Team Development in Visual FoxPro

By Ted Roche, Blackstone, Incorporated
mailto:tedroche@compuserve.com

Overview

A discussion of the processes and issues of developing software with a multi-person or multi-location development team. Covers defining the development process from specifications through documentation, development, testing, delivery, and maintenance. Drilldown into several tools, including Anomaly Tracking System and Visual SourceSafe.

"After more than 30 years of programming, we ought to know that the design of complex software is inherently difficult."

Niklaus Wirth

Introduction

I, like many xBase developers, started out my career in a more honest profession. I was the type of person knew how to change the printer ribbon, and wasn’t afraid to open up a laser printer with a paper jam. Before I knew it, I was drafted to write batch files, then spreadsheet macros, then quick-and-dirty data routines to print a set of address labels. Now, ten years later, I’m trying to manage team development of multi-user, multi-location applications used in real-time to steer the course of a business. What a long, strange trip it’s been…

The PC Revolution occurred with the introduction of cheap processing power into the office, combined with the growing dissatisfaction with the glacial response times of mainframe shops to the needs of a business. Over time, dependence on PC-based systems has grown with the increasing capabilities of those systems to reflect the company’s business, but without many of the checks and balances of the larger scale IS systems.

"Software engineering" in many shops, is an insult to the true engineers of our world.

Over the years, many successful PC shops have adopted the practices of their big brother mainframe shops, with the results that sometimes the PC shop ends up as unresponsive as the “glass house.” The goal of this discussion is to let you know that it is possible to introduce the reliability and repeatability of successes without a complete slowdown, by synthesizing the best practices of large-scale shops with the new tools of the 90s. It is urgent that we bring the reliability and robustness of our applications up to the level of solid mainframe applications without bogging down the development processes in unneeded bureaucracy.

There are no new problems under the sun, just new people encountering the same old problems, disguised in new terminology.
No discussion of development issues would be complete without some discussion of the Software Engineering Institute and the Capability Maturity Model (CMM) of software development. The Software Engineering Institute is a government-sponsored project at Carnegie Mellon University with a mission "to provide leadership in advancing the state of the practice of software engineering to improve the quality of systems that depend on software." CMM describes levels of achievement for a software development enterprise, with a focus on predicting the future achievements of the organization. One observation can be derived from the information provided: Tools are not the solution to the problem. Tools are a means to solve problems. Maturity in software development, similarly to life, can only be achieved as a gradual set of steps.

The CMM model presents a set of "levels of maturity," and, like their equivalence in real life - infancy, childhood, adolescence, adulthood, middle age and seniority - these are likely to be stages to grow through. The levels are described briefly in the table attached, but anything short of a book does not do the CMM justice. Check out writings by Watts Humphrey and the SEI (some of which are in the bibliography) for much more information on this important model.

<table>
<thead>
<tr>
<th>Level 1 a.k.a. Level 0</th>
<th>Initial, a.k.a. &quot;Chaos&quot;</th>
<th>The success of any one project is no indicator of future successes.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Level 2</td>
<td>Repeatable Process</td>
<td>The beginning stages of repeatable success: project plans, estimates, and concurrence from all parties, coordination and responsibility for fulfillment of estimates.</td>
</tr>
<tr>
<td>Level 3</td>
<td>Defined Process</td>
<td>Oversight and review of procedures with an aim at creating a defined process which encompasses all tasks, development of project risk plans, SCM controls, a standard process framework.</td>
</tr>
<tr>
<td>Level 4</td>
<td>Managed Process</td>
<td>Process management and analyses provide constant feedback and refinement of all phases of the development process, yielding corresponding product quality improvements.</td>
</tr>
<tr>
<td>Level 5</td>
<td>Optimizing Process</td>
<td>Productivity improvements are aimed primarily at the process itself.</td>
</tr>
</tbody>
</table>

Table 1: The Five Levels of the Capability Maturity Model

Unless you are a member of a rare organization, I would like to welcome you to level one. But like many 12-step programs, the first crucial step is the recognition that there is a problem. In this case, organizations that are aware of the SEI and CMM are almost always better than those who are not, even if the progress at the former companies is slow in coming.

Since tools are not the solution, how does an organization advance? Well, paradoxically, part of the answer is that advancement comes though the use of these tools. It is not the tool-usage that causes the advancement, however. Rather, it is the organization recognizing the need for the procedures that require these tools, and using the documents produced by these tools to better the development process, that causes advancement.

Too many organizations fall prey to the idea that these tools are the solution to their development dilemmas. Often referred to as the "Silver Bullet Syndrome," projects which make wildly optimistic estimates based on the imagined time-savings of a new promising tool are almost always doomed to failure. While a new tool may provide some time savings, the tool is far more likely to require extensive training, a few patches and a minor upgrade or two. The true value of the new tool is more often improved quality than reduced costs, and that on the second or third usage more often than the first.

While tools are not the answer, we can use tools to begin to document the process we hope to understand, standardize, measure and optimize. The remainder of this document will briefly examine the categories of some of these tools, and give you some insight into their usefulness.

Requirements Management

With one-client, one-developer projects, it is easy to scribble the requirements onto the back of an envelope, or mark up a set of paper forms and create a system to duplicate them. However, once development of larger scale systems begins, with requirements coming from different portions of an organization, and development being performed by different developers, there is a strong need to ensure that all parties understand and agree on what must be done. More formal control of design documents is essential.
One of the more difficult parts of requirements is writing them in such a way that they are concrete, measurable and achievable. An example of a poor requirement might be:

"The system will provide flexibility for later expansion and have all data available for reporting."

A much better requirement might read:

"The system will be capable of printing daily invoices (up to 500 per day) on pre-printed forms (see Appendix) within one half-hour."

The reality of the software business is that the requirements are very likely to change over the course of development. Developers should be thinking about systems and protocols to handle the fact that requirements are going to change. Change management software and requirements tracking systems, are appropriate for larger-scale systems. One interesting product is Requisite from PVCS, which creates an interface between requirements documents, maintained in MS Word, and a repository of requirements used for testing and evaluation. The impact across the system of a change to a requirement can be evaluated and appropriate documentation and testing procedures generated automatically.

**Project Management**

Tools such as MS Project are useful for creating time and material estimates, estimating time to completion, identifying dependencies. For more information on estimating, check out Whil Hentzen's session, [Managing the Application Development Process](#).

In addition to MS Project, another tool worth looking at is Microsoft Team Manager. This intriguing tool combines some of the capabilities of Project and MS Exchange and works with both of these tools. Team Manager is available as a stand-alone product and is also bundled with Office 97 Professional and Visual Studio 97.

For smaller project, the task management capabilities of Outlook or Excel may be sufficient for the task. But automating the time tracking, no matter how it is done, can simplify reporting, encourage reuse from project to project, and begin to accumulate metrics.

**Resources**

You've got to have the developers to develop the project. In addition, if the project is large enough, support staff for documentation, testing, project management will not only be helpful, but have the potential of bring more skillful completion to those tasks. Lister and DeMarco in "Peopleware" and Weinberg in "Psychology" cover a number of items important to developing and maintaining a good environment for programming work. Steve McConnell, in "Rapid Development," maintains that the primary product of software development is not so much a diskette than it is a piece of intellectual property, the developers of this property must be given proper respect and an environment in which to nurture and develop this material.

Microsoft certification may be helpful; like a college degree, not as an indication of development success, but rather as "learner’s permit" of basic mastery. I have also found that developers do benefit from small group study projects. A group I worked in reviewed "Code Complete" and learned a great deal from it. Another group I know of worked their way through "Personal Software Process. SM", a textbook becoming popular in college software engineering courses.

**Design and Analysis**

Larger scale systems require a smaller and smaller proportion of time spent performing the actual coding, and large amount of time developing the design. Consider design tools like xCase, Visual Modeler/Rational Rose, CRC tools, Flow chart and general graphics tools like Visio. In addition to providing better documentation of system design, these tools can be a great aid in analyzing the impact of changes to the system mid-course. These design tools are not primarily tools to generate documentation, piles of dead paper which sit around ignored, but rather are tools to be used interactively and often during the design cycle to help modify, simply, review and communicate a design.

**Communications**

The amount of communication needed by a group increases rapidly as the size of the group increases. While all of the developers need to be alerted when a new version of the database is produced, they should not all need to discuss the proper naming of each new field in an invoice table. Limiting communication to prevent overwhelming each developer while at the same time ensuring that developers are apprised of changes which affect them is one of the challenges of managing team projects. One technique to do this is to establish a series of mailing lists within your e-mail product. The Developers list gets the notice on all changes to major components, the General Ledger list on accounting issues, Management on project timelines, Testing on changes to testing protocols. Our team has used Exchange & Outlook with a great deal of success. We hope to evaluate Team Manager on an upcoming project.

**Coding and Review**

A standing policy for one other programmer to review all before it goes into production reduced our error rate significantly. Whether programmers were more careful when they knew they had to show their code to a co-worker, or whether the reviewers were good on catching simple logic errors doesn’t really matter.

There are many levels of code reviews and inspections. Formal inspection systems with many people poring over code, a scribe to
record the review process, and a formalized set of protocols to prepare, submit, review and re-review code is appropriate in situations where failure is proportionally undesirable, like life-support systems or nuclear weapons. A less stringent system is far more realistic for smaller development efforts.

In addition to its primary purpose of reducing errors in shipping code, a second form of code review can be used as a group educational project. A small group of programmers get together and reviews a particular module, algorithm or UDF. Discussion on the "right" ways to do things often lead to the staff adding a new trick to their repertoire, or discovering a flaw in well-known technique.

The other effect we see I've nicknamed the "CompuServe Effect," as it usually occurs to me while composing a message to post on CompuServe. Once a developer has tried everything they can think of and cannot get a piece of code to work, trying to explain the problem to a colleague often allows the teller to explain the answer to him- or herself. Many times I get halfway through composing a message on CompuServe, only to discard it because the answer is right in front of me. For this reason, we strongly encourage our developers to have some time that can be spent together reviewing problems or brainstorming, to take advantage of that synergistic effect.

Source Code Control

With Visual FoxPro 5.0, Microsoft introduced the ability to tie the native project manager directly to a source code control provider. Microsoft's Visual SourceSafe is one tool that can be used for this. Several third-party manufacturers also provide the interface needed to work with Microsoft's published Source Code Control Application Programming Interface.

When a source code control provider is available, having been properly installed and registered on the workstation, the Project Tab in the Tools/Options dialog will display the SCC provider in its dropdown (see Figure 1). A new option, "Join Source Code Control Project..." is added to the File menu, and new menu options are available in the Project menu.

In addition, the project manager itself displays a new set of icons next to the file list (see figure 2). These show which files are

![Figure 1: Source control options in the Tools/Options dialog](image1)

![Figure 2: FoxPro Project Manager with source-controlled files.](image2)

Click the right mouse button while over a source-controlled file provides many of the source control options:
While Visual SourceSafe and the Visual FoxPro 5.0 interface work very well under most conditions, I have worked in some environments where using Visual SourceSafe through the FoxPro interface was too slow to be usable. Opening a project could take three minutes, and refreshing the project just as long. Worse, if two users were to attempt to check out at the same time, their systems will appear to lock. Fortunately, there is hope: it is my understanding the Microsoft has fixed this problem in version 5.0a - I hope to soon test this and report my findings at this conference. As an alternative work-around, you could construct two projects for each developer, one linked into SourceSafe for daily check-ins and check-outs, and one for development use during the day.

SourceSafe will be demonstrated and reviewed in detail during the presentation.

Anomaly Tracking Systems

How many errors are typical of your development projects? Is there a member of your team who excels at I/O routines? How strong are your programmers at isolating and identifying bugs?

Most of us are unable to answer these questions authoritatively. We probably have a good grasp of generally how the team is doing, but it seems like no one is keeping score. This is where anomaly-tracking systems can come into play. An anomaly tracking system can help you understand where your team's strengths and weaknesses are, and help identify areas where you want your team to put more effort or care, or perhaps identifying areas where your team could use additional help. Check out products like Microsoft ATS, Visual Intercept, Track, and PVCS-Tracker for some ideas in doing this.

Microsoft's Anomaly Tracking System will also be demonstrated.

About the Author

Ted Roche is the director of development at Blackstone Incorporated, a Microsoft Solution Provider based in Arlington, Massachusetts specializing in database development and network infrastructure using Microsoft BackOffice and Visual Tools. He is co-author, with Tamar Granor, of the critically acclaimed "Hacker's Guide to Visual FoxPro 3.0" from Addison-Wesley. Ted is a Contributing Editor for FoxPro Advisor magazine and co-authors the "Ask Advisor" column. He is a Microsoft Certified Solution Developer, a Microsoft Support Most Valuable Professional and a CompuServe Support Partner. Email: tedroche@compuserve.com, phone: (617) 641-0400.

Bibliography

Brentnall, Savannah, Object Orientation in Visual FoxPro, 1996, Addison-Wesley, 0-201-47943-5
Brooks, Frederick P., Jr., The Mythical Man Month, 1975, Addison-Wesley, 0-201-00650-2
Cooper, Alan, About Face, IDG Books, 1995, 1-568843-22-4
DeMarco, Tom and Lister, Tim, Peopleware, Dorset House, 0-932633-14-5
Gamma, Erich, Richard Helm, Ralph Johnson, John Vlissides, Design Patterns: Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995, 0-201-63361-2 (NOTE: a CD version was due from Addison-Wesley in June 1997, but was not available for review in time for these notes.)
Humphrey, Watts S., Managing the Software Process, Addison-Wesley, 1996, 0-201-18095-2
Humphrey, Watts S., A Discipline for Software Engineering, Addison-Wesley, 1995, 0-201-54610-8
James, Geoffrey, The Tao of Programming, InfoBooks, 1987, 0-931137-07-1
Maguire, Steve, Writing Solid Code, Microsoft Press, 1993, 1-55615-551-4
McCarthy, Jim, Dynamics of Software Development, 1995, Microsoft Press, 1-55615-823-8
McConnell, Steve, Code Complete, Microsoft Press, 1993, 1-55615-484-4
Weinberg, Gerald M., *Understanding the Professional Programmer*, Dorset House, 1988, 0-932633-09-9

Articles

"A Few Words with Niklaus Wirth," Software Development magazine, June 1997, pp. 52-60

"Using Visual SourceSafe with Visual FoxPro 3.0" whitepaper from Microsoft Technet


Visual FoxPro Online Developer's Guide, Developing in Teams chapter, "Managing Visual FoxPro Projects Under Source Control"

Web sites of Interest

[www.acm.org](http://www.acm.org) — the Association for Computing Machinery, a 50 year old organization


[www.atl.sysguild.com](http://www.atl.sysguild.com) – Atlantic Systems Guild is the home of Tom DeMarco and Timothy Lister

[http://www.clark.net/pub/imv/qvcsman.htm](http://www.clark.net/pub/imv/qvcsman.htm) – check out a $25 shareware version control system!

[http://www.sei.cmu.edu](http://www.sei.cmu.edu) – The Software Engineering Institute at Carnegie Mellon University, with lots of information and numerous papers available for download.

[http://www.elsitech.com](http://www.elsitech.com) – Developers of Visual Intercept, a defect tracking system with tight integration to Visual SourceSafe, from the original developers of SourceSafe


[www.icca.org](http://www.icca.org) – Independent Computer Consultants Association


[www.microsoft.com/teammanager](http://www.microsoft.com/teammanager) – Microsoft's home page for Team Manager.

[www.pvcs.com](http://www.pvcs.com) – manufacturers of PVCS software configuration management, source code control and change management software, including RequisitePro®, Tracker® and Version Manager


[www.soffront.com](http://www.soffront.com) – manufacturers of Soffront Track, bug and issue tracking system.

[www.spr.com](http://www.spr.com) – location of Software Productivity Research, home of Capers Jones

[www.starbase.com](http://www.starbase.com) – source code management software, including Versions version control, sophisticate threaded conversations database, groupware, Internet support, more.

[www.visio.com](http://www.visio.com) - Visio is one of the best of the technical drawing packages.

[www.yourdan.com](http://www.yourdan.com) - home of one of the leading authorities in software development issues.

Newsgroups

On Microsoft's NNTP server, check out:

Team Manager: [news://microsoft.public.teammanager](news://microsoft.public.teammanager)

SourceSafe: [news://microsoft.public.sourcesafe](news://microsoft.public.sourcesafe)
FoxPro: twenty-one different groups, most active are

news://microsoft.public.fox.programmerexchange, news://microsoft.public.fox.forms, news://microsoft.public.fox.chatter