Il Metodo COSMIC di Misurazione

della Dimensione Funzionale

Versione 3.0

Linea Guida per Misurare il Software Applicativo Aziendale

VERSIONE 1.1

Maggio 2008

Ringraziamenti

Autori della Versione 1.0 e revisori 2005 (ordine alfabetico)		
Alain Abran, École de Technologie Supérieure, Université du Québec, Canada	Ton Dekkers Sogeti Olanda	Jean-Marc Desharnais, Software Engineering Lab in Applied Metrics – SELAM, Canada
Peter Fagg Regno Unito	Arlan Lesterhuis, Sogeti * Olanda	Roberto Meli, Data Processing Organization, Italia
Pam Morris, Total Metrics Australia	Serge Oligny, Bell Canada Canada	Marie O'Neill, Software Management Methods, Irlanda
Tony Rollo, Software Measurement Services, Regno Unito	Grant Rule, Software Measurement Services, United Kingdom	Luca Santillo, Agile Metrics Italia
Charles Symons * Regno Unito	Hannu Toivonen, Nokia Siemens Networks, Finlandia	Frank Vogelezang, Sogeti Olanda

Autori della Versione 1.1 e revisori 2008 (ordine alfabetico)		
Alain Abran, École de Technologie Supérieure, Univ. Québec, Canada	Arlan Lesterhuis, Sogeti * Olanda	Marie O'Neill, Software Management Methods, Irlanda
Luca Santillo, Agile Metrics Italia	Charles Symons * Regno Unito	Hannu Toivonen, Nokia Siemens Networks, Finlandia

*Autori delle versioni 1.0 e 1.1.

Oltre a quanto riportato, ulteriori commenti provenienti dal lavoro di traduzione in giapponese della v. 1.0 della Linea Guida sono stati di ausilio per la produzione della v. 1.1. Tali commenti sono stati sottomessi da:

- Shin-ichi Nagano, NTT East, Giappone
- Taku Fujii, OGIS-RI Co., Ltd, Giappone
- Noboru Hirabayashi, Fujitsu, Giappone
- Yasunori Nagatani, Kinkei System Corp., Giappone.

Gli autori intendono esprimere il loro apprezzamento a Rabobank Nederland e a Sogeti Nederland b.v. per la loro assistenza e contributo allo sviluppo della Linea Guida e a Software Measurement Services Ltd per l'uso di alcuni esempi per i casi di studio.

Traduttori e revisori della versione 1.1 in Italiano, 2011-2012 (ordine alfabetico)		
Luigi Buglione*, Engineering.IT	Luigi Francavilla*	Gianfranco Lanza, CSI Piemonte
Monica Lelli*	Andrea Salvatori *, NECS	Luca Santillo, Agile Metrics
Habib Sedehi, Univ. Roma I "La Sapienza"		

* Traduttori della versione italiana.

NdT L'espressione "Business Application Software" è stata tradotta come "Software Applicativo Aziendale". Può essere considerata come sinonimo di "Software (Applicativo) Gestionale". Si è preferito non utilizzare direttamente il termine "gestionale" per riservarlo al corrispondente termine inglese "management", come in "MIS (Management Information System)", che rappresenta un'altra tipica dizione per una simile categoria di software.

Controllo di Versione

La seguente tabella riporta la storia delle versioni di questo documento.

Data	**Revisore/i**	**Modifiche/Aggiunte**
05-12-01	COSMIC Measurement Practices Committee	Rilascio della prima versione pubblica 1.0
26-05-08	COSMIC Measurement Practices Committee	Revisione v1.1 per allineamento con il Metodo COSMIC v3.0

Traduzione in italiano		
Maggio 2012	V. pag. 2	Prima traduzione pubblicamente disponibile.

Prefazione

Scopo della Linea Guida e Relazioni con il Manuale di Misurazione

Lo scopo di questa Linea Guida è quello di fornire uno strumento addizionale al Manuale di Misurazione [3], su come applicare il metodo COSMIC v3.0 per la Misurazione della Dimensione Funzionale per dimensionare il software appartenente alla categoria "software applicativo aziendale".

Il Manuale di Misurazione contiene le definizioni dei concetti, principi, regole e processi di misurazione che si espandono sulla definizione di base del metodo così come definito nello standard ISO/IEC 19761:2003 [5] [*NdT Aggiornato nel 2011*]. Il manuale contiene inoltre ulteriore testo esplicativo sui concetti, ed inoltre esempi di applicazione del metodo a software appartenente a diversi domini applicativi.

Questa Linea Guida estende il testo esplicativo e fornisce una dettagliata guida addizionale e ulteriori esempi per dimensionare il software applicativo aziendale in aggiunta alle informazioni fornite nel Manuale di Misurazione. Questo dominio software risulta particolarmente importante poiché costituisce la categoria per la quale i metodi per la misurazione della dimensione funzionale c.d. di 'prima generazione', quali IFPUG, MkII e NESMA, sono stati progettati.

Destinatari della Linea Guida

Questa Linea Guida è destinata ad essere consultata da coloro che avranno il compito di misurare le dimensioni funzionali di software applicativo aziendale secondo il metodo COSMIC in ogni fase del ciclo di vita del software. Essi saranno identificati nella Linea Guida come i 'misuratori'. Tale documento dovrebbe essere di interesse anche per coloro che devono interpretare ed utilizzare i risultati di tali misure nel contesto di misure di performance di progetto, di controllo del contratto software, stima, ecc. La Linea Guida non è perciò legata a nessuna particolare metodologia di sviluppo o ciclo di vita.

I misuratori che hanno usato un metodo di 'prima generazione' per la misurazione della dimensione funzionale e coloro i quali desiderano migrare al metodo COSMIC possono apportare molta esperienza specifica al metodo più recente. Parte di questa esperienza è rilevante per COSMIC mentre un'altra parte non lo è. Molto deve essere riappreso. Qualcuno, passando da un metodo esistente, potrebbe essere familiare con un dettagliato 'libro di ricette' con esempi su come misurare la dimensione funzionale in diverse situazioni, mentre il Manuale di Misurazione COSMIC si concentra sulla definizione di principi e regole generali, e deliberatamente evita molti esempi relativi a specifici domini.

Per applicare il metodo COSMIC al software applicativo aziendale nella fase dei requisiti o in ogni altra fase del ciclo di vita del software, si richiede altresì una buona comprensione di alcuni metodi di analisi dei sistemi, in particolare la conoscenza di metodi per l'analisi dei dati. Una difficoltà è dovuta al fatto che tali metodi siano basati su concetti che non sempre hanno una corrispondenza puntuale con i concetti del metodo COSMIC. Inoltre, i metodi di analisi dei sistemi non sempre sono usati o interpretati nello stesso modo da differenti praticanti. Ad esempio, tali metodi possono propriamente essere usati a diversi livelli di granularità dei requisiti software. Ma se siamo consapevoli di avere affidabili e consistenti misure delle dimensione funzionale, è necessario avere regole che aiutino ad essere in accordo su soltanto una interpretazione di ogni pezzo di funzionalità assegnata. Per tale motivo, questa Linea Guida descrive la mappatura di alcuni concetti di certi metodi di analisi dei dati con i concetti del modello COSMIC. Essendo specifici di un dominio, tali mappature non appartengono al Manuale di Misurazione.

Si assume che i lettori di questa Linea Guida abbiano familiarità con il Manuale di Misurazione COSMIC, versione 3.0 e di ogni altro associato 'Bollettino di Aggiornamento del Metodo' (MUB, Method Update Bulletin) (ottenibili dal sito www.cosmicon.com). Per facilità di manutenzione, vi è una limitata duplicazione di contenuti tra il Manuale di Misurazione e questa Linea Guida.

Ambito di applicazione della presente Linea Guida

Il contenuto di questa Linea Guida è principalmente teso ad essere applicato al dominio del software applicativo aziendale. Tale dominio include software spesso descritto come 'applicazioni per l'elaborazione dei dati aziendali, 'per l'elaborazione delle transazioni applicative', 'sistemi per la gestione delle informazioni', 'sistemi di supporto alle decisioni'.

Per una descrizione completa di cosa caratterizzi il software applicativo aziendale, si veda la sezione 1.1[1]

Il contenuto della presente Linea Guida potrebbe essere applicabile ad una più vasta gamma di tipologie di software rispetto a quelle conosciute come 'applicazioni aziendali'. Questa gamma più vasta potrebbe essere descritta come 'ogni applicazione software progettata per l'uso degli utenti umani, ad eccezione dei tool software di tipo *general-purpose* (a volte anche descritti come 'applicazioni') quali elaboratori di testo, fogli di calcolo ed altri simili'. Esempi per i quali la Linea Guida potrebbe essere applicata potrebbero includere il software usato da operatori umani per impostare i principali parametri di controllo e monitorare le performance dei sistemi real-time, ad esempio per il processo di controllo o per i sistemi di telecomunicazione..

In ogni caso, fino a quando non sarà raggiunta una maggiore esperienza, il Common Software Measurement International Consortium (COSMIC) ha a cuore che le regole puntuali e dettagliate di questa Linea Guida siano applicabili al dominio del software applicativo. Feedback su esperienze pratiche su tali dominio e su altri domini saranno benvenuti (v. App. A, richieste modifica e commenti).

Introduzione ai contenuti della Linea Guida

La Linea Guida focalizza l'attenzione su esempi che illustrano i principi e le regole del Manuale di Misurazione e che aiutano ad interpretarle nel dominio del software applicativo aziendale. Tali principi e regole non sono duplicati nella Linea Guida, ad eccezione di dove ritenuto necessario al fine di supportare una nuova regola o a titolo di esempio nella Linea Guida.

In generale, per le definizioni dei termini del metodo COSMIC, si può fare riferimento al glossario nel documento Panoramica della Documentazione e Glossario [1]. Termini specifici per il dominio del software aziendale si possono trovare nel glossario alla fine della presente Linea Guida.

I capitoli 1 e 2 della Linea Guida forniscono materiale introduttivo progettato per assistere i misuratori nell'identificare i requisiti funzionali utente ed in particolare i requisiti relativi ai dati necessari per la misurazione partendo da artefatti software che si incontrano nella pratica professionale.

Il Capitolo 1 illustra cosa caratterizza la funzionalità del software applicativo aziendale e tratta alcuni aspetti di come i requisiti utente funzionali siano sviluppati e siano rilevanti per la misurazione.

Il Capitolo 2 definisce differenti livelli di analisi dei dati e descrive successivamente la mappatura tra i tre metodi di analisi dei dati e i concetti COSMIC più largamente usati, ovverosia l'Analisi Entità – Relazione (E/RA, o E/R), diagrammi delle classi dello Unified Modeling Language (UML) e l'Analisi Relazionale dei Dati (RDA). Sono inoltre presentate regole aggiuntive per l'identificazione degli oggetti di interesse.

Si consiglia vivamente ai misuratori di assicurarsi di aver ben compreso questi due capitoli prima di affrontare i capitoli 3 e 4. Questi ultimi forniscono una serie di indicazioni pratiche e svariati esempi su come applicare il processo di misurazione del metodo COSMIC al software applicativo aziendale, rispettivamente per le fasi di Strategia di Misurazione e di Mappatura e Misurazione.

[1] Un Rapporto Tecnico ISO [6] descrive due modelli formali che aiutano a distinguere il dominio del software applicativo aziendale da altri domnini del software.

Introduzione alla nuova versione 1.1 della presente Linea Guida

Questa nuova versione 1.1 allinea questa Linea Guida alla versione 3.0 del metodo COSMIC per la misurazione della dimensione funzionale, come definito nel documento 'COSMIC v3.0:Manuale di Misurazione', [3].

La versione 3.0 del metodo COSMIC introduce diverse semplificazioni. Ad esempio il suffisso '-FFP' è stato eliminato dal nome del metodo, e l'unità di misura è passata da 'Cfsu' a 'CFP' (COSMIC Function Point). E' stato introdotto il concetto di 'utente funzionale', una semplificazione che indica che il concetto di 'Punto di vista dell'utente finale della misurazione' potrebbero essere eliminato.

Tali cambi ed ulteriori suggerimenti per migliorie editoriali hanno reso necessario pubblicare una versione aggiornata di questa Linea Guida. Per un riassunto dei principali cambi apportati nella produzione della versione 1.1 dalla versione 1.0 della Linea Guida sulle applicazioni aziendali, si consulti l'Appendice B.

Va sicuramente enfatizzato che con l'aggiornamento della versione 3.0 del metodo COSMIC, non sono stati apportati cambiamenti alle regole di base per il dimensionamento funzionale. Di conseguenza, in questa versione 1.1 della Linea Guida sulle applicazioni aziendali, tutte le risposte agli esempi sono identiche a quelle date nella versione 1.0 della Linea Guida (sebbene la descrizione di alcuni esempi sia stata leggermente modificata al fine di renderli più chiari, in seguito ai commenti ricevuti dagli utilizzatori del metodo COSMIC).

Indice

1

LA FUNZIONALITÀ DEL SOFTWARE APPLICATIVO AZIENDALE

1.1 Caratterizzazione del software applicativo aziendale

Il 'software applicativo aziendale' si distingue per le seguenti caratteristiche:

- Lo scopo primario del software applicativo aziendale è quello di catturare, memorizzare e rendere disponibili i dati circa le proprietà e le transazioni nel mondo aziendale (sia nel settore privato che nel settore pubblico) in maniera tale da supportare le indagini e fornire informazioni per il processo decisionale.
- La funzionalità tende ad essere dominata dalla necessità di memorizzare i dati aziendali di diversa complessità strutturale, e a garantire l'integrità e la disponibilità di tali dati per lunghi periodi di tempo.
- Gli utenti funzionali del software applicativo aziendale sono in larga parte esseri umani, che interagiscono principalmente con il software tramite dispositivi di inserimento dati e di visualizzazione; questo significa che gran parte della funzionalità è dedicata alla gestione degli errori dell'utente umano e per fornire supporto per un uso efficiente del software. Oltre agli esseri umani, ogni altra applicazione simile, o componenti più grandi di altre simili applicazioni che interagiscono con l'applicazione che deve essere misurata saranno anch'esse utenti funzionali dell'applicazione. Si veda la sezione 3.2.1 per una caratterizzazione più precisa del significato di 'utente funzionale'.
- Diverse applicazioni aziendali possono interagire tra loro (ad esempio per scambio dati) sia in modalità on-line che batch.
- I dati vengono solitamente memorizzati in maniera storicizzata, ovverosia dopo eventi accaduti nel mondo reale, con tempi di risposta online appropriati per l'interazione umana. I dati possono essere anche processati in modalità batch. Tale dominio non include software usato per controllare eventi nel mondo reale in tempo reale. Comunque, i software aziendali possono ricevere dati in tempo reale, ad esempio i prezzi in un supermercato e possono essere costretti a rispondere rapidamente (da notare che i contenuti della presente Linea Guida non includono alcuna discussione sulla misurazione di specifici aspetti real-time del software aziendale).
- Sebbene le regole di business che governano la manipolazione di dati possano essere logicamente complesse, il software aziendale raramente coinvolge grandi moli di funzioni matematiche complesse.
- Il software applicativo aziendale è presente in uno strato (come definito nel metodo COSMIC), definito 'strato applicativo' (*application layer*). Invariabilmente, il software nello strato applicativo dipende dal software in altri strati, come ad esempio il sistema operativo, i driver dei dispositivi esterni, ecc., per il suo funzionamento.
- Una porzione di software che è considerata dai suoi utenti funzionali umani come un singolo software aziendale, potrebbe essere composta di diverse 'componenti alla pari' che potrebbero essere distribuiti su differenti elaboratori (dove ogni componente alla pari risiede nello strato applicativo del proprio elaboratore). Sebbene questa struttura sottostante non sia visibile agli utenti funzionali umani, se lo scopo delle misurazioni è quello di fornire le dimensioni come input ad un processo di stima, potrebbe essere necessario definire dei differenti scopi di misurazione per ogni componente alla pari, per esempio, quando differenti sviluppi o tecnologie di processore sono usate per ogni componente alla pari.

1.2 Requisiti Utente Funzionali (FUR)

Tutti i metodi per la misurazione della dimensione funzionale (FSM) mirano a misurare una dimensione dei 'requisiti utente funzionali' (FUR, Functional User Requirements). Eppure, nonostante le definizioni ISO, nella pratica c'è spesso incertezza rispetto al significato di cosa rappresentino i FUR. Anche i misuratori spesso hanno bisogno di una guida su questioni come 'chi sono gli utenti funzionali che dovrebbero essere presi in considerazione?', 'come si dovrebbero estrarre i FUR dagli artefatti del mondo reale?' e 'cosa fare se non fosse possibile determinarli in modo sufficientemente accurato per una data attività FSM?'

L'obiettivo di questo capitolo è quello di fornire indicazioni generali per aiutare a rispondere a domande che possono emergere nella pratica nell'applicare il metodo COSMIC nel dominio del software applicativo aziendale.

1.2.1 Il significato di 'funzionale'

'I requisiti utente funzionali' sono definiti in [13] come segue:

"Un sotto insieme dei Requisiti Utente". I requisiti che descrivono cosa il software dovrebbe fare, in termini di attività e servizi.

NOTA: I Requisiti Utente Funzionali includono, ma non sono limitati a:

- trasferimento di dati (per esempio dati del cliente in input, invio di segnali di controllo)
- trasformazione di dati (per esempio calcolo interessi bancari, derivare la temperatura media);
- memorizzazione di dati (per esempio memorizzazione degli ordini dei clienti, memorizzazione della temperatura ambientale nel tempo);
- recupero di dati (per esempio ottenere la lista degli impiegati correnti, reperire la posizione di un velivolo).

Esempi di Requisiti Utente Non Funzionali includono ma non sono limitati a:

- vincoli di qualità (per esempio usabilità, affidabilità, efficienza e portabilità);
- vincoli organizzativi (per esempio sedi di esercizio, hardware target e conformità agli standard);
- vincoli ambientali (per esempio interoperabilità, security, privacy e sicurezza);
- vincoli implementativi (per esempio linguaggio di sviluppo , delivery schedule)"

C'è anche una definizione ISO per 'requisito non-funzionale', definito come "un requisito che vincola la progettazione del software ma non descrive un servizio che il software deve fornire."

Queste due definizioni non sono sufficienti per tutti gli scopi della misurazione della dimensione funzionale. Un esempio di difficoltà che potrebbe essere incontrata nella pratica potrebbe essere un requisito relativo alla 'facilità d'uso' di un software aziendale. Questo potrebbe essere interpretato:

- come un requisito qualitativo (e quindi non come un FUR) o come un vincolo (e quindi non funzionale)
- come un requisito software implicito per un'interfaccia utente grafica (con FUR implicito) e pertanto anche come 'un servizio che il software deve fornire' (e perciò non-funzionale)

Seguendo la prima interpretazione, ogni funzionalità fornita puramente per *facilità d'uso* dovrebbe essere ignorata nella misura della dimensione funzionale del software ma con la seconda interpretazione alcune funzionalità risultanti dall'interfaccia GUI potrebbero essere considerate nella misura, in base alle regole del metodo FSM.

Ciascun metodo FSM deve perciò stabilire le proprie regole e fornire una guida su quali funzionalità dovrebbero essere misurate. Per il software aziendale, c'è un accordo generale che le principali

funzioni di elaborazione dei dati aziendali (inserimento, memorizzazione ed estrazione dei dati circa i clienti e gli ordini, ritorni delle polizze assicurative, interrogazioni sui saldi bancari, ecc.) risultano da il vero FUR e questo deve essere misurato. Tuttavia, l'incertezza può sorgere sull'opportunità o meno di misurare quelle che potrebbero essere definite funzioni del software 'secondarie' o 'generali' come ad esempio alcune funzioni di interfacce GUI, funzioni di controllo come ad esempio la funzione controllo accessi nel campo della sicurezza o di registrazione di dati, funzioni previste per facilitare la futura manutenzione del software, ecc.

Nel metodo COSMIC, tale incertezza nel definire se qualche requisito sia 'funzionale' o non, deve sempre e può essere risolta utilizzando le definizioni dei concetti basilari del metodo e dei suoi principi e regole:

Il misuratore deve sempre applicare il modello COSMIC che richiede che i FUR siano scomposti in 'processi funzionali', ciascuno consistente di 'movimenti dati', laddove un movimento dati muove un gruppo dati contenente attributi di un singolo 'oggetto di interesse'.

- I movimenti di dati:
  - ricevono in input un ‘gruppo dati’ da parte di un utente funzionale nel software (e ciò scatenare un processo funzionale), o forniscono in output un ‘gruppo dati’ dal software verso l’utente funzionale, o
  - spostano un ‘gruppo dati’ verso la memoria persistente o lo recuperano dalla memoria persistente.

  Ognuno di questi movimenti dati è da considerare ‘funzionale’ per il metodo COSMIC e dovrebbe perciò essere misurato.

- Al contrario, ogni movimento:
  - di attributi dati di qualcosa che non è un oggetto di interesse per l’utente funzionale,
  - o di qualsiasi attributo dati come passo intermedio in un processo funzionale per cui tale attributo non attraversa il confine tra il software ed i suoi utenti funzionali, né è mosso verso, o da, la memoria persistente,

  non può essere riconosciuto come movimento dati nel metodo COSMIC e deve pertanto essere ignorato per gli obiettivi della misurazione.

1.2.2 L’evoluzione dei FUR in un tipico ciclo di vita di un progetto di sviluppo software

Come indicato nella definizione di FUR nella sezione 1.2.1, una dichiarazione o derivazione dei FUR dovrebbe essere espressa al livello logico, ovverosia i FUR dovrebbero essere completamente distinti da ogni considerazione su fattori di implementazione fisica.

Nel mondo reale, è veramente infrequente cercare una definizione di FUR attuale, ‘pura’ che possa essere usata direttamente per misurare una dimensione funzionale. In genere in un progetto di sviluppo software il primo documento da produrre dovrebbe essere la definizione dei requisiti di alto livello (‘SOR’ – Statement of Requirements), contenente i requisiti utente funzionali insieme ai requisiti tecnici e qualitativi e forse supportati da un modello dati espresso a livello concettuale. Questa potrebbe essere vista come una prima descrizione del 'problema’ da risolvere.

Con l’avanzamento del progetto, nuovi fatti vengono alla luce, nuovi dettagli dei requisiti vengono trovati, le scelte economiche sono fatte su cosa includere o escludere e sul modo di soddisfare alcuni dei requisiti, ecc. Possono essere previste diverse iterazioni e ad ognuna, l'ultima dichiarazione concordata di ciò che deve essere fatto può essere vista come la 'soluzione' ultima alla comunicazione precedente del 'problema'; questa terminologia è particolarmente frequente nei metodi di sviluppo ‘agili’. Il livello di espansione della descrizione di una singola porzione di software prende il nome di 'livello di granularità'; si veda il Manuale di Misurazione per ulteriori dettagli.

Il compito degli utilizzatori di un metodo FSM non è quello di cercare di misurare sia il 'problema ' o la 'soluzione' poiché non vi è nulla di assoluto riguardo a questo modo di distinguere le fasi di un progetto software; il compito è quello di identificare e misurare i FUR del software durante la loro evoluzione nelle fasi del progetto. I metodi FSM misurano solo i FUR, non ogni aspetto di progettazione fisica o di implementazione tecnica dei FUR. Ad ogni fase è sempre possibile dedurre i

FUR del software riferiti a quella fase, anche molto tempo dopo che il software è stato installato, realizzato ed è regolarmente in uso.

Ciò significa che in ogni momento del ciclo di vita, il misuratore deve essere in grado di derivare i FUR da qualsiasi artefatto del software disponibile. Molto spesso, nelle fasi iniziali sarà disponibile solo una bozza del SOR. Successivamente quando il software comincia ad essere sviluppato ci sarà soltanto una specifica concordata e/o un documento di progetto sufficientemente aggiornato per essere utilizzato. Per una porzione di software implementata anni fa, potrebbe essere disponibile poca documentazione scarsamente aggiornata per tutte le possibili misurazioni, in genere solo il sistema installato più una guida utente.

Nulla di ciò ha a che vedere in linea di principio con i metodi FSM. Qualunque sia lo stato del software e dei suoi artefatti, è sempre possibile derivare una definizione corrispondente dei FUR *impliciti*, anche se la difficoltà di fare questo può essere in pratica alquanto variabile. Questo è il compito del misuratore, che deve avere una conoscenza approfondita dei concetti alla base del metodo FSM e di un processo per analizzare gli artefatti a disposizione e per mapparli ai concetti del metodo FSM.

Quando il processo viene applicato ad alcuni software già esistenti, esso comporta generalmente una forma di 'reverse engineering'. Un esempio tipico in cui solo il sistema installato è disponibile per la misurazione è che una singola schermata fisica può essere trovata per servire due o più separati processi funzionali, come ad esempio create e update. Un metodo FSM misura le diverse funzioni logiche del software e non i suoi artefatti fisici implementati.

Poiché i prodotti software possono esistere sotto molte forme, è impossibile descrivere un singolo processo di 'reverse engineering' dagli artefatti al modello COSMIC. In questa Linea Guida, quindi, si possono soltanto fornire diversi esempi su come interpretare requisiti software concreti e artefatti per ognuno dei concetti proposti dal modello COSMIC.

1.2.3 Requisiti qualitative e tecnici di sistema che evolvono in FUR software

Come affermato in precedenza, una tipica definizione dei requisiti di sistema contiene sia requisiti utente funzionali per il software, che requisiti tecnici e qualitativi (o 'non-funzionali') per il software e per altre risorse di sistema, e questi spesso sono derivabili dalle 'pieghe' dei requisiti generali. Un metodo FSM è soltanto interessato ai FUR del software, pertanto è necessario separare i FUR dai requisiti tecnici e di qualità durante la raccolta dei requisiti. Inoltre, poiché un progetto di sviluppo software progredisce, alcuni dei requisiti tecnici e qualitativi rimarranno invariati, mentre alcuni possono evolvere e diventare FUR del software. Gli esempi nel seguito illustrano entrambi i casi.

a) I requisiti tecnici per scrivere il software in un dato linguaggio di programmazione o per eseguirlo in un particolare centro dati rimarranno sempre requisiti tecnici. Un requisito qualitativo che il software abbia zero "difetti rilevanti" nel primo mese operativo rimarrà sempre un requisito qualitativo.

b) Una dichiarazione di FUR può definire (con molto dettaglio) che un nuovo sistema deve abilitare un cliente di un agente di borsa ad interrogare il proprio portafoglio investimenti e sul loro valore corrente via Internet. Nelle prime fasi del progetto, un tempo di risposta obiettivo è specificato quale requisito tecnico. Ulteriori studi mostrano che il requisito sul tempo di risposta può soltanto essere soddisfatto sviluppando il sistema per essere eseguito su una particolare piattaforma hardware ed anche fornendo continui *feed* dei prezzi del mercato azionario al portfolio del sistema di interrogazione. Il requisito tecnico originale è ancora valido, ma esso ora presenta un vincolo tecnico addizionale (hardware specificato) più i FUR di alcune nuove applicazioni software per ricevere i *feed* dei prezzi del mercato azionario. Quando si presenta un tale cambio, il numero di funzioni del software da realizzare aumenterà, e conseguentemente la dimensione funzionale del software che deve essere realizzato inevitabilmente crescerà[2].

[2] Assumiamo in questo esempio che la soluzione al problema di come raggiungere il tempo di risposta desiderato è stato concordato con l'utente e quindi i FUR aggiuntivi per provvedere a gestire continui *feed* dei prezzi d mercato siano veri FUR del software e non semplicemente una decisione implementativa dello sviluppatore. La dimensione die FUR aggiuntivi deve quindi essere inclusa nella dimensione funzionale dell'applicazione. La stessa considerazione è applicabile al successivo esempio c) relativamente alle funzionalità GUI.
Laddove le funzionalità si rivelano originate da una scelta implementativa dello sviluppatore, piuttosto che come il risultato diretto di un FUR implicito o esplicito, il misuratore deve determinare dallo scopo e dall'ambito stabiliti per la misurazione se è ragionevole includere la dimensione di tali funzionalità nella misurazione. In caso di dubbio, il misuratore dovrebbe sempre cercare di determinare 'i reali FUR degli utenti'. Ciò richiede un giudizio e non è un tema su cui un qualsiasi metodo FSM può fornire regole precise...

c) Il testo di un FUR definisce un nuovo sistema ed include un requisito qualitativo che afferma che esso debba essere 'facile da usare'. Come il progetto evolve, tale requisito qualitativo evolve nei FUR software di un'interfaccia utente grafica (o 'GUI'). (Una specifica dichiarazione dei FUR per la GUI in genere non viene quasi mai prodotta, dato che il team di sviluppo sa come interpretare tale requisito. Ma ciò è immateriale: se la GUI è fornita, successivamente i corrispondenti FUR possono essere ricavati.) Le caratteristiche della GUI, come liste a discesa, sono 'funzioni' del software disponibili per un utente e queste sono misurabili se le stesse coinvolgono movimenti dati relativi ad un oggetto di interesse. Il software con una GUI può avere maggiori funzionalità, e quindi una dimensione funzionale più grande rispetto ad un software che non includa tali funzionalità. Si vedano le sezioni 4.1.8 e 4.4.5 per una guida ulteriore su come poter dimensionare le funzionalità di una GUI.

Pertanto sebbene i requisiti evolvano durante l'avanzamento dei progetti e alcuni requisiti tecnici e qualitativi iniziali possano evolvere in un FUR aggiuntivo, *in ogni istante laddove una misura è necessaria* dobbiamo mirare ad essere in grado di dire "questi sono i FUR per questo software" o "questa è la sua funzionalità logica" e ciò implica che esistano taluni FUR in questo dato momento". Ad un FMS non importa che alcuni FUR del software siano stati formulati nello stesso momento dei requisiti di sistema tecnici e qualitativi.

Quando una misurazione è stata intrapresa nel contesto di un contratto software, e quando un requisito tecnico o qualitativo evolve risultando in una funzionalità software aggiunta, concordata e misurabile nell'ambito di misurazione concordato, le circostanze che conducono ad incrementare la dimensione funzionale dovrebbero essere documentate quale parte integrante della misurazione.

1.2.4 Quando i FUR non presentano sufficienti dettagli per applicare il metodo COSMIC

Nelle fasi iniziali del ciclo di vita di un progetto di sviluppo software, i requisiti tipicamente esistono soltanto ad un livello concettuale o ad un alto livello di granularità, ovverosia non molto dettagliati. Senza tale dettaglio può non essere possibile applicare il metodo COSMIC così come viene definito nel Manuale di Misurazione, ma una stima della dimensione funzionale può essere ancora necessaria, ad esempio per una decisione di investimento.

In un caso del genere, è possibile usare varianti che approssimino il metodo COSMIC. Possibili approcci per approssimare la dimensione usando COSMIC sono forniti nel documento 'Metodo COSMIC v3.0: Advanced and Related Topics' [4].

Quando si usa una variante del metodo COSMIC per una misurazione approssimata, è fortemente raccomandato di quotare la migliore stima della misura, insieme con i limiti superiori ed inferiori della misura. È una cattiva prassi. e può fornire un falso livello di confidenza, quotare un singolo numero per la dimensione quando vi è effettivamente notevole incertezza.

1.2.5 Misurazione quale controllo di qualità dei FUR

Fin troppo spesso, le dichiarazioni dei requisiti o delle specifiche non sono scritte in maniera molto chiara; esse possono essere ambigue, ci possono essere inconsistenze o omissioni e ci possono essere semplici errori. Molto spesso, il software viene costruito per la soddisfazione del cliente solo a causa di accordi verbali informali tra sviluppatori e cliente, e la documentazione non sarà probabilmente mai corretta per riflettere la realtà.

Questo stato delle cose rende il lavoro del misuratore più di una sfida ma anche potenzialmente più prezioso. Uno dei grandi benefici del metodo COSMIC è che la disciplina di applicare il metodo aiuta ad identificare le inconsistenze, ambiguità, errori ed omissioni. In altre parole il processo di misurazione è anche un processo di controllo della qualità veramente buono. Se una specifica non può essere misurata è quasi certo che in alcuni casi sia non rilasciabile. Il misuratore può poi aggiungere un gran valore tirando fuori i difetti (inconsistenze, anomalie per mancanza di chiarezza), dove essi si presentino e perché.

Frequentemente, la mancanza di chiarezza o anomalie nella documentazione, o incertezze su come interpretare il software fisicamente installato, può essere risolto parlando con un analista, un utente o con alcuni altri esperti sul software. Dove questo è impossibile, la dimensione misurata dovrebbe essere quotata con limite più alto o più basso di incertezza, come appena descritto sopra per il dimensionamento approssimato.

1.2.6 Chi può specificare i Requisiti Utente Funzionali?

Ognuna delle seguenti categorie di persone può potenzialmente generare FUR per un'applicazione aziendale.

- Utenti aziendali ('business users'), o sponsor o persone che effettivamente useranno il software, che possono specificare i FUR per le funzionalità aziendali e 'di supporto' e per fornire una futura flessibilità del business.
- Contabili e revisori dei conti, che possono specificare FUR per i criteri di validazione, funzionalità di log, controllo e tracciamento.
- Responsabili applicativi, che possono specificare FUR per il log, per l'autorizzazione di accessi sicuri, per fornire una futura facilità di manutenzione, ad esempio, richiedendo che alcuni dati che debbano essere variabili, i parametri mantenibile, e utente (errore / conferma) messaggi.
- Utenti 'non aziendali' ('non-business users'). Tali utenti possono specificare i FUR per processi funzionali di memorizzazione e/o per l'elaborazione dei dati circa gli oggetti di interesse per lo staff non aziendale, per esempio processi funzionali per mantenere i dati per il controllo dei media. I media fisici sono gli oggetti circa i quali i dati devono essere gestiti. Altri esempi potrebbero essere i FUR relativi ai processi funzionali per il backup o per la conversione e FUR per messaggi tecnici (errore/conferma) che sono rilevanti per gli utenti aziendali[3].

[3] Si noti che i FUR di un'applicazione aziendale non dovrebbero comprendere funzionalità che possono essere 'di interesse' per lo staff non aziendale, ma che forniscono una soluzione tecnica per un requisito di un utente aziendale. Un esempio di questo tipo potrebbe essere un processo funzionale che gestisce un file con conteggi intermedi, configurato per accelerare il processo di reportistica per gli utenti aziendali.

2

INTRODUZIONE ALL'ANALISI DEI DATI

Si assume che il lettore abbia familiarità con le definizioni di 'oggetto d'interesse', 'gruppo di dati', 'movimento di dati', e 'attributo di dati ' (o il suo sinonimo 'elemento di dati ') come indicato nel Manuale di Misurazione. Quando si usano questi termini occorre rammentare che i riferimenti di 'oggetto d'interesse', ecc. riguardano i tipi di questi , non gli eventi (o istanze). Solo quando sarà necessario distinguere i tipi di eventi, il manuale farà una esplicita distinzione.

Il metodo COSMIC si basa sulla identificazione dei 'movimenti dati' in ogni processo funzionale, dove ciascun 'movimento dati' muove un gruppo di attributi (un 'gruppo dati') descrivendo un singolo oggetto d'interesse. Pertanto è quasi inevitabile che alcune analisi dei dati dovranno essere utilizzate per individuare gli oggetti di interesse e da ciò i movimenti dati che saranno misurati in qualsiasi funzionalità stabilita.

2.1 'Livelli' di modellazione dei dati

Solitamente i metodi di analisi dei dati definiscono e distinguono 'modelli dati' a differenti livelli[4] [7]. Per i nostri scopi è sufficiente definire i seguenti tre livelli di modellazione:

- '"Concettuale". Questo livello mostra le cose[5] nel mondo reale che sono importanti per una parte del software e le relazioni fra loro.
- "Logico". Questo livello mostra le cose nel mondo reale e le loro relazioni che sono rilevanti per il software ed i gruppi di dati che descrivono queste cose e le loro relazioni.
- "Fisico". Questo livello mostra le effettive registrazioni dei dati e le loro relazioni, gli archivi e le basi dati che saranno gestite dal software che dovrà memorizzare le informazioni relative alle cose del mondo reale.

Nello sviluppo del software, i modelli concettuali sono vantaggiosi per estrarre i requisiti al fine di individuare le cose principali (o tipi di entità) che il software dovrà considerare. I modelli logici dei dati sono utilizzati in seguito nel processo di sviluppo per rappresentare i gruppi di dati che corrispondono ai tipi d'entità. Con il progredire delle specifiche, questi modelli esprimono maggiori dettagli dei quelli concettuali. Infine, i modelli fisici dei dati mostrano la struttura delle basi-dati o archivi di 'registrazioni fisiche' che dovranno essere sviluppati e, per completezza, i requisiti informativi delle mappe video e dei report.

I tipi-entità del modello concettuale saranno compresi nel modello logico, ma i gruppi dati del modello logico di solito non coincideranno con il modello fisico poiché questo ultimo dovrà essere progettato per tener conto delle prestazioni, della manutenibilità, degli accessi ed altri requisiti non-funzionali.

---

[4] Questi 'livelli' non sono differenti 'livelli di astrazione', poiché non sono viste differenti della medesima 'cosa'; essi sono modelli di tre 'cose' differenti (ma correlate).

[5] 'Cosa' è la parola che usiamo per rappresentare letteralmente qualsiasi cosa per la quale il software deve elaborare i dati, può essere un oggetto fisico o concettuale.

Il punto importante per i misuratori è di riconoscere che tutti i concetti definiti e necessari con i metodi FSM dovranno essere al livello di requisiti utente logico-funzionali, processi funzionali, oggetti di interesse, gruppi di dati, ecc. Ciò è rigorosamente vero per il metodo COSMIC.

2.2 Principi di analisi dei dati

Si presenteranno tre dei più diffusi modelli di analisi dei dati, denominati Entità-Relationship Analysis (E/RA), Diagramma delle Classi come indicati nello Unified Modeling Language (UML) e Relational Data Analysis (RDA). Ai lettori, che di solito utilizzano solo uno di questi metodi, si consiglia tuttavia la lettura dell'intero capitolo.

Tutti e tre i metodi hanno obiettivi simili, vale a dire produrre, per ciascuna parte di software da realizzare o gestire, un modello o modelli di 'cose' nel mondo reale, e le relazioni statiche tra queste per cui si richiede al software di memorizzare e/o elaborare i dati. Può sembrare disorientante che tre metodi usino termini diversi, abbiano differenti convenzioni grafiche e visualizzino dettagli diversi per queste stesse 'cose'.

Come regola generale, queste 'cose' corrispondono a 'oggetti d'interesse' nel metodo COSMIC, ma ciò non è sempre vero ed è di vitale importanza capire perché vi siano delle differenze. Per esempio, sia E/RA che Diagrammi delle Classi UML consentono all'analista di dati di evidenziare solo le 'cose' che sono d'interesse e solo nel dettaglio sono sufficiente per uno scopo specifico. Perciò i diagrammi prodotti con questi metodi possono non evidenziare tutti gli oggetti di interesse che il misuratore ha necessità di individuare con il metodo COSMIC. Si affronteranno altri esempi di disallineamenti nella sottostante sezione 2.6.

Nel metodo COSMIC, il termine generico 'oggetto d'interesse' è stato scelto al posto di 'tipo di entità', 'classe' o 'relazione 3NF (Third Normal Form, Terza Forma Normale)', rispettivamente per i tre metodi di analisi dei dati, per evitare l'utilizzo di un termine relativo ad uno specifico metodo.

In questa Guida è possibile indicare un solo breve profilo dei principi dei tre metodi per la misurazione delle dimensioni funzionali con il metodo COSMIC. Per maggiori informazioni si rimanda ai documenti [7], [8], [9] e/o [10].

2.3 Analisi E/R

Un tipo-entità è definito come 'qualcosa nel mondo reale per il quale si richiede che il software memorizzi e/o elabori le informazioni'. Pertanto il metodo E/RA può essere usato per realizzare un modello (a livello concettuale o logico) di queste cose nel mondo reale e le loro relazioni che è completamente indipendente da ogni considerazione imposta dall'implementazione del software. Le relazioni evidenziate sono 'statiche', in altre parole i diagrammi non mostrano le relazioni come variano nel tempo. Solitamente è evidenziato anche il 'grado' (o cardinalità) della relazione fra i tipi entità'. Per esempio in un sistema di elaborazione degli ordini si richiede di memorizzare le informazioni relative ai clienti ed agli ordini eseguiti. Un diagramma E/R può evidenziare questi requisiti nel seguente modo (le convenzioni grafiche possono essere diverse).

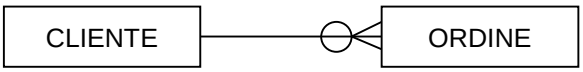

Figura 2.3.1 – Diagramma E/R che mostra informazioni relative ai clienti ed agli ordini con una relazione uno-a-molti.

Il simbolo sulla linea mostra la relazione fra i due tipi entità Clienti e Ordini ed indica che il Cliente può essere associato con zero, uno o più Ordini in un certo momento. Al contrario, un Ordine può essere associato ad uno ed un solo Cliente. A seguito di tale requisito entrambi i tipi entità saranno oggetto d'interesse.

E/RA prevede una tecnica specifica per l'identificazione di tipi entità addizionali (e quindi oggetti d'interesse) che sono importanti per il metodo COSMIC. Ciò è illustrato meglio con un esempio.

Si considerino i seguenti requisiti funzionali utente:

"Si devono memorizzare e gestire le informazioni relative agli impiegati e ai dipartimenti nell'ambito dell'organizzazione in cui questi hanno lavorato. Da queste informazioni, il software deve essere in grado di fornire la storia lavorativa di ciascun impiegato, vale a dire, l'elenco dei dipartimenti ove questo ha lavorato nel passato e la data di inizio e fine di ciascun impiego".

I due tipi di entità più evidenti (o oggetti di interesse) indicati in questi requisiti sono 'Impiegato' e 'Dipartimento', come mostrato nella figura sottostante. La figura indica anche le due relazioni che devono esistere fra questi due oggetti di interesse: un impiegato può essere assegnato a più reparti durante la sua carriera ed un reparto può comprendere più dipendenti ad una stessa data.

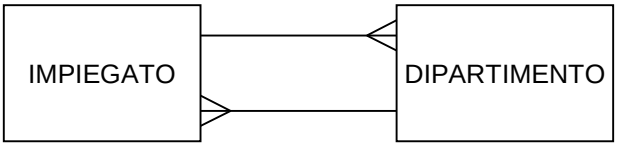

Figura 2.3.2 - Diagramma E/R che evidenzia tipi di entità e due relazioni uno-a-molti

Queste due relazioni 'uno-a-molti', quando combinate fra loro, confermano che la relazione Impiegato/Dipartimento è effettivamente 'molti-a-molti'. In pratica, una relazione molti-a-molti implica sempre l'esistenza di un altro oggetto di interesse che detiene le informazioni relative alla relazione fra un qualsiasi impiegato ed un qualsiasi dipartimento. Questa 'intersezione di entità' può essere definita come la 'Storia Dipartimento/Impiegato' e dai requisiti possiamo dedurre che deve contenere (almeno) due attributi propri, vale a dire la data di ingresso e di uscita dell'impiegato dal dipartimento.

Figura 2.3.3 - Diagramma E/R precedente, risolto con due relazioni uno-a-molti unite da una 'entità d'intersezione'

L'E/RA insegna che quando si individua una relazione molti-a-molti è necessario che questa sia risolta con due relazioni uno-a-molti unite da un'entità d'intersezione. Quest'ultima rappresenta sempre qualche cosa nel mondo reale e di solito ha attributi propri. Per misurare la dimensione funzionale di questo FUR, il metodo COSMIC distinguerebbe, in questo esempio, tre distinti oggetti d'interesse.

Si noti che nella maggior parte dei casi pratici, per un'entità d'intersezione che sia oggetto di interesse ci devono essere ulteriori attributi rispetto ai suoi attributi chiave. Gli attributi della chiave identificano la 'cosa' che esiste; gli altri attributi forniscono i dati concernente la cosa. La definizione di un oggetto d'interesse é logicamente soddisfatta quando sono presenti attributi non chiave – questa è una 'cosa' nel mondo dell'utente per la quale si richiede al software di elaborare e/o memorizzare i dati'. Nell'esempio precedente la 'cosa' è stata individuata come 'storia impiegato/dipartimento'. I suoi attributi chiave indicano solo che una relazione esiste tra un particolare 'impiegato' ed un particolare 'dipartimento'. Gli altri attributi: 'data di ingresso dell'impiegato nel dipartimento' e 'data di uscita dell'impiegato dal dipartimento', forniscono i dati della relazione. Quindi l'entità d'intersezione (cioè la relazione) è chiaramente un oggetto d'interesse.

È possibile, ma piuttosto raro, che un'entità d'intersezione possa essere oggetto d'interesse, anche se non possiede attributi che non siano chiave. Perché un caso del genere sia accettabile come oggetto di interesse, dovrebbe essere abbastanza chiaro dai FUR che vi è un requisito di business per cui il software deve registrare una relazione molti-a-molti tra due entità, ma che non c'è necessità di contenere altri dati sulla relazione. Un caso potrebbe essere il requisito di registrare in una matrice l'esistenza di una relazione tra alcune colonne e alcune righe, ad esempio per tenere traccia degli alunni e dei relativi compiti conclusi in un corso scolastico inserendo un segno di spunta per ogni compito completato da ciascun alunno. In questo esempio, la relazione 'alunno/compito' è un oggetto d'interesse.

2.4 Diagrammi delle classi UML (e casi d'uso)

L'UML (Unified Modeling Language) [10] definisce un insieme di diagrammi standard validi da utilizzare quando si procede dalla determinazione dei requisiti allo sviluppo di software orientato agli oggetti (OO). UML non prescrive alcun processo per realizzare i diversi diagrammi e, pertanto, non è realmente un metodo d'ingegneria del software.

(N.B. nel seguito NON va confusa la nozione COSMIC di 'oggetto d'interesse' con il concetto OO di 'oggetto' o 'classe di oggetti').

Quando si esegue la mappatura dei concetti UML con quelli del metodo COSMIC, la cosa più importante è la mappatura uno-a-uno di una classe UML con un 'oggetto d'interesse' (tipo di) COSMIC, anche se non sono lo stesso concetto. Una 'classe' è definita [10] come "una descrizione di un insieme di oggetti che condividono gli stessi attributi, operazioni, metodi, responsabilità e semantiche, dove un 'oggetto' è un istanza che origina da una classe". Quando una classe viene instanziata come un oggetto, gli attributi dell'oggetto sono valori assegnati, e l'insieme dei valori degli attributi è ciò che rende un oggetto diverso da un altro.

Contrasta con quanto sopra detto il fatto che un oggetto d'interesse (tipo di) sia 'una cosa … nel mondo dell'utente funzionale per la quale si elaborano e/o si memorizzano i dati'. Nel Modello Generale del Software COSMIC, un oggetto d'interesse è descritto soltanto dai suoi attributi dati.

In generale quindi, la parte di una definizione di classe UML con gli attributi corrisponde ad un oggetto d'interesse COSMIC ed ai suoi attributi di dati. Quando una classe è stata identificata, è stato identificato anche un oggetto d'interesse COSMIC. I 'metodi' definiti in una classe UML, sono anche una fonte di candidati per i processi funzionali come i metodi descrivono la funzionalità associata alla classe.

I diagrammi delle classi possono essere tratti da diversi 'punti di vista' e/o diverse fasi del ciclo di vita del software e mostrano diverse informazioni attraverso quelle fasi. Ai fini della misurazione della dimensione funzionale con il metodo COSMIC, i diagrammi di classe tratti dalla prospettiva della 'specifica' o dalla fase di 'analisi', corrispondono meglio a livello logico e sono quelli che contano.

Di solito i diagrammi delle classi mostrano le relazioni statiche fra classi (come da diagrammi E/R), gli attributi di ogni classe ed i vincoli che si applicano sulle modalità con cui gli oggetti sono collegati, vale a dire più dettagli di un diagramma E/R.

I diagrammi delle classi dovrebbero anche mostrare i 'sotto-tipi' se esistono. (E/RA ha lo stesso concetto, ma i sotto-tipi non sono sempre evidenziati nei diagrammi E/R). Un sotto-tipo di una classe (o tipo entità) eredita tutti gli attributi e le altre caratteristiche di quella classe ma possiede anche attributi univoci ed altre caratteristiche. Per maggiori informazioni su sotto-tipi e se i distinti oggetti d'interesse devono essere riconosciuti come sotto-tipi si consulti [11] ed anche l'esempio 5 nel sezione 4.2.3.

La questione dell'esistenza o meno di sotto-tipi mostrati nei diagrammi di classe o E/R dimostra che entrambe le tecniche possono essere utilizzate a diversi livelli di granularità. Ad esempio, nelle prime fasi di estrapolazione dei requisiti, si può constatare che devono essere gestiti i dati sui clienti e, quindi, 'cliente' è riconosciuto come un oggetto d'interesse. Successivamente durante il processo, si può riconoscere che vi sono importanti sotto-tipi di 'cliente' (ad esempio personale, dettagliante, grossista) ed in alcuni processi funzionali ed a seconda dei requisiti potrebbe risultare necessario che siano considerati come oggetti d'interesse separati.

Ne consegue inoltre che non può essere presa una decisioni definitiva se l'oggetto è d'interesse in ogni parte di software se non quando sono noti i requisiti per i processi funzionali. L'analisi E/R ed il disegno dei diagrammi di classe ULM non possono da soli determinare gli oggetti d'interesse. Devono essere conosciuti i processi funzionali necessari. Un accurato dimensionamento funzionale con COSMIC richiede che siano conosciuti i processi funzionali e le loro interazioni di dati.

In questo contesto, poiché i 'casi d'uso' UML sono ampiamente utilizzati, è importante sottolineare che un caso d'uso non corrisponde necessariamente ad un processo funzionale COSMIC. La costruzione di un 'caso d'uso' è 'utilizzata per definire il comportamento di un sistema o di altre entità semantiche,

senza rivelare la struttura interna dell'entità. Ciascun caso d'uso specifica una sequenza di azioni, comprese le varianti, che l'entità può compiere, quando interagisce con attori[6] dell'entità [10].

Gli utilizzatori dell'UML sono quindi liberi di scegliere, in base alle loro finalità, come definire un caso d'uso per un dialogo con un attore che comprende una sequenza di processi funzionali, un singolo processo funzionale, o solo una parte di un processo funzionale. L'UML ha il concetto di 'evento' con lo stesso significato del metodo COSMIC, vale a dire 'un evento è una descrizione dettagliata di un tipo di occorrenza osservabile [...] che non ha una durata' [10], ma l'UML non definisce relazioni fra un evento ed un caso d'uso. Pertanto per misurare un caso d'uso, gli utilizzatori dell'UML devono comprendere pienamente la definizione di 'processo funzionale' COSMIC e tutti i suoi elementi distintivi in modo che possano mappare le proprie procedure locali per mappare i casi d'uso in processi funzionali.

2.5 Il processo di normalizzazione della RDA

L'RDA è piuttosto differente. Si tratta di un metodo su base matematica che è spesso meglio considerato come una strategia 'dal basso verso l'alto' (bottom-up) piuttosto che 'dall'alto verso il basso' (top-down) come con UML (E/RA può essere usato sia top-down sia bottom-up). RDA individua un processo rigido per produrre un modello logico dei dati 'normalizzato' da un modello fisico dei dati di un archivio che in genere non è normalizzato. Per 'normalizzato' s'intende un modello in cui ciascuna 'relazione' dei dati che ne risulta è il più indipendente possibile da ogni altra relazione di dati nell'area osservata. (Una relazione normalizzata è un set di attributi di dati – un 'gruppo di dati' nella terminologia COSMIC – di un singolo oggetto d'interesse). Un database relazionale costruito per corrispondere ad un modello dei dati normalizzati sarà più semplice da gestire rispetto ad un qualsiasi database non normalizzato a causa di questa indipendenza delle relazioni.

RDA definisce le modalità che consentono la trasformazione, tramite tre forme relazionali fino alla terza forma normale (3NF), di una serie di dati non normalizzati (come i record fisici di un database non normalizzato). Le relazioni derivanti sono, per il metodo COSMIC, il punto di partenza per l'identificazione degli oggetti d'interesse e dei gruppi di dati.

Su un determinato gruppo di dati (ad esempio in un modulo, schermata, report, record di database o file) la normalizzazione si effettua in cinque passaggi:

1. rappresentare i dati come una tabella non normalizzata.
2. identificare la chiave per i dati non normalizzati.
3. prima forma normale (1NF): spostare i gruppi ripetitivi di dati in relazioni distinte.
4. seconda forma normale (2NF): spostare gli attributi dipendenti dei dati solo sulla parte chiave per separare le relazioni.
5. terza forma normale (3NF): spostare gli attributi dei dati che dipendono solo in parte dalla chiave in relazioni separate.

Le relazioni in 3NF sono chiamate 'normalizzate' ed i soggetti di tali rapporti sono quasi sempre oggetto d'interesse (per alcuni criteri aggiuntivi vedere il sezione 2.6.3). La proprietà tipica di una relazione in 3NF può essere riassunta come 'tutti i suoi attributi dipendono dalla chiave, l'intera chiave e nient'altro che la chiave'.

2.6 Dai tipi-entità, classi o relazioni agli oggetti di interesse

In quasi tutti i casi saranno considerati oggetti d'interesse COSMIC i tipi di entità che possono essere identificati da analisi E/R, le classi evidenziate nei diagrammi di classe UML ed i soggetti di relazione individuati da RDA. L'esperienza però dimostra che di norma i casi si presentano dove i metodi di

[6] Si noti che un 'attore' è definito come 'un'entità che avvia degli scenari e ne ottiene dei risultati'. Anche se questa definizione differisce da quella di 'utente funzionale', il suo intento sembra essere simile. Tuttavia gli utenti funzionali sono 'entità che possono solo inviare e/o ricevere dati da un processo funzionale ma non possono iniziarlo.

analisi dei dati non sempre conducono alla corretta identificazione degli oggetti di interesse. Altri criteri sono necessari in aggiunta alle norme contenute nel Manuale di misurazione. I casi sono descritti nelle successive due sezioni ed i criteri aggiuntivi sono descritti nel sezione 2.6.3.

2.6.1 Input, output e strutture dati transienti

L'analisi E/R, i diagrammi di classe UML e in qualche misura RDA, di solito sono applicati solo per comprendere la struttura dei dati permanenti (o archiviati). Applicando tali metodi ai dati permanenti, siamo in grado di identificare gli oggetti d'interesse che sono i soggetti dei gruppi dati relativi ai movimenti dati Read e Write.

Tuttavia, per gli scopi del metodo COSMIC, è necessario applicare questi stessi principi alle strutture dei dati delle componenti di input e di output dei processi funzionali al fine di identificare gli oggetti di interesse che rappresentano i soggetti dei gruppi di dati dei movimenti di Entry ed Exit. Per molti processi funzionali gli Entry e gli Exit saranno date dall'immissione o la visualizzazione dei dati che diventeranno o sono già permanenti (ad esempio, inserire i dati di un cliente, oppure chiedere informazioni su un impiegato, rispettivamente). Quindi in questi casi l'analisi dei dati di input e output di dati delle strutture è uguale a quello delle strutture persistenti dei dati.

Ma molti processi funzionali di interrogazione, ad esempio nei sistemi di gestione delle informazioni di segnalazione, generano dati derivati che non vengono memorizzati. L'Entry (a volte, gli Entry) dell'input ed una o più Exit che compongono i tipici risultati di tali processi funzionali produrranno strutture transitorie di gruppi dati, ovverosia gruppi dati che non sopravvivono al processo funzionale nel quale si manifestano.[7]

Si consideri il seguente esempio. Si supponga che vi sia un requisito funzionale dell'utente per sviluppare un'indagine per calcolare e visualizzare:

- il valore totale delle merci vendute in un dato periodo di tempo da parte del tipo-cliente (se quest'ultimo è codificato P = Privato, R = Rivenditore, o G = Grossista) e
- il valore totale delle merci vendute a tutti i clienti in un dato periodo di tempo.

('Tipo-cliente' è un attributo del Cliente, tutti i clienti hanno gli stessi attributi, così in questo esempio l'attributo tipo-cliente non significa che abbiamo sotto-tipi di clienti come descritto nel sezione 2.4.). Questi due livelli di aggregazione di dati delle vendite indica che ci sono dati transitori relativi a due oggetti di interesse e pertanto due Exit della visualizzazione. I due oggetti d'interesse sono:

- I prodotti venduti a clienti di un dato tipo cliente in un determinato periodo di tempo, e
- I prodotti venduti a tutti i clienti in un determinato periodo di tempo

Quale controllo incrociato su questa conclusione, l'analisi E/R direbbe che 'i prodotti venduti a qualsiasi tipo di cliente in un periodo 'sono una 'cosa' diversa da 'i prodotti venduti a tutti i clienti nel periodo'. In questo esempio ed in base alla definizione di un oggetto d'interesse, entrambe le 'cose' sono chiaramente differenti 'oggetti fisici nel mondo dell'utente per i quali si richiede al software di elaborare i dati'.

La stessa conclusione si raggiunge applicando l'analisi RDA all'output dell'interrogazione. Il risultato mostrerà, in un dato periodo, tre occorrenze di valore delle vendite per tipo cliente (distinte per cliente personale, dettagliante e grossista) ed un valore delle vendite per tutti i clienti nello stesso periodo. Il terzo passaggio del processo di normalizzazione RDA definirà che il dato relativo ai beni venduti ai tre tipi cliente è un gruppo ripetitivo ed è pertanto in una relazione diversa da quella del dato relativo a 'tutti i clienti'.

---

[7] Per chiarire una possibile fonte di confusione: il report di output o la videata di output che evidenziano il risultato di un'indagine ad hoc non è l'oggetto di interesse di cui stiamo discutendo qui. Sono i dati del report o della videata che devono essere analizzati per scoprire gli oggetti d'interesse. Nell'ambito del software applicativo aziendale, il misuratore dovrebbe concentrarsi sui dati dei FUR, non sui dati relativi ai dispositivi fisici, documenti o mezzi di comunicazione.

Va inoltre ricordato che l'Entry per questa interrogazione trasmette i parametri della ricerca, ad esempio, le date di inizio e fine, che comprendono anche un gruppo dati transienti. Questi sono gli attributi dell'oggetto di interesse 'periodo di tempo'.

2.6.2 Tabelle di parametri (decodifica) e oggetti di interesse

E' possibile che qualche 'cosa' che può rappresentare un oggetto d'interesse per alcuni tipi di utenti funzionali nei loro processi funzionali non lo sia per altri tipi di utenti funzionali nei loro processi funzionali. (Questi due insiemi di processi per tipi differenti di utenti funzionali possono o possono non essere definiti nell'ambito dello stesso campo di misurazione, poiché la definizione di un campo dipende solamente dalla finalità della misurazione stessa).

Questa situazione si verifica spesso nei software applicativi aziendali laddove si richiede una parametrizzazione elevata al fine di consentire una maggiore facilità di manutenzione. Il semplice utente applicativo di un'applicazione non considera i parametri come oggetti d'interesse, mentre tali parametri rappresentano oggetti d'interesse per coloro che li devono mantenere, ad esempio, un amministratore di sistema.

Si illustreranno i casi che possono verificarsi con alcuni semplici esempi. Per ulteriori chiarimenti si veda l'esempio 6 del sezione 4.2.3.

a) Si supponga che un'applicazione per l'elaborazione degli ordini memorizzi una grande quantità di dati sui clienti, ad esempio, ID-cliente, nome-cliente, indirizzo-cliente, contatto-telefonico- cliente, limite-di-credito-cliente, codice-tipo-cliente, data-ultimo-ordine-cliente, ecc, ognuno dei quali deve essere inserito.

   'Cliente' è chiaramente un oggetto d'interesse per molti utenti funzionali, ad esempio, in questo caso per lo staff dell'ufficio ordini. Così il processo funzionale più semplice per inserire i dati relativi al cliente dovrebbe prevedere un Entry ed un Write dei dati inseriti per l'oggetto d'interesse Cliente, e probabilmente un Exit per i messaggi di conferma/errore[8]. Dimensione totale: 3 CFP.

b) Si noti, nell'elenco di cui sopra, l'attributo codice-tipo-cliente. Può avere diversi valori, ad esempio P, R, W, ecc, che rappresentano, rispettivamente, Privato, Rivenditore, Grossista. Supponiamo che questa stessa applicazione di elaborazione degli ordini memorizzi anche altri dati, relativi a questa 'cosa' tipo di cliente ad esempio 'descrizione-tipo-cliente', '%-sconto-valore-ordine-tipo-cliente', 'termini-di-pagamento-tipo-cliente', ecc.

   Tipo-cliente è ora anche una 'cosa' con i suoi propri attributi ed in questo sistema è un oggetto di interesse per gli utenti funzionale che definiscono la politica commerciali e gestiscono gli attributi di ogni tipo di cliente ed è anche oggetto d'interesse per gli utenti funzionale dell'ufficio ordini.

   In questo caso, quindi, quando si immettono i dati relativi ad un nuovo cliente, il codice-tipo-cliente non è solo un attributo di Cliente ma fornisce un collegamento agli altri attributi come %-sconto-valore-ordine-tipo-cliente, ecc che sono molto rilevanti (cioè *d'interesse*) per ogni utente funzionale interessato ai rapporti con la clientela. Questi attributi dell'oggetto d'interesse codice-tipo-cliente diventano attributi indiretti dell'oggetto d'interesse Cliente.

   Pertanto il processo funzionale più semplice per immettere i dati relativi al nuovo cliente descritto nel caso a) dovrebbe essere ora nel caso b) includere un Read dei dati dell'oggetto di interesse tipo-cliente, in modo da convalidare che sia stato inserito un codice-tipo-cliente corretto.

   Si noti che, per immettere, in questo stesso processo funzionale, i dati relativi ad un singolo cliente, non va individuata alcun Entry separata quando si inserisce il codice-tipo-cliente poiché

---

[8] Per quasi tutto il software applicativo aziendale, per cui gli utenti umani inseriscono preparano dati per l'inserimento, si richiede di fornire messaggi di output per informare l'utente nel caso in cui i dati inseriti non passino un test di convalida e/i per conferma re che i dati sono stati appropriatamente elaborati. Tali messaggi sono tutte le occorrenze di un Exit, denotato in questa Linea Guida come 'messaggi di errore/conferma'. V. sez. 4.4.7 per una discussione di tali messaggi.

questo è inserito come attributo(cioè *relativo* ai dati) di un Cliente (in questo processo funzionale non stiamo inserendo i dati *relativi* al tipo-cliente).

Complessivamente il processo funzionale per l'inserimento dei dati relativi a un nuovo cliente, quando il tipo-cliente è un oggetto d'interesse per gli utenti funzionale richiede un Entry sui dati relativi al cliente, un Read per la convalida dei dati relativi al tipo-cliente, un Write dei dati relativi al cliente e probabilmente uno spostamento dei dati per i messaggi di conferma/errore. Dimensione totale: 4 CFP.

c) In alternativa al caso b), si supponga che la 'cosa' tipo-cliente abbia solo due attributi 'codice-tipo-cliente' e 'descrizione-tipo-cliente'. Questi due attributi potrebbero avere, rispettivamente, coppie di valori P, Privato; R, Rivenditore, ecc.

Quando si devono inserire i dati relativi ad un nuovo cliente ed in questo processo funzionale di immissione dei dati si raggiunge il punto dove viene inserito il 'codice-tipo-cliente', si supponga che il sistema visualizzi un elenco di 'descrizioni-tipo-cliente' valide. L'utente seleziona quindi la descrizione appropriata che produce l'immissione e la memorizzazione del corrispondente codice-tipo-cliente come un attributo del cliente.

In tal caso, nel processo di inserimento dei dati relativi a un nuovo cliente, non c'è ragione per l'utente funzionale di considerare come oggetto d'interesse il tipo-cliente, come nel caso b). Si richiede solo di memorizzare il 'codice-tipo-cliente' come un attributo di cliente ed in questo processo funzionale di inserimento dati del cliente non viene elaborato alcun dato per *tipo-cliente*. Il 'codice-tipo-cliente' è coinvolto solo nella fase di elaborazione dei dati sul *cliente*.

Ovviamente il software necessita di memorizzare i valori dell'attributo codice-tipo-cliente per esigenze applicative e l'equivalente della descrizione-tipo-cliente per le esigenze dell'utente, ma ciò é necessario solo per garantire che per ogni cliente sia inserito un valore valido di codice-tipo-cliente. (È irrilevante se i valori di questi attributi sono memorizzati ad esempio in una tabella di codici per la manutenzione del software o in memoria). In questo caso il tipo-cliente non è un oggetto d'interesse per l'utente applicativo di questo processo funzionale come nel caso b).

Per processi funzionali come l'immissione dei dati di questo cliente, non può essere individuato alcun movimento di dati per i riferimenti agli attributi dei dati del tipo-cliente e la dimensione totale è la stessa che per il caso a), ovverosia 3 CFP.

d) Continuando con l'esempio c), si supponga ora che i valori di questi due attributi 'codice-tipo-cliente' e 'descrizione-tipo-cliente' siano memorizzati in una tabella generale dei codici insieme ad altri sistemi di codifica e forse altri parametri per facilità di manutenzione da parte di un 'utente funzionale non aziendale'. (Tale termine include 'amministratori di sistema', 'responsabili applicativi', staff tecnico o di sviluppo, cioè tutti coloro che hanno il compito di gestire l'applicazione come ad esempio, il mantenimento di codici e descrizioni valide, ma che non sono dei normali 'utenti funzionali aziendali').

Questo software di manutenzione della tabella parametri dovrebbe includere processi funzionali per la creazione, l'aggiornamento, la cancellazione e la lettura di nuovi codici e descrizioni del tipo-cliente.

Chiaramente, per ciascuno di tali processi funzionali di creazione, aggiornamento, ecc degli attributi codice e descrizione del tipo-cliente, il software deve elaborare i dati relativi al tipo-cliente. In questi processi funzionali il tipo-cliente diviene, in tal modo, un oggetto d'interesse per l'utente 'non aziendale'.

Per tali processi funzionali, quindi, occorre individuare ogni trasferimento di dati (E, X, R, W) relativi al tipo-cliente. (Esempi dei processi funzionali che possono essere necessari per la manutenzione di tabelle di parametri sono indicati in sez. 4.2.3, Esempio 6.)

Nota: il fatto che il tipo-cliente non sia un oggetto d'interesse per i FUR del caso c), ma lo sia per i FUR del caso d) rimane più o meno valido a seconda che questi due tipi di FUR rientrino o non rientrino nello stesso ambito di misurazione.

In sintesi, da questi esempi è possibile notare che:

- Per l'utente funzionale aziendale il tipo-cliente è o non è un oggetto d'interesse, semplicemente se possiede (come nel caso b) o meno (come nei casi a) e c) propri attributi. Nei casi a) e c) il codice-tipo-cliente è solo un attributo di Cliente.
- Per l'utente funzionale non aziendale, quando gli attributi del tipo-cliente devono essere gestiti come parametri di sistema da processi funzionali, allora, in quei processi funzionali, il tipo-cliente è un oggetto d'interesse.
- Come conseguenza, tipi differenti di utenti funzionali possono 'vedere' oggetti d'interesse diversi nei loro propri processi funzionali e per la stessa applicazione. Vedere anche l'esempio 7 nel sezione 4.4.3.

I processi funzionali degli utenti applicativi e di quelli non applicativi possono essere definiti in ambiti separati di misurazione, ad esempio, quando l'obiettivo è la stima dell'impegno progettuale e, ove verranno utilizzate le diverse tecnologie per il software applicativo principale e per il software di manutenzione dei parametri. Ma non c'è nessun principio che imponga che questi due processi funzionali siano in ambiti separati. In un vecchio modo di dire dell'E/RA, 'un'entità per qualcuno è un attributo per qualcun altro'.

2.6.3 Ulteriori criteri per l'identificazione degli oggetti di interesse

Oltre ai casi di cui al punto 2.6.2 è necessario aggiungere alcuni criteri, oltre a quelli indicati nel Manuale di Misurazione, per il riconoscimento di un oggetto d'interesse, come di seguito riportato.

La chiave per capire se una 'cosa' è o non è un oggetto di interesse significa:

a) determinare se la 'cosa' è veramente 'd'interesse', ossia che nei Requisiti Funzionali Utente (FUR) è indicata come una cosa per la quale si richiede al software di elaborare (o memorizzare dati), e
b) riconoscere che se una 'cosa' è un oggetto d'interesse o no, può dipendere dal (tipo di) utente funzionale individuato dal FUR. Una 'cosa' può essere un oggetto d'interesse in alcuni FUR per determinati processi funzionali di determinati tipi di utenti funzionali, ma può non essere un oggetto d'interesse in altri FUR per alcuni processi funzionali per altri tipi di utenti funzionali. (Nota: è possibile che questi punti di vista diversi su cosa siano gli oggetti d'interesse possano esistere all'interno di un insieme di FUR che rientrano nello stesso ambito di misurazione.)

I Tipi Entità individuati in E/RA ed il soggetto delle relazioni in Terza Forma Normale prodotti dalla RDA sono quasi sempre destinati ad essere oggetto d'interesse. Anche i dati transitori delle componenti di input e di output dei processi funzionali devono essere analizzati così come i dati persistenti per identificare tutti gli oggetti di interesse. Solo un esame dei FUR (o, in assenza di documentazione dei FUR, i processi funzionali che sono stati attuati, facendo attenzione al significato di un oggetto d'interesse) è in grado di determinare se una qualche 'cosa' è un oggetto d'interesse o meno per gli utenti funzionale.

2.6.4 Riepilogo: tipi di oggetti di interesse

Di seguito si deve tener presente che alcune 'cose' sono un oggetto d'interesse solo se soddisfano la definizione di un oggetto d'interesse ed i criteri del sezione 2.6.3. Lo scopo di questi sezione è di focalizzare l'attenzione dei misuratori sulla molteplicità di tipi di oggetti d'interesse che si riscontrano nel software applicativo aziendale. Il dibattito è stata finora illustrato con esempi che sono 'dati', tutti chiaramente oggetti d'interesse. Si è quindi trattato:

- I primari (o fondamentali) oggetti di interesse su cui sono costruite delle applicazioni aziendali per mantenere i dati persistenti, clienti, impiegati, conti, tipi di prodotti, pagamenti, contratti, ecc. Le applicazioni aziendali possiedono alti volumi di questi dati, gli oggetti di interesse normalmente hanno molti attributi che congiuntamente cambiano molto spesso, come avviene in un normale processo aziendale.
- I secondari (o non fondamentali) oggetti di interesse su cui è necessario che le applicazioni aziendali mantengano dati persistenti a supporto degli oggetti di interesse primari. Noi abbiamo

visto un esempio nel sezione 2.6., caso b), dove un “tipo cliente” è un oggetto di interesse con alcuni importanti attributi, cioè “ tipo % di sconto applicato al volume degli ordini dei clienti. Altri esempi saranno:

- In una applicazione paghe e stipendi, le tabelle di fasce stipendiali per grado, o le fasce di imposte sul reddito comprese in date di validità,
- In un sistema di contabilità, le tabelle dei tassi di cambio, inscritti in bilancio, tra vari paesi,
- In un sistema di controllo della produzione, tabelle di centri di costo e la loro capacità produttiva, o tabelle di tipi di lavorazione e livelli richiesti di specializzazione,
- In un sistema bancario, tassi di interesse di risparmio per prodotto, per livello di deposito, con relative date di validità.

Gli oggetti “secondari” di interesse tendono ad avere bassi volumi di dati che cambiano raramente.

- Oggetti di interesse per i quali solo i dati transitori esistono sempre. I dati su tali oggetti di interesse sono sia immessi da un utente funzionale che prodotti da una applicazione; tali dati non sono memorizzati in maniera permanente e appaiono solo in input o output di interrogazioni,
- Oggetti di interesse derivanti dalla parametrizzazione di software applicativo aziendale flessibili e facili da mantenere. Queste “cose” non sono oggetti di interesse per il comune utente funzionale applicativo, ma possono essere oggetti di interesse per “utenti funzionali non applicativi”, come un amministratore di sistema se i suoi attributi sono memorizzati in maniera permanente e sono mantenuti da processi funzionali accessibili per utenti non applicativi. Noi abbiamo trattato esempi di semplici sistemi di codifica quando solo il codice e la descrizione di un oggetto di interesse sono memorizzati in tabelle gestionali (per un dettagliato trattamento delle tabelle di codifica vedere il sezione 4.2.3 esempio 6). Ma alcuni altri tipi di parametri possono essere trattati in tabelle gestionali, per esempio regole per il trattamento dei dati, o i parametri per insiemi di regole. Questa è una pratica comune per il software nel settore dei servizi finanziari dove vi è la necessità di cambiare o introdurre nuovi prodotti finanziari con nuove varianti delle norme ed essere in grado di farlo velocemente per motivi di competitività. Alcuni esempi includono le condizioni dei tassi di interesse sui prestiti ed i termini dei rimborsi, i termini delle provvigioni delle polizze di assicurazione sulla vita, le caratteristiche commerciali dei prodotti di risparmio o le condizioni di risoluzione contrattuale, ecc. Finché si richiede che tali tabelle di regole siano disponibili direttamente all’utente funzionale (vedere il successivo sezione 3.2.1) per il loro mantenimento attraverso processi funzionali, ogni tipologia di regola può essere considerato come un oggetto di interesse in questi processi funzionali.

Notare, tuttavia, che non vi è niente di categorico sulla distinzione tra questi vari tipi di oggetti di interesse e le distinzioni non sono vincolanti per gli scopi nell’applicare del metodo COSMIC. La distinzione può variare a seconda del settore dell’utente software. “Il Paese” può essere oggetto secondari o non essere un oggetto di interesse per una azienda manifatturiera ma può essere un oggetto primario per una azienda di spedizione merci. La “valuta” e d i “tassi di cambio” sono un oggetto primario per una banca o secondario o di nessun interresse per la pubblica amministrazione locale. I “Reparti” possono essere un oggetto primario per un’applicazione che mantiene una struttura organizzativa ma solo un parametro per una gestione contabile. Si fa menzione di queste distinzioni solo perché tali classificazioni sono spesso usate e questo ci aiuta a capire il peso di come va considerato l’oggetto di interesse.

Quando si discute di oggetti di interesse per cui sono trattate informazioni in un normale linguaggio, di solito si pensa come ad attributi di dati, come “identificativi”, cioè codici, nomi e descrizioni, numeri di riferimento, ecc. e come dati quantitativi, cioè calcoli, quantità, ammontare di denaro, rapporti, indici ecc. Ma bisogna anche considerare gli oggetti di interesse che hanno attributi contenti solo dati testuali, cioè contratti standard o clausole di polizze assicurative, descrizioni di lavori, testi di ausilio per applicazioni, ecc.

3

LA FASE DI STRATEGIA DELLA MISURAZIONE

Si assume che il lettore abbia familiarità con il capitolo sulla 'Strategia di Misurazione' del Manuale di Misurazione COSMIC.

L'importanza di convenire sulla 'Strategia di Misurazione', prima di iniziare una misurazione non può essere sovra-enfatizzata. In sintesi, la Strategia di Misurazione considera quattro parametri chiave della misurazione che devono essere considerati, ovverosia lo scopo e l'ambito della misurazione, l'identificazione degli utenti funzionali e il livello di granularità dei FUR che dovrebbe essere misurato. Tali parametri saranno discussi nelle sezioni seguenti.

L'ambito di una misurazione è definito in COSMIC come 'l'insieme dei FUR da dimensionare'. Dato che l'ambito dipende esclusivamente dallo scopo della misurazione, lo scopo e l'ambito di una misurazione sono strettamente collegati. Pertanto sono trattati congiuntamente nel seguito della trattazione.

3.1 Scopo e campo di applicazione della misura

Esistono diverse ragioni per misurare il software, in questa sezione alcuni esempi illustreranno come gli approcci possono variare.

3.1.1 Esempi di scopo e ambito

a) Se lo scopo è misurare il lavoro di un progetto di sviluppo applicativo ed anche di parte del software localizzato in strati intermedi, devono essere definiti diversi ambiti applicativi per ogni porzione di software in ogni strato.

b) Se lo scopo della misurazione è determinare la produttività del progetto software (o altro parametro di performance) e / o per sostenere la stima e, l'applicazione aziendale è un pezzo di software in esecuzione su un'unica piattaforma tecnica, allora l'ambito sarà l'intera applicazione.

   Un punto chiave da verificare, qualora lo scopo riguardi la misurazione delle prestazioni o la stima per un progetto di software, è che l'ambito di ogni parte delle funzionalità da misurare deve corrispondere esattamente ai dati di durata e impegno per il lavoro (da fare) fatto dal team di progetto su quella parte di funzionalità.

c) Se, come nell'esempio precedente, lo scopo della misurazione è relativo a determinare o stimare le performance e l'applicazione deve essere distribuita su diverse piattaforme tecniche, sarà opportuno misurare separatamente le dimensioni di ogni parte dell'applicazione che gira su una piattaforma diversa. (Ogni parte è chiamata 'componente alla pari' dell'applicazione). Ciò può essere fatto attraverso la definizione di un ambito distinto di misura per ciascuna delle componenti alla pari e la misurazione delle prestazioni o di stima può essere effettuata separatamente per ogni componente alla pari su ogni piattaforma. (La motivazione di fondo è che risulta molto più facile interpretare i dati delle prestazioni, o di stima, per un pezzo di software che è (o sarà) sviluppato su una piattaforma tecnica unica che per uno che è (o sarà) sviluppato per piattaforme multiple.) Si noti, tuttavia, che la misurazione delle prestazioni per piattaforma presuppone che lo sforzo per piattaforma così come la dimensione verranno (o sono stati) registrati separatamente Per ulteriori esempi di misure su componenti applicativi, vedere il paragrafo 4.3.

d) Se lo scopo è misurare la dimensione totale di un portafoglio di applicazioni (ad esclusione di qualsiasi software di infrastruttura), al fine di stabilire il corrispondente valore economico, come ad esempio il costo di sostituzione, potrebbe essere sufficiente una misurazione che non consideri le

funzionalità provenienti da un eventuale architettura distribuita. Un ambito separato deve essere definito per ogni applicazione da misurare. (Le dimensioni possono anche essere misurate con sufficiente precisione usando una variante di approssimazione del metodo COSMIC per velocizzare questo compito[9].) Un livello di produttività medio quindi può essere utilizzato per determinare il costo di sostituzione dell'intero portafoglio.

e) In ogni rapporto di fornitura di software tra cliente e fornitore è sempre possibile delimitare l'ambito della misurazione del software in modo tale da soddisfare l'obiettivo della misurazione.

   Per esempio, potrebbe essere che lo scopo del cliente è di controllare solo le dimensioni dei Requisiti Funzionali derivanti da pure "esigenze di business", ignorando eventuali requisiti utente funzionali risultanti da requisiti generali (che le parti dovranno definire). Oppure potrebbe essere che l'accordo sia di definire due ambiti, uno per misurare la dimensione del solo software applicativo aziendale, e l'altro per misurare un 'software di supporto', ad esempio per il controllo della sicurezza di accesso, la registrazione o il log, il mantenimento dei parametri di sistema e tabelle di codice, ecc (possibilmente a causa di diversi accordi sui prezzi per i due ambiti). Come già detto, il campo di applicazione dipende solo dalla finalità della misura.

3.1.2 Software in strati differenti

Le applicazioni aziendali risiedono nel cosiddetto 'strato applicativo' ('application layer'). Il software dello strato applicativo è supportato da software di 'infrastruttura' (sistemi operativi, software di gestione dati, servizi, drivers di periferiche, ecc) in altri strati dell'architettura al fine di poter essere operativo. I requisiti per questo tipo di funzionalità non sono pertanto parte dei requisiti utente funzionali di un'applicazione aziendale.

Qualora gli elaborati di un progetto di sviluppo di applicazioni comprendano le applicazioni aziendali principali e utilità di supporto in modo da permettere la registrazione di back-up e di ripristino, controllare l'accesso di sicurezza, ecc. può sorgere l'interrogativo se il software di supporto risieda nello strato applicativo ed è quindi 'alla pari' con l'applicazione aziendale, o se risieda in uno strato diverso. Perché il software risieda in strati diversi, tutti gli aspetti di definizione, i principi e le regole per distinguere gli strati devono essere soddisfatte (si veda il Manuale di Misurazione). I due aspetti più perché due porzioni di software risiedano su strati diversi sono che vi sia una dipendenza gerarchica 'superiore/subordinato' tra le due e che esse devono interpretare i dati scambiati in modo diverso.

Esempi:

a) Un'utility di copiatura di file può interpretare i dati di un'applicazione in modo diverso dall'applicazione stessa, ma può non esserci un rapporto di tipo gerarchico tra loro. Se l'applicazione può funzionare senza l'utility di copiatura di file e viceversa, entrambi i software sarebbero nel medesimo strato applicativo.

b) Un pezzo di generale software di controllo accessi di sicurezza possono determinare quali utenti possono accedere a quali applicazioni, e anche che le transazioni all'interno di ogni applicazione. La domanda può solo essere in grado di eseguire se il controllo viene passato dal software di controllo di sicurezza di accesso, quindi in un certo senso c'è un rapporto gerarchico tra i due pezzi di software. Ma se i dati scambiati tra il software di controllo accessi e sicurezza di ogni applicazione è vista allo stesso modo, allora devono essere considerati come nello strato stessa applicazione

c) Una funzione di log (a scopo di ripristino) può essere fornita come parte integrante del sistema operativo per ogni applicazione configurata per il suo utilizzo. La funzione di login può operare senza un'applicazione specifica ma non viceversa – un'applicazione configurata per l'utilizzo del login non può essere eseguita senza la funzionalità di login. Pertanto si presenta una relazione gerarchica tra loro dove ognuna interpreterà sicuramente in modo diverso i dati che vengono registrati.

[9] Si veda il documento 'Metodo COSMIC 3.0. Advanced & Related Topics' per varianti di dimensionamento approssimato del metodo COSMIC.

Per gli esempi a) e b) di cui sopra, vi è un interrogativo distinto sul come considerare queste utility di supporto, se all'interno di un'unico ambito di misurazione, unitamente all'applicativo aziendale o no. L'ambito dipende solo dallo scopo della misurazione. Lo scopo può essere di misurare solo la quantità dell'applicativo aziendale o può essere di misurare la quantità totale di tutti i deliverable del team di progetto, includendo ogni parte di software di tipo utility. Comunque, le parti di software che risiedono in strati diversi devono essere sempre definite con scopi (o obiettivi) di misurazione distinti.

3.2 Identificare degli utenti funzionali

3.2.1 Gli utenti funzionali di un software applicativo aziendale

Per un software applicativo, parte di esso o per una delle sue componenti principali (in un ambito definito) oggetto di misurazione, gli "utenti funzionali" che si identificano nei suoi Requisiti Funzionali Utente possono essere uno tra i seguenti:

- utenti umani che sono a conoscenza solo delle funzionalità applicative con cui possono interagire,
- altre componenti principali della stessa applicazione necessarie per lo scambio o la condivisione di dati con la componente principale che si sta misurando,
- altre applicazioni alla pari (o componenti principali di altre applicazione alla pari) necessarie per lo scambio o la condivisione di dati con l'applicazione (o con una delle sue componenti principali) che si sta misurando.

Si noti che il personale che mantiene il software, i dati memorizzati o che gestisca attraverso programmi di utilità, disponibili solo per il personale informatico, NON sono considerati 'utenti funzionali' del software.

3.2.2 Il confine

Nell'eseguire misurazioni di software su domini tipo applicativo aziendale si identificano i movimenti di dati attraverso le interfacce l'input / output ('I / O'), - il confine - si trova tra l'applicazione ei suoi utenti funzionali, come illustrato nella figura sottostante. I dispositivi di I / O e qualsiasi infrastruttura di supporto che consente agli utenti funzionali di comunicare con l'applicazione oggetto di misura sono 'invisibili' agli utenti funzionali e pertanto ignorati.

I movimenti di dati verso e da memorie di dati persistenti vengono anche identificati dal punto di vista di questi utenti ma queste avvengono 'dentro' il software, cioè non attraversano il confine. Gli utenti funzionali non sono a conoscenza di eventuali dispositivi di memorizzazione persistente; funzionalmente, tutti gli utenti funzionali richiedono che i dati siano persistenti, o che siano recuperati da archivi permanenti (persistenti).

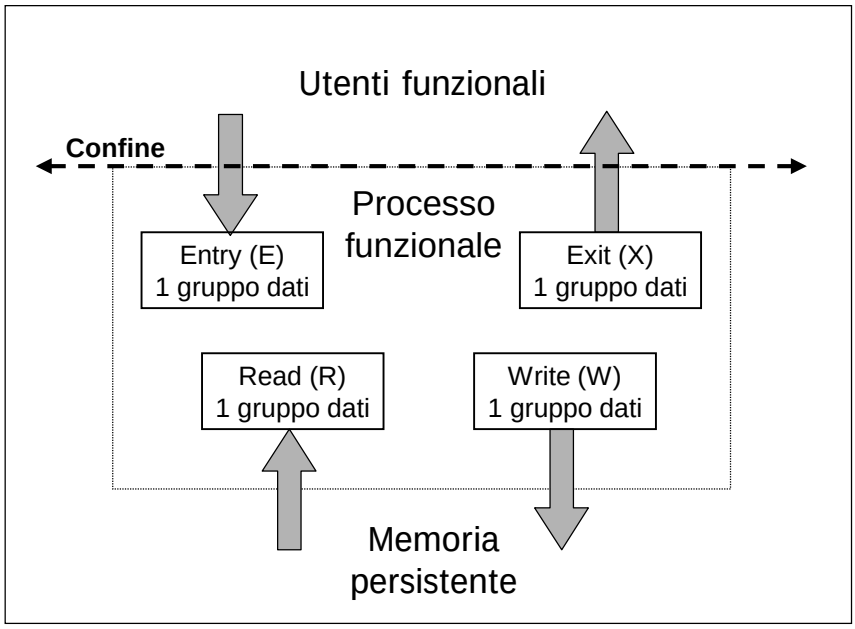

Figura 3.2.1 - Componenti di un processo funzionale e alcune possibili relazioni

3.3 Identificare il livello di granularità

Come indicato al punto 1.2.4, nelle prime fasi del ciclo di vita di un progetto di sviluppo software, i requisiti in genere esistono solo a livello concettuale o di 'alta' di granularità, cioè non sono dettagliati. Inoltre, i FUR, nella misura in cui sono prodotti, possono presentare diversi livelli di granularità.

Di conseguenza, il primo compito nella misurazione è di controllare il livello o i livelli dei requisiti funzionali. Poi, se i FUR non presentano il livello di granularità richiesta per una misura COSMIC accurata,, il misuratore dovrebbe usare una variante approssimativa del metodo in modo tale da poter per determinare la dimensione funzionale.

Il Manuale di Misurazione [3] fornisce un esempio dettagliato (il caso 'Everest') di situazioni che possono essere riscontrate quando una misurazione deve essere fatta su requisiti con specifiche a diversi livelli di granularità. Il documento [4] "Advanced and Related Topics" descrive diversi approcci per approssimare la misura dei requisiti ad alti livelli di granularità.

4

LE FASI DI MAPPATURA E MISURAZIONE

Nella fase di Mappatura, i FUR del software da misurare devono essere mappati su tre principali concetti del metodo COSMIC, nell'ordine processi funzionali, oggetti di interesse e gruppi dati. Nella successiva fase di Misurazione, conoscendo i processi funzionali e i gruppi dati, i singoli movimenti dati dei processi funzionali possono essere successivamente identificati, quale base per la misurazione della dimensione funzionale. Come quasi tutti gli esempi mostrano sia i processi funzionali che i movimenti dati insieme, entrambe le fasi vengono trattate in questo stesso capitolo.

4.1 Identificare i processi funzionali

Si suppone che il lettore abbia familiarità con la sezione 'Identificazione dei processi funzionali' presente sul Manuale di Misurazione [3].

Diversi casi possono aiutare a comprendere come distinguere I processi funzionali di un software applicativo aziendale.

4.1.1 Transazioni CRUD(L)

Nell'analisi e disegno di un software applicativo aziendale è buona pratica controllare le fasi del ciclo di vita di ciascun oggetto di interesse per il quale sono conservati dati persistenti, dato che ogni possibile passaggio da una fase all'altra (in termini UML la c.d. 'transizione di stato') dovrebbe corrispondere un processo funzionale. Questa regola è riassunta con l'acronimo 'CRUD' dove C=Create, R=Read, U=Update e D=Delete (talvolta conosciuta come 'CRUDL' dove L=List). I dati relativi a ciascun oggetto di interesse debbono essere creati, invariabilmente letti, e tipicamente saranno aggiornati e cancellati, e forse elencati.

Pertanto, quando un oggetto di interesse è identificato nei FUR, il misuratore dovrebbe in seguito determinare la parte del suo ciclo di vita che rientra nell'ambito della misurazione (es: identificando gli eventi che attivano le transizioni di stato), in modo da identificare quale dei cinque processi funzionali 'CRUDL' è richiesto nei FUR.

In pratica, per alcuni oggetti di interesse, i FUR potrebbero specificare diversi processi funzionali di tipo Update, corrispondenti a diverse fasi nel ciclo di vita dell'oggetto di interesse. Ad esempio, i FUR potrebbero specificare processi funzionali distinti per modificare i dati di una persona quando questa altera il proprio stato civile di celibe, sposato/a, vedovo/a e divorziato/a in funzione della necessità di aggiungere, modificare o cancellare alcune relazioni. In alternativa, i FUR potrebbero, naturalmente, specificare un processo funzionale per gestire tutti questi cambi di stato. Il numero dei processi funzionali di tipo Update dipende esclusivamente dai FUR.

A parte tali processi funzionali CRUDL che hanno a che fare con dati persistenti, il software applicativo aziendale in genere, naturalmente presenta anche requisiti per processi funzionali di richiesta o gestione del reporting che producono (o creano) dati transienti attraverso 'Read' dei dati persistenti.

4.1.2 Parti elementari di FUR e processi funzionali

Per identificare i processi funzionali nei Requisiti Utente Funzionali (FUR) è spesso utile iniziare dalla scomposizione dei FUR nelle loro 'parti elementari' o nelle 'più piccole unità di attività significative per l'utente/i' (ad esempio: definizione di schermate, o struttura dei report). Non è vero il contrario: non

ognuna delle parti elementari dei Requisiti Utente Funzionali (FUR) corrisponde ad un processo funzionale. Affinchè una porzione elementare dei FUR sia un processo funzionale deve necessariamente:

- essere eseguibile in modo indipendente e
- essere innescata da un evento nel 'mondo' degli utenti funzionali e
- quando questa porzione elementare è stata eseguita, il software ha 'eseguito tutto ciò che è richiesto di fare in risposta all'evento di innesco'.

Essere 'eseguibile in modo indipendente' non è sufficiente da solo. Se due parti dei FUR, A e B, sembrano essere eseguibili in modo indipendente, ma A è innescato da un evento esterno al confine, laddove B può solo essere innescato da A, allora A+B forma *un unico* processo funzionale. 'Innescato' significa che un utente funzionale – secondo le definizioni fuori dal confine – può essere identificato che individui un evento e poi inneschi il processo funzionale. Si veda anche la sezione 4.1.8 'Misurazione di processi funzionali (apparentemente) interdipendenti'.

4.1.3 Stile della progettazione delle videate e processi funzionali

Una singola videata fisica di transazione per l'inserimento di dati può, e spesso lo fa, fornire sia il processo funzionale di creazione dei dati relativi ad un oggetto di interesse che le funzionalità di aggiornamento per ciascuna fase del ciclo di vita dell'oggetto di interesse, poiché condividono una parte consistente delle funzionalità. Pertanto, in funzione dello stile di progettazione, una singola videata fisica può essere identificata per ciascuna di tali funzionalità logiche distinte, che sono menzionate nei Requisiti Utente Funzionali (FUR) come innescate da eventi separati.

4.1.4 Limitazioni fisiche delle videate

Rispetto al precedente caso, a causa dei limiti fisici di dimensione dello schermo, spesso capita che l'input o output di un processo funzionale debba essere sparso su più di una videata. Ci si assicuri di ignorare le limitazioni della videata fisica e i controlli navigazionali che attraversano le videate.

4.1.5 Processi funzionali distinti appartenenti a distinte decisioni di utenti funzionali

Più comunemente nel software applicativo aziendale on-line, i FUR per l'aggiornamento di dati persistenti richiede due processi funzionali separati. Il primo è una 'richiesta-pre-aggiornamento' nella quale i dati relativi all'oggetto/i di interesse da aggiornare è/sono prima letti (Read) e poi presentati. Ciò è seguito dal processo funzionale di 'aggiornamento' nel quale i dati modificati vengono resi persistenti da uno o più movimenti dati di tipo 'Write'.

Il processo funzionale 'richiesta-pre-aggiornamento' permette all'utente umano di recuperare e verificare che il record corretto/i sia/no stato/i selezionato/i per l'aggiornamento. Il successivo processo funzionale di 'aggiornamento' permette all'utente di inserire i dati che dovrebbero essere modificati, aggiunti, o cancellati e completare l'aggiornamento attraverso uno (o più) movimento/i dati di tipo Write.

Da un punto di vista logico, la 'richiesta-prima-dell'aggiornamento' e l'aggiornamento rappresentano due processi funzionali separati, auto-consistenti. L'utente può o può non decidere di procedere con il processo funzionale di aggiornamento in funzione del risultato della 'richiesta-pre-aggiornamento'. L'aggiornamento è interamente indipendente dalla richiesta.

I FUR per una 'richiesta-pre-aggiornamento' possono in ogni caso richiedere due processi funzionali distinti, come indicato nel seguito. L'attività è quella di aggiornare un record di un impiegato, laddove l'utente conosce il nome dell'impiegato ma non il suo codice matricola univoco. Primo, nel processo funzionale FP1, l'utente è invitato a elencare tutti gli impiegati per nome:

E Richiedi la 'lista degli impiegati'
R Dati degli impiegati
X Dati degli impiegati (es: solo nome e codice matricola, sufficienti a scegliere l'impiegato desiderato)
X Messaggi di errore/conferma

Dimensione del FP1 = 4 CFP

L'utente può ora, nel processo funzionale FP2, scegliere il particolare impiegato dalla lista e mostrare I dati che necessitano di aggiornamento:

E Cod. Matricola dell'impiegato (= selezione dell'impiegato desiderato)
R Dati dell'impiegato
X Dati dell'impiegato (modulo dettagliato, tutti gli attributi)
X Messaggi di errore/conferma

Dimensione del FP2 = 4 CFP

L'aggiornamento del processo funzionale, FP3, è poi (ignorando ogni funzionalità che possa richiedere la validazione dei dati aggiornati):

E Aggiornamento dei dati dell'impiegato
W Dati dell'impiegato
X Messaggi di errore/conferma

Dimensione del FP3 = 3 CFP

Dimensione di questo FUR = Dimensione (FP1) + Dimensione (FP2) + Dimensione (FP3) = 4 CFP+4 CFP+3 CFP = 11 CFP

In questo caso ci sono due processi funzionali di interrogazione (o reperimento) FP1 e FP2 seguiti dal processo funzionale di aggiornamento FP3. Ad ognuna delle tre fasi l'utente deve decidere in modo consapevole se continuare o fermarsi e fare altro. (L'utente potrebbe non trovare l'impiegato che sta cercando nella prima lista e potrebbe decidere di sperimentare un'altra tattica). Ogni necessità di decisioni consapevoli nel 'mondo' degli utenti funzionali implica un evento di innesco distinto, e pertanto un processo funzionale distinto.

Si noti inoltre che i FUR potrebbero richiedere diverse varianti di FP2 come sopra illustrato, che rappresenta il processo funzionale per mostrare tutti gli attributi di un dato impiegato. Se esaminiamo un tipico FUR per FP2 con maggior attenzione, si può osservare che FP2 deve essere limitato ad alcune figure di staff autorizzate all'aggiornamento dei dati degli impiegati e che può essere richiesto di mostrare alcuni campi in modalità protetta, che non debbono essere aggiornati.

In aggiunta al processo funzionale FP2, il FUR potrebbe specificare un processo funzionale di interrogazione generale disponibile per tutto il personale di staff del tipo FP4: 'mostra tutti i dati di un impiegato tranne lo stipendio attuale in una videata che non può essere modificata'. Potrebbero esserci molti requisiti di questo tipo. Ogni variazione di tali interrogazioni che ha dei FUR distinti dovrebbe essere considerata come un tipo di processo funzionale distinto e dovrebbe essere dimensionata separatamente.

4.1.6 Recupero e aggiornamento di dati in un singolo processo funzionale

Rispetto al caso precedente laddove si richiedono processi funzionali distinti per il recupero e l'aggiornamento dei dati, esistono alcune circostanze che possono portare a FUR di reperimento e aggiornamento dei dati di un oggetto di interesse tramite un solo processo funzionale. In tal caso, un Read è seguito da un Write degli 'stessi dati', senza alcun intervento dell'utente funzionale. Per 'stessi dati' si intende sia:

- il tipo di gruppi dato registrato è identico a quello letto, ma i suoi valori sono stati aggiornati, che
- Il tipo di gruppi dati scritto è per lo stesso oggetto di interesse come il tipo di gruppi dati letto, ma il gruppo dati scritto differisce da quello letto, ad esempio per l'aggiunta di attributi dati.

In questi casi un Read e un Write degli 'stessi dati' dovrebbero essere identificati in un solo processo funzionale. Ad esempio, in modalità batch processing, un singolo processo funzionale di aggiornamento può contenere un Read e successivamente un Write dello stesso oggetto di interesse.

4.1.7 Caselle di lista a discesa all'interno di processi funzionali

Quando durante un processo funzionale FP1 nell'Entry di un Gruppo dati che descrivono un oggetto di interesse A il valore di un attributo dati 'Z' può essere selezionato da una lista a discesa (drop-down list) (come in un'interfaccia grafica utente), sono tre i casi da considerare:

a) La lista a discesa mostra i valori validi dell'attributo Z dell'oggetto di interesse A dell'Entry, senza bisogno di referenziare i dati di ogni altro oggetto di interesse per ottenere tali valori. In questo caso la preparazione e la presentazione della lista a discesa non coinvolge alcun altro movimento dati e l'esistenza di una lista a discesa è ignorata durante l'analisi dell'Entry.

b) La lista a discesa presenta I valori validi dell'attributo Z dell'oggetto di interesse A dell'Entry. Tali valori validi sono ottenuti dall'attributo di un altro oggetto di interesse B. L'uso della lista a discesa per la selezione dei valori è obbligatoria. In questo caso si identifica un Read e un Exit per il recupero e presentazione della lista, es: un totale di 2 CFP aggiuntivi in questo processo funzionale FP1.

c) Viene offerto all'utente funzionale di poter scegliere tra due modalità per inserire il valore dell'attributo Z dell'Entry A, a seconda se l'utente funzionale conosca il valore valido da inserire o abbia bisogno di selezionare un valore da una lista di valori validi. Pertanto (i) sia l'utente funzionale inserisce il valore direttamente e il valore viene validato rispetto ad un attributo di un oggetto di interesse B, o (ii) l'utente funzionale può opzionalmente scegliere di mostrare i valori validi dell'attributo in una lista a discesa e poi selezionare il valore corretto, laddove tali valori validi sono ottenuti da un attributo di un altro oggetto di interesse B. In questo caso per l'alternativa (i) si identifica un Read aggiuntivo (dell'oggetto di interesse B) nel processo funzionale FP1 per la validazione dell'inserimento diretto dell'attributo dell'oggetto di interesse A dell'Entry E per l'alternativa (ii) si riconosce un processo funzionale distinto FP2 per la presentazione opzionale della lista di valori validi dell'attributo. Questo processo funzionale distinto FP2 presenta una dimensione funzionale di 3 CFP (1 Entry, 1 Read, 1 Write).

Nota 1: L'oggetto di interesse B deve essere 'genuino'. Si vedano le sezioni 2.6 e 4.2 per maggiori dettagli ed esempi su come distinguere gli oggetti di interesse.

Nota 2: Quando si clicca su una lista a discesa è improbabile (se non impossibile) ricevere un messaggio di errore/conferma. Pertanto nessun Exit viene identificato per 'messaggi' in relazion all'Entry di questo attributo Z.

Il caso b) è alquanto frequente nelle applicazioni di tipo web-based usate da un vasto pubblico laddove il controllo di qualità dei dati inseriti richiede un'attenzione particolare.

Il caso c) si riscontra laddove si forniscono delle facility sia per utenti esperti che inesperti. Primo, gli utenti esperti che applicano l'alternativa (i) possono inserire i valori dell'attributo direttamente e si deve identificare un Read per la sua validazione. Tale Read deve essere identificato per ciascun valore di attributo che è possibile inserire (che può essere validato rispetto ad un oggetto di interesse 'genuino') per ciascun processo funzionale.

Secondo, l'utente inesperto che decida di applicare l'alternativa (ii) deve prendere una decisione consapevole separata per mostrare la lista a discesa di valori validi per la selezione ed inserimento, es: l'utente funzionale innesca un processo funzionale FP2 distinto. Tale processo funzionale FP2, pari ad una dimensione di 3 CFP dovrebbe essere misurato una sola volta per l'intera applicazione. Comunque, ci saranno tanti processi funzionali di tipo FP2 nell'applicazione quanti sono gli attributi per i quali valori validi possono essere presentati in tal modo. L'uso di questa alternativa opzionale (ii) è simile all'uso di una funzionalità di help per ciascuna videata.

4.1.8 Misurazione di processi funzionali apparentemente inter-dipendenti

Il seguente esempio illustra alcune delle difficoltà che talvolta emergono quando i misuratori si confrontano con la necessità di distinguere tra vista logica e fisica del software per gli scopi delle misurazione dimensionale del software.

ESEMPIO: Un'applicazione presenta Requisiti Utente Funzionali (FUR) per due processi funzionali FP1 e FP2.

Il processo funzionale FP1 abilita l'utente funzionale a mantenere un 'indicatore del merito creditizio' per ciascun cliente, con tre possibili valori (livelli): 'eccellente', sufficiente', 'insufficiente'.

Il processo funzionale FP2 abilita un conto di un cliente a ricevere debito o crediti. I FUR stabiliscono che se il valore di un debito usando il processo funzionale FP2 porti ad un saldo negativo del conto, allora l'indicatore di merito creditizio esistente deve essere automaticamente ridotto di un livello.

Lo sviluppatore intravede l'opportunità di evitare la duplicazione del codice della stessa funzionalità. Egli implementa il processo funzionale FP1 in accordo al relativo FUR e successivamente implementa il processo funzionale FP2 usando una parte della funzionalità del processo funzionale FP1 relativa al settaggio dell'indicatore di merito creditizio.

Si identificano due processi funzionali:

- processo funzionale FP1 per mantenere l'indicatore del merito creditizio
- processo funzionale FP2 che include la funzionalità di abbassamento dell'indicatore di merito creditizio che è stato implementato nel processo funzionale FP1.

Discussione

a) La soluzione sopra esposta risulta in parte della funzionalità del processo funzionale FP1 misurato due volte. Ciò è corretto poiché quando si misura la dimensione funzionale di un software, si misura la dimensione dei FUR, non la dimensione della transazione fisica che lo sviluppatore ha scelto di implementare. Questo è un esempio di come lo sviluppatore tragga vantaggio da un riuso funzionale, che è alquanto comune nella progettazione del software.

b) Se l'ambito della misurazione nell'esempio sopra esposto prevede l'inclusione del processo funzionale FP2 ma non quello FP1, allora quando si dimensionerà il processo funzionale FP2, si dovrà misurare solo i movimenti dati del processo funzionale FP2 ricompresi nell'ambito più ogni messaggio di tipo Entry/Exit che attraversi il confine per mantenere l'indicatore di merito creditizio nel software al di fuori dell'ambito della misurazione. Si veda la sezione 4.3 per maggiori dettagli sulle componenti dimensionali degli applicativi aziendali.

4.1.9 La relazione 'molti-a-molti' tra tipi-evento e tipi di processo funzionale

La definizione di un processo funzionale riconosce che un singolo tipo di processo funzionale possa essere innescato da uno o più tipi-evento.

Nel dominio del software applicativo aziendale, un esempio potrebbe essere che un requisito per mostrare le ultime transazioni di un conto bancario 'on demand' possa essere innescato da un impiegato bancario che richieda la dichiarazione da un terminale presso lo sportello bancario con benestare del titolare del conto o da un impiegato in un call center durante una chiamata del titolare del conto stesso. Assumendo che il processo funzionale sia identico in entrambi i casi ed entrambi i casi siano ricompresi nello stesso ambito di misurazione, si misurerà solo un processo funzionale.

L'opposto di questa situazione è altresì possibile. Un singolo tipo-evento può innescare più di un processo funzionale. Questa situazione si verifica nei sistemi real-time laddove, ad esempio, l'individuazione di una condizione di emergenza può innescare molteplici processi funzionali distinti da eseguire in parallelo per fronteggiare l'emergenza. Questo tipo di situazione non è frequente nel dominio del software applicativo aziendale, non può essere esclusa a priori.

4.2 Identificazione di oggetti di interesse, gruppi dati e movimenti dati

Si suppone che il lettore abbia familiarità con la sezione 'Identificazione dei gruppi dati' e 'Identificazione degli attributi dei dati' presente sul Manuale di Misurazione.

4.2.1 Introduzione

Si assuma che per i FUR disponibili (espressi o impliciti), si sia fatto un primo passo verso l'identificazione dei processi funzionali usando la guida della sezione 4.1 e che si debba ora dimensionare ciascun processo funzionale attraverso l'identificazione dei movimenti dati delle sue fasi

di input, elaborazione ed output. In principio ciò può essere conseguito applicando il metodo di analisi dei dati preferito per le fasi di input, elaborazione e output di ciascun processo funzionale in cambio per identificare i diversi gruppi dati da muovere. In pratica, comunque, è maggiormente efficace procedere come indicato nel seguito:

- Primo, costruire un modello logico dei dati degli oggetti di interesse dei dati (memorizzati) persistenti che rientrano nell'ambito della misurazione, usando il metodo di analisi dei dati preferito.
- Successivamente, analizzare le fasi di input, elaborazione e output di ciascun processo funzionale per i gruppi dati da inviare o da cui ottenere memorizzazione persistente e per i gruppi dati transienti nell'input e output.
- Essere preparati a iterare questi due passi, dato che l'analisi dei dati delle fasi di input, elaborazione e output può comportare l'identificazione di nuovi oggetti di interesse, e l'identificazione di un nuovo oggetto di interesse può comportare l'identificazione di nuovi processi funzionali per la sua manutenzione (si veda la sezione 4.1.1 su 'Le transazioni CRUD(L)').

I primi due passi di questa procedura sono descritti con maggior dettaglio nella successiva sezione 4.2.2.

Avviso sui nomi di attributi fuorvianti

Si avvisa il lettore della seguente insidia. Specialmente quando si analizzano input e ouput di processi funzionali complessi o non familiari, è suggeribile esaminare tutti gli attributi al fine di identificare gli oggetti di interesse distinti e quindi i gruppi dati distinti. Ma i nomi dati in genere agli attributi negli artefatti software possono indicare in modo fuorviante più oggetti di interesse distinti di quanti ne esistano al momento. Si danno due esempi:

a) Nell'input per un semplice processo funzionale di 'creazione impiegato', un attributo può essere etichettato come 'livello' quando ciò che realmente significherebbe è 'livello contrattuale'. Questo è un attributo del gruppo dati in ingresso relativo ad un impiegato; sarebbe scorretto assumere che questo indica un gruppo dati 'livello' distinto. C'è solo un Entry 'dati dell'impiegato' in questo semplice esempio.

b) Per una interrogazione per calcolare le vendite ad un certo cliente di un certo prodotto durante un dato lasso di tempo, i parametri di input potrebbero essere 'Identificativo Cliente', 'Identificativo Prodotto', 'Inizio periodo' e 'Fine periodo'. Questi sono gli attributi principali di un gruppo dati transiente relativi ad un oggetto di interesse (le 'merci vendute ad un dato cliente di un dato prodotto durante un dato lasso di tempo'). Questa non è una interrogazione relativa ai dati persistenti di un oggetto di interesse 'Cliente' o 'Prodotto', ma relativi a dati che rappresentano il risultato di una intersezione di questi due e il dato lasso di tempo – un gruppo dati transiente.

   In questo esempio, i parametri di input dovrebbero essere veramente denominati come ad es: 'Identificativo del cliente le cui vendite sono state effettuate nel periodo', 'Identificativo del prodotto venduto al cliente nel periodo', 'Inizio periodo delle vendite', 'Fine periodo delle vendite'. Esiste solo un Entry per la interrogazione.

4.2.2 Identificazione di oggetti di interesse, gruppi dati e movimenti dati

Il processo suggerito sottolineato nella sezione 4.2.1 richiede in cambio di esaminare l'area del problema dalla prospettiva dei dati persistenti, e in seguito dalla prospettiva dei processi funzionali e di iterare tra queste due prospettive.

La scelta del metodo di analisi dei dati è lasciato all'analista di misurazione, secondo la sua esperienza e gli artefatti software disponibili per l'analisi. Il metodo Relational Data Analysis (RDA) è particolarmente utile per analizzare gli artefatti di software già esistenti.

Il modello logico dei dati persistenti

Si inizia dalla costruzione di un modello logico degli oggetti di interesse per i quali sono richiesti dati persistenti per l'esistenza di tutto il software nell'ambito della misurazione. Si esaminano i FUR per i nomi, ad es: nomi di 'cose' i cui dati debbano o sono conservati. Ci si assicura di seguire le indicazioni

fornite nelle sezioni 2.6.2 e 2.6.3 per identificare gli oggetti di interesse per i diversi tipi di utenti funzionali.

Dopodiché, per ciascun processo funzionale:

Per la fase di input di ciascun processo funzionale:

Si identifica un Entry per ogni gruppo dati unico nell'input, indipendentemente se alcuni o tutti i dati inseriti sia riferibili a memorizzazioni persistenti o meno, o se siano transienti (si ricordi che si stanno cercando tipi di gruppi dati; differenti frequenze di occorrenze indicano sempre diversi tipi di gruppi dati).

In aggiunta, si identifica un Read per ogni attributo di input il cui valore debba essere validato rispetto ai dati persistenti esistenti di un oggetto di interesse per questo processo funzionale, più un Exit se i valori validi dei dati persistenti esistenti di un oggetto di interesse debbano essere presentati per la selezione da parte dell'utente, ad es: in una lista a discesa.

Ogni processo funzionale deve avere almeno un Entry.

Per la fase di elaborazione di ciascun processo funzionale:

Per ciascun oggetto di interesse per il quale i dati persistenti sono richiesti per recuperare o per i quali dati è richiesto di essere resi persistenti, si identifica rispettivamente un Read o un Write (anche tenendo a mente l'unicità dei movimenti dati e le possibili eccezioni alle regole; si veda il Manuale di Misurazione).

Per la fase di output di ciascun processo funzionale:

Si identifica un Exit per ogni gruppo dati unico nell'output, indipendentemente se il Gruppo dati sia ottenuto direttamente dai dati nella memoria permanente o sia derivata e transiente (si ricordi che si stanno cercando tipi di gruppi dati; differenti frequenze di occorrenze indicano sempre diversi tipi di gruppi dati).

Ciascun processo funzionale deve avere un 'risultato' che si realizza almeno con un Write o un Exit.

I processi funzionali di applicazioni applicative molto comuni includono un Exit per messaggi di errore/conferma, sebbene ciò non sia sempre richiesto.

4.2.3 Esempi

Gli attributi principali sono sottolineati.

Esempio 1a – Interrogazione semplice

Si abbia un database con due oggetti di interesse:

Cliente (ID Cliente, nome cliente, indirizzo, …)
Ordine (ID Ordine, ID Cliente, ID prodotto, data di ordinazione, …)

Il FUR per l'esempio 1a riporta che "si richiede un'interrogazione per poter inserire la data di inizio e termine di un dato periodo e per ottenere tali dati più una lista di identificativi (ID) clienti per ciascun cliente che ha effettuato ordini in tale periodo, con il numero dei relativi ordini. Gli ordini sono per singolo articolo, es: un prodotto per ordine".

La soluzione per questo processo funzionale, ignorando possibili messaggi di errore/conferma è riportata nel seguito:

E Date di inizio e fine periodo
R Dati dell'ordine
X ID Cliente, date di inizio e fine periodo, numero di ordini.

In questo esempio, l'Entry muove un gruppo di dati transiente con gli attributi 'data_inizio_periodo_richiesta' e 'data_fine_periodo_richiesta' di un oggetto di interesse, 'periodo_temporale'. L'Exit muove un gruppo dati transiente relativo ad un oggetto di interesse che

potrebbe essere definito come “il business di un dato cliente in un dato periodo di tempo”. La chiave di questo oggetto di interesse è (data_inizio_periodo_richiesta, data_fine_periodo_richiesta, ID Cliente) e il suo solo attributo non-chiave è “il numero di ordine del cliente effettuati nel periodo”. Esiste una occorrenza dell’Exit per ciascun cliente che ha effettuato ordini nel periodo.

Si noti che in questo Esempio 1a, logicamente è solo necessario leggere l’oggetto di interesse dell’Ordine per ottenere i dati necessari per l’output, che è il motivo per cui non ci sono Read di Cliente da misurare. La dimensione di questo processo funzionale è pari a 3 CFP.

Esempio 1b – Interrogazione più complessa

Il FUR dell’esempio 1b è identico a quello dell’esempio 1a, ma con l’aggiunta “inoltre, si richiede in output il nome e l’indirizzo del Cliente e l’Identificativo Cliente per ciascun Cliente che abbia effettuato un ordine nel periodo”.

La soluzione per questo processo funzionale diventa ora:

E Date di inizio e fine periodo
R Dati dell’ordine
R Dati del cliente
X ID Cliente, nome Cliente, Indirizzo Cliente
X ID Cliente[10], date di inizio e fine periodo, numero di ordini.

In questo esempio è ora necessario leggere l’oggetto d’interesse del Cliente per poter ottenere il suo nome e indirizzo. Inoltre, applicando la RDA in 3NF ai dati richiesti per l’output ci dice che ora abbiamo due Exit, uno che muove dati persistenti relativi al cliente (un oggetto di interesse) e l’altro che muove dati transienti relativi al “business di un dato cliente in un dato periodo di tempo” (altro oggetto di interesse). Pertanto ci sono due Exit presenti ogni volta per ogni cliente.

La stessa soluzione si applica anche se lo sviluppatore decide di disporre i dati in un un modulo differentemente stampato come:

> Date di inizio e fine periodo (quale testata del report)
>
> ID Cliente, nome cliente, indirizzo cliente, numero di ordini (una linea per ogni occorrenza di Cliente)

La dimensione di questo processo funzionale è pari a 5 CFP.

Esempio 1c – Interrogazione ulteriormente complessa

Il FUR dell’esempio 1c è identico a quello dell’esempio 1b, ma il processo funzionale deve permettere l’inserimento fino a tre diversi periodi temporali e l’output deve mostrare il numero di ordini effettuati dal cliente in ogni periodo. Il modo naturale di presentare il report ora sarebbe:

ID Cliente, nome cliente, indirizzo cliente

> Date inizio e fine periodo 1, numero di ordini effettuati
> Date inizio e fine periodo 2, numero di ordini effettuati
> Date inizio e fine periodo 3, numero di ordini effettuati

Questi blocchi sono ripetuti per ciascun cliente.

[10] In questi esempi, per convenienza usiamo per gli attributi dati delle abbreviazioni e nomenclature comunemente usate. Risulta che lo stesso nome attributo ‘ID Cliente’, usato nelle due Exit in questo esempio può confondere e essere tecnicamente incorretto, dato che essi hanno diversi significati e rappresentano pertanto attributi differenti. ‘ID CIlente’ ‘è un buon nome nella prima Exit dato che è per questo gruppo dati un attributo chiave. Ma se si volesse essere veramente puntuali, un nome migliore per questo attributo nella seconda Exit potrebbe essere ad es: ‘ID del Cliente che ha effettuato un ordine’. Si vedano le ‘Avvertenze’ nella sezione 4.2.1 per altri esempi che possono causare simili difficoltà.

Ora, in questo esempio non è cambiato nulla fondamentalmente dall'esempio 1b. L'oggetto di interesse sia degli Entry che degli Exit sono le stesse dell'esempio 1b. Semplicemente ora si hanno occorrenze multiple degli stessi dati. La dimensione di questo processo funzionale è pari a 5 CFP.

Esempio 2 – Un processo funzionale di data entry

Il FUR richiede un processo funzionale che "abiliti l'inserimento di un ordine multi-articolo in un database che presenta già dati persistenti relativi a clienti e prodotti, dove

- gli ordini multi-articolo hanno i seguenti attributi:
  Intestazione ordine (ID Ordine, ID cliente, rif. ordine cliente, data consegna richiesta, ecc.)
  Articolo ordinato (ID Ordine, ID articolo, ID prodotto, quantità ordine, ecc.)
- gli identificativi ID ordine e ID articolo sono generate automaticamente dal software e
- l'ID cliente e l'ID prodotto devono essere validati all'inserimento."

Soluzione per questo processo funzionale

E Intestazione ordine
R Dati del cliente
E Articolo dell'ordine
R Dati del prodotto
W Intestazione ordine
W Articolo dell'ordine
X Messaggi di errore/conferma

In questo esempio, la chiave unica dell'intestazione ordine è il suo ID ordine, e la chiave unica dell'articolo-ordine è la combinazione (ID ordine, ID articolo). L'ID cliente nell'intestazione dell'ordine è "il cliente che effettua l'ordine". Esso rappresenta una parte essenziale dei dati relativi all'ordine. Non si stanno inserendo dati relativi al cliente. Pertanto non va identificato un Entry separato per il cliente. Analogamente l'ID prodotto rappresenta una parte essenziale dei dati dell'articolo-ordine, pertanto non va identificato un Entry separato per il Prodotto. I dati inseriti sono relativi esclusivamente a due oggetti di interesse, l'intestazione dell'ordine e l'articolo-ordine. I Read per i dati Cliente e Prodotto sono richieste per controllare che l'ID cliente e l'ID prodotto inseriti siano validi. La dimensione di questo processo funzionale è pari a 7 CFP.

Esempio 3 – Interrogazione semplice con condizione in input

Si consideri il seguente Requisito Utente Funzionale (FUR1):

"Il software deve supportare una interrogazione ad-hoc ad un database degli impiegati per estrarre una lista di nominativi di tutti gli impiegati più anziani di una data età, laddove tale età deve essere inserita".

La soluzione per il processo funzionale FUR1:

E Il limite di età dell'impiegato (della richiesta ad-hoc)
R Dati dell'impiegato
X Nominativi degli impiegati (per tutti coloro che abbiano superato tale limite di età)

Per poter comprendere questa soluzione, osserviamo i gruppi dati menzionati in questo requisito. Quello più ovvio è il gruppo dati persistente "Dati dell'impiegato".

Il parametro di selezione dell'interrogazione, rilasciato da un Entry, è il solo attributo di un secondo gruppo dati transiente, il cui oggetto di interesse è "l'insieme degli impiegati che hanno oltrepassato il limite fissato di età". Un terzo gruppo transiente di dati di interesse è 'un impiegato la cui età è entro il limite stabilito'. Si noti che l'oggetto di interesse della Entry e della Exit non è lo stesso. Un insieme, e un membro di tale insieme, non sono la stessa 'cosa'.

La dimensione del processo funzionale del FUR1 è pari a 3 CFP, ignorando ogni necessità di messaggi di errore/conferma.

Se ci fosse anche un requisito (FUR2) per presentare il limite di età in aggiunta al requisito del FUR1, allora ci sarebbe anche un extra Exit, poiché il 'limite di età degli impiegati' è un attributo dell'insieme degli impiegati della interrogazione. La dimensione totale di FUR1+FUR2 sarebbe quindi di 4 CFP.

Se ci fosse ora un ulteriore requisito (FUR3) per presentare il numero totale di impiegati che rientrano nei criteri del limite di età in aggiunta al limite di età (del FUR2), questo rappresenterebbe un secondo attributo dello stesso gruppo dati che contiene il limite di età e la dimensione del processo funzionale che intende soddisfare FUR1 + FUR 2 + FUR3 sarebbe quindi non modificata e pari a 4 CFP.

Esempio 4 – Report con aggregazioni multi-livello

Si consideri il seguente insieme di Requisiti Utente Funzionali (FUR):

- "Il software contiene tutti i dati del personale della compagnia in un archivio. Un attributo relativo a ciascun impiegato è l'ID dipartimento a cui appartiene alla data.
- Una tabella separata elenca tutti gli ID dipartimento assegnando anche il nome della divisione a cui il dipartimento appartiene.
- È richiesto un report che elenchi tutti gli impiegati della compagnia ordinate per nome, divisione e dipartimento all'interno della divisione. Tale report dovrebbe inoltre presentare I sotto-totali dei numeri di impiegati per ciascun ID dipartimento e per ogni divisione (per nome), e il numero totale di impiegati per l'intera compagnia".

La soluzione per il processo funzionale che soddisfa tali FUR è:

E Selezione del report
R Dati dell'impiegato
R Dati del dipartimento
X Nominativi degli impiegati (raggruppati per dipartimento all'interno delle divisioni)
X Sotto-totale degli impiegati per Dipartimento
X Sotto-totale degli impiegati per Divisione
X Totale degli impiegati della Compagnia

In questo esempio, per produrre questo report si richiede un processo funzionale che deve essere innescato da un Entry. Il processo funzionale deve coinvolgere due movimenti dati di tipo Read, degli oggetti di interesse 'Impiegato' e 'Dipartimento', al fine di ottenere i dati necessari per la memoria persistente ('Dipartimento' deve essere un oggetto di interesse per gli utenti funzionali di questa applicazione dato che effettivamente definisce la struttura della compagnia). Dato che il report presenterà i nomi di tutti gli impiegati, deve esserci anche un Exit per l'oggetto di interesse 'Impiegato', visto che tutti i nomi saranno stampati.

In ogni caso, il report deve anche mostrare il numero totale di impiegati per i tre livelli di aggregazione, espressamente per ogni dipartimento, divisione e per l'intera compagnia. Tali totali sono attributi dei tre oggetti di interesse, rispettivamente:

- l'insieme degli impiegati per ogni dipartimento;
- l'insieme degli impiegati per ogni divisione;
- l'insieme degli impiegati della compagnia.

Pertanto, è necessario identificare un Exit per ciascuna di queste (sebbene l'oggetto di interesse 'totale impiegati della compagnia' si verifichi solo una volta nell'output). Perciò in totale questo processo funzionale presenta 1 Entry, 2 Read e 4 Exit, es: la sua dimensione è di 7 CFP (ignorando la possibile richiesta di messaggi di errore/conferma, che non sono stati espressamente richiesti nel FUR).

Esempio 5 – Processi funzionali con sotto-tipi di oggetti di interesse

A volte si dovrebbe riconoscere oggetti di interesse distinti quali sotto-tipi di un particolare oggetto di interesse.

a) Si supponga (FUR1) che la classe-oggetto o entità 'Cliente' abbia un attributo che indichi la sua tipologia come 'Personale', 'Rivenditore', o 'Grossista'. Se le regole che governano l'elaborazione dei dati di tutti i clienti sono le stesse, e tutti i clienti hanno gli stessi attributi dati, allora questi tre tipi NON sono classificabili quali sotto-tipi di 'Cliente'. 'Tipo-Cliente' è semplicemente un attributo di 'Cliente'.
b) In alternativa, si immagini il seguente FUR2.

- “Ci sarà un database clienti dove tutti i clienti hanno gli attributi ID Cliente, Nome Cliente, Indirizzo, Num. telefono e Tipo-Cliente.
- ‘Tipo-Cliente’ può assumere tre valori: P = Personale, D = Dettagliante, G = Grossista
- I clienti di tipo ‘Personale’ non presentano attributi aggiuntivi.
- I clienti di tipo ‘Dettagliante’ possiedono attributi addizionali: Area di Vendita, Limiti di Credito, Fatturato dell’ultimo anno, la tabella di sconto del prezzo al cliente ‘Dettagliante’.
- I clienti di tipo ‘Grossista’ possiedono attributi addizionali: Referente, Limiti di Credito, Fatturato dell’ultimo anno, la tabella di sconto del prezzo al cliente ‘Grossista’, Termini di pagamento.

Dato che i Clienti presentano alcuni attributi comuni, ma ne hanno anche di differenti in relazione alla loro Tipologia-Cliente, ne consegue che Personale, Dettagliante e Grossista siano sotto-tipi di Cliente. Pertanto da un punto di vista logico l’oggetto di interesse ‘Cliente’ ha tre sotto-tipi di oggetti di interesse (‘Cliente Personale’, ‘Cliente Dettagliante’, ‘Cliente Grossista’).

La struttura fisica del database sarebbe probabilmente come illustrato nella seguente figura. Non c’è bisogno di un record distinto del database per i clienti personali, dato che essi hanno solo gli attributi comuni a tutti i Clienti.

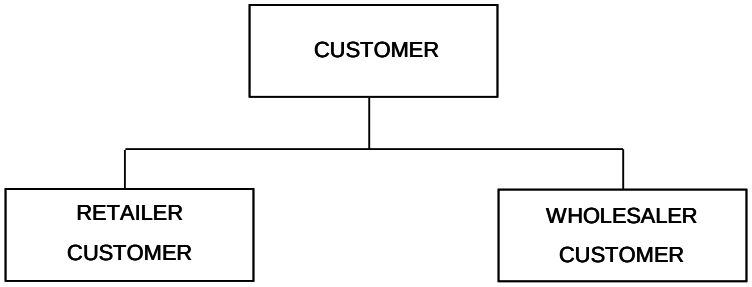

Figura 4.2.3 – Tipo di oggetto di interesse e relative sotto-tipi

Si introducono ora dei FUR aggiuntivi per I diversi tipi di processi funzionali che illustreranno gli effetti della necessità di referenziare tali tipi e sotto-tipi di oggetto di interesse.

FUR3. “Un processo funzionale è richiesto per produrre le etichette nome-e-indirizzo per tutti i Clienti”. Questo processo funzionale leggerebbe (Read) solo l’oggetto di interesse ‘Cliente’.

FUR4. “Un processo funzionale è richiesto che abiliti gli inserimenti di ordini per un Cliente Personale”. Questo processo funzionale necessiterebbe solo di leggere (Read) l’oggetto di interesse ‘Cliente Personale’ al fine di validare l’esistenza del particolare Cliente Personale che effettua l’ordine.

FUR5. “Un singolo processo funzionale è richiesto per abilitare l’inserimento di ordini per ogni Cliente Rivenditore o Grossista”. Questo processo funzionale avrebbe bisogno di distinguere due oggetti di interesse separati (Cliente Rivenditore e Cliente Grossista) al fine di validare che il Cliente esiste e che gli ordini possano essere accettati, e successivamente applicare le corrette regole di sconto per assegnare un prezzo all’ordine. Pertanto questo processo funzionale FP3 dovrebbe avere due Read per questi due sotto-tipi.

FUR6. “Un singolo processo funzionale è richiesto per inserire i dati per un nuovo cliente di qualsiasi tipologia”. Questo processo funzionale avrebbe bisogno di avere Entry e Write separati per ciascuno dei tre sotto-tipi Personale, Rivenditore e Grossista, es: con un totale di sei CFP per tali sotto-processi.

Il principio generale è che laddove c’è necessità di distinguere più di un sotto-tipo nello stesso processo funzionale, ogni sotto-tipi deve essere trattato come un oggetto di interesse separato.

Esempio 6 – Manutenzione di tabelle di parametri

Al fine di migliorare la manutenibilità, è pratica comune quella di memorizzare gli attributi di sistemi di codifica (es: codici e nomi degli stati), liste di nomi standard (es: fornitori di carte di credito accettate), clausole testuali (es: formule standard nelle lettere), i parametri delle regole di elaborazione, ecc. in tabelle e fornire software che aiuti a mantenere tali dati. Ciascun tipo di dato – ci si riferisce ad essi complessivamente come ai 'parametri (tipi di)' – tipicamente ha pochi attributi. Ad esempio, un sistema di codifica può avere solo codici e descrizioni e possibilmente date di validità iniziali e finali quali attributi.

La maggioranza di tali tipi di parametro non saranno oggetti di *interesse* per gli utenti funzionali aziendali del software applicativo che usa i parametri, ma alcuni potrerebbero esserlo. Come illustrato nella sezione 2.6.2, un parametro per poter essere un oggetto di interesse deve avere propri attributi che siano di interesse e che in genere siano mantenuti dall'utente aziendale. Il misuratore deve perciò normalmente esaminare tutte le tabelle dei parametri definite nell'ambito della misurazione al fine di determinare se esistano parametri riconoscibili quali oggetti di interesse per l'utente aziendale.

In questa sezione la misurazione delle funzioni per mantenere tabelle di parametri viene esaminata laddove la manutenzione è effettuata da un utente funzionale che è un 'utente non-aziendale'. (in precedenza si è usato questo termine per includere 'amministratori di sistema', 'responsabili applicativi', staff tecnico o di sviluppo, es: tutti coloro le cui attività supportino l'applicazione mantenendo i parametri, ad esempio codici e descrizioni valide, ma chi non è un 'utente aziendale' normale, autorizzato). Per un utente funzionale non-aziendale che deve mantenere le tabelle di parametri, gli attributi dei parametri descrivono gli oggetti di *interesse*.

Possono verificarsi diverse situazioni.

a) Le tabelle dei parametri sono tipicamente fornite con un insieme di 'processi funzionali CRUD' (Create, Read, Update, Delete) per mantenere i valori del parametro, che devono essere disponibili per l'utente non-aziendale.

   Nel caso più semplice possibile, un insieme di processi funzionali CRUD dovrebbe essere rilasciato per mantenere tutti i parametri in una tabella. Ad esempio, per modificare ciascuna occorrenza di un parametro esistente, l'utente non-aziendale dovrebbe visualizzare i contenuti della tabella dei parametri e la pagina attraverso i quali trovare la particolare occorrenza di parametro e i suoi attributi da mantenere. Effettivamente esiste un unico oggetto di interesse per l'utente non-aziendale, ovverosia il 'parametro'. Ciascuno dei quattro processi funzionali CRUD dovrebbe avere la dimensione di 3 CFP (1 Entry, 1 Write o Read, 1 Exit), per un totale di 12 CFP per l'insieme CRUD, assumendo che non ci siano messaggi di errore/conferma.

b) Una situazione più realistica di quella descritta in a) è quella in cui uno speciale software di utilità venga fornito per mantenere i parametri. Sfortunatamente è impossibile generalizzare il dimensionamento dei processi funzionali di tale software dato che possono essere molteplici potenziali variazioni.

   All'altro estremo del caso a), potrebbe accadere che gli attributi e le regole di validazione per ciascun tipo-parametro siano così differenti che ciascun tipo-parametro abbia bisogno del proprio insieme di processi funzionali CRUD per mantenere i propri valori di occorrenza e ogni tipo-parametro rappresenta un oggetto di interesse separato. La dimensione di questa utility sarebbe quindi almeno pari al numero dei tipi-parametro x 4 processi funzionali x 3 CFP per processo funzionale.

   Più probabilmente ci sarebbe qualche comunanza di attributi e regole di validazione attraverso i vari tipi-parametro. Ad esempio, per sistemi di codifica, ciascun sistema deve essere associato un insieme di regole di convalida che impattano i processi funzionali di 'Create' e 'Update'; tali regole saranno identiche per alcuni sistemi di codifica, ma è improbabile che possano essere le stesse per tutti.

   Si supponga di considerare sette sistemi di codifica i cui attributi codice/descrizione abbiano le stesse esigenze di validazione, in modo che i loro valori possano essere mantenuti da un insieme di processi funzionali CRUD. Si consideri il processo funzionale 'Create'. Si supponga che l'utente non aziendale richiami questo processo funzionale e debba innanzitutto selezionare da un elenco il particolare sistema di codifica che ha bisogno di nuovi valori. L'utente può quindi inserire una o più coppie di nuovi attributi codice/descrizione. Questo processo crea funzionale presenta 2 Entry

(uno per la scelta del sistema di codifica, uno per l'inserimento del codice/descrizione), 1 Write e 1 Exit. Totale 4 CFP. Potrebbe sembrare che dovrebbero esserci 7 Write , ma in questo caso i soggetti di queste sette sistemi di codifica sono realmente stati conglobati in un unico oggetto di interesse, quindi vi è solo un solo 1 Write.

Per quanto riguarda i requisiti per i processi funzionali di tipo 'Read', esistono infinite possibilità, ad es:.

- Richiedere che la descrizione corrisponda da un dato codice,
- Presentare tutti I codici/descrizione per un dato sistema di codifica.

Probabilmente tutti questi requisiti potrebbe essere soddisfatti da uno o due processi funzionali per tutti I sistemi di codifica di alcuni software per la manutenzione delle tabelle dati di decodifica generali.

L'utilità generale dell'esempio b) potrebbe tornare utile per molte applicazioni aziendali. Se le sue dimensioni dovessero essere incluse con quelle di una qualsiasi delle applicazioni, rimane ancora aperta la questione relativa alla definizione dell'ambito della misurazione.

4.3 Dimensionare le componenti di applicazioni aziendali

Quando un'applicazione aziendale è distribuita su due o più piattaforme tecniche e lo scopo di una misurazione è quello di misurare la dimensione di ogni componente dell'applicazione su ciascuna piattaforma, deve essere definito un ambito di misurazione distinto per ciascuna componente dell'applicazione. In tal caso, il dimensionamento dei processi funzionali di ciascuna componente segue tutte le normali regole, come descritto finora. L'unico nuovo punto riguarda il come gestire i movimenti dati coinvolti nelle comunicazioni tra componenti.

Dal processo per ciascuna misurazione (... definire l'ambito, quindi gli utenti funzionali e il confine, ecc.) consegue che se un'applicazione è costituita da due o più componenti, non ci può essere alcuna sovrapposizione tra gli ambiti di misurazione delle varie componenti. L'ambito di misurazione per ogni componente deve definire un insieme completo di processi funzionali, ad esempio non può esserci un processo funzionale in parte nell'ambito di un'applicazione e in parte in un altro. Allo stesso modo, i processi funzionali all'interno dell'ambito di misurazione di una componente non hanno alcuna conoscenza dei processi funzionali (oggetti di interesse inclusi) nell'ambito di un'altra componente, anche se le due componenti scambiano messaggi.

Gli utenti funzionali di ciascuna componente sono determinati al fine di esaminare dove accade che gli eventi inneschino processi funzionali nella componente sotto esame (gli eventi di innesco possono verificarsi solo nel mondo dell'utente funzionale).

Negli esempi che seguono, si discute dapprima il caso (a) dove lo scopo è quello di dimensionare un'intera applicazione ignorando che questa sia suddivisa in due componenti su piattaforme tecniche distinte e poi i casi (b) e (c) dove lo scopo è quello di dimensionare separatamente ciascuna componente dell'applicazione.

Esempio 1

Si supponga di avere un'applicazione semplice client-server, con due componenti, ad esempio la componente A gira su un front-end PC che comunica con una componente B su un computer main-frame che detiene alcuni dati 'legacy' che descrivono un oggetto di interesse. L'utente (umano) funzionale innesca un processo funzionale sul PC che richiede questi dati per essere letti dal main-frame e presentati sul PC.

Caso (a) L'ambito della misurazione è l'intera applicazione

L'utente (umano) funzionale dell'applicazione non ha conoscenza di come l'applicazione sia fisicamente distribuita tra PC e main-frame. La distribuzione dell'applicazione tra le due componenti è invisibile, così come i movimenti di dati tra le due componenti. Analogamente, ogni funzionalità aggiuntiva negli strati più bassi necessaria per poter comunicare tra PC e main-frame è invisibile e non viene considerata. I movimenti dati di questo processo funzionale sono rappresentabili come illustrato nella seguente figura 4.3.1 (totale: 3 CFP).

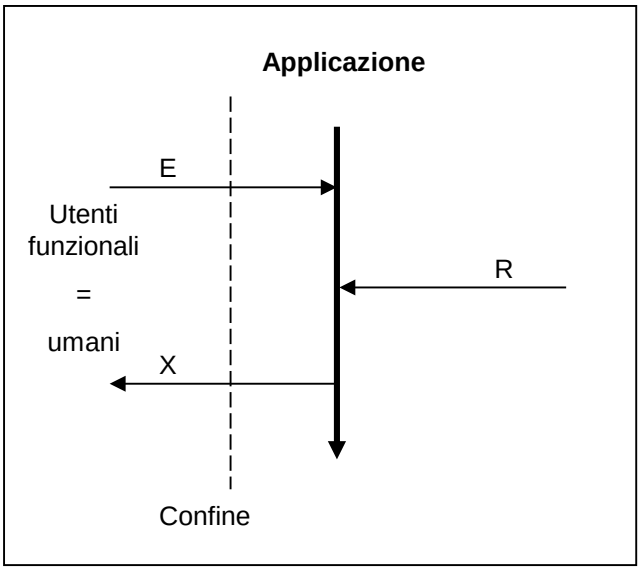

Figura 4.3.1 - L'ambito della misurazione è l'intera applicazione

Caso (b) L'ambito della misurazione è la componente A

Dal punto di vista degli utenti (umani) funzionali, il caso (b) della componente A del PC è identica al caso (a) per l'intera applicazione. Gli utenti funzionali (umani) della componente del PC non hanno conoscenza di come venga eseguito un Read sul PC, ad es: in modo locale o remoto. Dal loro punto di vista, essi innescano un processo funzionale per recuperare i dati da una memoria persistente e ricevere in ritorno i risultati.

Ma delimitando l'ambito della misurazione alla componente A nella funzionalità rivela in realtà la funzionalità effettivamente necessaria per ottenere i dati dalla componente B piuttosto che tramite un Read effettuato in locale - vedi la figura 4.3.2 seguente. La componente A deve emettere una 'richiesta per ottenere' (o un comando 'get') con criteri di selezione dei dati come un Exit verso la componente B e ricevere di ritorno i dati richiesti dalla componente B come un Entry. (La componente A non ha conoscenza di come la componente B ottiene i dati richiesti; potrebbe essere attraverso un Read, o attraverso un calcolo effettuato in locale o da qualche altro software.)

La componente B è diventata un altro utente funzionale della componente A, oltre agli utenti funzionali (umani) della componente A già discussi nel caso (a). I movimenti dati del caso (b) sono quindi riportati come illustrato nel seguito (4 CFP).

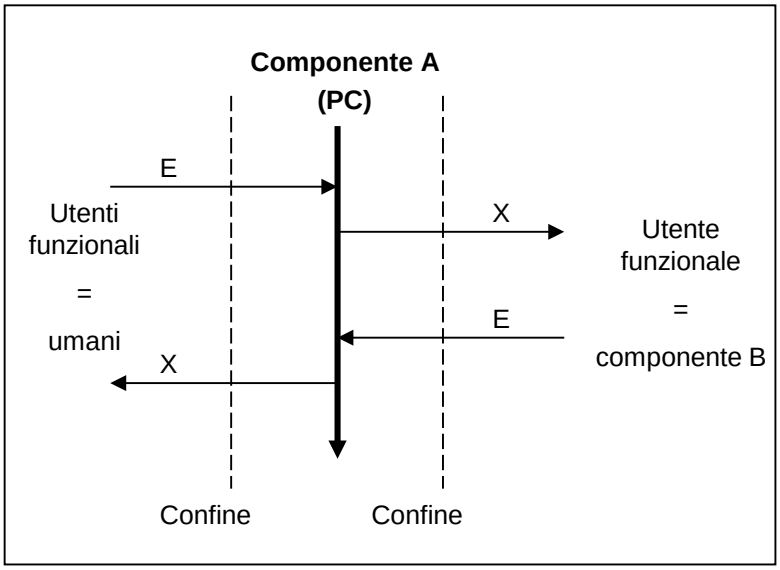

Figura 4.3.2 - L'ambito della misurazione è la componente A

Caso (c) L'ambito della misurazione è la componente B

Definire l'ambito come 'componente B' rivela che l'utente funzionale della componente B è ora la componente A dell'applicazione software PC, dove avviene l'evento di innesco di una richiesta di lettura dei dati. Si noti che le componenti A e B sono utenti funzionali l'una dell'altra; il loro scambio di messaggi 'peer-to-peer' avviene attraverso un confine comune.

Nella componente B, il processo funzionale 'Read dal database' è innescato dall'Entry 'richiesta di ottenere' dalla componente A con i parametri di lettura. La componente B esegue il Read e presenta un Exit verso la componente A, che contiene i dati richiesti e un codice di ritorno.

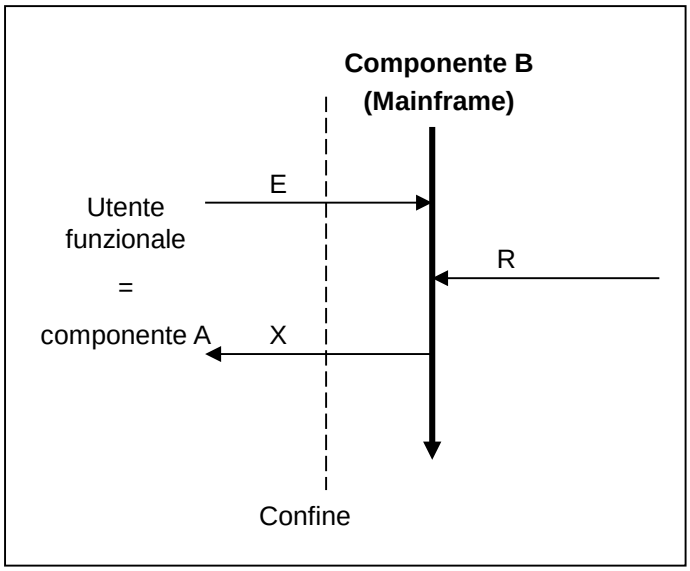

Figura 4.3.3 - L'ambito della misurazione è il componente B

Si può osservare che quando si misura separatamente, la dimensione (del processo funzionale) della componente B è pari a 3 CFP.

Ne consegue che l'incremento nella dimensione totale dell'applicazione quando lo scopo è quello di misurare separatamente la dimensione delle due componenti dell'applicazione è (4 + 3 – 3) = 4 CFP rispetto al caso (a).

Un ulteriore controllo incrociato per questa conclusione, e come sottolineato nella sezione 3.1.1, esempio c) sopra illustrato, la dimensione dell'applicazione che ignora la separazione fisica in componenti dovrebbe essere (la dimensione totale ottenuta attraverso la somma della dimensione di ogni componente misurata separatamente, es: 7 CFP) *meno* (la dimensione dei movimenti dati inter-componente, es: 4 CFP) pari a 3 CFP, come nel caso (a).

Esempio 2

Si supponga ora di avere un'applicazione con due componenti distribuite tra piattaforme tecniche distinte, laddove ogni componente può innescare processi funzionali nell'altra componente. Un esempio potrebbe essere quello di un'applicazione che ha una componente A su un laptop che abilita il personale dei punti vendita ad inserire dati via internet verso un'altra componente B su un server. Quest'ultima può a sua volta innescare processi funzionali per re-inviare i dati al laptop, indipendentemente dai processi funzionali della componente A.

Caso (a) L'ambito di misurazione è l'intera applicazione

Questo caso differisce ora dal precedente esempio 1 del caso (a). L'applicazione ha due utenti funzionali, gli utenti funzionali umani (es: il personale dei punti vendita) e 'qualcosa' (es: un clock) che innesca la trasmissione dei dati dal server ai laptop. Ma la funzionalità che abilita le due componenti perché comunichino tra di loro rimane invisibile.

Così, ad esempio, un processo funzionale che è innescato su un server per inviare dati ai laptop dovrebbe avere:

- un Entry dal clock per innescare il processo (fisicamente sul server)
- uno o più Read (fisicamente sul server) per recuperare I dati dalla memoria persistente
- uno o più Write per rendere i dati persistenti e/o Exit per presentare i dati (fisicamente sui laptop)

Caso (b) L'ambito della misurazione è la componente A

La componente A ora ha due utenti funzionali, ovverosia l'utente umano (es: il personale dei punti vendita) e la componente B, laddove entrambi possono innescare i processi funzionali della componente A. Tali processi funzionali dovrebbero essere analizzati come di consueto.

Caso (c) L'ambito della misurazione è la componente B

La componente B ha altresì due utenti funzionali, ovverosia la componente A e il 'qualcosa' (es: un clock) che innesca i suoi processi funzionali. Tali processi funzionali della componente B dovrebbero essere analizzati come di consueto.

L'effetto sulla dimensione complessiva dell'applicazione per questo esempio 2 con lo scopo di dimensionare separatamente ciascuna componente, come nei casi (b) e (c) è probabile che rappresenti un numero significativo di CFP aggiuntivi, dovuto ai movimenti dati inter-componente rispetto al caso (a), laddove lo scopo era quello di dimensionare l'applicazione ignorando la separazione in componenti distinte.

4.4 Altre convenzioni di misura

4.4.1 Comandi di controllo e dati generici-applicativi

I 'comandi di controllo' sono definiti come 'comandi che permettono all'utente funzionale di controllare il proprio uso del software ma che non comportano alcun movimento di dati su un oggetto di interesse'. I comandi di controllo devono essere ignorati nell'ambito del dominio applicativo aziendale.

Esempi di comandi di controllo sono le funzioni che consentono ad un utente funzionale di visualizzare/non visualizzare un'intestazione oppure visualizzare/non visualizzare (sotto-)totali calcolati, navigare su e giù e navigare tra schermi fisici, cliccare su 'OK' per riconoscere un messaggio di errore o per confermare alcuni dati inseriti, ecc. I comandi di controllo, pertanto comprendono anche i comandi di menu che permettono all'utente di navigare verso uno o più processi funzionali specifici, che di per sè non avviano alcun processo funzionale. Si veda anche il paragrafo 4.4.2.

Analogamente, 'dati generici-applicativi', vale a dire i dati relativi all'applicazione in generale e non legati ad un oggetto di interesse di uno specifico processo funzionale, devono essere ignorati. Quindi i dati di intestazione e piè di pagina (nome azienda, nome dell'applicazione, data di sistema, ecc.) che compaiono su tutte le schermate e i report, che non sono specificamente correlati agli oggetti di interesse su questi schermi o report, non vengono misurati.

4.4.2 Menu e eventi di innesco

Come detto sopra, un comando di menu che consente all'utente di navigare nel software, ma che non lancia alcun processo funzionale (per esempio, consente all'utente unicamente di navigare verso altri sottomenu) deve essere considerato come un semplice comando di controllo e deve essere ignorato nel dominio del software applicativo aziendale. Analogamente, si ignora il comando di menu che produce come risultato la visualizzazione di una schermata 'vuota' per l'inserimento dati di uno specifico processo funzionale. Lo/gli Entry di questo processo funzionale è/sono nello schermata compilata e il processo funzionale si considera innecsato attivato quando riceve il primo Entry.

Spesso tuttavia un comando di menu avvierà un processo funzionale specifico, con o senza dati di input per tale processo, per esempio un processo funzionale di interrogazione. Quando un utente preme un pulsante su un menu per lanciare uno specifico processo funzionale di interrogazione P,

viene inviato un messaggio al software indicando 'avvia l'interrogazione P '. Se l'interrogazione richiede anche dei parametri di selezione che l'utente deve specificare prima di premere 'Invio', questi sono attributi aggiuntivi dello stesso gruppo di dati (a questo punto il messaggio è: 'iniziare l'interrogazione P con questi parametri'). In entrambi i casi il messaggio è quello attivato dal Entry per lo specifico processo funzionale. L'oggetto di interesse è 'l'oggetto di interrogazione P '.

Ogni processo funzionale deve avere una modalità di prima attivazione, l'ultima solitamente è associata alla manipolazione dei dati - in questo caso si tratta dell'inizializzazione del processo funzionale. Quindi anche quando un processo funzionale di interrogazione non ha bisogno di dati in ingresso, viene attivato da un unico click su un pulsante del menu appare come se fosse un comando di controllo ma ci sarà sempre qualche manipolazione dei dati per la richiesta specifica. Pertanto, anche se 'nessun dato' è inserito, consideriamo questo l'utilizzo del pulsante del menu come voce di attivazione del processo funzionale.

4.4.3 Applicazioni elaborate in modalità batch

Fondamentalmente, nella misurazione della dimensione dei processi funzionali, non si deve fare alcuna differenza tra i processi funzionali che devono essere elaborati in modalità "on-line o in modalità batch. Tuttavia i requisiti utente funzionali per l'elaborazione in modalità batch sono talvolta necessariamente diversi dai loro equivalenti per elaborazioni on-line. Per esempio le caratteristiche dell'interfaccia utente (GUI) riguardano unicamente l'elaborazione on-line, mentre in un flusso di elaborazione batch, tutti i processi funzionali possono dover produrre dei messaggi di errore su un unico file comune stampato come report comune degli errori.

Le attività principali del misuratore nell'analizzare flussi applicativi batch sono nel decidere quali sono le azioni distinte di avvio e, pertanto, i diversi processi funzionali e gli utenti funzionali coinvolti nei processi batch.

La questione su chi o cosa sono gli utenti funzionali dei processi funzionali da elaborare in modalità batch, può essere illustrata con tre casi.

a) I dati di input per il processo batch sono stati inseriti on-line da persone e sono stati accorpati in un file per l'elaborazione batch. L'elaborazione del file può essere analizzata considerando le persone come utenti funzionali.
b) Un file di dati viene trasmesso in modalità batch dall'applicazione A come input per l'applicazione B per l'elaborazione. Le applicazioni A e B sono considerate reciprocamente utenti funzionali una dell'altra.
c) Un processo batch che non richiede dati di input, ad esempio un processo batch che produce un report di fine mese e viene fisicamente innescato da un clock o da un comando di un operatore. Il processo batch può essere analizzato come se un utente funzionale umano provvedesse all'azione di innesco.

Nota: il segnale di un clock o un comando operatore che avvia l'elaborazione di un flusso batch è un comando di controllo e non viene mai misurato nel dominio del software applicativo aziendale. Questo comando lancia il flusso batch, non un singolo processo funzionale.

ESEMPIO 1 per Caso a) I dati di input per il 'job' può consistere tipicamente degli Entry per numerosi processi funzionali differenti. Ad esempio, il flusso di batch per un sistema di elaborazione notturno degli ordini potrebbe contenere processi funzionali per aggiungere nuovi clienti, aggiungere nuovi prodotti ed eliminare gli obsoleti, inserire nuovi ordini, cancellare ordini, ecc Ognuno di questi diversi processi funzionali dovrebbero essere analizzati 'end-to-end' e indipendentemente da qualsiasi altro processo funzionale nello stesso flusso. Ogni processo funzionale che viene eseguito in sequenza, una volta che il processo è avviato, viene innescato dai proprio input, e deve essere analizzato come se i dati fossero stati inseriti on-line da parte dell'utente funzionale umano.

ESEMPIO 2 per il Caso b) L'applicazione A, il software da misurare, è necessaria per la trasmissione di un file di dati persistenti all'applicazione B in modalità batch (le applicazioni A e B sono utenti funzionali una dell'altra). Il processo funzionale dell'applicazione A che trasmette i dati deve avere:

- Un Entry per avviare il processo;
- Un Read ed un Exit per ogni oggetto di interesse la cui persistenza dei dati deve essere trasmessa all'esterno.

Nota:. Questo processo funzionale ha gli stessi movimenti di dati indipendentemente dal fatto che la trasmissione dei dati da A verso B avviene in modalità batch o se i record di dati vengono inviati individualmente).

ESEMPIO 3 per il Caso c) Supponiamo che un processo batch che non richiede alcun dato di input. Un esempio potrebbe essere un 'job' batch per produrre un insieme standard di report, nessuno dei quali necessita di alcun input esterno. Il misuratore deve prima decidere se il 'job' consiste di uno o più processi funzionali, notando che ogni processo funzionale in un flusso batch deve avere una propria voce di attivazione. Ad esempio, un criterio per distinguere processi funzionali separati potrebbe essere che diversi report o gruppi di report siano prodotti per diverse tipologie di utenti funzionali o sono richiesti con frequenze diverse, ad esempio, rapporti settimanali contro rapporti mensili nello stesso flusso. Ci deve essere una ragione di business perché un flusso batch sia diviso in più di un processo funzionale. Ogni report o un insieme di reports la cui produzione viene considerata su processi funzionali separati devono avere un Entry e tanti Read e Exit come necessario per la produzione ed emissione dei report. L'Entry può non trasmettere dati attraverso il confine, ma trasmette il segnale di 'avvio dello specifico processo funzionale' e può comportare l'inizializzazione per la manipolazione dei dati.

Altri esempi che possono aiutare all'analisi dei processi batch sono i seguenti;

ESEMPIO 4 Spesso, un singolo report di sintesi viene prodotto a seguito dell'elaborazione del flusso batch. Normalmente sarà rilevata la presenza di almeno un (tipo di) Exit per ogni (tipo di) processo funzionale nel flusso. Ad esempio, se il report mostra, per uno specifico processo funzionale, il conteggio del numero di volte che è stato eseguito, più una lista di messaggi di errore in relazione ad ogni non riuscita del processo funzionale, si identificano due Exit per il processo funzionale. Quindi ripetere la misurazione per ogni processo funzionale, la cui sintesi dei dati può essere visualizzata sul report in modo da sommare il numero totale degli Exit. In alternativa al singolo report di sintesi, ogni processo funzionale potrebbe emettere il suo proprio report. In generale questo non dovrebbe influenzare la grandezza misurata.

ESEMPIO 5 in modalità batch, un unico processo funzionale di aggiornamento normalmente richiede la lettura di record esistenti su un unico oggetto di interesse prima di aggiornarlo e registrarlo (Write) aggiornato. (Ciò è in contrasto con l'elaborazione on-line, dove di solito un processo funzionale di aggiornamento è preceduto da un separato processo funzionale di interrogazione.) Inoltre, in modalità batch, normalmente un processo funzionale di cancellazione necessita solo di un Write per eliminare il record.

ESEMPIO 6 Sub-processi o "passi" di programma che possono essere necessari nel mezzo di un flusso di elaborazione batch, come l'ordinamento, o un punto di controllo/re-start 'salva' (che sono funzioni di 'controllo' che implementano requisiti tecnici di roll-back) devono essere ignorati nel dominio del software applicativo aziendale.

ESEMPIO 7 Per l'applicazione da misurare supponiamo di avere un requisito aziendale che riguarda l'importazione di alcuni dati dei degli impiegati tramite un file di interfaccia da un'altra applicazione con un flusso batch. Supponiamo che successivamente, il direttore di produzione aggiunge un requisito generico di utilità di spostare qualsiasi versione particolare di qualsiasi tipo di file per garantire che non sia elaborato due volte. Come risultato, un file di intestazione standard è aggiunto a ogni file di interfaccia per nell'organizzazione. L'intestazione del file contiene i dati relativi al file (tipo di file, l'ID del file, data di elaborazione, il numero di record, i totali hash di campi specifici, ecc.). Questi dati descrivono un oggetto di interesse per il direttore di produzione chiamato 'file di interfaccia'.

In questa situazione, ogni volta che l'applicazione di importazione deve elaborare un file di interfaccia fisico, due tipi di processi funzionali sono coinvolti, in particolare:

- Il processo funzionale di utilità generale che elabora il file di intestazione standard e controlla che il file non sia stato già elaborato prima di passarlo all'applicazione di importazione.
- Il processo funzionale che è specifico per l'applicazione di importazione e al tipo di file in elaborazione; in questo esempio l'archivio degli impiegati contiene dati che descrivono un oggetto di interesse per gli utenti aziendali ('Impiegato').

Questi due processi funzionali potrebbero avere i seguenti movimenti di dati, rispettivamente, ad esempio:

Per l'elaborazione dell'intestazione:

E file di dati di interfaccia
R file di dati di interfaccia storico (file già elaborati?)
W file di dati di interfaccia storico (memorizza il risultato dell'elaborazione)
X Messaggi

Per l'elaborazione dei dati degli impiegati:

E dati degli impiegati
W dati degli impiegati (dati di un impiegato esistente che vengono sovrascritti)
X Messaggi

Nota: Quando si dimensiona un'applicazione che richiede dati da inserire tramite uno a più file batch di interfaccia, il tutto utilizzando l'utility:

- il processo funzionale di utilità deve essere contato una sola volta per l'applicazione
- ogni processo funzionale che inserisce e scrive uno specifico di tipo di file dovrebbe essere contato, ovvero, tanti processi funzionali quanti tipi di file di interfaccia da inserire nell'applicazione (assumendo che la convalida e il trattamento è diverso per ogni tipo-file).

4.4.4 Fonti, destinazioni e formati multipli di un movimento dati – applicazioni della regola di 'unicità'

Il Manuale di Misurazione afferma[11] che, eccezionalmente, diverse tipologie di gruppi di dati che descrivono un determinato oggetto di interesse possono essere richiesti (nei FUR) di essere spostati in un movimento di dati dello stesso tipo nello stesso processo funzionale. In alternativa, e di nuovo in via eccezionale, il gruppo di dati può essere richiesto di essere spostato nella stessa tipologia di movimento di dati per lo stesso processo funzionale, ma con diverse manipolazione dei dati associati.

Come conseguenza di questa regola, i requisiti per diversi formati o manipolazioni diverse di un determinato gruppo di dati quando vengono inviati a più destinazioni o ricevuti da fonti multipli devono essere trattati come diversi movimenti di dati. Non è la molteplicità delle destinazioni o le sorgenti, in quanto tali, che determinano se si hanno diversi movimenti di dati, ma la molteplicità di formati o manipolazioni richieste (in genere da parte di diversi utenti funzionali) nelle FUR. Per esempio, se lo stesso Exit è inviato a due dispositivi fisici o verso due destinazioni, solo un Exit è identificato. Ma se gli Exit verso i due dispositivi o destinazioni sono non banalmente diversi (cioè, comportano ulteriore analisi e progettazione), vengono individuati due Exit.

Esempio 1

Un gruppo di dati è visualizzato come output sullo schermo e, stampato nello stesso formato. Viene identificato un Exit.

Esempio 2

Come per l'esempio 1. L'output contiene i medesimi attributi mostrati a video, però il layout stampato è significativamente diverso. Vengono identificati due Exit. Un esempio potrebbe essere il caso un gruppo di dati mostrati sullo schermo in modalità grafica ma stampati in modalità numerica. La manipolazione dei dati e la formattazione differiscono nelle due presentazioni del gruppo di dati, pertanto si misurano 2 CFP.

[11] Si veda la regola di 'Unicità dei movimenti di dati (e possibili eccezioni)' nel Manuale di Misurazione per la versione 3.0 del metodo COSMIC, precedentemente nota come la regola di 'De-duplicazione dei dati' nella versione 2.2.

Esempio 3

Un file (ovvero, uno o più gruppi di dati di origine) deve essere distribuito da un applicativo aziendale verso una o più destinazioni (= utente funzionale) e le FUR dell'applicazione specifica delle differenze nell'elaborazione dei raggruppamenti di dati in uscita (ovvero, che si traducono in diverse manipolazioni dei dati per gli Exit) verso utenti funzionali distinti: un Exit per oggetto di interesse viene identificato per ogni utente funzionale per il quale una diversa elaborazione è richiesta.

Esempio 4

Un insieme di FUR stabiliscono: "i dati del cliente sono mostrati dopo aver inserito il nome del cliente. Se vi è un solo cliente con il nome indicato, i dettagli del cliente vengono mostrati immediatamente. Se ci sono più clienti con lo stesso nome, viene visualizzata una lista dei clienti con il nome specificato, più altri dati sufficienti (es. l'indirizzo) per distinguerli. Il cliente corretto può in seguito essere selezionato con i rispettivi dati di dettaglio visualizzati."

Soluzione: sono identificati due processi funzionali. Il primo processo funzionale mostra i dati di dettaglio del cliente che corrispondono al nome indicato o in alternativa la lista dei clienti contenenti il nome, più i dati distintivi (es. il loro indirizzo). Il secondo processo funzionale è necessario per selezionare dalla lista ed inserire l'id-cliente nel caso in cui ci sono più clienti con il nome inserito, come segue:

Processo funzionale 1, mostra i dettagli e/o la lista:

E nome del cliente
R dati del cliente
X dati del cliente (uno è stato trovato)
X lista dei dati del cliente (più di uno sono stati trovati)
X messaggi di errore/ conferma

Processo funzionale 2, per la selezione dalla lista + visualizzazione dei dettagli:

E id-cliente
R dati del cliente
X dati del cliente
X messaggi di errore/ conferma

Il processo funzionale 1 illustra la situazione eccezionale riferita nel Manuale di Misurazione dove due raggruppamenti di dati distinti ('dati cliente' e 'lista dati cliente') descrivono lo stesso oggetto di interesse ('cliente') che deve essere spostato con movimenti di dati dello stesso tipo (Exit), nello stesso processo funzionale.

4.4.5 Elementi dell'interfaccia utente grafica (GUI)

Per gli elementi di interfaccia utente grafica (GUI) che trasportano dati di controllo, si applicano le regole per i comandi di controllo (vedasi sezione 4.4.1), ovvero, vengono ignorati.

Per gli elementi GUI che trasportano dati relativi ad oggetti di interesse, si applicano le regole per l'identificazione dei processi funzionali e dei movimenti dei dati. Vedi in particolare la sezione 4.1.8 'elenchi a discesa derivanti da processi funzionali'.

4.4.6 Autorizzazioni, aiuti e funzionalità di log

Le autorizzazioni, gli aiuti e le funzionalità di log possono essere misurate se la loro funzionalità o le modifiche alla loro funzionalità sono esplicitamente descritte nelle FUR, le funzionalità sono nello strato applicativo e si è concordato che devono rientrare nell'ambito di misurazione dell'applicazione. Normalmente si ignorano le funzionalità di autorizzazione, aiuto o log dall'ambito della misurazione di un'applicazione che le utilizza e che non sono specifiche dell'applicazione stessa.

Un tasto di Help fornito su ogni schermo e che offre lo stesso servizio dovunque esso viene premuto deve essere misurato come un unico processo funzionale per l'intera applicazione. (Si assume che il tipo di processo funzionale sia lo stesso per tutte le schermate dell'applicazione e che il risultato fornito sia dipendente dal contesto rappresentando semplicemente occorrenze diverse dello stesso tipo di output. Naturalmente la funzione Help, una volta invocata, può offrire altri processi funzionali per ulteriori interrogazioni di ricerca. In questo caso ognuno dei processi funzionali devono essere analizzati in base alle regole, assumendo che il sotto-sistema di Help sia nell'ambito di misurazione dell'applicazione aziendale.)

A seconda dei FUR, le funzionalità di help e di log possono essere richieste come processi funzionali dell'applicazione separati (in questo caso questi processi funzionali vengono misurati), oppure come parte di vari o tutti i processi funzionali dell'applicazione. Per esempio, la funzione di log può essere richiesta nei FUR come un processo funzionale distinto, oppure come parte integrante di qualsiasi altro processo funzionale. Nel secondo caso, si deve identificare un solo Write per ogni oggetto d'interesse per il quale sono salvati i dati di log.

4.4.7 Messaggi di errore e di conferma

Ad un'applicazione aziendale solitamente viene richiesto di rilasciare molti tipi di messaggi di errore per i suoi utenti funzionali (persone). Tutti i messaggi di errore sono generati all'interno del software applicativo. Alcuni messaggi scaturiscono in seguito ad errori di imputazione umani. Altri derivano dall'interpretazione di messaggi provenienti da software in altri strati o altre porzioni alla pari, ad esempio, un codice di ritorno che indica 'il cliente non esiste'. Tutti i messaggi di errore applicativi sono considerati come occorrenze diverse dei movimenti di dati chiamati 'messaggio di errore'. Quindi, si individua un unico Exit per tutti i messaggi di errore applicativi su ogni processo funzionale. (Vedi anche la regola per gli Exit nel Manuale di Misurazione).

La ragione di questa regola è che il messaggio di errore deriva da un requisito utente per qualche output alternativo (un Exit) quando un processo funzionale fallisce nella produzione dell'output normalmente previsto. I messaggi di errore derivano da un movimento di dati che è fallito (es.: 'cliente non presente', o 'codice non valido') e quindi è richiesto unicamente il conteggio di un CFP addizionale per il messaggio di errore per ogni processo funzionale. Il messaggio di errore scaturisce dal movimento di dati che è fallito. Quando, ad esempio, il messaggio di errore deriva dal fallimento di un Read, non sono necessari ulteriori Read per la generazione del messaggio di errore.

Seguendo lo stesso ragionamento, qualsiasi messaggio che confermi che un movimento di dati è stato elaborato con successo, ad esempio, 'Aggiornamento OK', dovrebbe essere trattato allo stesso modo come se fosse emerso un messaggio di errore, cioè che esso deve essere considerato come messaggio di 'errore / messaggio di conferma' conteggiato dal singolo Exit.

Ignorare i messaggi di errore derivanti da software non-applicativo, ad esempio 'il server non risponde'.

Una funzione che fornisce la possibilità di ri-provare a seguito di una condizione di errore deve essere considerata come un comando di controllo e ignorata. Ma movimenti di dati aggiuntivi derivanti da una condizione di errore in un processo funzionale, ad esempio, per un percorso di elaborazione alternativo, devono naturalmente essere misurati. Ad esempio: in un processo funzionale per l'inserimento di un ordine, la produzione automatica di una lettera quando l'ordine non viene accettato a causa di un errore dovuto all'esito negativo da un controllo sul credito. Si identifica 1 Exit.

4.5 Misurazione della dimensione delle modifiche funzionali al software

Si assume che il lettore abbia familiarità con le sezioni '*Identificare gli oggetti di interesse e i gruppi di dati*', '*Identificare gli attributi dei dati*' e '*Dimensione del software funzionalmente modificato*' del Manuale di Misurazione [3]. L'approccio per il dimensionamento delle modifiche di un processo funzionale modifiche è illustrata da due esempi.

4.5.1 Esempi di modifica funzionale di un processo funzionale

Esempio 1

L'oggetto di interesse 'Impiegato' contiene l'attributo 'num. di dipendenti'. Si è deciso di archiviare ulteriori dati sui dipendenti. Di conseguenza, l'oggetto di interesse 'Dipendente' viene aggiunto e collegato all'Impiegato, i dati sui singoli dipendenti devono ora essere inclusi nell'inserimento 'crea impiegato', e l'attributo 'num. di dipendenti' viene rimosso dall'oggetto di interesse Impiegato.

Situazione precedente: ci sono dati persistenti su un oggetto di interesse

Impiegato (id-Imp, …, num. di dipendenti, …)

Il processo funzionale 'crea impiegato' è (ignorando il dettaglio dei Read a fini della convalida che sarebbe necessaria in pratica):

E dati Impiegato
W dati Impiegato
X Messaggi di errore/conferma

Dimensione totale: 3 CFP

Nuova situazione: c'è la persistenza dei dati per i due oggetti di interesse

Impiegato (id-Imp, …) ('num. di dipendenti' eliminato)
Dipendente (id-Dip, id-Imp, …)

Il processo funzionale 'crea impiegato' sarà, a seguito dei cambiamenti:

E	dati Impiegato	movimento di dati modificati (attributo rimosso)
E	dati Dipendente	movimento di dati aggiunto
W	dati Impiegato	movimento di dati modificato (attributo rimosso)
W	dati Dipendente	movimento di dati aggiunto
X	Messaggi di errore/conferma	(non variato)[12]

Pertanto la dimensione del cambiamento funzionale al processo funzionale 'crea impiegato' è di 2 Entry + 2 Write = 4 CFP. Probabilmente più processi funzionali subiscono un impatto per la modifica o l'aggiunta dei gruppi di dati e anche le modifiche a questi processi funzionali devono essere misurate.

Esempio 2

Un estratto conto indica gli interessi da pagare ogni mese sui saldi positivi. L'algoritmo per calcolare l'interesse deve essere modificato, anche se non vi è un cambiamento funzionale nei dati necessari come input per il calcolo degli interessi. L'estratto conto è invariato nel contenuto e layout, ma la manipolazione dei dati associati all'attributo è modificata. Il cambiamento funzionale viene misurato come 1 CFP per l'Exit modificato che mostra l'interesse mensile.

[12] Ci possono essere occorrenze di messaggi di errore/conferma nuove o modificate, senza cambiamenti nel movimento di dati

4.5.2 Software di conversione dati

Quando una nuova applicazione sostituisce un vecchio sistema o un sistema manuale, spesso si rende necessario lo sviluppo di software per la conversione dei dati dal formato o dalla tecnologia utilizzata nella vecchia applicazione a quella nuova, o da recuperare dal sistema manuale. Questo software 'di conversione dati' in genere viene utilizzato solo una volta. Poiché tale software risponde alle necessità degli utenti funzionali può essere considerato come un software applicativo aziendale e i suoi FUR possono essere misurati.

Se includerlo o meno nell'ambito di una particolare misurazione dipende dallo scopo del dimensionamento. Se lo scopo è quello di misurare il rendimento del lavoro di un team di progetto la dimensione di qualsiasi software di conversione dati dovrebbe essere inclusa nell'ambito dell'applicazione. Se lo scopo è quello di misurare la dimensione dell'applicazione rilasciata, la conversione dei dati del software sarebbe esclusa dall'ambito della misurazione.

4.5.3 Misura del software funzionalmente modificato

In seguito a una modifica funzionale di software, la sua nuova dimensione totale è pari alla dimensione originale, più la dimensione funzionale di tutti i movimenti dati aggiunti, meno la dimensione funzionale di tutti i movimenti dati rimossi. I movimenti di dati modificati non hanno alcuna influenza sulle dimensioni della porzione di software poiché questi esistono sia prima che dopo le modifiche che sono state fatte.

Nell'esempio 1 del paragrafo 4.5.1 di cui sopra, la variazione netta in termini dimensionali, a seguito della modifica funzionale, per il processo funzionale 'crea dipendente' è un aumento del 2 CFP (5 - 3 = 2).

Nell'esempio 2 del paragrafo 4.5.1 di cui sopra, la variazione netta in termini di dimensionali, a seguito del cambiamento funzionale, del processo che fornisce l'interesse mensile è 0 CFP (non risultano movimenti dei dati aggiunti o eliminati, solo la modifica di un movimento di dati).

RIFERIMENTI

[1] Metodo COSMIC 3.0: 'Panoramica della Documentazione e Glossario', 2007.

[2] Metodo COSMIC 3.0: 'Introduzione al Metodo', 2007.

[3] Metodo COSMIC 3.0: 'Manuale di Misurazione', 2007, Guida Implementativa COSMIC a ISO/IEC 19761:2003, Measurement Practices Committee COSMIC.

[4] Metodo COSMIC 3.0: 'Advanced and Related Topics', 2007.

[5] ISO/IEC 19761:2003, Software Engineering – COSMIC – A functional size measurement method.

[6] ISO/IEC TR 14143/5 'Information technology – Software measurement – Functional size measurement – Determination of Functional Domains for use with functional size measurement'.

[7] The entity-relationship model – toward a unified view of data, Peter Chen, ACM Transactions on Database Systems, Vol. 1, No. 1, March 1976.

[8] An Introduction to Database Systems, Date, C.J., Addison-Wesley, 1990.

[9] Data Models, Tsichritzis, D.C. and Lochovsky, F.H., Prentice-Hall, 1982.

[10] Unified Modeling Language Specification, March 2003, Version 1.5 (Version 1.4.2 of UML formal/05-04-01, published by the Object Management Group, which has been accepted as an ISO standard ISO/IEC 19501).

[11] Function Point: Manuale delle Regole di Conteggio 4.2, Part 4 'Appendices and Glossary', International Function Point User Group, 2004.

[12] Software Engineering, Sommerville, I., Addison-Wesley, 1996.

[13] ISO/IEC 14143/1:2007 'Information technology – Software measurement – Functional size measurement – Definition of concepts'.

APPENDICE A - PROCEDURA PER RICHIESTE DI MODIFICA E COMMENTI

Il COSMIC Measurement Practices Committee (MPC) è lieto di ricevere feedback, commenti e, se necessario, richieste di modifica (Change Request) riguardanti la documentazione COSMIC. Questa appendice illustra le modalità per comunicare con il COSMIC MPC.

Tutte le comunicazioni dirette al COSMIC MPC vanno inviate per e-mail al seguente indirizzo:

mpc-chair@cosmicon.com

Feedback e commenti generici informali

Commenti informali e feedback sulla documentazione COSMIC, come per es. difficoltà nella comprensione o nell'applicazione del metodo COSMIC, suggerimenti di miglioramento generale, ecc. dovrebbero essere inviati per e-mail all'indirizzo sopra riportato.

Si prenderà atto dei messaggi e sarà data di norma notifica entro due settimane dalla ricezione. L'MPC non può garantire di mettere necessariamente in pratica tali commenti generici.

Richieste di Modifica formali

Qualora il lettore della documentazione COSMIC ritenga che vi sia un errore nel testo o la necessità di un chiarimento, o che una porzione del testo richieda di essere migliorata, è possibile sottoporre una Richiesta di Modifica (Change Request, CR) formale.
Si prenderà atto delle CR formali e sarà data notifica entro due settimane dalla ricezione. A ogni CR sarà quindi assegnato un numero progressivo e essa sarà fatta circolare tra i membri del COSMIC MPC, un gruppo di esperti a livello mondiale del metodo COSMIC. Il normale ciclo di revisione richiede un minimo di un mese e può richiedere più tempo se la CR si rivela di difficile risoluzione.

L'esito dell'esame della CR può essere "accettata", "rifiutata" o "in attesa di ulteriore discussione" (nell'ultimo caso per es. se esiste una qualche relazione con un'altra CR) e sarà comunicato al mittente non appena possibile.

Una CR formale sarà accettata solo se corredata di tutte le seguenti informazioni.

- Nome, qualifica e organizzazione di appartenenza di chi sottopone la CR.
- Riferimenti di contatto di chi sottopone la CR.
- Data di invio.
- Scopo generale della CR (per es. "occorre migliorare il testo …").
- Testo esistente che richiede modifica, sostituzione o cancellazione (o chiaro riferimento ad esso).
- Testo aggiuntivo o sostitutivo proposto.
- Spiegazione esauriente del motivo per cui è necessaria la modifica.

Un modulo per sottoporre una CR è disponibile presso il sito web www.cosmicon.com.

La decisione del COSMIC MPC sull'esito dell'esame di una CR e, se accettata, sulla versione del Manuale di Misurazione dalla quale si applicherà la CR è definitiva.

Quesiti sull'applicazione del metodo COSMIC

Il COSMIC MPC si rammarica di non poter rispondere a quesiti relativi all'utilizzo o all'applicazione del metodo COSMIC. Esistono organizzazioni commerciali che possono fornire formazione e consulenza o strumenti di supporto sul metodo. Si prega di consultare il sito web www.cosmicon.com per ulteriori dettagli.

APPENDICE B - PRINCIPALI MODIFICHE NELLA V1.1 DALLA V1.0 DELLA LINEA GUIDA PER MISURARE IL SOFTWARE APPLICATIVO AZIENDALE

Nota. La natura di una modifica è indicata da:

- 'Metodo', quando è stato cambiato il *contenuto* del metodo COSMIC, o da
- 'Editoriale', quando è stata cambiata la *descrizione* del metodo (ma non il metodo stesso)

Riferimenti nella versione 1.0 / 1.1	Natura della modifica	Commento
-	Editoriale	La guida è stata ristrutturata in modo che tutta la materia generale di base venisse affrontata per prima, seguita dal materiale nella sequenza delle fasi come proposte del Manuale di Misurazione.
-	Metodo	Laddove appropriato, sono stati rimossi i riferimenti a 'punto di vista' quale principio astratto e al punto di vista dell'utente finale della misurazione
-	Metodo	Laddove appropriato 'utente finale' è stato sostituito dal termine 'utente funzionale'.
-	Editoriale	I Riferimenti ai Bollettini di Aggiornamento del Metodo (MUB – Method Update Bulletins) 1 e 2 sono stati eliminati e sono stati incorporati nei documenti del Metodo COSMIC v3.0
Prefazione	Editoriale	I dettagli che qualificano il 'dominio software delle applicazioni aziendali' sono stati spostati dalla Prefazione all'inizio della prima sezione del capitolo 2
4.1 / 1.2	Editoriale	La definizione di 'Requisiti Utente Funzionali' è stata aggiornata e allineata a quella proposta nello standard ISO/IEC 14143/1:2003 [13]
5.4.5 / 4.1.5	Metodo	La sezione 4.1.5 'Processi funzionali separati' é stata rinominata come 'Processi funzionali separati a seguito di decisioni distinte dell'utente funzionale'
5.4.6 / 4.1.6	Editoriale	La sezione 'Processi funzionali e relativi dati' è stata rinominata come 'Ricerca ed aggiornamento dei dati in un singolo processo funzionale'. Il precedente titolo non chiarisce quale sia il problema, né che si tratta di una prosecuzione della sezione precedente
5.6.3 / 4.4.3	Editoriale	Il testo sulle elaborazioni batch è stato allineato con il Manuale di Misurazione. I casi e gli esempi sono ora più comprensibili
5.6.4 / 4.4.4	Editoriale	Il testo sulle fonti multiple ecc. è stato esteso, allineato con il testo sulla 'regola di unicità dei dati' nel Manuale di Misurazione' ed è stato aggiunto un esempio
5.7 / 4.5	Editoriale	La definizione di una 'modifica ', i criteri per distinguere una modifica a uno spostamento dei dati e le regole per individuare e misurare le modifiche a un movimento di dati sono stati tutti rimossi in quanto descritti in dettaglio nel Manuale di Misurazione v3.0

GLOSSARIO

Il presente glossario contiene solo i termini e le abbreviazioni che sono definite in questa Guida e che sono specifiche del dominio software delle applicazioni aziendali. Per ulteriori termini e definizioni del metodo COSMIC, occorre riferirsi a 'Metodo COSMIC 3.0. Panoramica della Documentazione e Glossario' (Riferimenti, [1]).

E/RA

Abbreviazione per 'Entity Relationship Analysis' (Analisi Entità Relazione) – v. sez. 2.3 di questa Linea Guida.

Tipo di entità

Qualsiasi oggetto fisico o concettuale nel mondo reale per il quale si richiede al software di elaborare e/o memorizzare dati.

RDA

Abbreviazione di 'Relational Data Analysis' – v. sez. 2.5 della presente Linea Guida.

Relazione

Un qualunque insieme di attributi di dati

Nota: Una 'relazione in Terza Forma Normale' è un insieme di attributi di un singolo oggetto di interesse, cioè si tratta di un sinonimo di un 'gruppo di dati' COSMIC.

SOR

Abbreviazione per 'Statement of Requirements' (affermazione dei requisiti).

Affermazione dei requisiti

Un documento contenente tutti i requisiti di una porzione di software, ovvero Requisiti Utente Funzionali e Requisiti Non-Funzionali (cioè, requisiti tecnici e requisiti di qualità).

UML

Abbreviazione di 'Unified Modeling Language' – v. sez. 2.4 della presente Linea Guida.