asg technologies®

IT SYSTEMS MANAGEMENT

ASG-EXISTING SYSTEM WORKBENCH

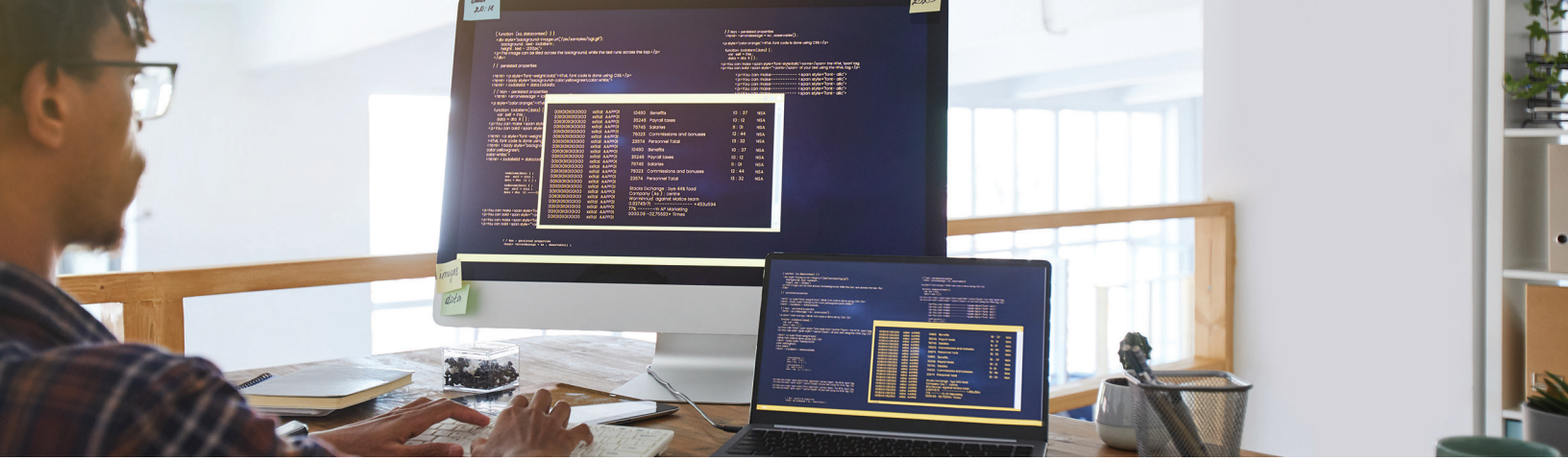
Enhanced Productivity for COBOL Developers
in the 21st Century

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

TABLE OF CONTENTS

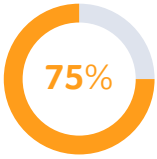


EXECUTIVE SUMMARY

The impact of the 2020 Coronavirus pandemic on mainframe systems has underscored how critical COBOL is to the functioning of many of the world's government agencies and enterprises and how extensively the language is used. It also highlights the challenge of managing COBOL code in an environment where the demand for change, and the consequential demand on the skills pool, is increasing beyond current supply. This results in a resource gap with a potential commercial impact that enterprises need to address. Moving away from COBOL is one answer, though it carries with it a material degree of risk, particularly if done in haste. Making such a move is difficult and costly. The better answer for many is to manage COBOL more efficiently.

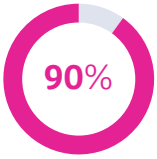
The risk, noted above, results from how vast and complex COBOL programs have become over time, such that more of a skilled expert's time is consumed in just gaining an understanding of the existing code rather than making necessary changes. The cost of errors in changed code can be significant, in terms of money spent on fixing those errors and damage to the organization's reputation if customers are affected.

ASG's COBOL analysis and understanding software (ASG-Existing Systems Workbench, or ASG-ESW) mitigates those risks. In this discussion, we describe how these products can help manage COBOL maintenance during the software development life cycle and illustrate the benefits to be gained by their deployment:



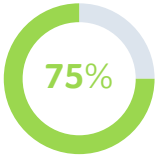
APPLICATION-LEVEL ANALYSIS

Application-level analysis and understanding tools for COBOL have been used to reduce analysis of data usage across an entire application by **up to 75%.**



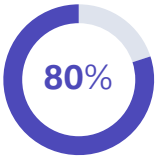
TOOLS FOR PROGRAM CHANGE

A COBOL intelligent editor for the ISPF environment can reduce the time and effort required during changes to COBOL application programs by **up to 90%**.



PROGRAM-LEVEL CODE UNDERSTANDING

Program-level analysis and understanding tools for COBOL have been used to reduce logic analysis by up to 50% and data usage analysis by **up to 75%**.



TOOLS FOR APPLICATION TESTING

Interactive testing and debugging tools with code coverage analysis can make savings during testing **as high as 80%** with improved accuracy and confidence that testing is complete.

To conclude, use of ASG's ESW products at proper points in the software development life cycle can help organizations realize an overall productivity gain of 20% or more. This helps organizations meet the parallel challenges of maintaining existing systems while developing new initiatives with the existing skilled resource.

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

INTRODUCTION

COBOL is still as relevant today, as we move into the third decade of the 21st century, as it has been for the last 50-plus years. The economy of many organizations is dependent on COBOL, and critical COBOL applications are still driving the activities of prominent businesses who spend many millions on development and maintenance of these applications. The demand for COBOL-skilled resources is as high as ever, but there is a limited supply of those resources. The sheer cost and effort to replace COBOL applications with something else rarely produces an acceptable return on the investment required. Furthermore, as we progress in the 21st century, new IBM software can take these COBOL applications to the cloud, making it even more flexible to handle new technologies and practices.

Today though, COBOL is a hot topic in news stories about the status of government IT issues. A portion of these stories focus on the age of the COBOL language, and while it is true that COBOL has recently passed its 60th birthday, the "old age of COBOL" is fake news! New COBOL compilers are released every few years to keep up with the computer hardware. The vast bulk of enterprise application code today is still written in COBOL and continues to be written in COBOL. Due to lack of knowledge, many current media reports combine refer to COBOL as old in a negative way in order to sell an incorrect story that "old" must be a bad thing. So, to understand the real problem, we must take a wider look at the issue.

Let us start by looking at what the underlying computer programs do. The programs simply apply business rules to the data to produce useful output. Governmental regulations supply the rules, in some cases such as the recent national pandemic stimulus packages, and such regulations can and do sometimes change both suddenly and dramatically. So, the developers must make significant changes to the code in a hurry to fit the code to the new rules. It does not matter whether the programs are new or old, written using COBOL, Java, or any other language; the changes must be made to include the new rules. So why isn't this simple? The problem is that the current developers are not the original authors of the programs, and they do not always fully understand the code in order to make the changes necessary for the new regulations to be applied. This problem is very often true, whatever the programming language that has been used, and is not a problem unique to programs written in COBOL. But because so much code is written in COBOL (more than all the other languages put together), it is making headline news.

Much of this code was developed over many years with many authors involved. Good documentation practices may not have been enforced and there may be significant technical debt caused by choosing an easy solution instead of using a better approach that would have taken longer. Choosing an easy solution typically happened because more attention was paid to expediently adding new features and less so to improving the design of the code, or refactoring. At the time, this was perhaps less important because the authors themselves understood the code, understood the changes, tested them and verified the results. Now, however, many of these original authors have moved on by advancement or retirement and are no longer available to perform new modifications. The people who are left, often have difficulty understanding the code in order to make the new required changes.

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

So, we are in the situation where we have the code built over a period of years by a variety of engineers, the new regulations and current developers. What is needed to effect the changes? The answer is "understanding." The current developers need to overcome a legacy of technical debt and understand the code so that they can adapt it to the new regulations as quickly as possible.

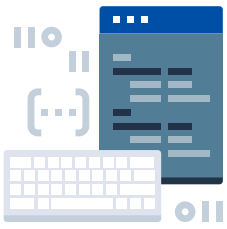
The purpose of this discussion is to help guide professional information technology managers, project leaders and developers who may be faced with these kinds of problems and are looking for ways to overcome their challenges. Demand for skilled resources has unexpectedly spiked in recent months, the supply is finite, replacement of COBOL applications is difficult and risky, and a good solution is to find ways to manage COBOL more efficiently – by improving productivity. Here, we will introduce ASG's Existing Systems Workbench (ESW) and describe how the different facilities can address the need for understanding at different stages of the lifecycle and illustrate the productivity gains that can be found.

MAINFRAMES AND COBOL TODAY

Mainframe computing and COBOL remain distinctly relevant today, regardless of pundits' comments:

MAINFRAMES STATISTICS

- 92 of the world's top 100 banks use mainframes
- 23 of the world's top 25 retailers use mainframes
- All of the world's top 10 insurers use mainframes
- 71% of Fortune 500 companies use mainframes
- Mainframes run 30 billion transactions per day
- Mainframes hold 80% of the world's business data
- Mainframes handle over 85% of all credit card transactions
- Mainframes process over 29 billion ATM transactions annually.
- Mainframes handle 68% of IT Production Workloads, yet account for just 6% of IT costs.



COBOL STATISTICS

- 90% of Fortune 500 business systems today are supported by COBOL
- 65% of active code today is COBOL
- 200 billion lines of COBOL are in use today
- 70% of all critical business logic and information are written in COBOL
- Between 60% and 80% of all transactions conducted worldwide are done in COBOL.

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

THE UNDERSTANDING GAP

Traditional application systems have been developed over many years and represent a considerable investment by organizations. Just like equipment and facilities, these application systems are important assets to the organization, are essential to the day-to-day operation and are very often at the heart of the enterprise's ability to provide unique value to customers.

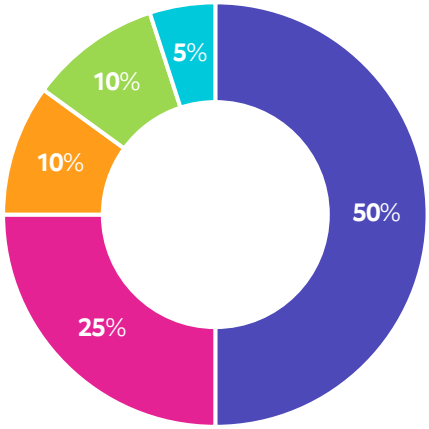
Maintaining this investment can be a significant cost in the organization's budget. IT departments must balance maintenance of existing systems against the development of new initiatives. The existing systems are what keep the lights on, and the new initiatives are where business growth comes from. This is a constant "tug-of-war."

It is important that maintenance of existing systems demands the attention of the people that understand it, but often the original authors have moved on either to new roles or have left the organization. Other challenges are that these systems may be poorly structured and complex, and there may be little or no current documentation.

By far, the largest effort goes into studying and understanding the existing code, the data and how that data is processed by the code. To understand why this is the case, if two or three different developers are given the same programming requirement, each developer will present a differently coded program. This is because each one uses different coding techniques to achieve the desired result. Some will be more complex and convoluted than others, while all still achieve the desired outcome.

When a program is changed, complexity inevitably increases unless the organization proactively works against this, a process known as software entropy. After many years of change after change to what was originally well-structured programs, and these changes reflecting different standards or evolution of best practices, the understanding difficulty is significantly increased. This difficulty in understanding is not a problem unique to COBOL, it is equally true for any programming language used for an application that is subject to being changed.

Studies have shown that just trying to understand a program can take up to 50% of the effort when making a change. Planning and making code changes is a relatively small part of the overall effort. The amount of time required by testing depends largely on the quality of the code changes, which depends very much on how well the program has been understood before new changes are coded.



WHERE THE EFFORT GOES

- **Understanding Code** (50%)
- **Testing Changes** (25%)
- **Planning Changes** (10%)
- **Making Changes** (10%)
- **Documenting Changes** (5%)

The ramifications of errors made early in the life cycle can be significant, with up to a tenfold increase in the cost of correcting an error the further along in the development life cycle a project is before the error is identified.

For example, if an error costs 10 units of work to correct when encountered in the analysis phase due to a failure in proper understanding, it could cost up to 100 units in the design phase, 1,000 units in the coding phase, 10,000 units in test phase and up to 100,000 units if not found until the production phase. Worse, an error that propagates all the way through to production can have a significant negative impact to the business or customer satisfaction.

Examples of this that have been observed include:

- A service charge calculation that was inadvertently bypassed for several months.
- A major distribution warehouse that was rendered inactive for over eight hours just before the busiest shopping day before Christmas, leaving stores short on stock
- An instance where the inability to debit customers for deliveries extended for two days.

There is a lot of industry focus on testing and testing tools but much less on tools that help understand the code in the first place. Using development tools earlier during the software development life cycle to support the phase that consumes the greatest part of the effort will greatly reduce errors from occurring during testing and in production. This ensures that the overall quality of programs implemented will be improved.

INTRODUCING ASG EXISTING SYSTEMS WORKBENCH

ASG-Existing Systems Workbench (ESW) is a suite of products specifically designed to address the problems of maintaining and changing COBOL code.

The key ingredient in ASG-ESW, and related products, is a feature called “COBOL Program Intelligence.” The Program Intelligence embedded in the ESW components can provide enhanced COBOL code understanding, which will allow a developer who is new to a program to better understand the code in a minimal amount of time. It can assist with making new changes to the code, and it can assist in testing those changes to verify that the changes produce the intended results, again, in a minimal amount of time. It provides this assistance using multiple components that cover all stages of the program Development life cycle.

ASG-ESW is composed of the following components:

- ASG-Insight
- ASG-SmartEdit
- ASG-Encore
- ASG-Alliance
- ASG-SmartDoc
- ASG-SmartTest
- ASG-Recap

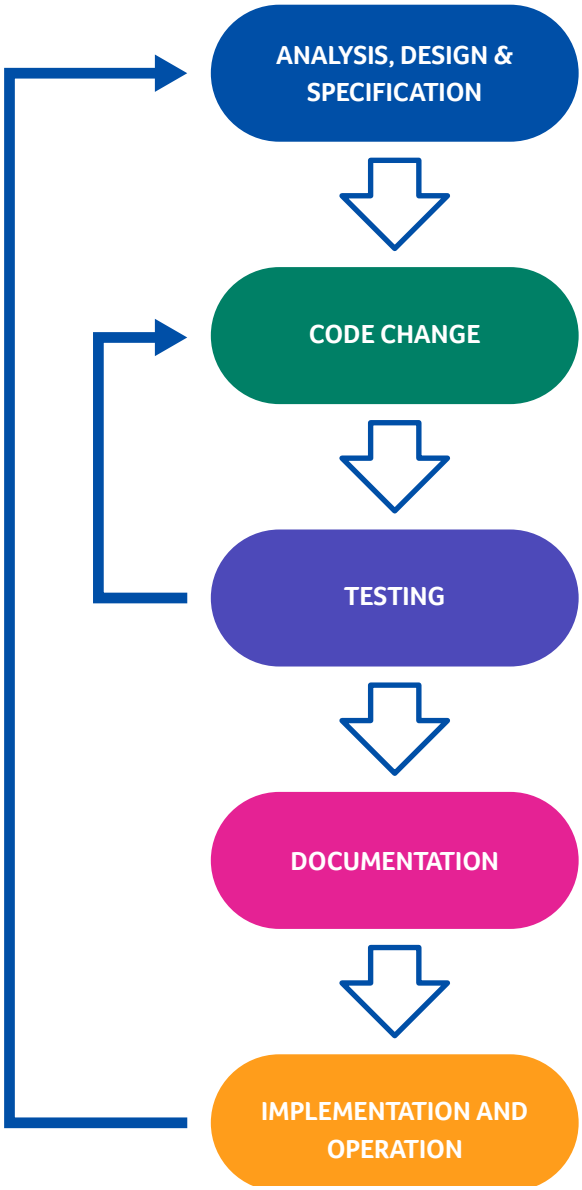
ASG also has other related products to assist the COBOL developer. These are ASG-SmartScope, ASG-SmartFile, ASG-TriTune, ASG-becubic, and ASG-Life Cycle Manager (LCM).

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

Where these products are applicable in the development/maintenance life cycle is shown in the following diagram.



ASG-Insight, ASG-SmartDoc, ASG-Alliance, ASG-Recap
(related: ASG-becubic)

ASG-SmartEdit, ASG-Encore
(related: ASG-SmartFile)

ASG-SmartTest
(related: ASG-SmartFile, ASG-SmartScope, ASG-TriTune, ASG-ValidDate)

ASG-SmartDoc, ASG-Recap, ASG-Alliance
(related: ASG-becubic)

(related: ASG-LCM, ASG-TriTune)

COBOL UNDERSTANDING WITH ESW COMPONENTS

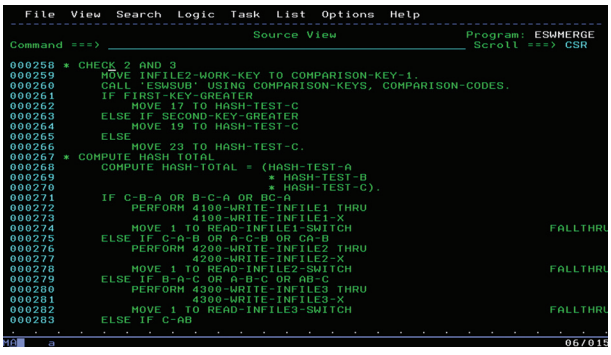
A typical way of viewing program code is through an editor (or viewer) that displays the lines of code in the sequence in which they are written.

ASG-ESW has an analytical engine that analyzes the COBOL code and stores the results of this analysis in an Application Knowledge Repository (AKR). The analytical engine connects each element of a line of code not only to the other elements on the line but also to other lines of code. So, for example the analytical engine recognizes where a data element is updated within the program and recognizes every other reference to that data element even if that data element is named differently. The analytical engine also recognizes the sequence by which the program code is executed (which may not be the same as the sequence it is coded). It is then able to recognize the destination statement executed as the result of tests or other conditional statements. External references across programs are identified where data is passed between one program and another program.

CODE VIEWING WITHOUT ESW

Here we see a code snippet as a developer would see it via a typical code editor or code viewer.

This is the normal view. To work out which section of code follows which other section, which section of code this belongs to etc., the developer must scroll through the lines of code, typically making handwritten notes, checking where a test causes the program to take a different path, following returns, and so on. This is very time consuming and prone to errors and oversights. To discover where data is used and updated in programs, the developer must take note of the input name, find statements using that name, repeat those finds ensuring that any renames or group references are remembered and possibly look up program copybooks in a separate viewing screen. During this process, the developer must take copious handwritten notes, which again is very time consuming and prone to further errors and oversight.



asg technologies®

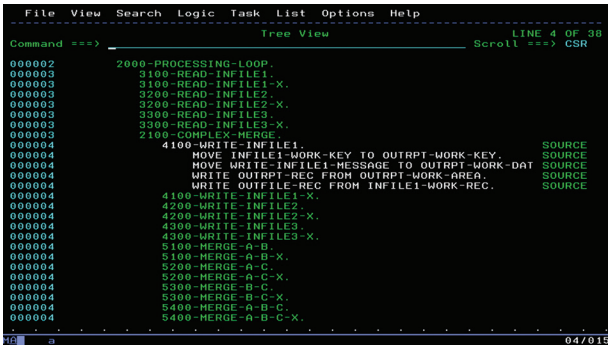
ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

CODE VIEWING WITH ESW

During analysis of the program code by ASG-ESW, the developer may wish to see which sections of code follow which other section. This is easy with the ASG-ESW products.

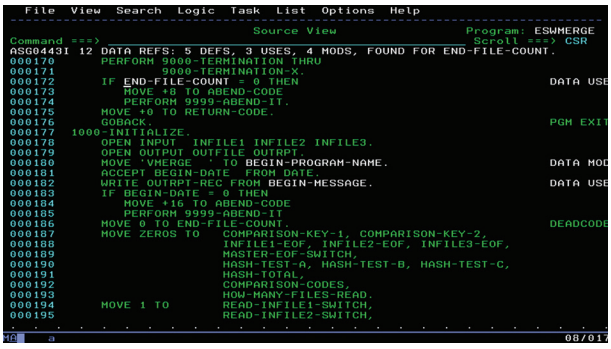
The results of analysis by the ASG-ESW Analytical Engine allow the product to sequence the display that shows each section of code placed correctly in its logical position in the program. The developer can select any section and view just the program code for that section. A simple command allows the developer to switch between normal and the tree view at any time.



With another command in ASG-ESW, the developer can easily discover all the instances in a program where data is used and updated regardless of name changes or definition level.

CODE SEARCHING WITH ESW

Performing a search on a data name reveals there are 12 references with five definitions, three uses and four modifications. The multiple definitions are the result of the practice in COBOL of defining data elements as part of group definitions and then referencing the data area at both the individual definition level and at the group level. In the example shown here, the target of the search item “END-FILE-COUNT” is defined in a redefined group item as shown below.




```
01 FIRST-LAST-LINE-AREA.
  05 BEGIN-MESSAGE.
    10 BEGIN-PROGRAM-NAME       PIC X(8).
    10 FILLER                PIC X(6).
    10 BEGIN-DATE             PIC 9(6).
  05 END-MESSAGE REDEFINES BEGIN-MESSAGE.
    10 END-FILE-COUNT          PIC 9(8).
    10 END-MESSAGE-TEXT        PIC X(10).
    10 FILLER                PIC X(2).
```

Using the COBOL Understanding technology, ASG-ESW is able recognize that the field “END-FILE-COUNT” occupies the same storage location as the definition “BEGIN-PROGRAM-NAME” and modification of this name will modify “END-FILE-COUNT.” ASG-ESW will display this for the developer as well as displaying references to the parent group level definitions. Using this kind of technology will reveal to the developer every statement in a program where a storage location is either defined, used or modified. Completing this kind of search without using technology requires repeated text searches and handwritten notes where the developer will manually link data definition to data usage by executable statements. Often these definitions are located in COBOL copybooks which necessitates opening a separate viewing screen. ASG-ESW brings together the program source code and the included COBOL copybooks into a single analysis result. ASG-ESW is also able to identify statements in the program that will not be executed, which are labelled as “DEADCODE.”

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

ESW PRODUCT FEATURES AND BENEFITS

- **ASG-Insight™** addresses the issue of code understanding by completely automating the program investigation process. It slashes the time needed for program understanding, thus enabling more effective management of existing systems.

- **ASG-SmartDoc™** is a unique static analysis and documentation generator that completely automates this important process. It provides valuable information for routine maintenance or planning and conducting program re-engineering, system consolidation and migration projects.

- **ASG-SmartEdit™** is a unique interactive COBOL editor for ISPF that adds automation and COBOL intelligence to the code change process. Its powerful facilities enable fast response to change requests, and high confidence in the results.

- **The ASG-SmartTest™** family of testing and debugging tools provides language intelligence with automated testing and debugging facilities to help accelerate the return of application programs to production.

  The ASG-SmartTest family includes support for COBOL, Assembler and PL/I programs running in TSO, Batch, ISPF, CICS and IMS environments. ASG-SmartTest products provide a single CUA compliant user interface to support all languages and environments, increasing productivity and reducing the learning curves associated with disparate products.

  ASG-SmartTest is an integral part of ASG-ESW. ASG-SmartTest leverages the power of ASG-ESW to provide the best in class testing and debugging solution.

- **ASG-Encore™** is a component of ESW that is an integrated re-engineering environment for COBOL programs facilitating program code refactoring. ASG-Encore includes analysis facilities and allows you to extract code based on the most frequently used re-engineering criteria. The code generation facilities allow you to use the results of the extract to generate a standalone program, a callable module, a complement module and a CICS server. Prior to code generation, you can view and modify the extracted Logic Segment using the COBOL editor.

- **ASG-Recap™** assists IS professionals in conducting a portfolio analysis of their COBOL applications. ASG-Recap measures how well COBOL applications are designed and implemented from a technology viewpoint (this is referred to as technical quality). ASG-Recap reports provide function point analysis and metrics information, program quality assessments, intra-application and inter-application comparisons and summaries and historical reporting of function point and metrics information. You can view this information interactively or export it to a database, spreadsheet or graphics package.

- **ASG-Alliance™** is the application-understanding component of ESW that is used by IT professionals to conduct an analysis of every application in their environment. ASG-Alliance supports the analysis and assessment of the impact of change requests upon an entire application.

  ASG-Alliance allows the developer or analyst to accurately perform application analysis tasks in a fraction of the time it would take to perform these tasks without an automated analysis tool. The impact analysis from ASG-Alliance provides application management with additional information for determining the resources required for application changes.

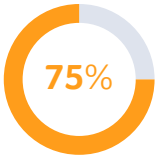
asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

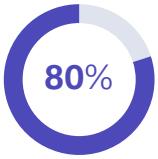
ASG-ESW BENEFITS AND ROI

ASG-ESW will deliver the best ROI when it is implemented as part of an overall application management process, with savings of:



Up to 75% on application level analysis time

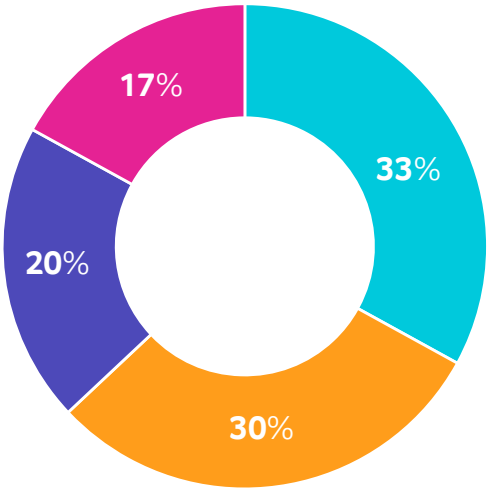
Up to 50% on program code analysis



Up to 80% on COBOL code syntax checking

Between 50% and 80% on COBOL code refactoring

Taking a closer look at different activities during application maintenance and the savings that can be achieved, industry analysts break down application maintenance into four primary activities. This chart shows the average time and effort required for each maintenance activity required to support existing applications.



APPLICATION MAINTENANCE ACTIVITY

- **Analysis** (33%)
- **Testing** (30%)
- **Charge** (20%)
- **Deployment** (17%)

ESW tools provide support for the following application maintenance activities:

Application-Level Analysis (ASG-Alliance and ASG-Recap):

Data is stored in files and databases, which are then used by programs and utilities across an application. Analyzing this usage is a complex task requiring examination of multiple components across the application. ASG-Alliance gathers metadata from an entire application to build an Application Knowledge Repository (AKR), and from this provides application level analysis and understanding. ASG-Alliance has been used to reduce data use analysis across an entire application by as much as 75%.

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

ASG-Recap provides application-level analysis of program metrics and anomalies. You cannot monitor what you cannot measure. It is known that increased code complexity requires more effort to maintain and change. ASG-Recap provides the measurements that show code complexity, e.g. Cyclomatic Complexity, Program Size. Likewise, it shows code quality, e.g., Dead Code, Live Exits or Recursion. Programs that have a poor metric or indicate program anomalies can be singled out for proactive maintenance activities and code refactoring.

Program-Level Code Understanding (ASG-Insight and SmartDoc):

Understanding program logic flows can be a complex task especially in large programs. ASG-Insight has been used to reduce program logic analysis and data usage analysis typically by:

- Logic analysis: 50%
- Data usage analysis: 75%

The combination of ASG-Alliance and ASG-Insight provides a feature-rich application understanding and analysis toolset that will reduce application analysis effort by as much as 50%.

ASG-SmartDoc automatically provides current COBOL documentation required by quality assurance for turnover and deployment. ASG-SmartDoc reduces the time and effort required to understand and document a COBOL application program by as much as 90%.

Changing Programs (ASG-SmartEdit and ASG-Encore):

ASG-SmartEdit provides a COBOL intelligent editor for the ISPF environment and will reduce the time and effort required to make changes to COBOL application programs. Some typical savings include:

- Find impacted data: 95%
- Check COBOL syntax: 80%
- Display copybooks: 75%
- Code walk through: 60%

ASG-Encore provides the ability to leverage existing COBOL application programs by providing code extraction features. ASG-Encore can extract valuable business functions from COBOL to create COBOL components that can be used by existing applications or in new development initiatives. Some typical savings include:

- File I/O extraction: 90%
- User interface extraction: 50%
- COBOL segment extraction: 85%
- Data item extraction: 60%

Application Testing (ASG-SmartTest):

ASG-SmartTest provides interactive testing and debugging functions. It enables the user to control program execution and view changes to data as the program executes. Some typical savings include:

- Diagnosing program exceptions (0C1, 0C7, and 0C4s): 80%
- Diagnosing data corruption errors: 70%
- Diagnosing program logic errors: 60%

The illustrated time and effort savings were gathered by ASG's Professional Services organization and were based on their experience over a three-year period and included many varied projects. The best results were reported by organizations that implemented the ESW product suite into their application maintenance process and provided ongoing training opportunities for their developers.

asg technologies®

ASG-EXISTING SYSTEM WORKBENCH

Enhanced Productivity for COBOL Developers in the 21st Century

CONCLUSION

Industry studies estimate that 80% of IT budget is allocated to maintenance and support of existing systems and applications. **This leaves just 20% for new initiatives.**

The goal of the ESW product line is to reduce the resources required to maintain traditional applications while at the same time improving quality. This empowers our customers to use liberated resources on new initiatives that help them move ahead of their competition.

ASG-ESW provides automated COBOL Program Intelligence across the application development lifecycle to assist developers in understanding COBOL code more fully, which leads to improved productivity, reducing the resources required for maintenance of existing applications. The tools provided by ESW perform at their very best as part of an integrated application management process.

In a team of just ten application developers, a 20% improvement in productivity is like adding two new resources for new initiatives—for free!

Contact ASG to learn more about how ASG-ESW can help your organization overcome the growing challenges with helping newer and fewer developers understand critical COBOL applications.

FOLLOW US

www.asg.com

ASG Technologies is a global software company providing the only integrated platform and flexible end to end solution for the information powered enterprise. ASG is the only solutions provider for both Information Management and IT Systems and has over 3,500 customers worldwide. To learn more visit www.asg.com.

ASG Technologies | 1.239.435.2200 or 1.800.932.5536 | 708 Goodlette Road North, Naples, Florida USA 34102 | www.asg.com

Whitepaper-ASG-ESW_20201014en