

Visual Studio Team System: coniugare metodi agili e certificazione di qualità

Come costruire un processo per il ciclo di vita del software basato su metodi agili e conforme alle norme Vision 2000 utilizzando il sistema di gestione dei team integrato nel nuovo Visual Studio 2005

di **Francesco Arrigo**

Allo stato attuale esistono parecchie aziende che, per vari motivi, possiedono processi certificati ISO9001: 2000 (Vision 2000) per l'analisi, progettazione e sviluppo software e parecchie altre che, adottando i metodi agili (extreme programming, SCRUM, ecc.), dicono di essersi liberate dall'oppressione della documentazione a tutti i costi a vantaggio della velocità, produttività e soprattutto dell'abbattimento dei costi. Non sono tante, invece, quelle che riescono ad adottare metodologie di sviluppo agili, che nello stesso tempo fanno parte integrante di un sistema conforme alle norme Vision 2000. In questo articolo vedremo come raggiungere questo obiettivo in maniera tale che si adatti naturalmente alle esigenze dei componenti di

un team di progetto software. Si vedrà inoltre come gli stessi componenti possono applicare in maniera trasparente il Microsoft Solution Framework nella versione 4.0 una volta definita una metodologia basata sui paradigmi di programmazione agili utilizzando Visual Studio Team System.

Vision 2000 e metodi agili: convergenza o contraddizione?

Quando si parla della certificazione ISO 9001 si pensa subito che questa serva per poter dimostrare pubblicamente che l'azienda segue un processo documentato, indipendentemente dal fatto che questo, poi, risulti burocraticamente

Francesco Arrigo è laureato in Scienze dell'Informazione presso l'Università di Catania. È team leader del software Eraclito® per la progettazione e simulazione di reti di fluidi sviluppato in C++ e prodotto dalla Proteo S.p.A. di Catania. È responsabile della qualità nel software nella stessa azienda certificata ISO9001:2000 dal 1999.

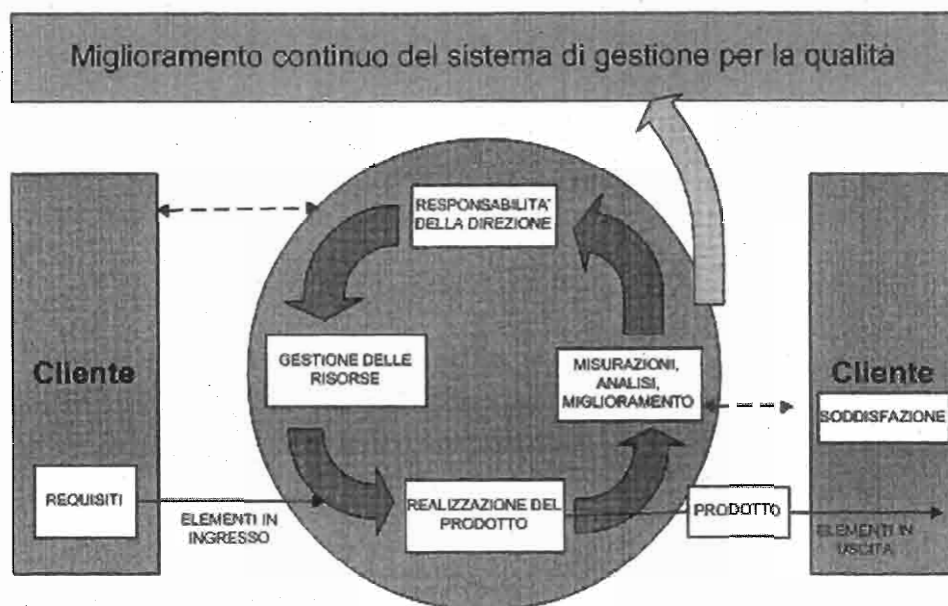


Figura 1 Modello di Sistema di gestione per la qualità basato sui processi

insostenibile. D'altra parte all'interno delle comunità sostenitrici dei metodi agili c'è chi dice che la quantità di documentazione richiesta per ottenere la certificazione contraddice i principi fondamentali, perchè un processo documentato è inerentemente non agile. Spesso, però non si comprende che essa riguarda l'intera azienda e non solo il processo di sviluppo software, anche nel caso in cui questo rappresenti il suo core-business. Tali opinioni nascono soprattutto dal fatto che le prime versioni dello standard ISO davano risalto alla documentazione delle procedure piuttosto che alla qualità delle stesse. Inoltre erano piuttosto generali in modo da poter essere utilizzate nell'ambito di un processo produttivo qualunque. Le cose però sono cambiate da quando lo standard corrente è basato sulla qualità all'interno di un processo, infatti le norme ISO 9001:2000 – dette anche Vision 2000 – sono intensivamente orientate ai processi e richiedono una organizzazione che li identifica, gestisce e migliora continuamente, come è illustrato nel modello proposto in **Figura 1**. Successi-

vamente sono state introdotte anche delle linee guida ISO IEC 90003 2004, pubblicate ufficialmente il 15 febbraio del 2004, che aiutano ad applicare i requisiti Vision 2000 alla progettazione e sviluppo del software.

Le norme ISO 9001:2000 dette anche Vision 2000 sono intensivamente orientate ai processi e richiedono una organizzazione che li identifica, gestisce e migliora continuamente

La principale novità introdotta è un più rilevante coinvolgimento del top management aziendale nello stabilire la politica della qualità e gli obiettivi per assicurare che le necessità del cliente siano sod-

disfatte. Il miglioramento continuo, secondo i sostenitori dei metodi agili, può rappresentare ancora un ostacolo perché aggiunge "overhead" all'agilità del processo di sviluppo. Il punto chiave, però, non è il processo ma l'abilità di dimostrare che esso può raggiungere gli obiettivi delle Vision 2000 e cioè gestire e migliorare la qualità. Non c'è nulla di esplicito o implicito nei metodi agili che precluda tutto questo. Anzi si può dire che le Vision 2000 forniscono chiarezza e stabilità ai metodi agili e dimostreremo più avanti come la loro agilità fornisca naturalmente e senza appesantimenti gli elementi necessari per il miglioramento del processo.

Modelli di ciclo di vita del software: stato dell'arte ed applicabilità in contesti concreti

Sebbene le Vision 2000 non definiscano un particolare modello di ciclo di vita, esse sembrano fare riferimento in qualche modo al vecchio processo di sviluppo a cascata del software (V-model). Questo perché la classificazione delle norme ci porta a pensare ad un'esecuzione sequenziale delle fasi di analisi dei requisiti, design, implementazione, testing e manutenzione. La storia dei progetti software ci dimostra però che la sequenzialità è un'utopia, in quanto si basa sulla stabilità dei requisiti. Il continuo cambiamento di questi ultimi ha consentito l'affermazione dei modelli evolutivi che hanno aggiunto al modello a cascata una fase di prototipazione in grado di ottenere migliori requisiti.

Da qui la nascita dei metodi iterativi e quindi agili, che si sono rivelati i più vicini al processo di sviluppo reale e tra di essi possiamo distinguere diversi gradi di agilità, che vanno da DSDM (Dynamic System Development Method) ad XP (Extreme Programming). L'eccessiva agilità può condurre, però, anche alla perdita del controllo del progetto, soprattutto nell'applicazione di una metodologia come XP, in cui compiti e ruoli si scambiano all'interno del team di sviluppo.

Bisogna trovare, quindi, un equilibrio tra produttività e gestione del processo ed in questo compito ci aiuterà il MSF (Microsoft Solution Framework) che discuteremo nel prossimo paragrafo.

Il nucleo di ogni processo di sviluppo agile è il modo in cui si partiziona e si pianifica il lavoro

MSF (Microsoft Solution Framework)

Il Microsoft Solution Framework (MSF) è un template di metodologie (metamodello) che consente alle organizzazioni di definire una metodologia propria, a partire da un insieme di linee guida di base, per la gestione dell'intero ciclo di vita del software. MSF, giunto alla versione 4.0, rappresenta un cambiamento rispetto a quello che erano state le strategie Microsoft del passato. Il colosso di Redmond non era certo stato un riferimento per le metodologie di sviluppo software poiché la precedente versione del MSF era legata alla sequenzialità delle fasi. La filosofia di MSF è fornire linee guida per il processo di sviluppo che siano produttive, integrate negli strumenti ed estendibili. La mossa vincente, però, è stata quella di trasformare un modello in un metamodello e soprattutto nel fornire un'applicazione che consenta di utilizzarlo, cosa essenziale e non banale per evitare che le linee guida teoriche rischiarono di rimanere tali. Nei prossimi paragrafi vedremo i Team Foundation Services di Visual Studio Team System. Per ora diciamo che Microsoft fornisce due metodologie di default insieme al framework e cioè "MSF for Agile Software Development" e

“MSF for CMMI Process Improvement”. La prima è destinata a team di sviluppo da tre a venti persone, mentre la seconda è più rigida ed è pensata per team più numerosi e fornisce un modello per migliorare continuamente un processo (CMMI infatti sta per Capability Maturity Model Integration). Comunque anche se le due metodologie hanno in comune l’approccio agile, prenderemo in considerazione “MSF for Agile Development” perché è la metodologia che più si addice alla maggior parte dei team di sviluppo. Il nucleo di ogni processo di sviluppo agile è il modo in cui si partiziona e si pianifica il lavoro. Inizialmente, un progetto è suddiviso in periodi fissati nei quali si svolge lo sviluppo. Questi periodi fissati sono le iterazioni. La lunghezza di una iterazione è solitamente fissata tra le due e le otto settimane sebbene possa capitare che, per piccoli progetti, le iterazioni possano essere lunghe giorni o addirittura ore. La prima iterazione (iterazione zero) è quella in cui sono fissati gli scenari ed i requisiti di qualità del servizio (per esempio le performance, la piattaforma richiesta, i requisiti di sicurezza). Inoltre la prima iterazione è quella in cui gli sviluppatori forniscono una stima di massima degli scenari e dove vengono stabilite le priorità. All’iterazione successiva gli scenari vengono dettagliati maggiormente e tarati sulla velocità dell’iterazione precedente, mentre gli sviluppatori dividono gli scenari in task e forniscono stime più dettagliate per ciascun task. L’aspetto che riguarda la stima è molto importante: l’errore nella stima del software è sempre stato uno dei problemi critici che le metodologie agili risolvono con brevi iterazioni e raffinamenti successivi dei requisiti. Oltre agli sviluppatori, anche gli architetti ed i tester creano task all’interno di una iterazione ed alla fine di essa c’è sempre una fase di integrazione (*continuous integration*). Comunque, i tradizionali ruoli nel ciclo di vita del software, all’interno di un processo agile

MSF, sono racchiusi in poche persone che hanno uguale importanza. Grande importanza ha il cliente ed il suo rapporto con il team di sviluppo. I metodi agili dicono che il cliente deve lavorare fianco a fianco con il team di sviluppo, ma questa situazione ideale quasi mai può avvenire. Microsoft all’interno dei propri team ha introdotto il concetto di “persona” che svolge il ruolo di cliente interno e cioè la persona che rappresenta un gruppo di clienti “reali”. Inoltre i metodi agili possono essere applicati ai grandi progetti nel momento in cui anche gli architetti forniscono un design per componenti del progetto che viene via via raffinato. Si parla infatti di “shadow” design che rimane disallineato con la codifica durante l’iterazione per poi riallinearsi alla fine. Infine, all’interno di una iterazione, MSF richiede le unità di test (unit test) come parte delle attività di sviluppo.

Microsoft all’interno dei propri team ha introdotto il concetto di “persona” che svolge il ruolo di cliente interno e cioè la persona che rappresenta un gruppo di clienti “reali”

Architettura di Visual Studio Team Foundation

Per poter applicare concretamente MSF dobbiamo compiere il primo passo e cioè l’installazione di Visual Studio 2005 sulle macchine client degli utenti del team di sviluppo e l’installazione di Team Foundation Server su un server Windows 2003

Listato 1 Esempio di ProcessTemplate.xml

```

<?xml version="1.0" encoding="utf-8"?>
<ProcessTemplate>
  <metadata>
    <name>Agile Software Development with Scrum</name>
    <description>Scrum is an agile, lightweight process
      that can be used to manage and control software
      and product development using iterative, incremental
      practices. Wrapping existing engineering
      practices, including Extreme Programming and RUP,
      Scrum generates the benefits of agile development with the
      advantages of a simple implementation.
      Scrum significantly increases productivity and reduces time
      to benefits while
      facilitating adaptive, empirical systems development.
    </description>
  </metadata>
  <plugins>
    <plugin name="Microsoft.ProjectCreationWizard.
      Classification" wizardPage="false" />
    <plugin name="Microsoft.ProjectCreationWizard.
      WorkItemTracking" wizardPage="false" />
    <plugin name="Microsoft.ProjectCreationWizard.
      Reporting" wizardPage="false" />
    <plugin name="Microsoft.ProjectCreationWizard.Portal"
      wizardPage="true" />
    <plugin name="Microsoft.ProjectCreationWizard.Groups"
      wizardPage="false" />
    <plugin name="Microsoft.ProjectCreationWizard.Version
      Control" wizardPage="true" />
  </plugins>
  <metadata>
  </metadata>
  <groups>
    <group id="Classification" description="Structure
      definition for the project."
      completionMessage="Project Structure uploaded.">
      <dependencies>
      </dependencies>
      <taskList filename="Classification\Classification.xml" />
    </group>
    <group id="Groups" description="Create Groups and
      assign Permissions."
      completionMessage="Groups created and Permissions assigned.">
      <dependencies>
        <dependency groupId="Classification" />
      </dependencies>
      <taskList filename="Groups and Permissions\Groupsand
        Permissions.xml" />
    </group>
    <group id="WorkItemTracking" description="Workitem
      definitions uploading."
      completionMessage="Workitem definitions uploaded.">
      <dependencies>
        <dependency groupId="Classification" />
      </dependencies>
      <taskList filename="WorkItemTracking\WorkItems.xml" />
    </group>
    <group id="Portal" description="Creating project Site"
      completionMessage="Project site created.">
      <dependencies>
        <dependency groupId="Classification" />
        <dependency groupId="WorkItemTracking" />
        <dependency groupId="VersionControl" />
      </dependencies>
      <taskList filename="Wss\Wss.xml" />
    </group>
    <group id="Reporting" description="Project reports
      uploading."
      completionMessage="Project reports uploaded.">
      <dependencies>
        <dependency groupId="Classification" />
        <dependency groupId="Portal" />
      </dependencies>
      <taskList filename="Reports\Reports.xml" />
    </group>
    <group id="VersionControl" description="Creating
      Version control."
      completionMessage="Version control task completed.">
      <dependencies>
        <dependency groupId="Classification" />
        <dependency groupId="WorkItemTracking" />
      </dependencies>
      <taskList filename="VersionControl\VersionControl.xml" />
    </group>
  </groups>
</ProcessTemplate>

```

SP 1 (edizione Standard o Enterprise) con SQL Server 2005 SP1 e gli SharePoint Services 2.0 SP2 o superiore. Sorvoleremo, in questo articolo, sui dettagli di configurazione perchè questi richiederebbero un articolo dedicato. Diciamo soltanto che il server non deve essere un domain controller e può far parte sia di una architettura

client-server che di una a tre livelli in cui il server contiene solo i dati ed i Team Foundation Services (web service a cui i client richiedono i servizi disponibili) si trovano su server localizzati su aree geografiche diverse. Tipicamente, per piccoli team (da tre a venti persone), è sufficiente un'architettura client-server.

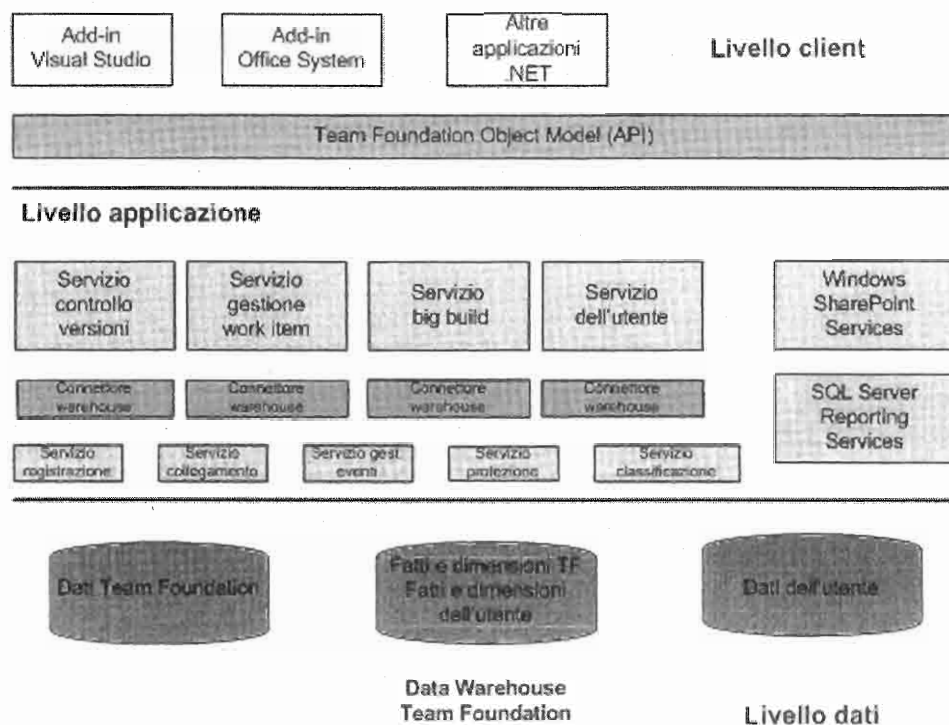


Figura 2 Architettura di Visual Studio Team Foundation

Sicuramente introdurre Visual Studio Team System all'interno di un'organizzazione non è cosa semplice e comporta l'assunzione di decisioni che influenzano l'amministrazione dell'intero sistema informativo aziendale.

Premesso ciò, supponiamo di aver terminato la fase di installazione e di voler creare un nuovo "team project" su Team Foundation Server. Abbiamo la seguente partizione logica del server:

- Database server SQL Server 2005 necessario per gestire e tracciare tutte le attività pianificate per il team di progetto. Tali attività sono chiamate "elementi di lavoro" o "work item" e su di esse è costruito un datawarehouse ibrido relazionale ed OLAP per metriche ed indicatori sul progetto e di processo. Tutti i dati non possono essere acceduti direttamente e tra di essi sono inclusi i dati per il controllo delle versioni,

quelli dei risultati dei test e della generazione del codice ("team build").

- Application server, in cui sono esposti i web service dei diversi strumenti e attraverso i quali è possibile accedere ai dati archiviati. Inoltre questo livello gestisce l'hosting del sito Windows SharePoint Services per il portale del progetto necessario per presentare documenti, fogli di calcolo e piani di attività favorendo in tal modo la comunicazione tra i membri del team. I servizi di report di SQL Server, infine, consentono in qualunque momento di avere a disposizione la situazione dettagliata del progetto.

Il livello client di Team Foundation, invece, utilizza Team Explorer, un add-in dell'ambiente di sviluppo di Visual Studio, per accedere al livello di applicazione, ma non solo. Esistono add-in per il pacchetto Office, in particolare

per Excel, Word e Project 2003. Essi permettono ai membri non tecnici del team di progetto di lavorare nella maniera in cui sono soliti fare, accedendo a tutte le funzionalità senza necessità di installare un ambiente IDE come Visual Studio che potrebbe rivelarsi piuttosto ostico da utilizzare. Gli sviluppatori che

intendono avvalersi delle caratteristiche di estensibilità di Team Foundation, possono accedere ai web service del livello di applicazione attraverso un modello ad oggetti ed un set di API che fungono da proxy avanzati come vedremo nel prossimo paragrafo. L'intera architettura è illustrata in **Figura 2**.

Listato 2 Codice C# per aggiungere un nuovo work item in TFS

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Collections;
using Microsoft.TeamFoundation.Client;
using Microsoft.TeamFoundation.WorkItemTracking.Client;

namespace Microsoft.Samples.TeamFoundation
{
    class Attachments
    {
        static void Main()
        {
            string server = GetServer();

            try
            {
                // Ottiene il work item store da TeamFoundationServer
                Console.WriteLine("Connessione a {0}...",
                    server);
                TeamFoundationServer tfs = TeamFoundationServer
                    Factory.GetServer(server);
                WorkItemStore store =
                    (WorkItemStore)tfs.GetService(typeof(WorkItemStore));

                // Crea un nuovo work item
                if (store.Projects.Count <= 0)
                {
                    Console.WriteLine("Non ci sono progetti
                        nel server");
                    return;
                }

                WorkItemTypeCollection workItemTypes = store
                    Projects[0].WorkItemTypes;
                if (workItemTypes.Count <= 0)
                {
                    Console.WriteLine("Non c'è nessun tipo di
                        work item in questo progetto");
                    return;
                }

                WorkItem workItem = new WorkItem(workItemTypes[0]);
                Console.WriteLine("Creato un nuovo work item di
                    tipo {0}", workItem.Type.Name);

                workItem.Title = "Creato per esempio";

                // Salva il workitem
                workItem.Save();
                Console.WriteLine("Creato work item: {0}",
                    workItem.Id);

                Console.WriteLine("Salvato work item.");
            }
            catch (Exception e)
            {
                Console.WriteLine("Errore: {0}", e.Message);
            }

            // Metodo di ausilio per ottenere il nome del server
            // di TFS server a cui si è connessi.
            private static string GetServer()
            {
                string server =
                    Environment.GetEnvironmentVariable
                        ("TEAM_FOUNDATION_SERVER_NAME");
                if (server != null)
                {
                    server = server.Trim();
                }

                while (server == null || server.Length == 0)
                {
                    Console.WriteLine("Inserisci il nome del Team
                        Foundation Server, \n oppure impostalo nella variabile di
                        ambiente TEAM_FOUNDATION_SERVER_NAME:");
                    server = Console.ReadLine();
                    server = server.Trim();
                }
                return server;
            }
        }
    }
}
```

ISO 9001:2000	Sviluppo agile con Visual Studio Team System
7.3 Progettazione e sviluppo	
<p>7.3.1 Pianificazione della progettazione e dello sviluppo L'organizzazione deve pianificare e tenere sotto controllo lo sviluppo del prodotto, stabilire le fasi, attività di riesame, verifica, validazione, ruoli e responsabilità per la progettazione. Inoltre deve assicurare comunicazioni efficaci e garantire che i risultati della pianificazione siano aggiornati con il progredire della progettazione e dello sviluppo.</p>	<p>I work item che danno evidenza della pianificazione sono definiti nella metodologia applicata dal team di sviluppo e memorizzata in TFS. In questa metodologia sono definite le fasi (iterazioni), stabilite le regole per le attività di riesame, verifica, validazione. Le group policy definiscono ruoli e responsabilità. Lo sviluppo del prodotto è tenuto sotto controllo mediante il sistema di controllo delle versioni di TFS. Le comunicazioni efficaci e aggiornate sono assicurate in ogni momento mediante il portale SharePoint.</p>
<p>7.3.2 Input di design e sviluppo Gli elementi in ingresso riguardanti i requisiti del prodotto devono essere definiti e conservati. Essi devono comprendere requisiti funzionali, prestazionali, cogenti applicabili, informazioni da progettazioni precedenti. Gli elementi in ingresso devono essere riesaminati, completi, non ambigui, non in conflitto tra loro.</p>	<p>I requisiti sono definiti mediante i work item e conservati in TFS. È possibile definire una tipologia di work item per ogni tipologia di requisito. Ogni work item può contenere ogni tipo di informazione aggiuntiva ed il riesame avviene stabilendo opportune politiche di check-in per TFS. Ogni work item è dotato di identificatore univoco e di numero di revisione.</p>
<p>7.3.3 Output di design e sviluppo Gli output di design e sviluppo devono essere forniti in una forma che consenta le verifiche secondo gli input e devono essere approvati prima del loro rilascio. Gli output di design e sviluppo devono soddisfare i requisiti in ingresso, fornire informazioni per l'approvvigionamento, la produzione ed erogazione di servizi, contenere o richiamare i criteri di accettazione per i prodotti e precisare le caratteristiche dei prodotti essenziali per una sicura e adeguata utilizzazione.</p>	<p>Gli output di design e sviluppo sono rappresentati dal codice eseguibile e da tutto quello che sta a corredo (help, documentazione, ecc.). TFS gestisce la cronologia dei work item oltre alle revisioni. I work item possono contenere riferimenti ad altri work item, al codice sorgente ed agli unit test necessari a completarli. Le politiche di check-in in TFS consentono di raggruppare in un unico blocco (changeset) un insieme di file associati e di stabilire il routing di approvazione. I report di SQL Server consentono di fornire tutte le informazioni sul progetto.</p>
<p>7.3.4 Riesame del design e dello sviluppo In fasi opportune devono essere effettuati riesami sistematici del design e dello sviluppo in accordo a quanto pianificato (vedi 7.3.1), al fine di valutare le capacità dei risultati del design e dello sviluppo di ottemperare ai requisiti, individuare tutti i problemi e proporre le azioni necessarie. A tale riesame devono partecipare rappresentanti delle funzioni coinvolte nel design e sviluppo oggetto del riesame. Risultati e azioni susseguenti devono essere conservati.</p>	<p>Al termine di ogni iterazione stabilita viene effettuata una fase di integrazione al fine di valutare il completamento dei work item. Ogni iterazione consente il raffinamento dei work item precedenti o l'aggiunta di nuovi di tipo bug legati ai precedenti. Ovviamente i responsabili dei work item sono coinvolti e tutte le azioni sono tracciate e mantenute in TFS.</p>
<p>7.3.5 Verifica del design e dello sviluppo Devono essere effettuate verifiche in accordo con quanto pianificato (7.3.1) per assicurare che i risultati di output del design e dello sviluppo siano compatibili con gli elementi di input. Le registrazioni dei risultati delle verifiche e delle eventuali azioni intraprese devono essere conservate.</p>	<p>Le verifiche del design e dello sviluppo sono effettuate mediante gli unit test legati al codice sorgente e quindi ad uno o più work item. Possono essere stabilite delle politiche che impediscono il check-in dell'intero changeset (codice-work item-test) che non abbia superato lo unit test. Tutte le attività vecchie e nuove sono memorizzate in TFS.</p>
<p>7.3.6 Validazione del design e dello sviluppo Deve essere effettuata la validazione in accordo con quanto pianificato (7.3.1) per assicurare che i risultati di output del design e dello sviluppo siano compatibili con gli elementi di input. Le registrazioni dei risultati della validazione e delle eventuali azioni intraprese devono essere conservate.</p>	<p>Come nel punto precedente ad un intero gruppo di work item possono essere associati test di regressione in modo da coprire l'intero prodotto software. Il superamento dei test, memorizzati in TFS, è condizione necessaria per il completamento dei work item.</p>
8.2 Monitoraggi e misurazioni	
<p>8.2.3 Monitoraggio e misurazione dei processi L'organizzazione deve adottare adeguati metodi per monitorare e misurare i processi in modo tale da dimostrare di ottenere i risultati pianificati.</p>	<p>Il datawarehouse di TFS consente di definire opportune misure per l'analisi del processo. La visualizzazione dei risultati è effettuata mediante i servizi di reporting di SQL Server 2005.</p>
<p>8.2.4 Monitoraggio e misurazione dei prodotti L'organizzazione deve monitorare e misurare le caratteristiche dei prodotti per verificare che i relativi requisiti siano stati soddisfatti. Deve essere documentata l'evidenza della conformità ai criteri di accettazione.</p>	<p>Il datawarehouse di TFS consente di definire opportune misure per l'analisi del prodotto. La visualizzazione dei risultati è effettuata mediante i servizi di reporting di SQL Server 2005.</p>

Tabella 1 Mappatura norme ISO 9001:2000 / Azioni in VSTS

Applicazione concreta di MSF Agile: adattare il metodo al team e non viceversa

Una volta definita l'architettura di Team Foundation abbiamo a disposizione un sistema integrato per applicare una metodologia agile nel modo più naturale possibile. Fino a questo momento sia per poter adottare una metodologia agile che per seguire un processo ISO9001:2000 era necessario muoversi tra diversi strumenti o documenti in varie forme con l'inevitabile conseguenza di compromettere sia l'agilità che il rispetto delle norme. In questo paragrafo illustreremo come l'utilizzo degli strumenti di estensibilità di Visual Studio Team System (disponibili scaricando ed installando il Visual Studio 2005 SDK di settembre 2006) ci consente comunque di muoverci tra diversi strumenti, ma solo per la volontà di adottare MSF Agile senza sconvolgere le abitudini del team. Allo stesso tempo, però, Team Foundation Server (TFS) li integra, fornendoci la possibilità di definire un metodo personalizzato o applicarne uno esistente, tracciare il workflow ed ottenere indicatori di processo per poter rispondere alle norme. Utilizziamo come punto di partenza "MSF for Agile Development" e proviamo a scaricare il metodo dall'IDE Visual Studio 2005. Per farlo è necessario connettersi al server di Team Foundation e, attraverso Team Explorer, utilizzare un wizard detto "Gestore dei template di processo" per avere a disposizione un insieme di file XML, i "Process Definition Files". Tali file possono essere modificati per personalizzare il metodo secondo le nostre esigenze definendo utenti, gruppi, permessi, politiche per il controllo delle versioni e tipi di query per i work item. Definiremo così un nostro metodo che può essere importato in TFS per intero sia mediante il wizard, sia utilizzando il prompt dei comandi di Visual Studio (utile per esempio nel caso

in cui si discosti da uno predefinito di un solo file) con la seguente sintassi:

```
witimport / f nomefile.xml / t <nome server> /p nomeprogetto
```

Il file XML che guida l'intera creazione e deployment del metodo è *ProcessTemplate.xml*, un esempio del quale è fornito nel **Listato 1**.

Una volta personalizzato il metodo è possibile costruire applicazioni .NET per l'integrazione con TFS. In questo compito ci supporta il modello ad oggetti di Team Foundation (TFOM) appositamente studiato da Microsoft per accedere ai dati di SQL Server 2005 e per poter costruire gli add-in personalizzati per il pacchetto Office e Visual Studio stesso. Inoltre è possibile utilizzare TFOM per importare i dati di task e bug da altri sistemi esistenti verso TFS. In questo modo si ottiene l'unica fonte dati dalla quale attingono strumenti di sviluppo ed applicazioni fornendo il punto di partenza per il sistema integrato. Il **Listato 2** descrive un esempio di utilizzo realizzato creando lo scheletro di un'applicazione in C# mediante il wizard delle applicazioni di Visual Studio.

Le Vision 2000 forniscono
chiarezza e stabilità ai
metodi agili

Per poter utilizzare gli oggetti del TFOM bisogna aver cura di aggiungere i riferimenti agli assembly (.dll) di Team Foundation (i cui nomi iniziano per *Microsoft.TeamFoundation*), visibili nel browser dei componenti una volta installato Visual Studio SDK. Nell'esempio si vede

come aggiungere un nuovo work item in un progetto di Team System. TFOM consente inoltre di personalizzare l'integrazione con il sistema di controllo delle versioni.

Mappatura di una metodologia agile alle norme Vision 2000

Dopo aver adeguato gli strumenti alle attività di un team di sviluppo agile, la **Tabella 1** descrive come il compimento delle attività mediante gli strumenti possa essere conforme alle norme. Sono stati presi in considerazione alcuni dei requisiti più significativi (7.3 e 8.2) relativi rispettivamente alla progettazione e sviluppo del prodotto e ai monitoraggi e misurazioni. I requisiti 8.4 e 8.5 riguardano l'analisi dei dati ed il miglioramento. È facilmente intuibile come la presenza del datawarehouse di TFS sia uno strumento essenziale.

Conclusioni

Abbiamo visto come agilità e qualità possono andare d'accordo senza che necessariamente l'una debba escludere l'altra. Sicuramente il rigore nel progettare e sviluppare un'applicazione è imprescindibile. Il segreto è tutto nel pensare che le normali attività di un project manager, di uno sviluppatore o di un tester possono creare valore aggiunto se ben indirizzate. Visual Studio Team System è solo uno strumento che ci ha aiutato a raggiungere l'obiettivo.

Riferimenti

- [1] Graham Wright - "Lecture Notes in Computer Science", Springer Berlin 2003
- [2] <http://www.stsc.hill.af.mil/crosstalk/2005/12/0512miller.html>
- [3] "Sistemi di gestione per la qualità - Requisiti (ISO9001:2000)"
- [4] http://www.microsoft.com/italy/msdn/library/vs2005/team_foundation.msp

Come collaborare con noi?

Computer Programming

- c'è un argomento che non è stato mai trattato
- hai sperimentato una nuova tecnologia
- hai risolto un problema o creato un componente interessante
- sei preparato su un aspetto teorico di programmazione



Progettazione del software - Grafica
 Sistemi Operativi - Networking - Algoritmi
 Database - Applicativi - Ambienti di sviluppo
 Linguaggi di programmazione
 Programmazione Low-Level
 Programmazione Web - Sicurezza...

Puoi scaricare il kit dell'articolista, contenente le norme tecniche, dal sito www.infomedia.it al link "Proponi il tuo articolo"