

A Five Step Approach to Protecting and Optimizing Software Investments

A SourceIQ White Paper

October 2004

Written By Ian S. Hayes, President, Clarity Consulting, Inc.

sourceIQ

TABLE OF CONTENTS

Section	Page
Introduction	3
Solution	5
Approach	6
Inventory	7
Control	7
Measure	8
Improve	9
Optimize	10
Conclusion	11
Acronym Glossary.....	12
References	13
About SourceIQ	14
About Clarity Consulting and Ian Hayes	14

© Copyright 2004 SourceIQ, Inc. All rights reserved. No portion of this document may be reproduced without authorized written permission of SourceIQ, Inc.

SourceIQ – A Five Step Approach to Protecting and Optimizing Software Investments

Introduction

Whether you are an IT or business executive, the challenge of creating, managing and operating software directly impacts you and your organization. Software brings enormous benefits -- facilitating key business processes, enabling new products and services, improving our productivity and enhancing competitiveness -- but our growing dependence on software has a price. We devote an unprecedented share of our budgets to developing, maintaining and supporting software, and then run the risk of suffering costly, high impact errors and outages from malfunctions. We must continually monitor and protect our software from intentional and unintentional harm. And, we are often hamstrung in our ability to react quickly to new opportunities and threats by the difficulty of adapting our existing software.

Complexity is partly to blame for this state of affairs. Today's software is far more complex than its predecessors. A typical business process relies upon numerous applications from many sources to automate activities and exchange critical data. These applications have an increasing number of "moving parts" -- program code, objects, components, documentation, test cases, spreadsheets and other scaffolding and supporting functionality. Watts Humphrey of the Software Engineering Institute notes that the size of software used for any given function has been growing by roughly 10 times every 5 years.¹ These parts may be new or old, unique or shared among many systems, purchased or custom developed, supported internally or outsourced onshore or offshore. The complexity and volume of parts makes supporting and modifying software difficult, error prone and expensive. There are more breakage points to compromise stability, a higher likelihood of introducing errors and greater difficulty to audit and verify results. Not surprisingly, a study performed by Capers Jones found that the risk of project schedule delays and outright cancellations is directly related to the size and complexity of the application being developed.² With duplicate, overlapping and obsolete parts and versions abounding, simply finding the right pieces for a given application or product is a daunting chore.

Challenges aside, companies still need to produce and maintain software, and they need to do it more productively. Competitive and financial pressures demand that software be delivered faster and cheaper, yet our productivity in doing so lags significantly behind the leaps made in hardware performance. We must find innovative ways to improve productivity and focus development efforts on activities that provide the greatest business value. We must eliminate time-robbing overhead and low-value efforts, and reduce the 40 to 50% of development effort spent on avoidable reworks.³ We need to be able to benchmark our productivity and compare it across internal, onshore and offshore sourcing options. We need tools and controls to ensure the quality and integrity of the software produced by dispersed teams.

Source code is at the heart of all of these issues. It is the medium for translating business logic and requirements into computer executable software. **It is an irreplaceable repository of accumulated business knowledge, and often the only accurate and complete documentation of the rules by which the business functions.** A company's portfolio of source code represents an enormous investment of human and financial resources, is a valuable asset in its own right, and a crucial competitive differentiator.

Source code is both the embodiment of the challenges described above and the key to solving them. By improving source code quality, and the development and management processes around it, companies can better direct investments to high ROI areas, improve their productivity in delivering and maintaining software, and lower their associated costs and risks. Undoubtedly, your company already has tools for managing and protecting its source code investments. This white paper, however, proposes a new approach to understanding and managing your critical source code investments, one that can yield substantial improvements in both IT and business performance.

Solution

Traditionally, companies have taken a minimalist approach to managing their source code, consolidating code assets into shared libraries and/or Software Configuration Management (SCM) products and relying on developers to use control procedures when accessing those assets. When scrupulously followed, this approach provides a measure of protection and change auditability but does not address the multitude of issues affecting software costs and reliability. Taking a higher level, long-term view of “how, where, when and by whom” source code is used, provides companies with a wealth of information that:

- significantly strengthens monitoring and compliance efforts,
- highlights ways to improve quality and stability,
- exposes potential human and technical risks
- provides insights on how to obtain breakthrough improvements in development productivity,
- ensures that financial and people resources are directed to areas where they provide the greatest value, and
- accelerates internal and external knowledge transfer.

SourceIQ’s flagship product, **Source Code Portfolio Manager (SCPM)** helps companies achieve this higher-level view. SCPM provides a comprehensive portal for analyzing, managing and reporting on a company’s portfolio of source code assets, enabling developers, QA engineers, users and executives to optimize the creation, maintenance and business value of those assets. Through its seamless integration with existing SCM and version control tools, SCPM offers a consistent model for measuring and monitoring source code located anywhere in a company’s portfolio as it progresses through the software development lifecycle. Moreover, SCPM’s deep, broad analytical and dashboard capabilities can be personalized to meet the needs of stakeholders throughout the organization, allowing critical information to be viewed on a real-time or historic basis, and organized and analyzed at departmental, team, project, application or individual levels.

A fundamentally different approach to source code asset management, SCPM gives executives and managers an entirely new set of capabilities to address their software challenges. SCPM can provide immediate benefit by solving current source code management issues; however, it can deliver even greater value when its sophisticated capabilities are harnessed to improve the productivity and business alignment of development efforts.

Approach

SourceIQ views source code management as the best means of protecting and optimizing software investments as well as a foundation for improving the processes used to develop, enhance and support software. SourceIQ’s approach for implementing optimal source code management relies on a combination of five increasingly powerful levels of capabilities as shown in the figure 1 below. These levels build from basic capabilities -- creating a complete inventory of source code assets and establishing control over multiple SCM environments – to more sophisticated capabilities for improving quality, productivity and business alignment. Each capability builds upon its predecessors and offers an ever larger set of benefits. SCPM supports actions in all five levels and provides the platform and insights to enable companies to progress to higher capabilities levels.

Companies differ widely in terms of benefit needs and existing source code management capabilities. You can establish your company’s position within this model based on the strength of your current source code management tools and practices, and your specific improvement SLAs. You can use SCPM to strengthen and gain greater benefits from your current capability level, and you can leverage SCPM to move to higher levels of capabilities at your own pace.

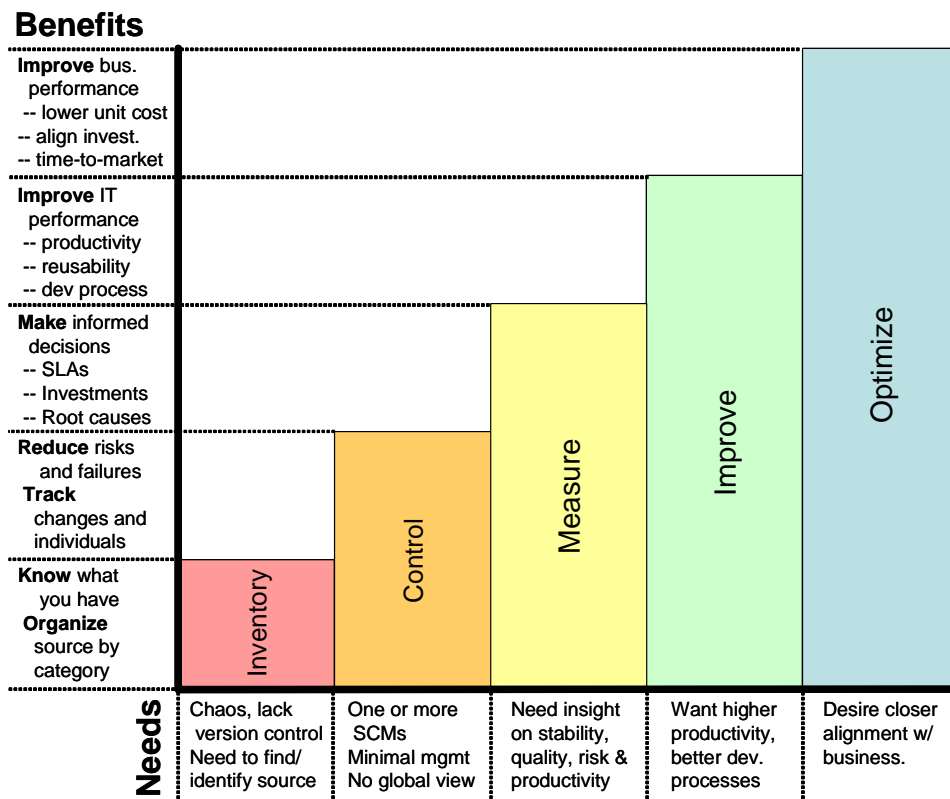


Figure 1 – SCPM’s Five Levels of Source Code Management Capabilities

1. Inventory – *Set the Foundation*

Goal: Find and organize your organization's source code

The first step to optimal source code management is creating an inventory of those assets. As shown by past Year 2000 remediation efforts, this step is often surprisingly difficult to accomplish. Source code assets exist in many versions and are typically scattered across SCM tools, shared libraries and developer's personal files. Components may also be mismatched, missing or incomplete. Capturing non-traditional code assets, such as Excel files, is also important, yet standard version control tools and practices often overlook them. The next step is to organize and categorize these code components into more meaningful higher level units, such as applications, products, services, processes, etc. for reporting and management purposes. This categorization identifies version dependencies, uncovers components shared by multiple applications, enables impact analysis and ensures all necessary components can be found when modifications are required.

An organization can implement the inventory step tactically or strategically. If your organization frequently loses time or suffers from production errors due to missing source code or version mismatches, the inventory process and basic source code controls will immediately cut development and production failure costs and lower risks of future failures. Developing an inventory is also a valuable tactic to ensure that all necessary source components are found and turned over in the case of an application outsourcing engagement or as part of a merger or acquisition. Strategically, a complete inventory is the foundation for elevating all source code management practices, and if all source code is included, provides a company with a full view of its entire source code investment.

SCPM facilitates all aspects of this step, capturing information from network files and SCMs if they exist. Where version control systems are absent, SCPM puts business logic under active management ensuring back up/recovery and audit trails for source code assets. SCPM's metadata capabilities enable organizations to categorize and track source in any necessary manner.

2. Control – *Manage Components*

Goal: Institutionalize active management of source code

Companies at this level typically have version control practices in place and one or more SCMs deployed. Specific practices tend to vary by team, and the company lacks a global view across all of its source code assets. Cross team and SCM insight is critical for compliance efforts (such as providing audit trails for Sarbanes-Oxley); it ensures the integrity of business processes that involve multiple applications from multiple teams, and supports outsourced/offshored development where work efforts are distributed across teams and time zones. By providing non-disruptive integration with existing SCMs, SCPM centralizes source code management while allowing teams to continue using existing tools and practices. This capability provides transparency across all development and support efforts and establishes a chain of connection between development teams, applications and business processes.

Once the connections are in place, SCPM's analysis and reporting capabilities enable active control and management of source code, controlling access to source code, capturing information on who is changing what code, creating an audit trail and identifying potential risks. Managers can confirm code origination before sign-off, flag non-qualified code insertions and identify unnecessary churn in the code base. These controls are especially important when managing efforts that cross organizational boundaries and involve multiple versions of source code for development, testing, QA and production. SCPM helps to identify and prevent code conflicts and incompatibilities. It highlights potential risks resulting from hasty changes such as emergency production fixes and last minute modifications to pending software releases. Complete control and reporting avoids surprises and minimizes risk when developing, maintaining, testing and upgrading applications. The result is higher levels of business application availability from more stable applications, fewer reworks and production errors, and lower risk from accidental and malicious code modifications.

3. Measure – *Gain Insight*

Goal: Find issues and opportunities

The ability to use metrics to gain factual information to better manage software portfolios and software development and support processes provides breakthrough benefits, and is a key differentiator between SCPM and standard methods of source code management. The previous levels solve immediate issues by establishing foundational controls, but metrics enable proactive management and targeted improvements to achieve sustainable, long-term gains in productivity, quality, and cost efficiency.

By sitting above and across existing SCMs and centralizing and aggregating information from many sources, SCPM delivers a big picture view of performance across teams, applications and locations. Through a series of customized tools, dashboards and reports, it provides objective insights into areas such as development team productivity and code quality, allowing users to identify and assess risks early on, verify their perceptions with factual data, and target efforts to the areas that will provide the greatest benefit.

Monitoring code changes provides a means of measuring productivity, which in turn is useful for evaluating performance, comparing sourcing alternatives, identifying bottlenecks and setting performance objectives for outsourcing service level agreements (SLAs). As shown in figure 2, productivity numbers enable managers to monitor and compare progress by teams, individuals and delivery partners. Knowledge of historical performance provides a benchmark for SLAs, and is a predictor of future performance. It also helps minimize unexpected events and issues. For example, if a team has historically delivered an average of 350 lines of fully debugged code (LOCs) per week, is it reasonable to assume that the team could produce the 800+ LOC per week needed to meet a looming deadline? Should the manager add resources to boost productivity or shift the deadline? Productivity information is also useful for:

- Identifying and understanding dependence on specific individuals
- Comparing the relative cost/performance of offshore development to internal development
- Validating progress against development milestones

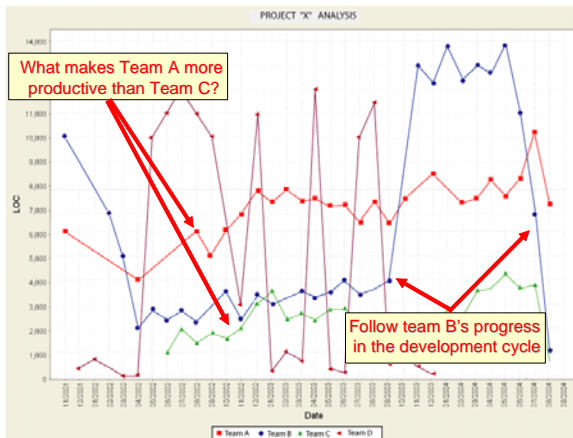


Figure 2: Productivity Comparison



Figure 3: Evaluating Release Stability

Equally important is the ability to measure volatility, which is the level of change to a unit of source code. Volatility is an indicator of quality, stability, risk and potential reusability, and is a valuable adjunct to application portfolio assessments. High volatility indicates “hot spots” of potential risk and targets for root cause assessments, code improvements and increased attention in test plans. Studies have shown that approximately 80% of defects come from 20% of the modules, about half the modules are error free, and the amount of change is a key contributor to error-proneness.³ As shown in figure 3, volatility, or the lack of it, is also a prime indicator of stability when deciding whether a software release is ready to ship. By monitoring volatility, SCPM can also alert managers when code that should be “locked down” isn’t. On the positive side, volatility can highlight particularly important modules as candidates for software reuse efforts.

4. Improve – Enhance Performance

Goal: Making meaningful improvements to the development process, individual and team productivity and code quality

This stage moves beyond basic source code management to improve the efficiency and effectiveness of IT and development processes, people productivity and utilization, and enhance the quality of the company’s application portfolio. SCPM provides abundant current and historical information to identify and rank improvement opportunities, and can verify if expected benefits were received after implementing the improvements. For example, past productivity trends provide a solid baseline for validating development estimates, while current data verifies whether those estimates are being met during project execution. If used correctly to improve the estimation process, the variance between estimates and actual performance should decrease over time, leading to greater predictability and more accurate project budgeting. Contrasting team and individual productivity is an excellent means for identifying, harvesting and propagating best practices, and highlights which tools, training, technologies and skill sets are more effective than others. Individuals can strengthen their own skills and estimating abilities by using SCPM to better understand their own productivity characteristics and support disciplined personal practice efforts such

as the Personal Software Process developed by Watts Humphrey in conjunction with the Software Engineering Institute.⁴

Reducing resources spent on activities that provide marginal benefit is a highly effective way of enhancing productivity and business value delivery. For instance, one research study shows that as little as 3% of the programs in a typical application account for as much as 80% of the overall costs of supporting that system.⁵ Improving the quality of this small percentage of programs should reduce support costs and provide significant financial and productivity benefits; spending resources on the remaining programs would offer little or even negative return on investment. Correlating volatility data (rate of change) with production defect data identifies lucrative opportunities to address root cause issues in code quality as well as in the processes used to develop, modify and test problematic modules. The result is fewer production errors, higher system stability and significant resource savings from reducing reworks.

5. Optimize – *Manage for the Long Term*

Goal: Institutionalize source code controls

The highest level in source code management is achieving alignment between source code investments and the business objectives for those investments. While few executives are interested in the technical aspects of source code, abstracting, organizing and presenting critical performance information in a graphical and metrics-oriented format permits meaningful communication between executives and technology professionals. Aggregating source code productivity and volatility impacts up to the business process level helps executives to make informed decisions on issues such as replacing existing systems, determining whether to request last minute development changes, approving additional resources, and investing in new tools, processes and the use of offshore outsourcing. SCPM provides solid data to support development estimates and budget requests as well as proof of performance by the IT organization and their partners.

SCPM supports strategic project and application portfolio management initiatives by supplying a means to connect technology investments and their manifestations in source code. It provides cost, quality and risk information that can be aggregated at various levels within a portfolio assessment. The connection between these initiatives allows companies to assign a specific business value to a unit of source code and use that information to drive decisions on reuse, additional investments and risk mitigation strategies. Business directed code reuse shortens time-to-market for new functionality, reduces risks and defects, and generates sustainable decreases in development and long-term support costs.⁶

Where the Improve level identifies and implements specific improvements to development practices, the Optimize level institutionalizes the monitoring, managing and enhancing of those practices as part of a metrics-based continuous improvement process. SCPM captures the historical data and provides the analytical capabilities to support continuous improvement efforts on its own or as part of formal SEI CMMi or Six Sigma initiatives.

In Conclusion

Source code is simply too valuable an asset to go without active management. From supporting mission-critical processes to enabling rapid response to high-value business opportunities, companies are utterly dependent on their source code. Further, effective source code management is the foundation for improving the processes by which software is developed, enhanced and maintained. Basic controls are essential to reduce operational risks, ensure production stability, safeguard intellectual property, and protect against malicious attacks. More sophisticated management is needed to support auditing and compliance efforts, manage and monitor distributed development and offshore outsourcing efforts, and improve development and support productivity. Improving software development and support processes reduces time-to-market for new functionality while cutting the total cost of ownership over the life of each software application. Finally and most importantly, companies need a means to extract greater business value from their existing application investments while responding more swiftly and cost effectively to competitive challenges, business opportunities and regulatory requirements.

Typical source code management approaches solve only the most basic issues and fail to capitalize on knowledge associated with the code and its development to gain high-value benefits in productivity, quality and improved business performance. Further, most companies use a mixture of isolated tools and practices that provide local benefit, but cannot provide seamless control across all assets or an overall view of development and support performance. Hence, they do little to reduce the complexity of source code management nor do they provide a means of addressing the challenges and risks of increasing levels of software integration.

By offering a new approach for measuring, monitoring and managing source code, SCPM capitalizes on your existing SCM investments to provide the capabilities and insight to gain breakthrough improvements in development and business performance. As shown in figure 4, SCPM supports all levels of source code management from collecting inventories and establishing basic controls to providing the measurements and insights to manage and continuously improve development practices, productivity and source code value. SCPM delivers immediate value and enables companies to capitalize on its sophisticated capabilities and accelerating benefits at their own pace. Regardless of your company's current sophistication in source code management, SCPM can improve your performance while reducing unit costs of development and support.

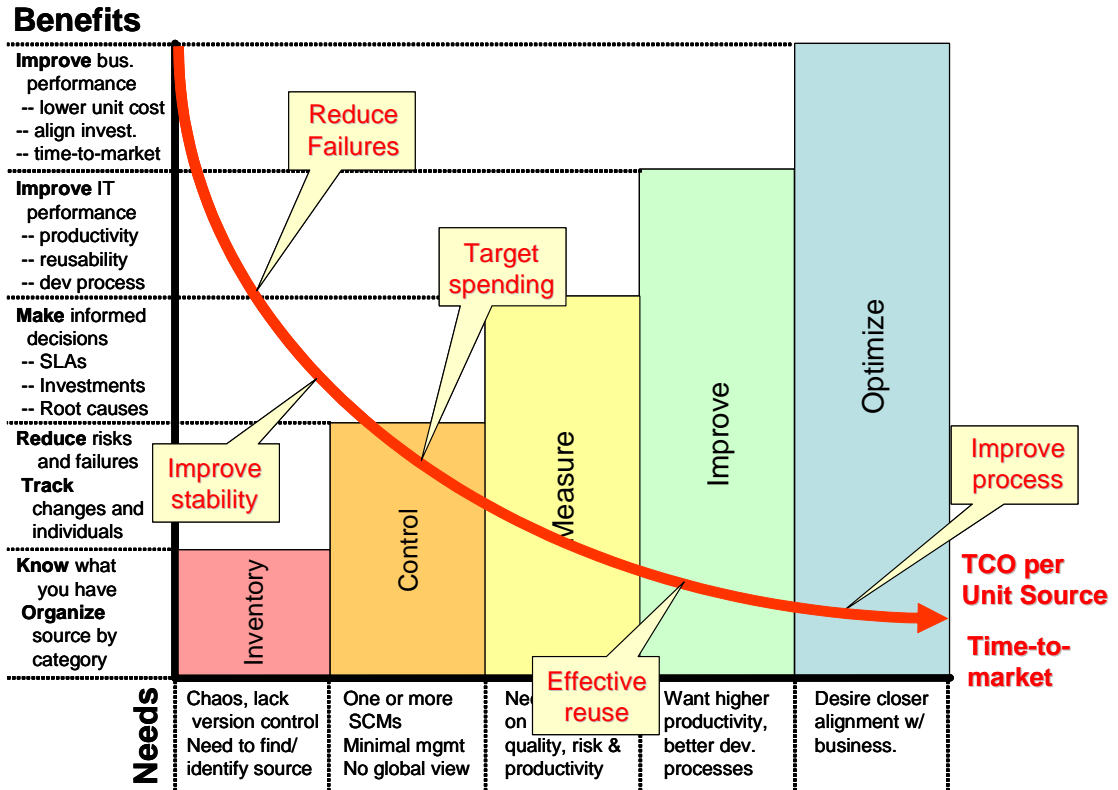


Figure 4 – SCPM’s Sophisticated Capabilities Result in Declining Total Cost of Ownership (TCO) per Unit of Source Code and Faster Time-to-Market

Acronym Glossary

- LOC -- Lines of Code
- SCM – Software Configuration Management
- SCPM – Source Code Portfolio Manager
- SLA – Service Level Agreement
- TCO – Total Cost of Ownership

References

1. “The Future of Software Engineering: Part 1,” Humphrey, Watts, published in The Watts New? Collection: Columns by the SEI’s Watts Humphrey, 2004, www.sei.cmu.edu/news-at-sei.
2. “Conflict and Litigation between Software Clients and Developers,” Jones, Capers, Version 8, September 2000
3. “Software Defect Reduction Top 10 List,” Barry Boehm and Victor R. Basili, January 2001, published by the Center for Empirically Based Software Engineering, www.cebase.org
- 4.. Information about the Personal Software Process is available at the Software Engineering Institute website, www.sei.org
5. “Improving Quality with Software Metrics,” Hayes, Ian S., Handbook of IS Management, Fifth Edition, Umbaugh, R.E., Editor, Auerbach, page 673
6. “Software reuse: From Library to Factory – Technical,” Griss, M.L., IBM Systems Journal 1993

About SourceIQ

SourceIQ is a fast-growing software and services firm with a mission to help companies understand their core business logic. Founded in 2002, SourceIQ is pioneering the development of software methodologies and applications that provide Global 2000 companies with strategic and tactical insights into their portfolio of source code assets.

About Clarity Consulting and Ian S. Hayes

Ian Hayes is President of Clarity Consulting, a management consulting and analyst firm that focuses on IT strategy and improving the business returns generated by IT investments. Ian has advised dozens of Fortune 1000 companies on how to increase the value of projects and services by better targeting IT investments, improving the effectiveness of IT execution, optimizing the sourcing of IT activities and establishing measurement programs that tie IT performance to business value delivered. Author of three books, he has chaired numerous industry conferences, and his articles have appeared in publications such as Business Week, Computerworld, Better Management, the Cutter IT Journal, AD/Trends, Enterprise Application Integration Journal, The Manufacturer and Software Magazine. You can contact him at 1-978-927-0313 or visit his web site at www.clarity-consulting.com.