

# MICROSOFT SOLUTIONS FRAMEWORK

Pierre-Andre Vungoc  
#260149525  
petevungoc@videotron.ca

ECSE 428  
Software Engineering Practice  
Robert Sabourin

## ABSTRACT

In this paper, the Microsoft Solutions Framework is described. The description of the framework will be explained from its foundational principles, the phases of its process model, the teams of its team model as well as the new processes emerging from its most recent version.

## Keywords

Microsoft Solutions Framework. Software Engineering Practice.

## 1. INTRODUCTION

### 1.1 Overview

The “Microsoft Solutions Framework (MSF) is a highly extensible, scalable, fully integrated set of software development processes, principles, and proven practices within Visual Studio Team System, guiding software project teams to deliver enterprise ready solutions.”<sup>1</sup> In other words, it is a collection of software engineering principles and steps to follow in order to help developers to achieve quality software in an effective manner. These principles and steps have been proven by Microsoft and its partners. The eight foundational principles of MSF are:

1. Foster open communication
2. Work towards a shared vision
3. Empower team members
4. Clear accountability and shared responsibility
5. Focus on delivering business value
6. Stay agile, expect change
7. Investing quality
8. Learn from all experiences<sup>2</sup>

MSF now consists also of two models. The MSF Team Model describes the role of the various team members in a software development project. On the other hand, the MSF Process Model describes the different stages in processing for a project.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

Vungoc, Month 4–7, 2008, Montreal, Quebec, Canada.  
Copyright 2004 ACM 1-58113-000-0/00/0004 ...\$5.00

<sup>1</sup> FAQ: Microsoft Solution Framework

<sup>2</sup> GM Solutions: Microsoft Solutions Framework

## 1.2 History of MSF

### Version 1.0

Released in 1993.

### Version 2.0

Released in 1997. The framework is divided into two courses: Solutions Development Discipline (SDD) and Designing Component Solutions (DCS).

### Version 3.0

Released in 2002. The major new updates were the combination of the previous separate models into new Team and Process Models. The new Process Model is made out of 5 phases: Envisioning, Planning, Developing, Deploying (for infrastructure projects) and Stabilizing (for development projects). The new Team model consists of 6 roles: Product Management, Program Management, Development, Testing, Release Management (Logistics Management) and User Experience (User Education).

### Version 4.0 (current)

Released in 2005. This is the most recent version and it updates the 3.0 version by adding two new software methodologies: Agile Software Development (MSF4ASD) and Capability Maturity Model Integration Process Improvement (MSF4CMMI).

## 2. FOUNDATIONAL PRINCIPLES

### 2.1 Foster Open Communication

Communication is vital within and outside of a team. “Historically, many organizations and projects have operated purely on a need-to-know basis, which frequently leads to misunderstandings and impairs the ability of a team to deliver a successful solution.”<sup>3</sup>

There is a need for free-flow communication within the team and with stakeholders for good software development. It is needed also to focus on deliveries and market values.

This way, a free-flow of information will improve software process efficiency by reducing misunderstandings among team members and stakeholders as well as decreasing uncertainties surrounding the project.

### 2.2 Work Towards a Shared Vision

It is important for team members to possess a clear and precise understanding of what the goals and objectives are for the project or process. A shared vision among members help them to

<sup>3</sup> MSF Team Model v.3.1

accomplish the same goal. Without a shared vision, people in a team may not perceive a project in the same manner and might have competing ideas of development, making it harder to develop as a cohesive group.

### 2.3 Empower Team Members

Empowering team members helps them to meet and achieve better their commitments to the project. However, that also means that each member has the ability to take the reasonable effort to achieve a commitment and to communicate as soon as possible if a commitment can possibly not be achieved.

### 2.4 Clear Accountability, Shared Responsibility

In order for a project to work well, responsibilities have to be divided equally among members of a team. However, they are also shared among members because work cannot always be perfectly independent and because members can have a clearer view of what is going on within the project. This way, one individual cannot be accounted for all the goals of a project. Talking about accountability, each role is accountable to the team itself for achieving their role's quality goal.

### 2.5 Focus on Delivering Business Value

It is important that the product management and stakeholders orient a team to define goals. Key project decisions are based on a sound business understanding. This way, a feature or a goal of a project can be dropped if it has no business value anymore or focus can be put on a goal that has a greater market priority.

### 2.6 Stay Agile, Expect Change

The world of software engineering and information technologies is constantly changing and no project is completely isolated from this phenomenon. In a software development process it is necessary to make sure that the designated persons are always present during the project in order to make decisions arising from these changes.

### 2.7 Investing Quality

The quality of a project is a key factor that can literally change the business value of a project or its viability. The team has to make sure that the project complies to quality standards established for a specific type of project. Nowadays, many industries, such as the domain of aircrafts, underly very strict software quality standards.

### 2.8 Learn from all Experiences

A team can constantly improve over time. In life it is always said that we learn from our mistakes. The same way, a team learn from all experiences in order to not commit the same mistakes or to improve its efficiency. It will also be able with time to adapt a process model for developing its software over time that will suit similar types of projects.

## 3. PROCESS MODEL

This process model is mainly used with Waterfall and Spiral software life cycles.



Figure 1: Phases of Process Model

### 3.1 Envisioning

Every project starts somewhere, but it is important that it starts in a good way. The main issue with the Envisioning phase is to make sure that all members of the team share the same vision of the project. The team must have a clear vision of what the customer wants to be accomplished. The business requirements are defined mostly here.

### 3.2 Planning

This phase sets up the development process. The team prepares functional requirements specifications, works through the design process, estimates costs (money and time) and establishes schedules and deadlines for the various deliverables. The requirements are made of four general categories: business requirements, user requirements, operational requirements and system requirements. The design process is split into conceptual design, logical design and physical design.

### 3.3 Developing

The Developing phase is the principal phase of the software development. This does not mean that the other phases are not important, even critical. The team realizes most of the building of the software and documentation needed. However it is important to notice that some building of the software can be made in the Stabilization phase after when testing necessitates code revisions.

### 3.4 Stabilizing

The main activity of the Stabilizing phase is the testing of completed features. The team makes sure that it meets all requirements. The team also focuses on fixing bugs that are prioritized by their importance. Once the project passes testing, it is considered stable and is then deployed (released).

### 3.5 Deploying

The Deploying phase consists of releasing the software by delivering it to the customer. The team must still support the customer and make sure this one is satisfied with the product.

## 4. TEAM MODEL



Figure 2: Overview of Team Model

### 4.1 Product Management

The main goal of the Project Management team is to satisfy the customer. To do so, the team develops and maintains business cases, deals with customer expectations and manages marketing and public relations.

### 4.2 Program Management

The Program Management team ensures that the software is delivered within the constraints of time and requirements. They facilitate the communication and negotiation within the team in case of a conflict, for example. They also conceive the software architecture most of the time.

### 4.3 Development

The Development team is the one that builds the product to the required specifications. They decide within specifications and constraints imposed how to build the product. They can also estimate cost and time.

### 4.4 Testing

The Test team approves the product for release when they have accomplished all the testing required in order to meet all specific requirements and quality standards. Code revisions can be made or they are sent to the Development team.

### 4.5 User Experience

The User Experience team centralizes on enhancing user effectiveness. To do so, the team trains the user to use the software, elicits the user requirements and acts as the user advocate on the team.

### 4.6 Release Management

Finally the Release Management team ensures the smooth deployment of the software by advocating operations, support and delivery channels as well as managing product deployment, of course. They can be seen as the team that installs the software for the client, making a transition from a testing environment to the real operational environment.

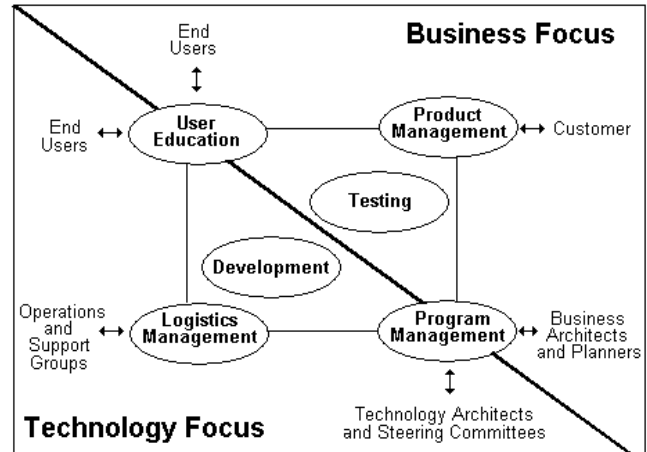


Figure 3: Detailed View of Team Model

## 5. MSF V.4.0 UPDATES

### 5.1 Agile Software Development (MSF4ASD)

Agile Software Development can be considered as a recent technique of software development. Considering that the requirements for a software are always changing and that sometimes software is developed without definitive specifications, Agile is the way to go. “MSF for Agile Software Development makes iterative software development enterprise ready by providing features like risk management, release management and design for operations.”<sup>47</sup> By being a lightweight, iterative and adaptable process, it provides guidance which focuses on changes.

### 5.2 Capability Maturity Model (MSF4CMMI)

The MSF for Capability Maturity Model Integration Process Improvement (MSF4CMMI) has more artifacts, more processes, more sign offs and more planning. It is intended for projects that require a higher degree of formality and ceremony. This is commonly seen in domains where requirements are known in advance and where standards in terms of quality are very high, such as in the aircraft industry, or for any domain that does require mission critical software, for example.

## 6. CONCLUSION

The Microsoft Solutions Framework provides guidance, based upon experiences and best practices from inside Microsoft and its partners, to increase the chance of successful delivery of an information technology solution to the customer by working faster, decreasing the number of people on the project team, preventing some risks, while enabling high quality results.

However, the MSF may not be adaptable or suitable to all software projects. Some phases in the process model or teams in the team model may not be necessary. However for most projects this is a very strong guide. Startup companies can use this model to do effective software development. With time, they will learn from the foundational principles how to model their own process.

<sup>4</sup> Visual Studio Team System.

## 7. REFERENCES

- [1] Microsoft Corporation. FAQ: Microsoft Solutions Framework. Visual Studio Team 2005 Developer Center. DOI= <http://msdn2.microsoft.com/en-us/teamsystem/aa718928.aspx>
- [2] GM Solutions Software Development and Consulting. Microsoft Solutions Framework. DOI= [http://www.gmsolutions.net/microsoft\\_solutions\\_framework.aspx](http://www.gmsolutions.net/microsoft_solutions_framework.aspx)
- [3] Microsoft Corporation. Microsoft Solutions Framework: White Paper. MSF Process Model v.3.1. June 2002. DOI= <http://download.microsoft.com/download/2/3/f/23f13f70-8e46-4f44-97f6-7dfb45010859/MSF%20Process%20Model%20v.%203.1.pdf>
- [4] Microsoft Corporation. Visual Studio Team System – Process Templates and Tools. Visual Studio Team 2005 Developer Center. DOI= <http://msdn2.microsoft.com/en-us/teamsystem/aa718795.aspx>
- [5] Malhotra, S. 2002. Microsoft.NET Framework security. Collaboration NIIT. Premier Press. Cincinnati, Ohio. 408 p.
- [6] Microsoft Corporation. Microsoft Solutions Framework: White Paper. MSF Team Model v.3.1. June 2002. DOI= <http://download.microsoft.com/download/8/7/e/87eef7e-05d2-418a-900d-4896ae4e20db/MSF%20Team%20Model%20v.3.1.pdf>
- [7] Microsoft Corporation. MSF for Agile Software Development Process Template v4.2. Microsoft Download Center. DOI = <http://www.microsoft.com/downloads/details.aspx?familyid=EA75784E-3A3F-48FB-824E-828BF593C34D&displaylang=en>
- [8] Microsoft Corporation. MSF for Agile Software Development Process Guidance. Microsoft Download Center. DOI = <http://www.microsoft.com/downloads/details.aspx?familyid=9F3EA426-C2B2-4264-BA0F-35A021D85234&displaylang=en>
- [9] ApSCo. Microware. Microsoft Solutions Framework. DOI= <http://www.echoes.com/index.html>