UNISYS

Developing Agility

A Quarterly Newsletter for Unisys EAE and Agile Business Suite Customers

July 2012

Contents

How Do You Improve Developer Productivity?

By Maarten Schneider, Worldwide Marketing Manager, Enterprise Application Environment and Agile Business Suite, Unisys TCIS

"Do more with less." It's on everyone's to-do list – especially when it comes to IT. Naturally, this means being more productive. But how do you measure productivity or know if it's improved? This is a question software development teams have struggled with for many years, and one that's more relevant today than ever before.

When I speak with software development managers, most say they have a good sense of their team's productivity, but admit that it's difficult to measure. Some calculate "hours per function point" after a project is finished, and a few use this data to estimate the size of new development projects. Others track the number of lines of code written in a given period of time.

In my view, measuring productivity is more important than the method you use. You have to begin somewhere and gradually refine your approach. For example, start with "X hours per function point," and then see how well that matches up to the actual effort.

When I look back at the many projects I've seen from Enterprise Application Environment (EAE) and Agile Business Suite (AB Suite) customers, one thing stands out: The larger the team, the more time its members spend communicating – and the lower the productivity per developer. In fact, I have even seen cases where adding extra developers to a lagging project actually slowed it down even more. >>

The lesson? You are better off having a few very good developers following a sound process supported by the best possible tool.

Therefore, I would argue that when you start to look for productivity improvements, you should consider three unique elements:

1. Your developers' technical skill sets and business knowledge
2. Your development process
3. Your development tool

Continuous training of your developers pays off. And that training should happen in three key areas:

1. Your organization's business objectives, processes, and end-user requirements
2. New application development processes
3. New features/enhancements to your application development tool

Bridge the Business-IT Communication Divide

Let's be honest: Detailed requirement documents are difficult to develop, hard to read, and nearly impossible to get approved. Therefore, small "use case" descriptions, limited-scope projects, and routine prototyping are excellent ways to speed up development and deliver faster. In addition, the more familiar your developers are with your business processes, the easier they can communicate with business users – and the better chance that what's developed matches their expectations.

Leverage the Best of New Development Approaches

Application development processes have changed in the last couple of years. Today, most organizations are using approaches that are positioned somewhere between the traditional Waterfall and Rational Unified Process (RUP) or Scrum.

RUP and Scrum are agile and iterative approaches, and both use timeboxing as a way to set an end date, communicate it to everyone involved, and hold to that timeline.

RUP teams have regular project meetings, and Scrum teams take it even further – gathering daily for 15-minute "stand up" meetings. Staying close to the business requirements and users, and following strict project-management techniques are key for both approaches.

Many organizations report better results with smaller developer groups using RUP, Scrum, or a combination of both. And, many of the developers I speak with confirm that smaller projects reach a successful result more often than large ones.

Use Highly Productive Development Tools

The last area for improving productivity is the development tool. The lines of code a developer must write are important, not only for the initial development of the software, but also for long-term maintenance. The more code, the more complex the environment – and the harder it is to support and enhance.

In this regard, EAE and AB Suite users have the ultimate productivity tool – a high-level language that dramatically reduces lines of code. The AB Suite environment has an additional advantage because the Microsoft® Visual Studio® framework enables developers to see the application more clearly and leverage an easier debugging facility.

So, the next time you're looking for ways to do more with less, consider how you can do more with the latest capabilities in EAE and AB Suite. For example, take a look at the ClearPath ePortal article in this issue to see how it's evolved to support EAE and AB Suite. And, think about ways you can update your developers' skills – with both technical and business training. Finally, look at your development process and see if you can adopt elements of RUP and Scrum to further speed time to delivery.

How are you doing more with less? I'd like to hear from you. Please send me a note at ABSuite@unisys.com.

New ePortal Integration Brings Mobility to EAE and AB Suite

Already a standard feature of all new ClearPath systems, the ClearPath ePortal specialty engine is a great tool for web- and mobile-enabling ClearPath applications of all kinds.

In June of 2012, we released changes that made the use of ClearPath ePortal even easier, and more powerful, when it comes to EAE and Agile Business Suite.

With the new release, you can import your existing client interface into the ePortal generator and efficiently extend your EAE and AB Suite applications to the Web and mobile devices – all without touching the core application. And because this happens natively within your ClearPath environment, you can execute these projects while leveraging the same levels of security, flexibility, and scalability you've come to appreciate in your other development efforts.

Getting Started

The first step in creating an ePortal client application is to define the data source project. Simply select "AB Suite EAE Data Source" from the ClearPath data source project templates in Visual Studio, and let the wizard do the rest. Then, import the Ispec definitions – there's a wizard for that, too – by pointing it to the output from your EAE/AB Suite ePortal client generator.

Next, define the client types that will access the Ispec transactions by creating a ClearPath Presentation project using one of the supplied templates. Each client project can use either the same data source definitions or different ones, depending on how much access you want to give users.

Today there are ClearPath Presentation project templates for Mobile, Web, and Web Services.

Mobile Made Easy

When you choose "Mobile" as the ClearPath Presentation type, the solution automatically builds the mobile interface for Apple® iOS, Android™, and BlackBerry® smartphones, and iOS or Android tablets – all without you ever having to worry about learning a specific mobile programming language. The look and feel will change to match the device connecting at runtime.

You may want to tweak the presentation to eliminate data fields that are unnecessary in a mobile interface, or take advantage of specific Graphical User Interface (GUI) controls.

Mobile is all about speed and efficiency, so the fewer transactions your mobile application has to perform, the better. Rather than have users navigate a logon page and several menus, a mobile app should take them where they want to go in just a step or two.

The orchestration feature in ePortal is extremely helpful in this regard. Orchestration essentially creates a macro that presents multiple transactions to the user as a single action. Moreover, it can further streamline the end-user experience by helping you selectively import the screens and images you wish to include in the mobile application.

Should the user interface (UI) of the main application change at any point, ePortal has a reconcile function you can invoke to keep what mobile users see synced up with what's happening on the back end. What's more, if you start using ePortal with EAE, but are planning to migrate to AB Suite, there's no reason to worry about losing any of your hard work. Your UI will seamlessly move over to the new environment, and you'll be set.

To learn more about everything you can do with the ClearPath ePortal, check out the playlists on the ClearPath YouTube channel.

CSC Spotlight: Grant Paine

This article is the latest in a series showcasing the Unisys Customer Support Center (CSC) Analysts who support EAE and Agile Business Suite. Interested in seeing a support analyst featured? Send us your nomination: ABSuite@unisys.com.

Grant Paine arrived at Unisys (then Burroughs) in 1983. Nearly three decades later, Grant is a key member of the CSC team, providing support for ClearPath MCP software, EAE, and Agile Business Suite to customers in the Asia-Pacific (APAC) region.

Developing *Agility* recently talked with Grant about his role as a CSC analyst, what he likes best about the job, and the insights he's gained from working with EAE and AB Suite users for nearly 30 years.

Developing *Agility*: Tell us a bit about the services you perform and the customers you support.

Grant Paine: I've been involved in a wide range of support services over the years – from performing MCP operating system upgrades, installing new MCP systems, and training users, to helping customers migrate to AB Suite. I also do customer demos and provide consulting services.

I work mostly in the APAC region, though I also assist customers in the United States, United Kingdom, and Europe more frequently now that the CSC functions as a global team.

DA: What do you like best about being a CSC analyst?

GP: I enjoy sitting with customers, learning more about their environments, and working through issues together. So I really like to meet face-to-face because it allows me to be more familiar with a customer's needs and support them better.

Along those lines, I also enjoy the wide variety of customers we support and the different work they are doing. I'm happy to see that the majority of customers are eager to adopt new technologies and new ways of doing things, too.

DA: What were some highlights of the last year?

GP: A big highlight of 2011 was the work I did to help the Alliance Group in New Zealand implement a Microsoft Visual Basic® .NET version of Client Tools to replace an aging graphical workbench product on over 700 workstations.

The other highlight was when I traveled to Manila, in the Philippines, to migrate a customer from EAE 3R2 to EAE 3R3 on Microsoft Windows®, and help them implement an ASP.Net frontend using Client Tools. This interaction renewed our relationship with the customer and resulted in an engagement to migrate them to AB Suite.

DA: What do you think customers don't know about the CSC that they should?

GP: Customers may not know that all CSC locations now work as a global team, so analysts from all over the world will often collaborate and assist one another on a customer support request (SR). This structure helps us share knowledge and results in improved customer service. >>

DA: If there was one thing customers could do to make your response more effective and efficient, what would it be?

GP: Provide as much detail as possible when creating electronic SRs, and be sure to note the software/interim correction (IC) levels being used. The more information we have available the better, as it will allow us to investigate the situation and provide a resolution much faster.

DA: Is there anything else you'd like to tell our readers?

GP: The "Documentation" page of our support site features a wealth of information, including the latest manuals, white papers, and tutorials about AB Suite. You can access it by going to support.unisys.com, choosing "Documentation" from the top-left side of the screen, and clicking the AB Suite link under "Application Development Solutions." You can access information about EAE by clicking the EAE link.

Engineering Corner: Debugger Tips and Tricks

By Suresh Ananthan, Team Lead, Agile Business Suite Debugger, Unisys TCIS; and Paul Bourke Subject Matter Expert, Agile Business Suite Debugger, Unisys TCIS

There's a lot of great functionality packed into the Debugger module within Agile Business Suite Developer – some of which you may not even know. So to help you maximize the value of Debugger, we've compiled this list of helpful tips and tricks.

Tip #1: Use a *Copy* of Your Production Database with Host Database Access (HDBA)

Using valid data while debugging is important because it's the best way to identify data-related problems. However, we recommend that you steer clear of debugging using HDBA against your production database. Besides the risk of corrupting live data, Debugger could inadvertently trigger lock situations that may hold up the production system.

That's why you should always use a copy of your production database in the HDBA function. This way, you'll be working with valid data, but won't accidentally impact anything that's currently in production.

Please note that in MCP systems you'll have to set "Enable Host Database Access" to "true" to access the host database in Debugger. Learn more about setting up HDBA on MCP systems in this how-to document.

You'll need to switch it on in Windows environments, as well, but you'll also have to point Debugger to a Windows runtime database or a copy of it.

Tip #2: Use Multiple Watch Windows

Debugger features a number of watch windows where you can monitor the values of certain variables. Because AB Suite allows you to have multiple variables with the same name, copying the fully qualified variable and pasting from logic is the best way to guarantee you're watching the right one.

In addition, the "autos" watch window shows all of the variables for your current method. It automatically updates as your Debugger session rolls on, so as you switch methods and the set of variables in use changes, you'll see them reflected in the content of this window.

Tip #3: Add "GLB.STATUS" to the Watch Window

A good way to uncover unexpected errors is by keeping an eye on GLB.STATUS during the debugging the session. Adding "GLB.STATUS" to your watch window will help you detect any failures in the LDL+ code while debugging, such as adding a record that failed due to a duplicate record in the database, or trying to delete records that don't exist in the database.

Tip #4: Remember the LDL Verb "LOG"

One of the more useful LDL verbs, "LOG" helps you log different levels of messages, such as info, warning, or error, in the Visual Studio output window. Plus, it has an option to halt the program execution while running the debugger. When "If LOG DEBUG halt <message>" is used, Debugger will stop logic execution after the LOG LDL+ line and display a message in the output window. This will log a message only in the debug session. >>

If you want to log the information in Runtime and debugger, you should substitute "ALWAYS" for "DEBUG." This is especially helpful when you want to identify and weed out any specific conditions that should never happen.

Tip #5: Don't Forget About the Tracepoint Option

A feature introduced in Microsoft Visual Studio 2005, tracepoint provides a different way to use breakpoints. Tracepoint is a breakpoint **with a custom action associated with it**. Unlike LOG commands, which become a permanent part of the code, tracepoints are visible only to the developer who creates them. And just like breakpoints, they remain until you delete them from the breakpoints window.

A tracepoint causes Debugger to perform a specified action, rather than simply pausing program execution (although this is an option, too). Instead of halting, the tracepoint writes information to the Visual Studio output window, allowing you to watch your message while debugging.

Moreover, you can customize the message displayed in the output window to include anything you want displayed, such as variable names, function names, or any other block of text. This makes tracepoint extremely helpful when debugging a big loop, where you'll need to track thousands of values.

Please note that inserting a tracepoint must be done from the breakpoints section of the shortcuts menu.

Tip #6: The Call Stack Window Shows Which Method is Called

When you're in a situation where one method invokes another, the call stack window helps you see under what circumstances something is invoked.

This is useful when you reach a breakpoint deep in your debug session and want to see how you got there. In the call stack window, you'll see that the top entry is the method you're currently executing, the following one will be the method that called it, and so on. If you double click on any entry in the call stack, it will take you to the code from where the call was made.

The call stack window is also good for any methods in the AB Suite language that are indirectly invoked. For example, if you choose to print a Frame, the call stack window will show what method invoked the print method on your Frame below the Frame's main method.

Tip #7: Use the Right Configuration for the Right Reasons

While you can create different configurations for specific purposes, when it comes to Debugger, we recommend using the same configuration you'd use in your runtime deployment. If you don't, you'll need to apply any changes that happen in runtime to the Debugger configuration. But if you use the same configuration, you'll always be in sync.

There are a couple of instances where this isn't possible. If you have multiple Debugger users working concurrently in a terminal server environment, then you can't use the same configuration. And if you want to deploy the Windows runtime to the same machine on which you are running production, you'll have to make changes to avoid clashes between the two. >>

For more information about configurations, and how you can use them to speed up Debugger build and start times, check out our how-to document.

Tip #8: Automate Automates Routine Steps

Similar to the Animate feature in EAE Developer Test, the Automate macro in Debugger performs repeated steps for you, so all you have to do is sit and watch the path it takes through the code as each line is executed – until it hits a breakpoint. This comes in handy when you're analyzing a looping code, but aren't getting the results you expect. You can switch on the automate macro and watch it step through the logic without ever having to touch a keyboard or mouse.

Please note that Automate is not available right out of the box. You'll need to create the macro first, and then set up a tool bar. For pointers on this process, check out our how-to document.

Tip #9: Edit and Continue Enables on-the-fly Changes

The Edit and Continue feature allows testing and development to happen in unison, giving you immediate feedback about whether or not that logic is performing as desired. As its name implies, edit and continue gives you the option to correct any invalid LDL+ logic you find, resume testing, and validate the fix on the fly – all without ever ending your Debugger session.

Got a tip that isn't covered here? Drop us a line, and we'll add it to our next list of tips and tricks.

AB Suite Migrations are Hot in France

There's a hot new trend taking France by storm, and this time it has nothing to do with cutting-edge fashions or art-house cinema. It's Agile Business Suite migrations, and they're all the rage among French Unisys customers.

We recently spoke with members of our French team to find out what's made migrations so popular there, and the answer came down to a few simple points.

First and foremost, every migration follows a defined, step-by-step process that divides the project into logical phases designed to show continuous progress. It all begins with an analysis of the customer's existing EAE environment. Our French team reviews the application's source code, performs a migration into a test AB Suite environment, and provides the customer with a report detailing their findings and the best path forward.

Once these details have been discussed and agreed upon, the team builds a formal plan for migrating the customer's environment to AB Suite.

This plan follows a proven path, where every phase builds on the last, so the organization completes the migration well prepared to maximize their investment in AB Suite. Steps include:

- Train developers in Visual Studio, new development concepts, and AB Suite
- Install AB Suite on the customer's development system
- Migrate each application's model
- Test and validate migrated applications
- Build a plan to move migrated applications into production
- Conduct more in-depth developer training to promote fluency in the new environment
- Assist with the move into production, both before and after the applications go live
- Monitor the customer's progress post migration to ensure AB Suite is functioning as intended

To support the process, there are two specific training courses our French team routinely recommends to any organization ready to migrate. These courses revolve around developing and building applications with AB Suite, and include modules on Visual Studio and Object-Oriented (OO) concepts.

For customers moving to AB Suite on Windows, the team also provides a specialized, two-day course for administration and operations personnel. This course is particularly important because the AB Suite Runtime for Windows environment is significantly different than that of EAE.

Unique Projects, Unique Lessons Learned

Every customer our French team works with is unique, so each AB Suite migration provides a new insight or kernel of knowledge they didn't have before. And after successfully migrating seven organizations to AB Suite, the team has some valuable advice for any company thinking about making the move: Don't hesitate.

While they understand that organizations may have some trepidation about migrating, the benefits are far too substantial to deny. They note that even if the organization's developers initially intend to continue working as they did in EAE, moving to AB Suite introduces a host of new capabilities they can leverage in the future. >>

And although there is a bit of a learning curve with AB Suite, the team has found that organizations are more than comfortable with the new development environment in just a month or two. In particular, the OO programming concepts have stood out as one of the new capabilities developers are most excited to put into use.

One Approach Across the Globe

So what's most noteworthy about the success our French team has had migrating organizations to AB Suite? The migration process they employ is by no means exclusive to Unisys customers in France.

Instead, it is a standardized methodology followed by all Unisys teams across the globe. So no matter where you're located, your migration – from initial analysis to the formulation of the migration plan to the training program – will unfold along the same proven path we've highlighted in this article.

If you have questions about the migration process, or would like to get started planning yours, please contact Diane McGonigle.

Info Center

New additions to our libraries of How To documents, white papers, and other useful information in the Public Information section of the Unisys Support Site include:

- **How To:** How To Automate Logic Execution in AB Suite Debugger (**NEW**)
- **How To:** Use MULTI in Windows Runtime (**NEW**)
- **How To:** Use Non-Phased SQL in Windows Runtime (**NEW**)
- **White Paper:** External Classes in AB Suite 2.0 (**NEW**)
- **Utility:** ABSLogAnalysis (for analyzing the performance of online transactions in runtime systems) (**UPDATED**)
- **Demonstration:** Business Integrator (**NEW**)