APPLICATION TRANSFORMATION

Making Existing Applications More Competitive

Ian S. Hayes
President,
Clarity Consulting, Inc.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Executive Summary</td>
<td>1</td>
</tr>
<tr>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>When to Address a Legacy Application</td>
<td>6</td>
</tr>
<tr>
<td>Options for Making Existing Applications More Competitive</td>
<td>7</td>
</tr>
<tr>
<td>Starting an Application Transformation Project</td>
<td>10</td>
</tr>
<tr>
<td>How Progress Supports Its Partners’ Transformation Efforts</td>
<td>11</td>
</tr>
<tr>
<td>OpenEdge Reference Architecture</td>
<td>11</td>
</tr>
<tr>
<td>Application Transformation Approach (ATA)</td>
<td>16</td>
</tr>
<tr>
<td>Why Start Now?</td>
<td>19</td>
</tr>
<tr>
<td>Conclusion</td>
<td>21</td>
</tr>
<tr>
<td>About the Author</td>
<td>21</td>
</tr>
</tbody>
</table>
EXECUTIVE SUMMARY

Application software is the core asset of an independent software vendor (ISV). It is the major source of revenue and market value, and its capabilities define the company’s competitive advantages. Much capital and effort are devoted to develop, enhance and support it. Like any other asset, the goal is to extract maximum value from an application software investment by increasing the revenue it generates, improving its profitability and extending its useful life.

Ironically, the enhancements and customizations that improve an application’s business value often contribute to its technical decline. Evolving technologies, design limitations, continual maintenance and marketplace changes take their toll. The application may become increasingly difficult to extend, support and sell, reducing its profitability and asset value and hurting the ISV’s ability to grow its business.

Fortunately, an approach called “application transformation” can cost-effectively reverse this decline while adding new capabilities that increase the application’s market attractiveness and extend its productive and profitable life. Unlike financially costly and risky new development strategies, application transformation leverages investments made in existing applications, modernizes their architecture and interfaces to support new technologies, and adds desirable new features.

A transformed application is more easily extended to respond to customer requirements and more quickly adapted to exploit new market opportunities, leading to increased sales and higher revenues. A transformed application is also less costly to support and maintain, lowering overhead and increasing profit margins. In short, a transformed application helps an ISV expand its business while improving its productivity.

Intended for CTOs, CIOs, senior architects and others wrestling with the issues of aging systems, this paper reviews the Progress Software approach to application transformation. The strong, comprehensive Progress approach is a valuable model for any company considering transformation. Progress has researched and assembled best practices for application design and development, and combined them with first-rate transformation techniques,
to produce a superior transformation approach that many partners are already using to great success.

The heart of the Progress transformation program is its robust reference architecture—the Progress® OpenEdge® Reference Architecture (OERA)—a model for designing modern applications that reflects industry standards and leading thinking about application architectures. To bring the benefits of the OERA to existing applications, Progress created the Application Transformation Approach (ATA). The ATA is an iterative, phased process for transforming existing applications, with a fundamental goal of harvesting and re-using functionality where feasible. Together, the OERA and ATA are instrumental in transforming applications to make them more competitive and drive greater business benefits. Progress partners are fortunate to have these strong tools and assistance and should inquire today about the potential value that transformation can bring to their businesses.
INTRODUCTION

Software applications are the lifeblood of an independent software vendor (ISV). They define the nature of the company, establish its image, and determine its fortunes. They generate revenues through license sales, customizations and maintenance. They embody the company’s intellectual property, domain expertise and unique business logic. They are a valuable asset in their own right, serving as the source for spin-off products and contributing to the company’s market value.

ISVs clearly have a vested interest in the continuing health and viability of their application software. Yet, as shown in Figure 1, all software is susceptible to a natural decline in quality and value over time without continual, balanced investment in functions, features, architecture and technologies. Without balanced investments, the functional and technological modifications made to an application during its lifespan increase the complexity and cost of customization and support. Ironically, constant functional modification often increases the business value of the application while simultaneously decreasing its technical quality, making future changes more difficult.

Many external and internal factors combine to erode software quality, eventually compromising its value and usability.

---

1 Other aspects of software quality include functionality, reliability, usability, maintainability, portability and efficiency, among others. See ISO/IEC 9126 standard.
Business and competitive landscapes change, surpassing application capabilities—Business models are constantly in flux, affected by a range of external factors, industry trends and competitive pressures. Globalization, deregulation, supply chain integration, rationalization, mergers and acquisitions, mobile workforces and technology innovations profoundly affect the larger business landscape. To remain relevant and vital, ISVs, and their customers, must adapt their businesses to these new realities. To retain market value and appeal, application offerings must also evolve along with these business changes and innovations and do so in “business time.” As the pace of business change has quickened, ISVs must somehow keep their older applications flexible, agile and responsive to avoid falling into a game of constant catch-up.

Technologies change, outpacing older ones—The technologies that interoperate with software applications inevitably change, offering better performance, enabling more sophisticated application designs, and even spawning entirely new business models. Distributed n-tier architectures, Microsoft .NET and J2EE, web services, service-oriented architectures (SOA), XML, rich web-enabled interfaces, handheld devices and wireless technologies are all examples of technological changes that profoundly impact application functionality and operation.

Application design limitations are reached—Created for a specific purpose, most applications find their original intent and form changed over time in response to external customer demands, competitive pressures and business innovation. Yet few applications are built from the outset to accommodate infinite re-design. Monolithic program structures, little separation of functional logic, and embedded, technology-dependent code make it difficult to substantively change older applications. Most eventually reach their architectural limits, inhibiting further extensions. Propositions such as web-enabling a user interface, adding local language and currency support for new geographic areas, or integrating with other systems and partners—capabilities deemed baseline today—become monumental undertakings for older applications.
Continuous maintenance increases complexity—Years of accumulated maintenance take their toll, compromising application integrity, introducing layers of complexity, and making it exceedingly costly, time-consuming and risky to introduce changes. Developers are hard-pressed to figure out what code is doing and where and how to make changes, especially when documentation is outdated or missing. Sporadic reuse leads to redundant and duplicative code, magnifying the distribution and potential impact of maintenance changes and requiring higher degrees of coordination and testing. Simple tweaks cause unintended consequences, and seemingly straightforward bugs defy quick analysis and resolution. ISVs become increasingly reliant on application gurus and less able to attract and train new technologists to tackle application maintenance.

The ISV’s dilemma is to find a way to modernize its existing applications while keeping its business running. In most cases, immediate revenue needs will trump longer term software investment initiatives, and understandably so. Left unaddressed, however, growing code complexity and architectural compromises will eventually lead an application to its end of life, impairing an ISV’s ability to grow its business and compete.

Fortunately, an ISV can improve the quality and value of its applications without sacrificing business fundamentals. Transforming aging applications, using a phased, architecture-driven approach can stem and even reverse declining software quality. Application transformation helps extend the life and value of existing applications, renovating problematic components to improve productivity and save maintenance and support costs. More important, from a business standpoint, transformed applications are a springboard for an ISV to:

- Grow its business
- Generate more leads and add more prospects to the pipeline
- Enter new or synergistic markets or geographies

---

> Respond more quickly to market demands with new offerings
> Meet new customer requirements
> Beat competitors, regionally and globally
> Attract and retain skilled technical personnel
> Shift technical resources to value-adding tasks and new technology deployments

This paper examines the difficulties posed by legacy applications and options for tackling those difficulties from an ISV’s perspective. It is intended for any company that wrestles with the business ramifications of aging software systems. In particular, the paper explores the option of legacy application transformation—leveraging the value inherent in existing applications by taking steps to improve them—and focuses on what Progress Software is doing to help its partners adapt and transform their applications to make them more competitive and drive in more business.

WHEN TO ADDRESS A LEGACY APPLICATION

As used in the IT industry today, every application, without exception, becomes a “legacy” application gradually over time. “Legacy” refers to a natural lifecycle stage that begins once an application moves from the development phase of its lifecycle into the successive phase—production. For an ISV, this stage starts once the application is packaged, released and licensed.

When does a legacy application need attention? The conditions that merit attention will appear and grow over time rather than manifesting themselves all at once and may not seem obvious, especially when a company is used to living with them. An ISV should consider addressing its legacy applications if one or more of the following conditions apply:

> Sales of the application are lagging whether due to waning market interest, an inability to satisfy market requirements or competitive pressures.

“Legacy systems are not defined by age, language, platform, or data structure type. If an application is functioning in a production environment within an enterprise, it can be considered a legacy system.” Ulrich, William M., “Legacy Systems Transformation Strategies,” Prentice Hall PTR, 2002, page 21.
The application is increasingly costly to enhance and support, cutting into the profitability of maintenance streams.

The time and effort needed to customize the application for new buyers are affecting the ability to price competitively.

Architectural or design limitations hinder the addition of desired new functions that would increase the application’s value and help win more business.

OPTIONS FOR MAKING EXISTING APPLICATIONS MORE COMPETITIVE

An ISV charged with supporting and maintaining existing applications has three basic options. In order of declining business risk they are: do nothing, develop a new application or transform the existing application into a more competitive one.

OPTION ONE: DO NOTHING

Although technologically simple and requiring no outlay of funds, this option bears the highest business risk of all. Over time, income from new license sales drops as the application loses ground competitively; profits decline as the application becomes more costly to support and customize; and recurring maintenance revenues dry up as customers defect to newer product offerings. This option applies only for ISVs interested in winding down their businesses.

OPTION TWO: DEVELOP A NEW APPLICATION

This approach advocates throwing away the burdensome legacy application and creating a new application from scratch. The new development option is tempting, as it allows an ISV to theoretically abandon older, problematic applications, and is generally preferred by developers. But new development projects pose many risks, especially for ISVs that depend on their software products to sustain their ongoing businesses.

Financial risks—Developing a new application is a “go for broke” strategy with nothing to show unless the project is finished and is a commercial success. Direct costs from technology acquisitions and staffing a large-scale project can be high, and an anticipatory drop in
sales may occur if customers defer buying decisions pending release. But high financial risks are also incurred due to the time it takes to introduce a new product to market. Market needs and desires can change precipitously and significantly between the time a new project is conceived and the time it is delivered, creating the risk that the final application will be out-of-step with its intended market.

**Functionality/intellectual property risks**—New application development requires a thorough understanding of the existing application. Legacy applications often are the sole, complete source of business knowledge, containing not only the major logic but, more critically, the subtle changes, special conditions and hard-earned insight discovered over time. Most customers will expect functional equivalence between the old application and the new, making it essential to capture and replicate business logic in the existing application.

**Failure risks**—Perhaps most troubling of all, new development projects face a significantly high probability of failure. IT research firm, The Standish Group, found that a mere 28% of new development projects complete successfully, defined as on time, on budget and with all originally specified features and functionality. ² Twenty-three percent of new development projects fail completely, and another 49% end up “challenged”—completed but over-budget, over time and without all the originally specified features and functions.

New development is appropriate for launching new products or in cases where an existing product is functionally as well as technologically obsolete. If the current application retains significant functional and intellectual property value, however, the costs and risks of this approach are substantial.

**OPTION THREE: TRANSFORM THE EXISTING APPLICATION**

Compared to the alternatives, transforming existing application assets is a very attractive proposition. The philosophy of transformation is to reuse existing application functionality where possible to extend the life and value of the initial application investment. Inherently different from new

development, transformation places a premium on existing application assets, preferring not to jettison them but to salvage their most valuable pieces. For example, transformation may involve replacing a character-based front end with a web interface to increase an application’s appeal and revenue-generating capacity. Or it may entail harvesting and reusing critical business logic in a service-oriented architecture (SOA) on a new platform.

Unlike a new development project, transformation efforts can avoid risky “big bang” implementations and their associated financial risks. Transformation projects are usually amenable to iterative cycles, with each one delivering distinct business or technical advantages and executed as warranted by budgets and opportunities. For example, if a business goal is to reduce maintenance and support costs, transformation can target the typical 3% to 5% of programs that account for 50% to 80% of overall maintenance effort. New, revenue-generating functionality and extensions can also be planned and added as part of a transformation effort.

Transformation is not without hurdles. A total transformation is a lengthy process, requires reskilling and training technical resources in transformation and modern application design techniques, and demands rigorous project management. When compared to the other options, however, transformation is the superior choice if the existing application functionality remains suited to its market but could benefit from extensions and modernization.

Options for Making Existing Applications More Competitive

<table>
<thead>
<tr>
<th></th>
<th>Do Nothing</th>
<th>New Development</th>
<th>Transformation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Technical challenge</td>
<td>Low</td>
<td>High</td>
<td>High</td>
</tr>
<tr>
<td>Direct investments</td>
<td>Low</td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>Financial risk</td>
<td>High</td>
<td>High</td>
<td>Medium</td>
</tr>
<tr>
<td>Functionality/IP risk</td>
<td>Low</td>
<td>High</td>
<td>Low</td>
</tr>
<tr>
<td>Project failure risk</td>
<td>N/A</td>
<td>High</td>
<td>Medium</td>
</tr>
</tbody>
</table>

8 In the context of SOA, Gartner states that, “there is no better environment in which to source services than the installed base of working, time-tested business functions implemented in legacy systems.” Vecchio, Dale, “CIO Update: Extend Mainframe Life by Adopting a Service-Oriented Architecture,” Gartner, March 19, 2003.

STARTING AN APPLICATION TRANSFORMATION PROJECT

Application transformation has proven to be a sensible, effective option for many companies, but it can also be intimidating when just starting out. A wide range of technology alternatives and approaches exist, yet few answer an ISV’s more pressing questions: What is the right path to take my existing application to my desired end goal? How far do I have to go down that path?

Successful transformation projects share an important characteristic—excellent preparation. Before diving into technical work, these projects lay the foundation for success by:

- **Understanding business objectives**—what does the business need from the application and how much should be invested to get there\(^{10}\)
- **Creating a product roadmap**—a step-by-step plan to align product strategies and business goals
- **Devising the transformation plan**—a plan for transforming the current application to fulfill the product roadmap
- **Identifying a transformation methodology**—a how-to-guide describing the process and steps to transform the application
- **Seeking outside expertise, as needed**—for insight and assistance to jumpstart the effort, determine training and skills needs, and develop and mature new, internal capabilities

In the remainder of this paper, we examine the Progress Software approach to application transformation. This solid, industry-leading approach is noteworthy in its ability to meet the criteria for success outlined above and for the valuable, real-life benefits that it is delivering to Progress application partners. Many methodologies focus on the abstract and are difficult to implement in real-world scenarios. In contrast, the Progress approach, rooted

---

\(^{10}\) Just as all applications do not have the same investment needs, they also do not have the same modernization needs… unless enterprises have identified their business strategies, key initiatives and related application systems, and have prioritized investment decisions, any modernization project would be an exercise in futility. Vecchio, Dale, "Portfolio Management: Step 1 in Architectural Modernization,” Gartner, May 15, 2002.
in reality and industry best practices, is delivering very strong, credible results in the field today.

HOW PROGRESS SUPPORTS ITS PARTNERS’ TRANSFORMATION EFFORTS

Progress Software sets itself apart from other software providers with its comprehensive set of partner support programs, programs that have earned industry-wide recognition. The degree and quality of these programs and the investments that Progress has made to date, rival and exceed those of other technology vendors. With these programs, Progress is acting as a partner in the true meaning of the word. The stated Progress goal is to help partners expand and grow their businesses and develop sustaining technical skills, processes and know-how to transform their applications. According to Progress, hundreds of partners have participated in these programs, and have witnessed more than 20% annual revenue growth thereafter.

The devotion of Progress to its partners is manifested in numerous programs, workshops, tools and reference materials. The two with perhaps the greatest impact on application transformation are the Open Edge Reference Architecture and the Application Transformation Approach.

OPENEDGE REFERENCE ARCHITECTURE (OERA)

The OERA encapsulates the Progress Software vision of modern application design and development using the Progress® OpenEdge® platform. It incorporates best practices and know-how that Progress has gleaned through its own work, its partners’ work and independent research. Its intent is to provide software development organizations with a best practice approach to building modern, service-oriented business applications and, thereby, avoid many of the problems and issues associated with legacy applications cited earlier in this paper. The Progress OERA is an excellent example of a complete yet flexible architecture that accurately incorporates current standards and leading thinking about modern application architecture and design.

The OERA is a logical application architecture. To physically implement the architecture, software development organizations use the tools and infrastructure provided by the Progress OpenEdge platform. These
unified, synergistic tools help developers create, deploy, integrate and manage applications that conform to the OERA architecture, throughout all phases of their lifecycles.

**What Is an Application Architecture?**

Software development organizations use application architectures to guide the design, creation and deployment of application software. The concept of using architectures gained favor in the 1990s when distributed applications began to replace monolithic mainframe-based ones. Most legacy applications, however, were not designed according to a pre-defined architecture, and years of maintenance changes have obscured their original designs.  

An application architecture is a set of practices, principles and models that describe how an application should be built. It is used by developers when constructing applications and also serves as a source of after-the-fact documentation to aid later understanding. A well-crafted application architecture benefits an organization by:

> Promoting more structured application design, aiding in understanding and easing future maintainability

> Speeding the development process by removing many decision points and providing developers with a view of the “end goal”

> Extending the lifespan of applications by advocating logically sound designs that can accommodate extensions and customizations

> Lowering development risks by supplying a validated design prior to coding and reducing application failures and re-works

**The Characteristics of the OERA**

The Progress OERA is a reference architecture or “master blueprint” for modern application design. In practice, an application architect uses the OERA as a guide when designing an application architecture with the goal of creating more extensible, interoperable, long-lived software. Organizations that adopt the OERA will find it helpful not only in transforming existing applications but in developing new ones.

As a reference architecture, the OERA is necessarily high-level. To be useful, however, it must have the right balance of specificity—to ensure that an architect can understand and use it—and flexibility—to ensure that it can accommodate a wide variety of application types across the Progress partner base. The OERA succeeds on both counts. A multi-tiered, component-based, service-oriented architecture, the OERA is characterized by four layers:

- **Business services**: encapsulates the core business logic and business rules in specific workflow, tasks and entities that drive the application
- **Data access**: retrieves and manipulates data in the underlying data stores
- **Presentation**: specifies user interfaces and interaction points—the visual parts of the application
- **Integration**: supports external access to data and business processes, providing interoperability with other services
The most salient and beneficial characteristics of the OERA, those that will “future proof” transformed applications, are highlighted below.

**Technology neutral**

Although based upon Progress OpenEdge platform, the OERA does not advocate the use or inclusion of any particular technology. It anticipates that applications will be built to operate in a variety of technical environments, using a range of technology implementations. By adhering to a layered approach and insulating functionality at each layer, technology platforms, operating systems, data stores, integration technologies and interfaces can vary without requiring re-design at the other layers. A technology neutral architecture, the OERA allows applications to be built that support a range of technologies from Microsoft .NET, J2EE, HTML, XML, JMS, ESB, web services, SmartObjects, Progress ProDataSet, to a variety of RDBMS, flat files and more.

**Incorporates industry standards**

The OERA reflects and combines current industry standards and best practices for application design and development. A distributed, component-based, multi-tiered architecture, it also heavily incorporates web services, SOA and SOBA principles and uses modern techniques to correct many of the deficiencies commonly found in legacy application architectures.

**Component-based**

The OERA contemplates a component-based architecture that fosters reuse of business logic, data access, presentation and integration components from application to application. Component-based design offers numerous advantages. It speeds development cycles and eases future maintenance by eliminating duplicate and overlapping coding—typical attributes of legacy systems. It enables plug-and-play applications in which individual pieces can be swapped out and replaced without affecting the whole. It serves to contain modifications and customizations, limiting their ripple effect across the application.

---


User Interface independence

Using the OERA, companies can add a variety of rich, cutting-edge interfaces to support multi-channel access without impacting the design or structure of the guts of the application. Web-based interfaces, for example, can be designed to support activities ranging from eCommerce to business intelligence reporting, and interfaces can be tailored to different user groups and device types from desktop computers to mobile handheld units.

Data independence

By providing for a separate data access layer, the OERA achieves data independence and a consistent method of accessing data. The business portion of the application contains no direct database calls or file access, which allows for a variety of underlying data stores and technologies including RDBMS, XML, flat files, etc.

Integration support

The OERA avoids the limitations of traditional “stovepipe” architectures by making integration an inherent capability of applications built using the OERA. Applications can more easily expose and share their business logic with other applications regardless of the technology or platform on which they are developed. Integration services are encapsulated in an integration layer that can readily support web services and JMS.

Distributed access

The OERA makes no assumptions of how applications, data or business logic will be accessed, from where, or using what technology. By separating the data access, presentation and integration layers, OERA-designed applications allow for distributed, even remote, access, using wired and wireless devices, no matter the degree of session connectivity (always connected, intermittently connected or disconnected).

Scalable

The OERA supports the construction of all types and sizes of applications from very small applications to those containing dozens or hundreds of modules. In fact, organizations can use as much of the architecture as needed, based on application requirements. As solution capabilities and functionality increase in scope, and as the audience of users
expands, the OERA foundational principles and structure remain relevant and capable of supporting virtually unlimited growth.

**Extensible**

By adhering to a component-based, layered design, the OERA makes it easier to add features and functionality to an application. The logical design allows developers to more quickly determine where to make changes and reduces the odds of unintended side affects. Component-based applications simplify maintenance and support chores, speed development cycles, and enable quicker customizations in response to customer and new market needs.

**APPLICATION TRANSFORMATION APPROACH (ATA)**

Application transformation is a “process for applying changes to the form, design, and/or function of one or more legacy applications.” Working hand-in-hand with the OERA, the Progress ATA is a high-level approach or methodology that provides a process for transforming existing applications into more modern ones. Whereas the OERA describes the end goal—how the application should look—the ATA describes the process for getting there. In practice, the ATA is a flexible template or guide rather than a rigid process: it is meant to be tailored to meet the unique requirements of a customer’s situation while still preserving the integrity of the approach.

Like other development methodologies, the ATA aims to reduce the risk and improve the efficiency of producing software. Where other software development methodologies focus on creating new applications from scratch, the ATA is designed to transform existing ones. That said, the ATA does incorporate best practices from several leading software development methodologies and standards, including Agile, RAD, UML, code reuse and legacy extension.

The ATA does not necessarily require that an entire application be transformed or that transformation activities occur in a prescribed order. Companies use the ATA to transform applications according to their larger product and business goals, which may call for anything from a partial to

---

full transformation. Example transformation projects may focus on migrating to a new database technology; adding new, competitive features; building richer user interfaces; or all of the foregoing. Because the ATA embraces the concept of iterative transformations, companies can perform their transformation projects successively, as desired, to build upon and extend earlier implementations.

**The Characteristics of the ATA**

Progress describes application transformation as “changing organizations, people, processes, architecture and technologies.” Reflecting this belief, the ATA also covers areas such as project management, skills training and release management—all areas that contribute to a successful transformation.

![The ATA](image)

Depending on an organization’s goals, transformation can be a lengthy process with associated risks. One of the ways that the ATA helps minimize risks is to begin with a thorough current state assessment followed by iterative transformation cycles to break down the process and speed attainment of benefits. Together, there are four distinct phases, of

---

which the last three are performed iteratively, and a set of tasks to perform at each stage:

> **Assessment:** In this phase, Progress assesses the current business position of the company, application requirements, and current infrastructure and formulates a transformation plan that aligns with business needs.

> **Analysis and Modeling:** In the analysis and modeling phase, a set of logical (requirements, use case, business rules, data, user interface) and physical models (deployment, data, source code, components) are created to represent the form and function of the existing application and reflect any new requirements or improvements. An inventory of physical artifacts is compiled identifying items that can be feasibly harvested and re-worked for use in the transformed application.

> **Redesign and Harvesting:** In this phase, models are created representing the target application. The models captured from the existing application are transformed into models for the new application, and new models are created. The harvestable artifacts identified in the prior stage are extracted and re-worked for use in the new application.

> **Build and Test:** In the last phase, the new, modern application is constructed using the redesigned models and the harvested components.

According to Progress, the initial phase of a transformation project could take from 4-to-12 weeks while a complete application transformation could range from 8-to-36 months. Progress consultants are available to help partners begin their transformation project and develop the required skills, and to mentor them on an as-needed basis during subsequent iterations. The initial transformation project is generally designed to be small enough to complete in a reasonable amount of time, yet broad enough to gain exposure to a number of transformation techniques and intended to deliver business value upon its completion.

The ATA applies the fundamental design principles of the OERA and brings its benefits to fruition. Components are harvested and reused

www.progress.com
to leverage the value and knowledge of existing applications, speed development and reduce later maintenance and support burdens. Application functionality is componentized to localize changes and enable the introduction of new interfaces and technologies without impacting business logic or other functional areas.

**WHY START NOW?**

Any discussion of existing systems and the transformation process necessarily revolves around technology. But framing the discussion purely in terms of technology misses the point. The real drivers for transformation are economic ones: the lagging sales, decreasing profits and shrinking revenue base symptomatic of applications that have lost their appeal and competitiveness. The real value to be obtained from transformation is also economic: the true, business-based elements of ROI that every ISV desires. Transformation is a path to:

> Enhanced competitiveness, improved feature sets and faster response to threats and opportunities
> Increased revenues through more product sales
> Higher profitability through reduced support and customization costs
> Greater asset value from product investments
> Expanded market share through new functions and product spin-offs

The concept of transforming legacy applications is not a new or untried one. Companies have sought to transform existing applications for over 20 years, and techniques and knowledge have matured and evolved during that time. Impressively, Progress Software has researched and harnessed the best of those techniques and combined them with a superior architecture and process to create a leading transformation approach. Even more impressive is that the Progress transformation approach is not just theoretical. Many Progress partners have already started using the OERA and ATA to transform their existing applications.
Integrated Warehousing Solutions

Integrated Warehousing Solutions of Downers Grove, Illinois, develops warehouse, distribution and supply chain solutions for large-scale operations such as Sears.com, Sanmina-SCI and the State of Ohio Department of Health vaccine distribution Program. Integrated Warehousing Solutions used the Progress Business Empowerment Program to brainstorm about its business and discover process improvements, and the Technical Empowerment Program to transform its existing application. According to Carl Brewer, President of IWS, “We needed the transformation to be quick and augment our existing client deliverables. With Progress guidance we utilized an iterative transformation approach that allowed our organization to realize value along the way. As a result, we re-architected our application on time and under original budget.” Result: Integrated Warehousing Solutions’ success with its transformed application is allowing it to continue to bid on large-scale projects with much more opportunity to achieve margin-rich revenue due to a simplified architecture.

proALPHA Software

Based in Germany, proALPHA designs, develops and implements business solutions and services for mid-sized manufacturing companies. It needed to redesign its flagship product, proALPHA, to support a multitude of hardware and operating systems. proALPHA originally sought to take a “big bang” approach and completely rewrite the application, but with over 5M lines of 4GL code, it realized the enormity of the task. After conferring with Progress, proALPHA chose to use the ATA’s iterative approach to conduct a coordinated, six-month transformation process. proALPHA VP of Development & Technology Martin Wolf describes the transformation project as an “extremely positive experience” and credits Progress for “streamlining the process, keeping bureaucracy and paperwork to a minimum and getting the work done as fast as possible.” Result: proALPHA gained a transformed application, reducing its code by a “huge amount” and resulting in key operations being executed in “minutes, not days.”
CONCLUSION

Software providers, analysts and industry experts all talk about legacy application issues and transformation, but few do anything concrete to tackle the problem. That is where Progress diverges from the pack. Progress has not only developed a cutting-edge transformation architecture and approach, it is freely sharing it with its partners. Far from being all talk and no action, Progress is actively helping its partners modernize their applications and revitalize their businesses. At a minimum, every partner should consider an application transformation assessment to discover the potential value that transformation can bring to its business.

To keep growing and thriving, businesses must keep their applications compelling, innovative and agile. Salvaging the enormous value contained within legacy applications and turning it into a renewed, revenue-generating engine are a critical priority and singular advantage of ISVs. Application transformation is key to extending the life and value of an ISV’s most important investments and helping it compete more effectively in “business time.” To that end, Progress partners have a golden opportunity to tap into powerful transformation programs that will help them exploit and grow the inherent value in their applications starting today.

ABOUT THE AUTHOR

Ian S. Hayes is President of Clarity Consulting (www.clarity-consulting.com), a management consulting and analyst firm that focuses on IT strategy and improving the business returns generated by IT investments. Ian has over 20 years of application transformation experience, having co-founded Language Technology, a pioneering software redevelopment company, and managed the software re-engineering consulting practice at Keane, Inc., prior to founding Clarity Consulting in 1993. As an analyst and consultant, Ian has advised dozens of companies in legacy strategies. Author of three books, Ian chairs numerous industry conferences, and his articles appear in publications such as Business Week, Computerworld, Better Management, the Cutter IT Journal, AD/Trends, Enterprise Application Integration Journal, The Manufacturer and Software Magazine.
Progress Software Corporation (NASDAQ: PRGS) is a global software company that enables enterprises to be operationally responsive to changing conditions and customer interactions as they occur. Our goal is to enable our customers to capitalize on new opportunities, drive greater efficiencies, and reduce risk. Progress offers a comprehensive portfolio of best-in-class infrastructure software spanning event-driven visibility and real-time response, open integration, data access and integration, and application development and management—all supporting on-premises and SaaS/cloud deployments. Progress maximizes the benefits of operational responsiveness while minimizing IT complexity and total cost of ownership.

WORLDWIDE HEADQUARTERS
Progress Software Corporation, 14 Oak Park, Bedford, MA 01730 USA
Tel: +1 781 280-4000   Fax: +1 781 280-4095   On the Web at: www.progress.com

Find us on facebook.com/progresssw  twitter.com/progresssw  youtube.com/progresssw

For regional international office locations and contact information, please refer to the Web page below:
www.progress.com/worldwide

Progress, OpenEdge, ProDataSet and Business Making Progress are trademarks or registered trademarks of Progress Software Corporation or one of its affiliates or subsidiaries in the U.S. and other countries. Any other marks contained herein may be trademarks of their respective owners. Specifications subject to change without notice.

© 2005, 2012 Progress Software Corporation and/or its subsidiaries or affiliates. All rights reserved.
Rev. 01/12 | 111202-0103