Bridging the Knowledge Gap

Enterprise Analyzer Technical White Paper

"For every complex problem, there is a solution that is simple, neat, and wrong."

-H.L. Mencken
THE ENTERPRISE APPLICATION PROBLEM

For over 50 years, software applications have been developed to serve the needs of businesses and other organizations. Unlike capital investments in physical assets, the majority of these applications were never overhauled or replaced. Over time, while continuing to deliver core value, they have also been consuming the majority of the IT budget.

The growing IT debt – a result of business changing faster than its underlying software assets - is a well-recognized issue with industry analysts and IT leaders. To various degrees, organizations are unable to deliver new requirements in time, and experience high costs when attempting to do so. Common approaches to dealing with the issue are myriad – including internal application overhaul projects, replacement by packages and outsourcing to professional services organizations. Whichever option is chosen, the ability to appreciate and understand the existing applications and how they currently operate is a primary obstacle in determining how and what to change.

Staff turnover, poor documentation, changes in responsibilities and other factors have led to a dearth of appropriate skills and awareness in key incumbent IT systems. We will refer to this lack of insight as the “Knowledge Gap”. In this context, Micro Focus’ Enterprise Analyzer helps support business initiatives requiring “invasive procedures” into existing program code. This paper discusses the scenarios where organizations are looking to undertake efficient application maintenance work by tackling this knowledge gap.

APPLICATION PORTFOLIO MANAGEMENT SCENARIO

Delivering value at an atomic level

Application Portfolio Management (APM) advocates support a top-down, IT governance-oriented approach. How can you effectively maintain your existing applications if you don’t have a clear view of your IT estate and the relative cost or benefits of each application? Following this path, the solution is to first map the existing applications to their business functions, then assign values and costs to each group by combining internal financial data, survey results and technical complexity metrics. These are communicated through graphical and tabular visualizations, from which an analyst can dig deeper to assess the points of failure and where the best return on investment can be obtained. Micro Focus Enterprise View, a web-based product integrated with Enterprise Analyzer, aids this approach.

The quest for better application responsiveness and quality doesn’t stop there. Once identified, modernization or replacement targets must be understood in greater detail. A few of the questions raised are:

- Which modules join together to provide a business service?
- What are the dependencies between applications – including external interfaces – which we must consider before making changes?
- What collection of languages and platforms must we address with the proper skillsets?

Questions of this nature commonly reside at the intermediate level of application analysis. A common approach for several vendors in this domain is to build a repository of modeled application objects with their physical and logical relationships. A selection of source code would then be collected and registered to it, and then interactive tools for search, report and visualization purposes would be applied. Micro Focus Enterprise Analyzer, with its industry leading capability to share application knowledge on multiple layers, delivers on this scenario in a manner that is well integrated with the APM scenario mentioned on page 1.

All of the above, however, is not enough to support the individual developer in his quest to deliver changes in an efficient, risk free manner. Once the application has been decomposed and there is a sufficient understanding of each program’s role, the developer is usually tasked with making changes to parts of a program, making sure that the changes don’t break dependent processes. Effective knowledge tooling that makes it possible to execute on the changes safely, is required - without occupying too much of the workday on research. This goes to the heart of the Knowledge Maintenance scenario.

MAINTENANCE KNOWLEDGE SCENARIO

The Maintenance Knowledge Scenario drives efficiency and reduces risk at an atomic level of application modernization: the single developer or small team.

In summary, Maintenance Knowledge delivers the ability to quickly locate objects of interest within an application and accurately determine the size of a desired change request. Enterprise Analyzer Developer Edition is the product used to deliver this scenario.
In a real life example, a change to the length of ICD (International Classification of Diseases) from 9 to 10 digits necessitated a shift in all software applications handling this code. At a leading healthcare insurance provider, Enterprise Analyzer was used to locate all instances of ICD in the application portfolio and to ensure full coverage of all impacted constructs. There was no possibility of completing this task by other means, since the instances were widely dispersed throughout the code and documentation was outdated.

In another case, a medical and surgical supplies distributor wanted to expand a number of fields spread across millions of lines of code in its custom Enterprise Resource Planning (ERP) application. The company was looking for a solution that would allow it to carry out the analysis with greater accuracy, in less time, cost-effectively and at low risk. Using Enterprise Analyzer, the assignment was completed at 15% of the estimated cost (saving 85%) and 33% of the estimated schedule (saving 66%).

Enterprise Analyzer offers unrivalled capability in this category. Other solutions typically provide reports and diagrams to the intermediate level of analysis discussed above. But the missing component is the ability to effectively show a developer the true impact of a single field change across the application, including the full path from a starting object of interest to other impacted points. Impact analysis is typically provided in a report or list output. It does not address false positives, which are cases where an impact is claimed but does not exist in reality.

DETAILS

Let’s describe each step in the Maintenance Knowledge scenario in some further detail.

Part 1 – Application Level Understanding

Advance steps relevant to any application modernization project are outlined below. These can be performed at the administrator / systems analyst / architect level and may not necessarily involve the individual developer. But when it does, it is critical that these tasks have been properly carried out in advance.

Enterprise Analyzer Repository – build and parse (Administrator Task)

Depending upon organizational preferences, the repository build phase may preclude or follow the modernization selection phase (below). In the first and more typical case, we will initially want to build a repository of existing applications and then perform some preliminary analysis on them in order to decide on what to modernize. In other cases, the application, or applications, selected for modernization is already known, and it is only this that we will populate the repository with.

The mechanics of building a repository are well-defined: pointing Enterprise Analyzer to source code, registering it to a database and then parsing the code using static analysis algorithms similar to compilation. For those languages and platforms that are supported to an advanced level, this includes the building of control flow and data impact models, allowing for subsequent analysis to take place to the most detailed level possible.
Select modernization/maintenance targets (Business Analyst / Architect task)

This step involves mapping your application portfolio and deciding upon the highest priority candidates for modernization/overhaul, or replacement. An optional product like Micro Focus Enterprise View can facilitate this step by allowing you to collect technical metrics and business information which is then used as described in the Micro Focus APM scenario mentioned on page 1. Some organizations opt for homemade solutions such as Excel or “intuitive” decisions, which may suffice if the problem set is small, or there is a sound understanding and alignment between business and IT. Ultimately, from the single developer’s perspective - before being asked to change an application or sub-system - this step should have been taken so that highest priority applications are dealt with first.

Application Decomposition - segmentation and tagging (Systems Analyst / Architect task)

Attempting to gain a full understanding of a range of applications in one exercise can be daunting – given its size, complexity and number of interfaces. A useful approach is to decompose the applications so that the highest priority ones, i.e. the modernization targets, can be fully understood.

Within Enterprise Analyzer there is a capability to divide the parsed application into sub-systems or “Projects”. Each Project is a logical grouping of application objects and can be defined or deleted on the fly. A user has control over which Projects will appear in her default Repository view.

Another important facility for the user is “Tagging”. Tagging allows for single or multiple textual values to be assigned to each application object. A tagging structure can be hierarchical, so that business areas such as Sales, Finance and HR are part of Enterprise and Order Entry, and CRM are part of Sales. Enterprise Analyzer even includes a “Tag Manager” which helps assign and manage tag values across a high number of objects.
Both features are utilized in the next step, which is Application Analysis.

Application Analysis (Systems Analyst task)

The Application Analysis step involves gaining a firm grasp over your sub-systems of interest, their content and inter-relationships.

The most common Enterprise Analyzer tool for this is Diagrammer – a visual analysis facility for programs, screens, files, transactions and their inter-relationships. Diagrammer operates within the confines of a Project whilst allowing a flexible arrangement of groups according to their tag values. Going far beyond the visualization capabilities of competing products, it is capable of displaying thousands of objects in a pane and zooming in and out of areas of interest on the fly. It also offers a powerful scope editing facility, allowing the definition of customized Diagrams based upon a selected collection of objects and relationships. For example, all screen, program, transaction and read/write to file relationships.

Hand in hand with visual analysis, Enterprise Analyzer provides industry standard metrics reports with complexity metrics. Obvious examples include lines of code and the number of independent logical paths in a program. Other notable instances include Halstead’s program volume, the Maintainability Index – an inverse function of the metric used to assess the ‘maintainability’ of programs – and McCabe’s cyclomatic complexity. This information will also help determine the more complex programs that should be tackled first.

Part 2 – Program Level Analysis

At this point, detailed program level analysis is performed by a developer looking to carry out selective maintenance. For example, a business driven decision may be: “Increase the size of all ICD codes from 10 to 11 in the health claims application”. The onus will now be on the developer to locate all the instances of the ICD code within the application and ensure that any change performed will not adversely impact any processes.

To achieve this goal, the Maintenance Knowledge Scenario supplies the detailed level information to assist the developer in his daily tasks. This is where Enterprise Analyzer differentiates itself from other products. They may also offer “impact analysis” but typically will produce a report with many false positives, where the developer cannot determine the true impacts and those which can be passed over.
Program analysis

This step involves taking application understanding to the next level: program understanding. The lines between an “application” and “program” are sometimes unclear. But to keep it simple, let’s assume that a program consists of source code and its associated “include” files: for example COBOL copybooks.

The developer will use Enterprise Analyzer’s Interactive Analysis facility to visualize the source code, its control flow and its data flows. For instance, you might right-click on a variable in the source pane, navigate to its declaration, right-click again and see all of its occurrences in the program. Or, you might view the control flow of paragraphs within the program as well as a flow chart of logic within a paragraph.

This level of understanding is important to a developer, because nobody wants to make changes in isolation. Locating a point of interest in a program and understanding the various ways it may have been possible to reach it, provides a perspective similar to that of a visitor to a foreign city, using a handheld GPS device to navigate from one place to the next.

Object location and listing

Let’s return to our example: locating all of the fields associated with the ICD code, so the scope of changes required can be estimated. At this point, the Advanced Querying tool, also known as "Clipper", will be used to locate all instances of data being written into a field with the naming convention fitting the ICD designation. The results of the query will be placed in a "list". A list is a collection of source code locations that can be saved for later reference. In our case, we have every instance that the ICD field is being written into.
Field change analysis

Finally, using the list created in the previous step, we would like to perform impact analysis to identify all possible dependent fields. For example, if one of the list members is named "Claim-code-ICD" and a working storage variable named "WS-Claim-code" passes data to it, we want to isolate the working storage variable and add it to the list of variables that must be changed.

We will not have located the "WS-Claim-code" variable in our initial query, because its name does not include the 'ICD' naming convention. By using Enterprise Analyzer’s Change Analyzer tool, we will be able to locate it.

Change Analyzer relies on a data impact algorithm. This takes advantage of the mapping of all application objects to their relative memory locations we carried out. It is able to traverse an “impact path” forward and backward from an established point of interest to all other impacted fields within a program, and between multiple programs within a Project. Once related variables and records - the “synonyms” - are located, they can be added to the list of items to be handled in the maintenance workflow.

CONCLUSION

The enterprise application portfolio presents challenges regarding responsiveness when evolving business needs. High level application portfolio management and analysis tools are an effective first step in helping organizations understand their applications and prioritize modernization efforts. Additionally, the individual developer requires in-depth insights that go beyond metrics and visualizations to analyzing the impact of each change request.

An ability to analyze and establish knowledge of mature technology across your whole zEnterprise portfolio is essential in the efficiency of an organization. It will form a bridge between absent understanding of existing applications and knowing how to modernize them, and knowing what to change in order to modernize. This will clear a path to innovation that is as smart as it is efficient.

Micro Focus Enterprise Analyzer delivers these vital insights, turning the individual developer into an application expert and increasing overall productivity by 40% and beyond.

Figure 7: Change Analyzer performs field level impact analysis
Resources

1. IT Debt: the backlog of work IT has been requested to undertake by the business, which has yet to be actioned.

2. IT Application Portfolio Management (APM) is a practice that has emerged in mid to large size Information Technology (IT) organizations since the mid-1990s. Application Portfolio Management attempts to use the lessons of financial portfolio management to justify and measure the financial benefits of each application, in comparison to the costs of the application’s maintenance and operations.

3. If that convention isn’t readily known, a useful method is to locate the output record containing at least one data item that depends on the ICD9 value, then run “backward impact analysis”, until finding the first ICD-related field.

4. Source: Micro Focus customer project results.

About Micro Focus

Micro Focus, a member of the FTSE 250, provides innovative software that allows companies to dramatically improve the business value of their enterprise applications. Micro Focus Enterprise Application Modernization, Testing and Management software enables customers’ business applications to respond rapidly to market changes and embrace modern architectures with reduced cost and risk.

For additional information please visit: www.microfocus.com