SYNOPSYS®

WHITE PAPER

Benefits of a Design Quality Audit in Software Due Diligence

Introduction

Technology merger and acquisition (M&A) due diligence demands a thorough and meticulous review of the target company's software architecture and code. This is especially critical in transactions where software is core to the valuation. Stakeholders require a shrewd evaluation not just of the code itself, but of the structure and architecture it's built on: a Design Quality Audit. This is a particularly daunting task given that the architecture rarely looks the same as the original design after months and years of development effort. But failure to perform adequate analysis can leave an organization unaware of the potential technical debt it's inheriting, and that can have significant ROI implications, because remediation efforts add time and money and directly impact future innovation.

The complexities of M&A due diligence

Manual code review provides key information about the software, but it doesn't address how code is structured and the implications on future development. During due diligence, the acquiring party has limited, if any, visibility into the source code of the target, and it's rarely able to access it until after the close of the acquisition. And even when a third-party code review takes place, it often skips over any analysis of the foundation of the application—the architecture. Acquirers typically review design plans but fail to consider the changes that can occur as developers speed toward each release. Without an architecture review, a critical gap in insight and analysis threatens the maintainability and scalability of the codebase. This obstacle is best overcome by implementing design audits performed by a mutually trusted third party, in tandem with manual code reviews. This dual-pronged approach provides the most comprehensive picture of the target software.

Addressing architecture and code review needs with Black Duck Audits

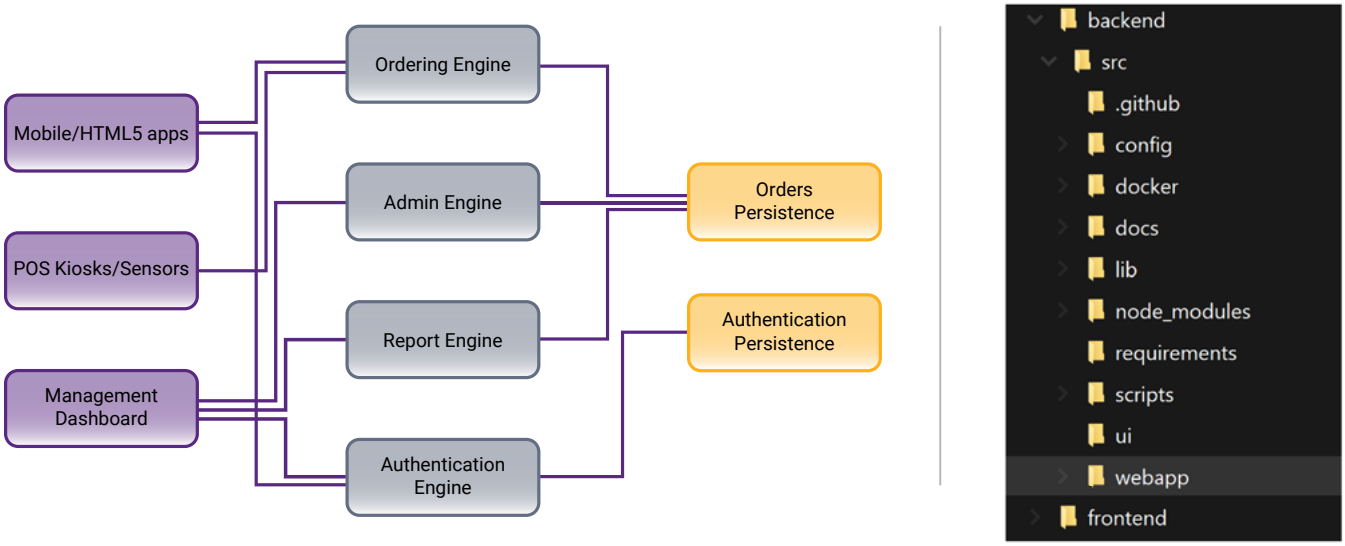
Manual code review allows an acquirer to understand how a product or service is designed to fulfill customer requirements. This exploration covers the overall layout of the various software components and their interactions that provide the business functionality.

This manual review of a software system's architecture is recommended for discovering the strengths and weaknesses of the system and identifying any roadblocks to future growth plans and maintenance efforts. The target's technical leadership will typically illustrate the functionality and design elements with flowcharts and high-level architecture diagrams.

What these reviews don't include is an analysis of the actual implementation of the architecture, which is just as important. Although the target's technical leaders may list the various programming languages, platforms, and tools used, they generally don't go into the details of the software implementation. Thus, the high-level manual review cannot easily identify problem areas of the codebase. Synopsys' Black Duck® Design Quality Audit fills the gap between this high-level review and the actual implementation details, providing a complete picture of the quality of the target company's software systems.

Key findings that support the need for code review

Synopsys auditors have examined the software systems and development practices of hundreds of companies and have observed certain trends emerging over the years. An important finding is that high-level design elements rarely tell the whole story. How this high-level architecture is translated by the developers into working pieces of code, and how the codebase evolves over time, can make or break the stability and performance of a system.



Architect's view vs. developer's view

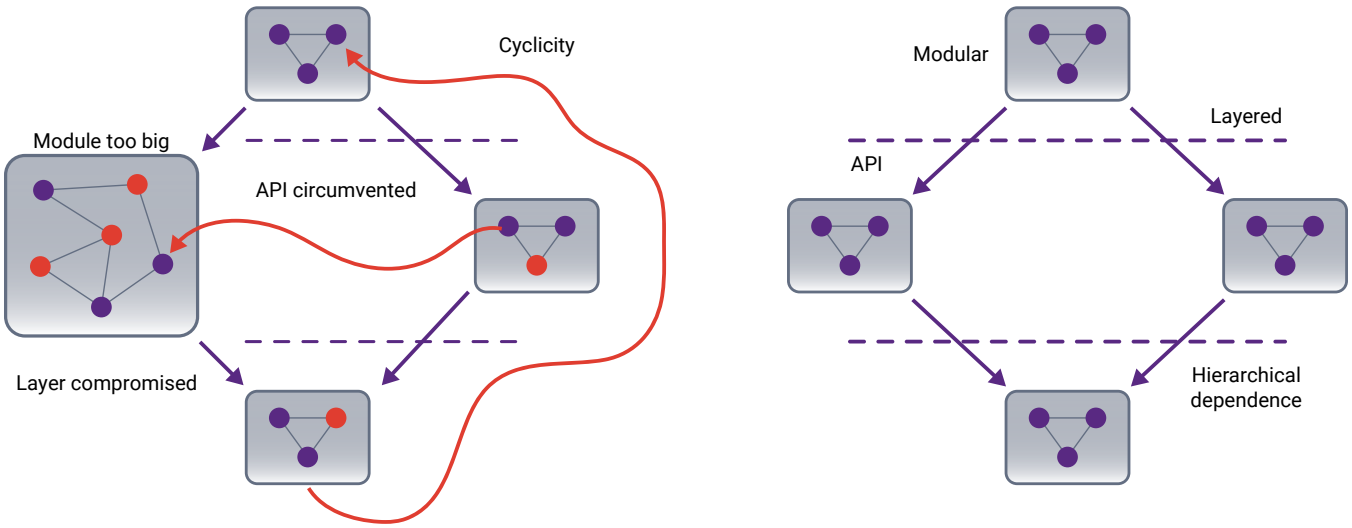
A key factor is that software systems are constantly being updated for various reasons including:

- Changing or new requirements
- Growth in demand
- Security threats
- Upgrades to third party components
- Vendor changes and other challenges

Over time, as the implementation details stray from the original intended architecture, the pictures shared by the target of the current system become increasingly inaccurate. Developers are often on tight deadlines to ship changes, and they are forced to skip best practices like code reviews, automated testing, or secure programming activities. Lack of testing and consistency leads to codebases that are hastily patched together. The codebase becomes messy and brittle, and testing messy code that is highly interdependent can be challenging. Even more challenging is maintaining a codebase that breaks with even minimal change.

How Black Duck Design Quality Audits work

A Black Duck Design Quality Audit can help measure the modularity of a system and identify problematic components. It provides insight at the source code level and assesses maintainability of the system as well as the underlying components. It's important to note that this differs from a code quality audit, which is focused on the bugginess of the code rather than how the code itself is structured. A code quality audit is also important, but it doesn't address how the structure of the code might impede developer productivity. Using Silverthread's CodeMRI tool, Black Duck scans the software and analyzes relationships between the code components, identifying highly interdependent pieces of code that can be difficult to maintain and enhance. These clusters of problematic files (referred to as critical cores) are often the root cause of persistent maintenance problems in a codebase. The more cyclical dependencies a codebase contains, the higher the chances are of defects being introduced as developers work on it.



Unhealthy vs. healthy architecture

Higher defect rates and diminished developer productivity lead to longer development times, delayed releases, and associated increases in overall costs. A Design Quality Audit can not only zero in on areas of concern, it also provides estimates for maintenance costs and expected defect rates as new features are added to the system. CodeMRI's statistical model, built from analyzing thousands of real-world codebases, provides acquirers with insight into future development costs.

In addition to identifying trouble spots, the audit report also helps inform the difficult decision of whether to refactor an existing codebase or rewrite components to improve overall quality. Expert review of these results can also help uncover underlying organizational issues such as lack of focus on testing or a shortage of experienced technical resources.

A valuable addition to the Design Quality Audit report is a technical health improvement plan. This additional analysis identifies specific sections of code that should be reviewed and refactored to eliminate or reduce cyclicality in a codebase. Acquirers find this information extremely useful when planning refactoring as part of onboarding the new software assets.

silverthread

CodeMRI® Health Diagnostics

CodeMRI GovernmentSystem7 Analysis

CodeMRI GovernmentSystem7 is an unhealthy codebase that requires attention.

CodeMRI GovernmentSystem7 contains 3 critical Cores and 5 emerging Cores — tangles of files intricately dependent on each other. Core #1 contains 695 files, and has the biggest impact on your codebase's cost, schedule, and performance. It also contains 623 files with high complexity scores. Complex files are problematic because they reduce reliability, safety, and dramatically increase the likelihood of defects. Talk with engineering about a refactoring initiative for this area of the codebase.

? Show details for: Overall System

About this Codebase

? Primary language:	C++
? Number of files:	9,184
? Lines of code (LOC):	4,332,000
? Average file size (LOC):	472
? Diagnostic confidence:	Moderate

Technical Health

Technical Health predicts both the Cost of Ownership, Quality and Risk associated with a particular program. Technical health is comprised of both Code Quality and Design Quality measures.

Design Quality

Design Quality is measured by the size, quantity, and composition of each Core.

	? Count	? Lines of Code
? Critical Cores:	3	1,088,236
? Emerging Cores:	5	113,970
? Files outside Cores:	7,109	3,129,794

Code Quality

Code Quality is measured by the number of logical paths through a file (reported as complexity).

	? Files	? Lines of code
? High complexity:	623	1,219,466
? Medium complexity:	868	878,895
? Low complexity:	7,693	2,233,639

Economic Outcomes

Based on the Technical Health of codebase , we predict the following outcomes:

*Feature is defined as 1000 lines of code.

Cost of Ownership

models.

? Cost to produce feature*:

Bottom 20% Top 20%

$36,381 $25,105 $11,520 $8,636

Choosing Black Duck to ease M&A stress

Clients faced with the difficult task of evaluating a target company's software systems should adopt a multipronged approach to analysis. Manual reviews should be complemented with an external audit of the architecture and processes, as well as a deep dive into the code. A Black Duck Design Quality Audit offers expert experience in these complex systems. It illuminates hidden trouble spots in the codebase and determines if critical components are well-designed or if they contain unhealthy areas that should be reviewed. Addressing these problem zones will lead to reduced defect rates and minimize roadblocks to deployment. The Design Quality Audit results provide added insight and help with crucial refactor/rewrite decisions. And the technical health improvement plan provides a detailed refactoring plan that will improve overall maintainability of the system.

During M&A due diligence, it's essential for acquirers to identify problematic code in the target's software before the transaction terms and integration timelines are set. Savvy acquirers perform software code audits whenever software assets are a significant part of the deal valuation.

Black Duck Audits will provide the advice your firm needs to quickly assess a broad range of software risks in your acquisition target's software. Get a complete picture of the target software so you can make informed decisions with confidence.

SYNOPSYS®

The Synopsys difference

Synopsys helps development teams build secure, high-quality software, minimizing risks while maximizing speed and productivity. Synopsys, a recognized leader in application security, provides static analysis, software composition analysis, and dynamic analysis solutions that enable teams to quickly find and fix vulnerabilities and defects in proprietary code, open source components, and application behavior. With a combination of industry-leading tools, services, and expertise, only Synopsys helps organizations optimize security and quality in DevSecOps and throughout the software development life cycle.

For more information, go to www.synopsys.com/software.

Synopsys, Inc.
690 E Middlefield Road
Mountain View, CA 94043 USA

Contact us:
U.S. Sales: 800.873.8193
International Sales: +1 415.321.5237
Email: sig-info@synopsys.com