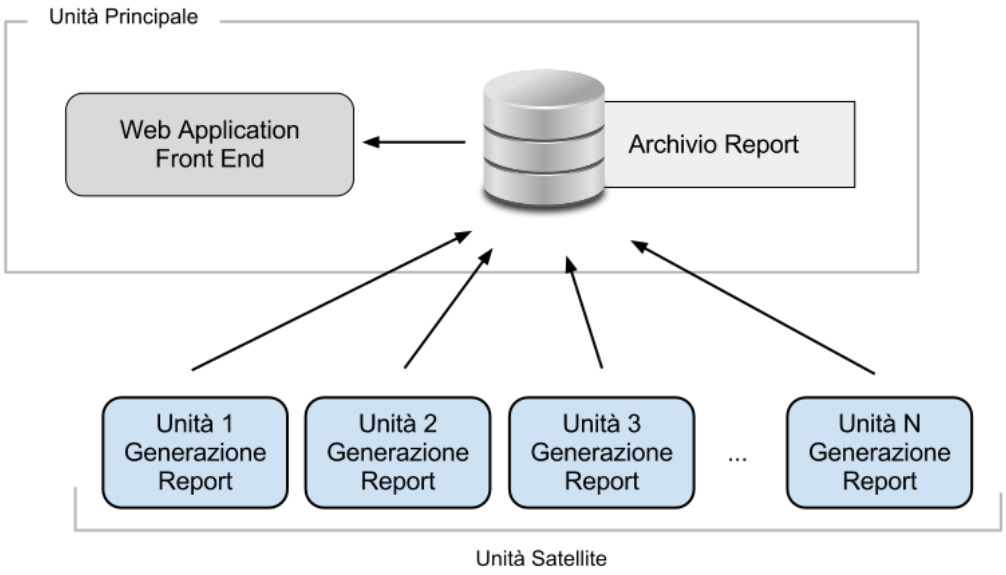

Figura 5.1. *Schema riassuntivo del sistema IDCReport.*

Ognuna delle unità satellite conteneva quattro cron jobs distinti, uno per i report che andavano eseguiti ogni ora, uno per i report giornalieri, uno per quelli settimanali e uno per quelli mensili. Ogni cron jobs avviava una procedura che inizializzava, in modo sequenziale, un'istanza del BIRT Engine per ognuno dei file rptDesign da processare appartenenti alla categoria indicata.

La scelta di usare più unità satellite era data dalla difficoltà di eseguire più istanze del BIRT engine sulla stessa macchina, dovuta a diversi fattori:

- Ogni istanza del BIRT engine era sostanzialmente un processo Java eseguito sulla Java Virtual Machine (Java prevede che ogni classe dell'applicativo venga compilata in un Bytecode, indipendente dalla piattaforma su cui è stato sviluppato, eseguibile poi su un qualunque sistema dotato della JVM). Per evitare problemi di esaurimento della memoria RAM è spesso utile allocare manualmente l'heap space, ovvero un'area dedicata alla sola JVM nella memoria fisica, tramite

parametrizzazioni al suo avvio (nel nostro caso la direttiva -Xms1024 –Xmx2048 data alla JVM consentiva di allocare un'heap space della dimensione minima di 1GB e massima di 2GB). In un sistema come il nostro, con soli 4GB di memoria RAM disponibile, è chiaro come la memoria occupata dal sistema operativo (che usa quasi 1.5GB) e quella assegnata per l'esecuzione di un'istanza del BIRT engine fosse sufficiente ad esaurire quasi completamente le risorse di memoria del sistema.

- I tempi di esecuzione su un database remoto di una query SQL descritta all'interno del file rptDesign e lanciata dal BIRT engine, specie nei casi di JOIN multiple tra 3 o più tabelle, erano mediamente nell'ordine dei 30-50 secondi. Inoltre, il risultato di queste query talvolta consisteva in migliaia di record, ad esempio un report che generava la lista dei singoli articoli su cui è stata fatta una segnalazione nell'ultimo anno e ordinati per priorità interna, consiste in più di 10 mila record, per un totale di circa 50MB di dati, il che significa ulteriori 30 secondi per la ricezione e la codifica del dato. Non tutti i report avevano questo grado di complessità, ma l'abbattimento dei tempi morti dovuti alle operazioni di I/O sarebbe stato possibile solo con l'esecuzione in parallelo di più BIRT engine, resa impossibile dalle condizioni esposte nel punto precedente.
  Considerando che in una singola unità il numero di report era tra i 20 e i 30 elementi, di cui circa 15 da generare ogni ora, un ciclo di esecuzione impiegava generalmente dai 20 ai 30 minuti.

- Un database remoto in particolare, Avaya, contenente i dati relativi alla durata delle conversazioni telefoniche degli operatori e usato in alcuni report, non permetteva di instaurare connessioni simultanee provenienti dal medesimo indirizzo IP. Anche questo era un motivo

per evitare l'esecuzione concorrente dei report da un'unità harware dotata solamente di un indirizzo IP.

Nell'unità centrale, la componente di interfaccia a front-end del sistema era costituita da una applicazione web in Java eseguita su un server Glassfish, un application server per Java EE open source.
L'intera applicazione web era composta da più moduli, o sezioni, contenenti file JSP (Java Server Pages) con i link a tutti i report afferenti alla sezione in oggetto, salvati sul disco rigido su cui è in esecuzione. Ad esempio la sezione "Nordics" contiene i link a tutti i report utili ai manager di reparto del delivery center dei pool Danesi, Svedesi, Norvegesi e Finlandesi, suddivisi per categorie e frequenza di aggiornamento, oppure sezioni specifiche per i pool dedicati ai "grandi clienti" come Board Gàis Ireland.
L'accesso alle varie sezioni, o anche ai singoli report, è regolato da un livello di autenticazione: ogni utente effettuava l'accesso al sistema tramite la propria login IBM, ed in base alle sue credenziali, ottenute interfacciandosi ai server blue pages IBM contenenti le informazioni relative a ciascun dipendente, il sistema abilita o meno le sezioni da visualizzare. Le regole di accesso sono impostate tramite un pannello di controllo, implementato nell'applicazione stessa, che consentiva ad alcuni utenti abilitati di impostare regole e restrizioni per gli altri utenti che avevano accesso al sistema sotto il suo controllo, ad esempio un utente che da credenziali IBM è manager di reparto potrà decidere quali report far visualizzare agli analisti suoi collaboratori.

I motivi dell'insorgere dell'esigenza di una migrazione del sistema sono stati principalmente la frequente comparsa, nelle macchine satellite, di errori relativi all'esaurimento di memoria RAM, probabilmente a causa dell'incremento del numero di report da eseguire sopraggiunto negli ultimi anni e di una maggiore complessità degli stessi, che portava al fallimento della loro esecuzione, ed in secondo luogo la comunicazione da parte del

management di IBM Global della volontà di terminare il supporto al sistema operativo su cui l'applicazione era in esecuzione (Windows 2000), pena l'introduzione di ammende a carico del Team.

5.3 Il processo di migrazione software in IBM: le diverse fasi

All'interno di IBM le operazioni di manutenzione sono regolate da delle cosiddette policy, ovvero delle linee guida, spesso obbligatorie, che definiscono la procedura da adottare.
In particolare nel caso della migrazione, viene richiesto di rispettare una serie di procedure per l'individuazione e l'approvazione all'uso del nuovo hardware, mediante una catena di richieste formali la cui formalizzazione può protrarsi anche per molti giorni.

Nel team di BI&M abbiamo fatto riferimento alle linee guida interne di IBM che possono essere sintetizzate in quattro punti chiave (Fig. 5.2):

- *Discover:* Studio della configurazione del server esistente, stilando un inventario delle dipendenze tra le applicazioni.

- *Map:* Individuazione dell'hardware di destinazione e formalizzazione della richiesta all'uso tramite il portale di gestione degli asset di IBM Global.

- *Provision, Migrate and Configure:* Approvvigionamento del sistema operativo e del software necessario e migrazione del sistema (software e dati) con le dovute configurazioni del caso. Per quanto riguarda questo step, IBM considera tassativo l'appoggio ad un team dedicato esterno (il Configuration Team), in quanto il setup di un server deve seguire parametri comuni a tutti i team di IBM, parametri di cui, normalmente, un team di sviluppo non è a conoscenza, come

ad esempio le versioni dei software da installare, la configurazioni per ottimizzare le prestazioni del database, l'installazione di interfacce per il backup automatico dei dati, e gli strumenti di rete per garantire e uniformare l'accessibilità ai servizi sia al nostro team che ad un eventuale nuovo team, qualora il progetto venisse decommissionato. Per garantire quindi l'omogeneità delle configurazioni software di base è necessario contattare un team di configurazione e concordare le caratteristiche hardware e software di cui si necessita.

- *Test and Go-live:* Test del sistema sulla base di casi pianificati, e accordo con il team di configurazione per una data di go-live (di rilascio).

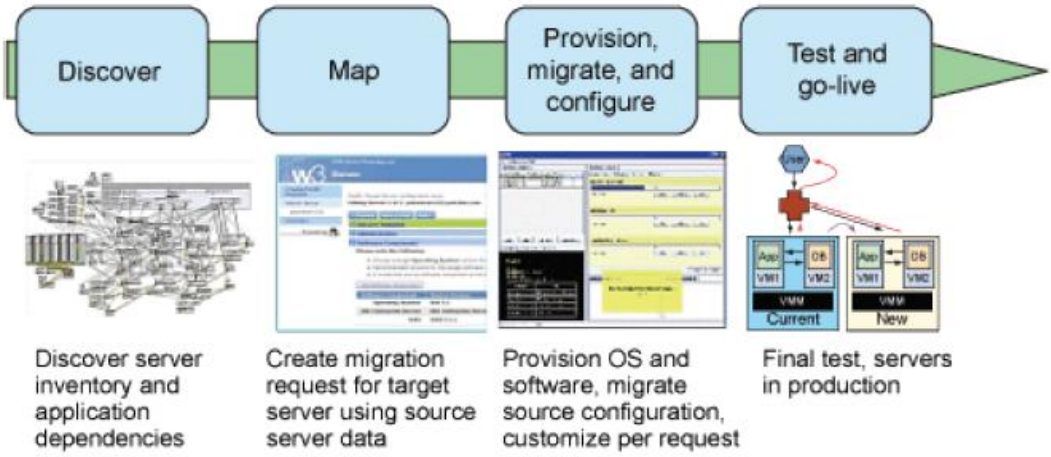

Figura 5.2. *Le diverse fasi del processo di migrazione in IBM.*

5.4 Requisiti e proposte di configurazione.

Come accennato alla fine del paragrafo 5.2, le ragioni dietro la necessità di migrare il sistema erano innanzitutto la scarsa disponibilità di memoria fisica nelle macchine satellite, e la sopraggiunta obsolescenza del sistema operativo su cui l'intero applicativo poggiava.

I requisiti quindi sono stati principalmente a livello hardware, evidenziando il bisogno di una maggiore dimensione della memoria RAM, e a livello software l'uso di un sistema operativo aggiornato.

Come team di sviluppo, le nostre richieste si sono concentrate inizialmente sulla possibilità di trasferire l'applicazione su una piattaforma di cloud computing, essendo a conoscenza dell'esistenza di una piattaforma interna ad IBM adatta a questo tipo di esigenze (Blue Cloud), che avrebbe consentito di ottenere una quantità di risorse potenzialmente illimitata pur mantenendo al minimo i costi di gestione e manutenzione. Una piattaforma di cloud computing permetterebbe di creare un'istanza virtuale di un server all'interno di un sistema performante i cui costi sono legati solamente all'effettivo utilizzo in termini di cicli di CPU, memoria utilizzata e quantità di dati trasferiti.

La nostra proposta però non trovò riscontro da parte del management del team per ragioni esclusivamente operative: al team era già stato assegnato un server dedicato, con hardware relativamente recente, per un progetto mai decollato e quindi in disuso, dal costo fisso e ben definito, in contrapposizione con un sistema di cloud computing dal costo variabile e difficilmente prevedibile.

5.5 Migrazione di IDCReport: Discover.

Operativamente, la difficoltà maggiore è stata rappresentata dalla fase di ricompilazione dell'applicazione web di front-end, e sicuramente quella a cui è stato dedicato più tempo rispetto all'intero processo di migrazione.

Il software mancava di alcune porzioni di codice sorgente che non erano presenti sul sistema di versionamento (dove vengono salvate le versioni di sviluppo del codice sorgente), e presentava dipendenze errate tra i progetti o librerie mancanti.
Inoltre, le librerie di connessione al database contenente i permessi utente andavano sostituite, e di conseguenza parte del codice riscritto, in quanto il nuovo sistema avrebbe utilizzato una nuova versione, più recente, del database IBM DB2.

5.6 Migrazione di IDCReport: Map.

Risolti, non senza difficoltà, i problemi di compilazione, abbiamo provveduto a contattare il team di configurazione per iniziare il setup del nuovo sistema.
Il nuovo Hardware messo a disposizione del team era sufficientemente performante per poter gestire la totalità del sistema, quindi unità centrale e unità satellite riposizionate su un'unica macchina, forti del fatto di avere a disposizione sul nuovo server (un IBM System X3550 M3) 64GB di RAM, più del doppio della quantità di memoria della somma delle memorie RAM installate sull'unità centrale e sulle sei unità satellite, e di un processore quad-core Intel Xeon a 3.60Ghz.
L'attività di mapping della risorse hardware si è limitata alla richiesta, tramite un apposito portale per i dipendenti IBM, di un cambio di scopo della macchina precedentemente assegnata al team.

5.7 Migrazione di IDCReport: Provision, migrate, and configure.

In accordo con il team di configurazione, è stato deciso di suddividere la capacità computazionale del server in più “fette” dedicate, ovvero delle virtualizzazioni completamente isolate tra loro, in modo da garantire completa indipendenza tra la componente di archiviazione dei dati sui permessi di accesso (il database DB2), l’applicazione web di front-end, e replicare su un singolo server la struttura a satelliti utilizzata per far generare i report tramite la componente di BIRT a runtime.

Su ognuno degli otto server virtuali, creati dal team di configurazione usando un software apposito (VMWare), sono state installate altrettante copie del nuovo sistema operativo. Ogni virtualizzazione funziona in tutto e per tutto come un server dedicato, solamente che le risorse hardware vengono suddivise tra le otto virtualizzazioni ed allocate dinamicamente in base alla richiesta delle applicazioni in esecuzione sul ciascun server virtuale.

La migrazione fisica dei dati è avvenuta consegnando i singoli progetti Java al team di configurazione e i dati di accesso al database preesistente, il quale si è occupato di installare il software all’interno delle virtualizzazioni e di creare una replica del database contenente le informazioni di accesso della componente a front-end.

5.8 Migrazione di IDCReport: Test e Go-live.

La scelta di utilizzare server virtuali si è rivelata ottimale per far fronte alle esigenze iniziali. Grazie alla virtualizzazione, che alloca dinamicamente le risorse di memoria ai processi che ne richiedono l’uso, abbiamo potuto configurare le JVM con dimensioni massime di heap space maggiori (3GB).

Le migliorate capacità computazionali del processore, ed il fatto che il nuovo server fosse collegato a una rete a più alte prestazioni, hanno permesso di abbattere i tempi di I/O e quelli necessari alla codifica dei dati, inoltre il mantenimento della struttura a unità satellite, seppur virtuali ma con diversi indirizzi IP ciascuna, ci ha consentito di poter ancora accedere ai database di Avaya, anche nel caso in cui due unità satellite distinte stiano usando contemporaneamente il database stesso.
Il nuovo sistema utilizza virtualizzazioni del sistema operativo Windows XP, ora conforme alle richieste di IBM Global.

Prima della data di go-live sono stati definiti una serie di test, per valutare sia le prestazioni dei singoli report (il tempo totale di esecuzione), sia la loro buona riuscita, e un controllo sulla consistenza dei dati ottenuti.
Abbiamo quindi creato uno spreadsheet excel contenente una riga per ogni report di cui eseguire il test, e nelle cui colonne evidenziavamo:

- Il nome del report.
- Breve descrizione del report.
- L'esito sul nuovo sistema (OK/FAIL)
- Tempo di esecuzione in secondi sul vecchio hardware
- Tempo di esecuzione in secondi sul nuovo hardware
- Congruenza dei dati ottenuti (OK/NO), ovvero un controllo che, per il dato report, i risultati ottenuti fossero i medesimi su entrambi i sistemi.
- Uno spazio per le note nel caso non fossero congruenti.

I test sono stati eseguiti dai rispettivi 'owner' dei report, ovvero il componente del team che si è inizialmente occupato dello sviluppo dello stesso.
I test hanno evidenziato un miglioramento generale delle prestazioni medio di circa il 25%, e la quasi totale scomparsa dei problemi dovuti all'esaurimento di memoria RAM disponibile.

6. Caso 2: Nordic Processor
Raccolta delle specifiche tecniche per l'implementazione di reportistica su Tivoli Maximo.

6.1 Obbiettivi

Il secondo caso di studio su cui vale la pena soffermarsi è quello dato dall'esperienza come analista di processo per Nordic Processor, una joint-venture tra IBM e un gruppo bancario operante nei paesi scandinavi, Nordea.

L'attività consisteva nel definire i requisiti relativi ad una serie di report precedentemente gestiti da un software di raccolta delle segnalazioni di problematiche utente e reportistica chiamato eESM, tali da poter essere riscritti in un formato adatto all'esecuzione su una piattaforma di gestione della clientela, il CRM Tivoli Maximo, con funzionalità in parte simili ma architettura sostanzialmente differente, e su cui i dati del vecchio sistema sarebbero stati migrati a breve.
La riscrittura dei nuovi report per Tivoli Maximo ha comportato la definizione di un data mapping, ovvero una mappatura dei campi all'interno del database usato dal sistema eESM con i corrispondenti campi del database di Tivoli Maximo.

Nonostante non si sia trattato di un ruolo prettamente tecnico, e sopraggiunto inizialmente come mansione 'riempitiva' tra un progetto e l'altro, si è rivelato interessante per capire quali siano i meccanismi che stanno alla base della raccolta di requisiti in ambito BI.

6.2 Requisiti funzionali di partenza

Il materiale inizialmente a nostra disposizione per poter cominciare a definire le specifiche tecniche dei vari report era sostanzialmente una serie di documenti di analisi funzionale, contenenti indicazioni sommarie sul tipo di informazioni richieste ed il tipo di layout desiderato per ogni singolo report, dove i valori indicati facevano riferimento ad attributi e definizioni proprie del software precedentemente in uso, eESM.
Ad esempio alcuni documenti funzionali contenevano descrizioni di tabelle sui trend di spesa di determinati asset, raggruppati per categoria, e con calcoli di valori limite per cui evidenziare trend particolarmente negativi.

A supporto di queste informazioni, il nostro team aveva ottenuto accesso diretto ai database di produzione del software eESM. Questo quantomeno ci ha permesso di ottenere visibilità delle tabelle e dei campi di eESM, facilitando l'interpretazione dei requisiti funzionali a nostra disposizione, pur non avendo alcuna visibilità della parte a front-end del software stesso, in quanto eESM è un software sviluppato da una società estranea ad IBM.

Per quanto riguarda il sistema di destinazione invece, Tivoli Maximo, avevamo a disposizione la documentazione completa, sia delle della struttura del data model del database interno, sia delle funzionalità accessibili a front-end dell'applicativo.

6.3 Differenze tra i data model di Tivoli Maximo e di eESM

Basandosi sui soli data model, è stato chiaro fin da subito che Tivoli Maximo presentava uno schema di tabelle molto più strutturato e vasto rispetto a quanto osservabile sul corrispettivo data model di eESM, le cui funzionalità, seppur simili, rappresentano verosimilmente un sottoinsieme di quelle disponibili nel CRM Tivoli Maximo.
Ciò è supportato anche dal numero di tabelle presenti nei due data model: il data model di Tivoli Maximo era composto da 905 tabelle, quello di eESM era formato da sole 94 tabelle.

Non tutte le 905 tabelle di Tivoli Maximo erano interessate dai report oggetto della raccolta di requisiti tecnici, come nemmeno tutte le 94 tabelle di eESM, ma le similitudini tra i due sistemi erano concentrate soprattutto su un piccolo insieme di tabelle principali.

Le tre tabelle di maggiore interesse nel sistema eESM erano:

- 'PROBLEMS', contenente in forma aggregata tutte le informazioni relative al ticket inserito: richiedente, tipologia della richiesta, date di apertura/risoluzione/chiusura, stato del ticket (aperto/chiuso/in attesa), priorità etc.
- 'CHANGE', contenente lo storico delle operazioni effettuate per la risoluzione del problema e indicazioni sui gruppi tecnici di manutenzione a cui la problematica veniva data in carico.
- 'INVENTORY', con i dettagli, qualora ammissibili, sulle singole unità hardware o software coinvolte nel problema.

A sua volta, il data model di Tivoli Maximo conteneva tabelle che ricalcavano i medesimi insiemi di dati, ma con una terminologia ed una organizzazione in termini di tabelle e chiavi di riferimento differente.

In Tivoli Maximo la tabella ‘PROBLEMS’ di eESM era riconducibile all’insieme delle tabelle ‘TICKET’ e ‘CLASSSTRUCTURE’, mentre i dettagli relativi alla segnalazione, anch’essi definiti nella tabella ‘PROBLEMS’ di eESM, venivano nettamente suddivisi in service requests e problemi, rispettivamente le tabelle ‘SR’ e ‘PROBLEM’ nel data model di Tivoli Maximo. (Tab. 6.1):

- ‘TICKET’ descrive i dettagli generali relativi alla segnalazione ricevuta: orario di creazione, orario di prevista ed effettiva chiusura, stato del ticket (aperto, chiuso, in attesa), priorità interna e riferimenti ad eventuali componenti hardware o software coinvolte.
- ‘CLASSTRUCTURE’ contiene informazioni relative alla tipologia del problema, decodificando lo schema SCIM di cui esiste un parallellismo nella tabella PROBLEMS di eESM. Lo schema SCIM definisce la tipologia del problema secondo i valori di System, Component, Item e Module (quest’ultimo opzionale). Le definizioni della tipologia SCIM possono essere ad esempio:

  “NETWORKS \ CONNECTIVITY \ Notes Database \ Time Out”
  “OS SOFTWARE \ OS AIX \ Tivoli License Compliance \ Manager”
  “HARDWARE \ MAINFRAME \ Physical Damage \ Peripheral”

- ‘SR’ acronimo di “Service Request”. E’ sostanzialmente una sottotipologia di ticket, da coinvolgere nel caso in cui la segnalazione non sia relativa ad un problema o malfunzionamento, ma piuttosto ad una richiesta di modifica di un sistema preesistente, come ad esempio un aggiornamento software.
- ‘PROBLEM’ invece contiene le informazioni specifiche su segnalazioni di malfunzionamenti, con riferimenti ai gruppi di lavoro dedicati alla risoluzione del problema.

La tabella ‘CHANGE’ in eESM invece aveva una corrispondenza nelle tabelle ‘WORKORDER’ e ‘WOCHANGE’ di Tivoli Maximo:

- ‘WORKORDER’ è un registro delle fasi di lavoro dei gruppi di assistenza tecnica, tipicamente operatori in loco, per cui è prevista una data di inizio e fine lavoro, costi orari, costi fissi, dettagli sul tipo di lavoro svolto e indirizzo fisico del luogo dove la prestazione è stata effettuata.
- WOCHANGE usata nel caso in cui una fase del lavoro di risoluzione del problema venga affidata ad un gruppo di lavoro diverso da quello originariamente assegnato. Questa tabella contiene i dettagli relativi al nuovo team assegnatario e le motivazioni del passaggio di mano.

La tabella ‘INVENTORY’ di eESM e la tabella ‘ASSET’ di Tivoli Maximo invece contengono in linea di massima lo stesso tipo di informazioni, ma con denominazioni diverse. Si tratta di dettagli sulle unità hardware o software coinvolte nella segnalazione.

eESM		***Tivoli Maximo***	
Tabella	***Descrizione***	***Tabella***	***Descrizione***
PROBLEMS	Contiene in forma aggregata le informazioni salienti relative ad una segnalazione pervenuta all'help desk.	TICKET	Informazioni generali sulla segnalazione ricevuta, provenienza e priorità interna.
		CLASSSTRUCTURE	Tabella contenente la tipologia del problema secondo lo schema SCIM (System, Component, Item, Module)
		SR	Service Requests, sottotipologia di ticket per richieste di prestazioni ordinarie non causate da problemi
		PROBLEM	Informazioni sul problema segnalato dall'utente.
CHANGE	Dettaglio degli eventi scatenati a fronte della ricevuta segnalazione, come ad esempio la presa in carico di un ticket da parte di un determinato team tecnico.	WORKORDERS	Registro delle fasi di lavoro dei gruppi di assistenza tecnica
		WOCHANGE	Registro passaggi di competenze tra i diversi team di assistenza tecnica assegnati alla risoluzione della segnalazione.
INVENTORY	Specifiche hardware/software delle unità oggetto della segnalazione.	ASSET	Specifiche hardware/software delle unità oggetto della segnalazione.

Tabella 6.1. *Schema corrispondenze tra le principali tabelle di eESM e di Tivoli Maximo*

L’attività di data mapping, che consisteva nel definire per ogni campo presente in un data model il corrispettivo campo (o l’insieme di campi) del data model di destinazione, ha poi coinvolto numerose altre tabelle dell’uno e dell’altro data model, ma quelle appena elencate sono sicuramente le più significative.
Per conoscere il significato dei singoli campi di ognuna delle tabelle considerate, in modo da creare delle associazioni logiche valide tra i campi dei due diversi data model, io e un collega abbiamo dovuto sostenere numerosi meeting nella sede di IBM di Stoccolma con i componenti del team della Nordic Processor responsabili della migrazione fisica dei dati pregressi presenti sul sistema eESM in via di dismissione nel nuovo CRM Tivoli Maximo, e che come noi stavano cercando di definire un data mapping, nel loro caso per poter trasferire su Tivoli Maximo la maggiore quantità di informazioni possibile trasferibile a partire da eESM.

6.4 Scopo dell’attività di raccolta requisiti tecnici

L’obbiettivo è quello di riuscire a definire, a partire dalle informazioni a nostra disposizione, le specifiche tecniche necessarie all’implementazione di tutti i report che venivano precedentemente eseguiti sul sistema eESM, formulate per essere codificate nel formato rptDesign di Eclipse BIRT. Il sistema di destinazione Tivoli Maximo infatti, è dotato di un BIRT engine interno in grado di generare report a partire da file di configurazione rptDesign caricati a sistema, e di gestire la loro visualizzazione all’interno dell’applicativo stesso, a patto che le sorgenti di dati siano le tabelle del database del software di CRM (Nell’eventualità che le sorgenti dei dati fossero esterne al sistema Tivoli Maximo ciò non sarebbe possibile, ma nel caso in esame il problema non si pone).

6.5 Metodologie di raccolta requisiti, l'IBM Global Delivery Framework.

L'attività di raccolta requisiti in IBM rientra in un quadro metodologico più generale denominato IBM Global Delivery Framework (GDF).
L'IBM GDF è, nella visione della società, un insieme di direttive di produzione che mirano a migliorare la qualità dell'offerta al cliente, tali da raggiungere risultati d'eccellenza e al tempo stesso ridurre i costi operativi.

Esiste un'ampia documentazione interna sull'argomento, in larga parte protetta da clausole di non diffusione, ma che verte principalmente su tre aree di interesse: i sistemi operativi (intesi in senso di operatività generale), i sistemi gestionali, ed un corollario su comportamenti, impostazione mentale e capacità richieste.

In particolare per l'area GTS - Global Technology Services - la strategia GDF si ramifica in più componenti: esistono indicazioni riguardanti la coordinazione tra gruppi produttivi (Cross-Pool Coordination), la standardizzazione del processo di preventivazione (Delivery Catalog component), il monitoraggio dei processi produttivi attraverso strumenti analitici (Analytics component), la prevenzione di bug e difetti (Defect Prevention component) e le regole da seguire per l'assegnazione e la suddivisione del carico di lavoro (Dispatch component). Anche la raccolta requisiti è definita da una sua componente, Requirements Gathering Component (RGC).

L'intero framework si basa sui concetti di lean development, ovvero un approccio iterativo alla risoluzione dei problemi, ma applicato ad ogni declinazione del processo produttivo, non solo sugli aspetti riguardanti lo sviluppo tecnologico.

6.6 Lean Development

La definizione di lean development, in riferimento allo sviluppo software, prevede una serie di linee guida che potrebbero essere sintetizzate come segue:

- Prioritarizzazione delle funzionalità chiave e graduale eliminazione del codice obsoleto o non necessario.
- Pianificazione, in fase di raccolta requisiti, anche dei relativi casi d'uso e test di integrità funzionale, onde evitare l'impatto del reworking sui tempi di consegna.
- Nei casi di indecisione sulla forma o sulla corretta interpretazione di un requisito funzionale, rimandare la decisione al più tardi possibile in modo tale da incrementare la base di informazioni utile a definire correttamente il requisito
- Definire obbiettivi misurabili. Uno dei fattori chiave della corretta catalogazione dei requisiti è la sottrazione di qualunque forma di astrazione dalla definizione dei requisiti, che devono presentarsi in maniera concreta e misurabile.
- Considerare l'implementazione di prototipi delle funzionalità. I prototipi, che possono essere costituiti da implementazioni a codice oppure da semplici descrizioni diagrammatiche, aiutano ad individuare sia le caratteristiche funzionali non immediatamente visibili che i limiti di fattibilità della funzionalità stessa.

6.7 Requirements Gathering Component

La componente sulla raccolta dei requisiti della GDF (RGC) contiene una serie di linee guida cercano di risolvere il problema dell'elusività dei requisiti architetturali tramite un approccio sistematico, innanzitutto distinguendo le varie fasi che portano alla definizione dei requisiti funzionali (Fig. 6.1):

- *Customer Requirement Analysis*: Comprensione e analisi dei requisiti del cliente.
- *User Requirements Specification*: Specifica a basso livello dei requisiti utente, determinando limiti e possibilità dell'ambiente di destinazione.
- *System Design*: Formulazione dei requisiti o delle modifiche architetturali (ad es. del data model) necessarie per far fronte al requisito.
- *Proposal*: La formulazione di una proposta della specifica tecnica.
- *Functional Design Specification*: A fronte dell'approvazione dell'utente, la stesura di una prima versione della specifica.

La stesura della specifica concordata non è un atto definitivo, ma potrebbe essere riconsiderata e modificata durante le fasi successive, dovessero sorgere complicanze che non erano state prese in considerazione in precedenza:

- *System Development*: lo sviluppo vero e proprio, a livello di codice, della specifica fornita.
- *Commissioning*: La fase di verifica che il risultato dello sviluppo sia conforme ad obbiettivi e specifiche.

Uno dei punti fondamentali inclusi nella componente di raccolta requisiti della GDF prevede l'estensione del ciclo di vita della raccolta dei requisiti

fino al termine della realizzazione dell'applicativo o del servizio, anziché limitare questa fase al solo lasso di tempo antecedente l'inizio dello sviluppo. Posticipando le decisioni su ipotesi incerte, si può dare il tempo all'utente di comprendere meglio le sue necessità, ed evitare di investire in soluzioni tecnologiche che sarebbero limitanti o complicate da sostituire in un momento successivo.

Inoltre, in linea con l'approccio di lean development a cui GDF si ispira, è prevista la definizione di diverse priorità tra i requisiti, in modo tale da poter avviare lo sviluppo nel più breve tempo possibile.

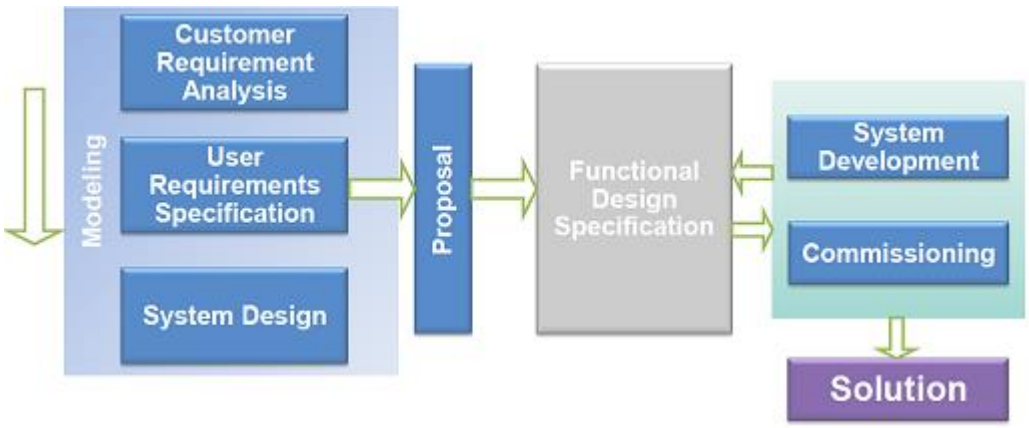

Figura 6.1. *Le fasi dell'acquisizione dei requisiti funzionali.*

6.8 Concretizzazione delle specifiche tecniche

Nel caso in esame le direttive suggerite dal Requirement Gathering Component del framework GDF sono state seguite in modo abbastanza fedele, in particolare i punti che prevedono di posticipare la formulazione della specifica tecnica fino alla totale e precisa definizione del requisito funzionale e quello che riguarda la pianificazione dei casi di test.

Inizialmente, in accordo con le informazioni raccolte con il team svedese che si occupava della migrazione dei dati pregressi da eESM a Tivoli Maximo, abbiamo proceduto progressivamente alla compilazione del data mapping, la tabella associativa tra i campi del data model di eESM e i corrispettivi in Tivoli Maximo.

In parallelo, abbiamo aggiornato un documento, detto 'Report Tracker', in cui venivano annotate tutte le richieste di chiarimenti sui singoli requisiti funzionali e le relative risposte.

Anche qui l'approccio lean nell'acquisizione dei requisiti funzionali e nella stesura del data mapping si è rivelato fondamentale in quanto, come previsto, le differenze tra i due sistemi erano sostanziali e le specifiche tecniche necessitavano di frequenti e talvolta radicali adattamenti che andavano continuamente discussi con l'utente finale.

Il risultato del lavoro di raccolta delle specifiche si è tramutato in 105 documenti, uno per ogni report richiesto, in cui si descriveva l'obbiettivo del report contestualizzato ai valori degli attributi e alla terminologia del CRM Tivoli Maximo, con descrizioni dettagliate dei prospetti delle tabelle risultanti dall'esecuzione dei report. A margine di ogni documento di specifica tecnica veniva fornita inoltre la lista degli attributi coinvolti e una bozza della (o

delle) query SQL da eseguire sul database interno di Tivoli Maximo per ottenere i dati grezzi.

Ogni documento di specifica tecnica si componeva di cinque parti.

- La descrizione del requisito così come fornito dal cliente
- Una spiegazione del significato degli acronimi, dei valori e dei concetti formulati nel requisito di cui sopra
- La formulazione della proposta di requisito tecnico, inclusa la lista dei campi e delle tabelle coinvolte in Tivoli Maximo.
- Un modello grafico del layout del report che si desiderava ottenere
- Una bozza della query SQL necessaria all'implementazione della specifica.

I documenti risultanti così potevano indirizzare lo sviluppatore che aveva in carico l'implementazione del report in maniera molto più chiara. Ad esempio un documento di cui inizialmente si conosceva solamente la dicitura interna (BI_100), una brevissima descrizione (IMAC weekly change reports) e una lista dei nomi dei campi risultanti così come disponibili sul report di eESM, è stato 'tradotto' in una specifica contenente una descrizione esplicativa:

"Weekly change report showing all planned IMAC changes with risk category 'critical', 'major' and 'medium' from Friday at 00:00 until next Thursday at 24:00"

Una spiegazione del significato dell'acronimo IMAC:

"IMAC changes are Service Requests assigned to an NDI group, NDI refers to licensed service groups."

La formulazione della proposta di requisite tecnico:

The report should gather the list of allowed NDI groups IDs from the IMAC database, but since there's no visibility of the IMAC DB from the Maximo report engine (local drivers only), and considering the overall un alteration of such groups over time, it's been agreed to hardcode the list in the report (35 items, see file attached) .

The "START_DATE" should be fetched only from the latest recorded item in the WOCHANGE table.

Risk code should be codified as follows:

- Medium -> 4
- Major -> 5
- Critical -> 6

Records involved must include a start date ranging from the latest Friday at 00.00 (compared to system time) to the next Thursday at 24.00.

Population of the other fields (CHANGE_ID, APPROVAL_STATUS, ASSIGNEE_NAME and CHANGE_ABSTRACT) should be a trivial task.

E un modello esemplificativo del report risultante:

Change ID	Risk	Approval Status	Start Date	Assignee Group	Assignee Name	Change Abstract
2945632	6	Approved	14/08/2012	NDI-I-NDSSOSEDB	JOHAN RAEDER	SE – APPLICATON CHANGES PPLEX
2930614	5	Approved	18/08/2012	NDI-I-NDMIDAS	ERIK STRÖMBERG	NO, SE: DEPLOY NEW VERSION OF NORA2 (1.3.0) ON MIDAS IN PRODUCTION
2589922	5	Approved	17/08/2012	NDI-I-NDSSOSEII	JON KETTEL THOMSEN	DK – 5258 / ONLINE INVESTERING RELEASE 7.1.1.0

Queste informazioni in genere erano sufficienti a far si che lo sviluppatore potesse proseguire all'implementazione del report senza particolari problemi

Completata la definizione del data mapping, ed ottenute tutte le informazioni sui requisiti, il lavoro di implementazione è stato relativamente semplice: si è trattato di perfezionare, caso per caso, le query SQL necessarie all'ottenimento dei dati necessari, usando poi il BIRT designer per procedere all'incrocio dei dati, la loro raffinazione e la formulazione dei calcoli sui dati aggregati per ottenere il risultato finale.

7. Conclusioni

Nel presente documento ho cercato di mettere in luce quanto l'esperienza lavorativa presso il team di Business Intelligence & Metrics IBM di Dublino sia servita per comprendere meglio il concetto di BI all'interno di una realtà lavorativa, e quanto l'insieme di metodologie proprie della BI venga di volta in volta plasmato alle esigenze del cliente e delle circostanze in cui esso opera.

Nonostante il limitato numero di casi affrontati, che sicuramente non possono e non vogliono coprire l'intera gamma di argomenti che afferiscono al tema della Business Intelligence, ho notato come spesso sia necessario lavorare ad un livello di astrazione superiore a quello a cui ero abituato, ma anche quanto sia importante disporre di una conoscenza approfondita dello strumento di Business Intelligence in uso nel team e di una gamma di abilità tecniche il più ampia possibile per potersi prendere carico della risoluzione dei problemi che possono sorgere a livello software.

Bibliografia

[1] Larissa T. Moss, S. Atre, “Business Intelligence Roadmap”, Addison-Wesley Professional, 2003

[2] M. Biere, “Business Intelligence for the Enterprise”, Prentice Hall IBM Press, 2003

[3] R. Battiti, M. Brunato, “Reactive Business Intelligence. From Data Models to Insight”, Reactive Search, 2011

[4] P. Bapoo, *The complete getting started guide to BIRT reporting*, birtreporting.com, 2009

[5] The Eclipse Foundation, *BIRT integration technical reference*, in www.eclipse.org/birt/phoenix/ref/

[6] P.N. Tan, M. Steinbach, V. Kumar, *Introduction to Data Mining*, Addison Wesley, 2006