Six Essential Elements of Product Modernization

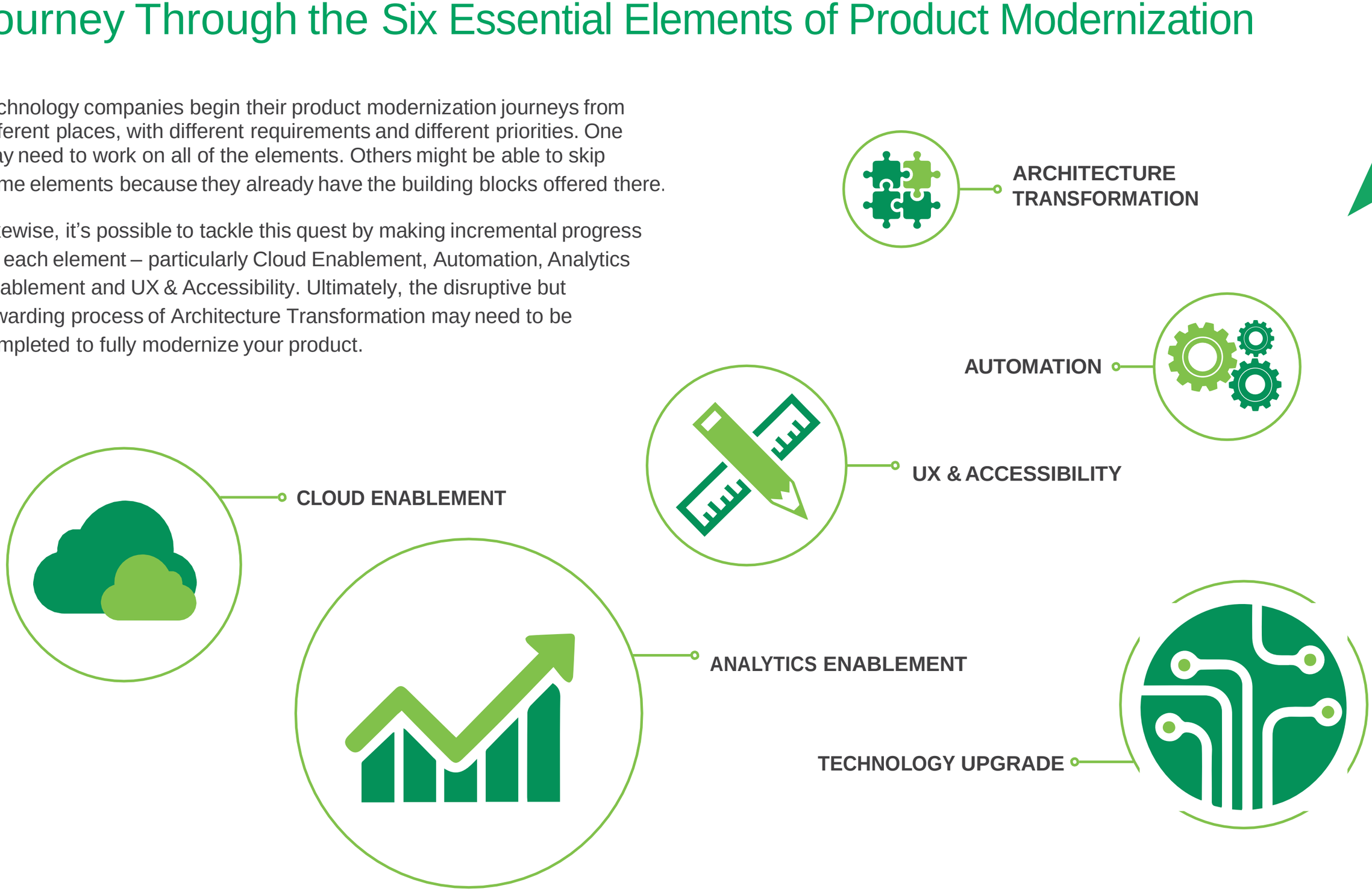
Enabling ISVs To Help Their Clients Achieve Digital Transformation

This e-book will guide you through the six elements of Product Modernization, explaining the benefits and risks encountered with each element, helping you navigate the risks and directing you towards opportunities throughout the product modernization journey.

Journey Through the Six Essential Elements of Product Modernization

Technology companies begin their product modernization journeys from different places, with different requirements and different priorities. One may need to work on all of the elements. Others might be able to skip some elements because they already have the building blocks offered there.

Likewise, it's possible to tackle this quest by making incremental progress on each element – particularly Cloud Enablement, Automation, Analytics Enablement and UX & Accessibility. Ultimately, the disruptive but rewarding process of Architecture Transformation may need to be completed to fully modernize your product.



Primary driver: Competitive advantage

A technology upgrade is often a prerequisite to leveraging modern technologies like cloud, Big Data & analytics and artificial intelligence. Because of that, and because your technology stack is the foundation upon which you modernize, a technology upgrade is often the first stop on the modernization journey.

According to TechRepublic, an IDC report found that by 2023, more than half of all worldwide GDP is predicted to be driven by products and services from digitally transformed industries (from TechRepublic).

What technology upgrade might entail

These are just a few of the elements that a technology upgrade might include:

Modern dev environments

Because your technology environment impacts your ability to modernize on so many levels, if you're running an older environment and want to modernize in certain areas – for example implementing microservices architecture – upgrading your environment to be on current versions may be a necessary prerequisite to modernization. For example, if you're using J2EE 1.4 you may in some circumstances consider upgrading to Java 8. If you're running ASP.NET you may consider upgrading to .NET Core. In some cases upgrading your dev environment can be an easy win though in other cases, it may end up requiring more fundamental architecture transformation.

Modern platforms and frameworks

Older architectures – for example, three-tier architectures with a relational database, a monolithic application server and a web front end – were well suited to that time. Today's platforms have to suit different needs – for example, unstructured data, and the speed with which products are released and iterated (which itself is a competitive advantage). So today's platforms require different architectures – for example, serverless, microservices-based, distributed architectures, with multiple forms of data.

There are many new platforms emerging, so the question is how to adopt and use them without breaking your product. The good news is that modern architectures are designed to deal with a world characterized by constant change.

Your clients will digitally transform. If you modernize your products, you can help them.

Modern infrastructure

When the products and platforms you build are the ones your customers deploy, then building on and being compatible with modern infrastructure could be imperative for enabling your customers' digital transformation. For example, simple network attached storage (NAS) was appropriate for enterprise-class applications. Today, new storage architectures are preferable to fully leverage cloud native architectures. For example, Amazon's S3

enables you to deal with cloud-friendly storage and toolsets provide the same or similar interfaces to private cloud storage. To accommodate those modern storage technologies, an infrastructure upgrade may be necessary.

Key considerations for technology upgrade

These are just a few of the key issues you might consider as you think about a technology upgrade:

Will your product be compatible with the upgraded technology?

Perhaps the most important question to ask when upgrading technology is 'What will break or stop working when I upgrade?' In some cases your product may have such a tight dependency on the old technology that code changes may be necessary for it to continue to function in the upgraded version of the technology.

If some of the features your customers use will not work after the upgrade, then you will have to make some hard decisions about whether to do a more fundamental architecture transformation. Sometimes there's no choice if the old technology will no longer be supported by the vendor. When it comes to making such hard decisions, the key is to ensure that you are in sync with your customers and their upgrade cycles.

How many versions of the code base will you have to maintain?

Sometimes you will have to maintain and support several versions of the code base in order to keep your legacy customers. Whether you do that, or convince your customers to upgrade – or when you can convince your customers to upgrade – of course is a decision you'll have to make in concert with the business side of the organization.

The CTO of a leading quality management software (QMS) company explained in a recent presentation how he worked with his legacy customers as he planned his technology upgrade: "We had this app that did a lot but in many cases was too complicated. So some features had to be simplified. Yet communicating that to customers who had become familiar with these features was a challenge. We had to show customers the tradeoff – the other benefits they'd get with the new product."

How will you mitigate the risks of upgrading?

In some cases a technology upgrade is straightforward. Other times the first step is to do a complete audit of your technology stack, which is a good opportunity to consider your architecture. A methodology that has served Xoriant well combines a simple empirical iterative approach with a more in-depth, inside-out audit that includes a complete review, full mapping and recommendations for fixing.

You may find that your product will run just fine with a technology upgrade, in which case you may decide to bide your time before a product re-architecture decision. Or, the audit process may force a decision to transform your architecture.

> ### Technology upgrade in action
>
> "Our new modernized application has reduced IT (licensing and infrastructure) costs by 50%, reduced production downtime and improved control over the environment – improving operational efficiencies by 25% overall."
>
> – CTO at FSQA Management Solution Provider

CLOUD ENABLEMENT

Primary drivers: High availability, cost efficiency and scalability

You know that enabling your product to work within cloud environments can in itself be tremendously beneficial – with benefits like high availability, cost efficiency and scalability. You also know that your customers are moving to the cloud (that's not a new phenomenon). What is new is the rising prevalence of hybrid cloud and multi-cloud approaches among enterprises. So your products need to work within and across multiple public clouds and private clouds. The good news: You may be able to achieve that without a major redesign, depending on your product.

What cloud enablement might entail

These are just a few of the elements that cloud enablement might include:

Cloud native approach

As enterprises move to the cloud, many ISVs developing enterprise products are transitioning to a "cloud native" approach to application development. Here, products are built specifically to exist within cloud environments and to take advantage of the availability, efficiency and scalability that the cloud offers. The 12-factor approach to developing cloud native applications is a good guide. Software products that enterprises incorporate into their environments will need to be compatible and follow such best practices.

The benefits of adopting a cloud native approach can be significant. It was for the leading provider of food safety and quality management solutions who needed to find a way to implement the equivalent of on-premises procedures – backups, security checks, patches – through configuration level changes. (They also wanted to mobilize their solution to increase reach and serve their customers better.) We re-architected the application and migrated the client from Windstream to Azure. We developed a mobile application along with new WCF REST service. This new approach enabled the client to increase their customer base by 20%.

Platform-as-a-Service

Because of the rise of platform-as-a-service (PaaS), you have several choices of platforms on which your products can run in the cloud. Programming on top of a PaaS like Cloud Foundry, for example, enables you to leverage in-built capabilities like CI/CD and containerization. Depending on your architecture, you may be able to take advantage of some capabilities of such platforms without major redesign and hence quickly achieve their benefits. Fully leveraging them, however, may require re-architecture and re-work.

Hybrid cloud

By far the most common enterprise infrastructure environment is a hybrid of multiple clouds, including private and public. Many enterprises build private clouds on-premise then turn to the public cloud for expansion, or keep certain applications or data in the private cloud and others in the public cloud (an approach that's common in highly regulated industries like financial services and healthcare). Today companies are also looking at how can they further optimize through multi-public cloud environments. So your products must be configured to work seamlessly in these new multi- hybrid cloud environments.

Key considerations for cloud enablement

These are just a few of the key issues you might consider as you think about cloud enablement:

Do you have the right tools to manage a hybrid, multi-cloud environment?

A best practices approach ensures that your product can run in any and all cloud environments, whether or not you are building your products as native cloud applications. Best practices for building cloud native applications are defined in resources like 12factor.net. These may include microservices architecture, containerization, and implementing monitoring and logging capabilities for the software. (It's also important that your product can be accessed and monitored by other applications within the ecosystem.)

What's your best path to the cloud?

There are broadly three different ways that you can migrate your applications and data to the cloud:

- Lift and shift applies in circumstances where the entire application can be deployed to run in a cloud environment without major changes – for example within containers.
- Partial refactoring may be sufficient with reasonably modern structures where systemic capabilities are well separated from functionality in your software, and cloud enablement consequently does not require a complete rewrite.
- Complete refactoring may be necessary if you are confronted with a massive migration and complete architectural incompatibility (e.g., moving from mainframes to a cloud native environment).
- As you think about the best path to the cloud, consider performance, reliability, and vendor lock-in.

How will you protect data in transit and at rest in a new cloud environment?

Security and privacy concerns remain top-of-mind for enterprises considering cloud adoption, especially in regulated industries. Indeed, cloud adoption can create new entry points with vulnerabilities at all layers, situations previously masked by fenced-in environments and monolithic software.

As such, security and privacy should continue to be important factors as you consider how your software would run in the cloud. With more distributed architectures and new platforms, security is important at all layers of the product lifecycle when the cloud comes into the picture – not just during product design but also on an ongoing basis during its operation.

Cloud enablement in action

"Xoriant re-architected our desktop application for migration to a cloud native application on Amazon AWS, which enabled us to increase revenue by quickly providing relevant loan information at the borrower's premise."

– Leader at On-demand Mortgage Origination Solution Provider

AUTOMATION

Primary drivers: Maximize speed and productivity; minimize cost and risk

Today's business environment demands rapid responsiveness, and software development and deployment cycles have had to keep pace. Where production cycles used to be measured in years, now they are measured in months or even weeks. Speed to market is the key competitive advantage; in the digital world this means rapid changes to production with the ability to rollback.

As the complexity of software stacks continues to increase, and as software gets more distributed with more moving parts, automation is the only way to stay alive, let alone stay ahead. Continuous Integration/Continuous Delivery (CI/CD) is the ideal. To fully achieve it, you may need to transform your product architecture. But any level of automation is a step forward, and with the use of appropriate tools and methodologies available today, you can achieve significant wins without having to modify your core assets.

What automation might entail

These are just a few of the elements that might be involved in automation:

Development and operations

Agile programming enables faster deployment and higher quality, but it also increases complexity. When you're continuously making changes and the product is growing in an amorphous manner, with a big team of people deploying and testing code, the number of builds and test runs correspondingly increases. Add a modern, distributed architecture to the picture, and the number of different pieces that need to be kept in sync for each build can quickly become difficult to manage.

Modern distributed products have many moving parts, and the cadence of production deployment can vary for each set (e.g., the UI may be updated daily, functional capabilities updated weekly, etc.). The ability to put together the right versions of each component into the deployment candidate and to roll it out and roll it back as needed, immediately and without error, can only be achieved with automation. Fortunately, there are various tools and technologies available today to help (a major improvement from just a decade ago), and significant progress can be made without affecting the software itself.

The ultimate goal is to build automation and DevOps into your code, with self-deploying capabilities to achieve the true promise of CI/CD. Modern, cloud native applications with reactive capabilities are able to scale specific pieces of the software based on the load or such criteria, with automatic deployment and undeployment.

Infrastructure

Automation extends not just to the software that comprises the product, but also to the virtual infrastructure that most products run on. When your product is part of an enterprise's application set, it will need to be managed along with that set, and respond to the management tools that deploy, administer and undeploy it. If your product runs as a cloud-based service, then you will need to be able to manage the infrastructure it runs on. Virtualized infrastructures, especially container-based structures, are increasingly used today and can be spun up and down as need arises. Automation of these processes is more frequently being integrated into the software itself.

The benefits of automation can be significant. Automation saves labor (and costs), reduces errors, and improves reliability. For example, one client, a leading networked communication solutions provider, was running a Windows application with a huge set of regression test cases, which increased time to market due to manual QA efforts. And the client still faced production bug leakage and customer dissatisfaction. So we developed a solution for faster test data generation and integrated a test management tool and CI tool. As a result of implementing the automation framework, testing efforts were reduced by 70% and bug leakage was reduced by 40%.

Key considerations for automation

These are just a few of the key issues you might consider as you think about automation:

What is your process for automation?

Before automation tool selection and actual automation mechanisms can begin, it is important to develop and document a process for automation. Many elements of your product lifecycle process are unique to your business and your engineering organization. Process boundaries also tend to break at team boundaries, so you may have overall processes governing handoff between teams and internal processes within teams. Ultimately, what you are automating is a set of processes and activities, during which artifacts are created, modified and processed in steps, some of which are serial and some of which can be parallelized.

So implementing automation involves first a full understanding and documentation of the existing manual and semi-automated processes and sub-processes, intrinsic serialization patterns, artifacts that are involved, and all checkpoints – as well as the desired post-automation state. And because there is continuous evolution in the technology, artifacts and processes, automation is not an end-state. Your automation process should also include the ongoing efforts necessary to maintain and build on automation as your product grows.

Have you considered automation across the entire product lifecycle?

The ideal modern software product is automated throughout its lifecycle – from build, test, and deployment to various staging levels through production, and in the case of SaaS products, through operation. Modern software principles advocate building automation into the software itself so that the relevant portions can reactively deploy themselves (and undeploy) as needed. Automation also applies to key elements of the product's environment – e.g., automated management of virtual infrastructure.

Have you defined an automation roadmap?

Automation should be done in line with known and described policies, with audit mechanisms at each stage. The roadmap to achieve this can be a set of discrete procedures for each element (e.g., build automation, deployment automation, etc.). You can make significant progress without needing to re-architect the software.

> ### Automation in action
>
> "As a result of implementing the automation framework, testing efforts for regression test cases were reduced by 70% and bug leakage was reduced by 40%."
>
> – Leader at Networked Communication Solutions Provider

ANALYTICS ENABLEMENT

Primary drivers: Competitive advantage

Because analytics is critical for turning data into action, your customers are demanding analytics capabilities in the products and platforms they leverage. Four years ago, this is how Grand View Research sized up the Big Data market from 2014 to 2025.

A modern enterprise application incorporates analytics functionality that integrates with other analytics tools. Furthermore, modern products themselves are measurable and self-measuring, producing logs that can be analyzed to reveal powerful insights about user behavior as well as information about the products' own condition (e.g., performance, systemic health, etc.).

It is possible to achieve some quick wins by adding or increasing the analytics capabilities of your product without needing to make major architectural changes. As such, analytics enablement could be one of the early steps in the modernization process, and help buy you time before full architecture transformation.

What analytics enablement might entail

These are just a few of the elements that might be involved in analytics enablement:

Business visualization

In a digital business, the software is the entity interacting with the customer, and with other parts of the business, which are manifested by other software. (And increasingly, all business is digital business.) In these cases, the software needs to be measurable itself and needs to be able to produce the information that ultimately is used to run and grow the business it operates in.

Effective business visualization is about presenting the right information in the right way for the right people. (Because, for example, a CEO interacts with data in a different way than the CMO or the head of security does.) Business visualization should encompass a variety of visual and integration points,from mobile devices to other applications.

Data visualization in itself can serve as a competitive advantage. For example, we helped a global consulting firm build a risk assessment platform that the client then sold to its customers. The platform aggregates data from diverse sources, including social media and news channels, applies risk models in real time and visually presents the data on iOS and web devices to enable decision-making.

Predictive analytics

Products have traditionally provided some standard reports, which could be customized to get more information and/or information in a different format. Typically, these reports were based on some static data, which was generated by normal operation of the business. Yet in today's data-driven organizations, product users demand sophisticated analytics and dynamic reports.

Sophisticated analytics and dynamic reports rely on a variety of statistical techniques – including data mining, statistics, modeling, machine learning and artificial intelligence – to enable users to not only measure trends for the past business operations, but also make prediction and probabilistic

determination of future events. They also enable what-if analysis while changing one or more of the parameters.

Machine learning

An application of artificial intelligence, machine learning enables technology to learn, grow and change in a world where independent adaption is a valuable business tool. The iterative nature of machine learning is key. As models are exposed to new data, they are able to independently adapt, learning from previous computations to produce reliable, repeatable decisions and results. Today's tools give machine learning a fresh momentum, enabling systems to learn from data, identify patterns and make decisions with minimal human intervention.

Key considerations for analytics enablement

Can your current databases and reporting technologies handle much larger data sets?

Traditionally, reporting was done on a data warehouse or a transaction system and the datasets required for creating specific reports were small and manageable. Today, because the accuracy of predictive or probabilistic models improves as the size of the data set increases, modern data analytics requires a large amount of historical data.

Traditional databases and reporting technologies are not capable of handling the operations required for prediction, probabilistic analysis and what-if considerations. Much bigger and more powerful data infrastructure and visualization mechanisms are needed for the larger data sizes characteristic of a modern analytics environment.

Does your product enable real-time analytics?

Users expect modern products to enable them to react in real-time based on the conditions of the relevant parameters, as soon as a certain condition is reached. That means the product has to run analytics in real time, as events occur. There can be significant benefits in moving analytics "closer to the action" – users can decrease waste, increase efficiency, or enhance reliability, as the situation dictates. Yet historically, typical reporting and analysis systems have been based on static data, which is taken as a snapshot of historical business operations. These systems do not scale up to perform real-time analytics. Special systems, such as IoT and sensors, and special techniques are required to successfully deliver real-time analytics.

Can you provide analytics on a broad variety of data?

Traditional reporting and analytics systems were based on operational data, stored in a structured format in a static manner, with different data slices viewed at different angles. Traditional analytics environments comprised a traditional data store and a traditional reporting tool.

The new analytics data environment includes traditional data sources, as well as data from sensors and communications tools like emails, texts, news feeds and social media. The data volumes generated by these new data types and sources are much larger than the traditional data sources. And they need different kinds of data stores.

Mixing the traditional data and the new data becomes very challenging from the technical standpoint. Visualization of the combined data also is very challenging. So fully enabling modern analytics requires modernization of the data environment.

Analytics enablement in action

"Working with Xoriant as they developed a real-time monitoring system to identify geo-political risk of third-party suppliers from different countries gave us renewed confidence about the benefits we are able to pass along to our customers."

– Leader at Big 4 Consulting Firm

UX & ACCESSIBILTY

Primary drivers: Enhancing the customer experience and driving top line growth

The advent of the first iPhone in 2007 completely changed the concept of UI/UX in day-to-day technology products. Users became so used to these easy-to-configure, easy-to-use devices that they started demanding similar user interfaces for all the products/devices/software they touched. New technologies and techniques have made it possible to deliver that kind of user experience across websites, mobile applications and other products in general.

In fact, user experience has become so important that instead of the traditional sequence of building the functionality first, followed by the user interface as an afterthought, the sequence for building a modern application is to define the user experience/interface first and then build the functionality within the UX framework. So effective modern UX demands a customer-first approach rather than a feature-first approach. Yet adding new UX capabilities or channels to your existing product can be a quick win, and a strong step on the product modernization journey.

What modernizing UX & accessibility might entail

Emerging interfaces (voice, video)

From voice command and input to integrating digital screens into daily life, modern products push the boundaries of next-generation human/machine interfaces. In many cases, emerging interfaces can be built into existing products to generate significant benefit. For example, when we updated the user interface for a machine learning solution provider, adding multi-channel enablement, enhancing communications displays and improving search, the result was a 30% reduction in the clicks required to complete a task and a significantly shorter user learning curve. Ultimately, the enhancements improved decision-making for end users as the application is now more accessible and available on mobile with voice input capability.

Omnichannel UX (web, mobile AR, VR)

Omnichannel has been a challenge since the rise of different mobile devices and operating systems. Today the number and types of devices and systems is even more varied, with new channels coming online every day. Retaining context across channels – to provide users with a seamless experience – is as important as ever, but more difficult than in the past. Key is to consider how your product will work not only across specific new channels like AR and VR but whether your product is flexible enough to add channels as they arise.

Conversational UX (chatbots)

The rise of artificial conversational tools provides a significant opportunity for companies to provide enhanced support to their customers without the associated increased costs. Integrating such technology into existing support mechanisms can be done without too much disruption and can be another quick win without major architecture transformation needed.

APIs

APIs extend the reach of your products and services. For a leading provider of loan origination solutions, for example, adding an API enabled the company to expose some of the product's capabilities and expand their customer base significantly. Making the product into a platform enabled this company to get to non-core markets, with other people writing programs using their capabilities.

Providing APIs or adding the capability to existing products could be of critical importance to ISVs whose products have to exist within the context of other enterprise software. Building APIs to key portions of software products could be done without full architecture transformation and either as a prior step or in parallel. If designed well, the interfaces could remain post-modernization.

Key considerations for UX & accessibility

How will you ensure high-quality aesthetics and responsiveness?

The beautification of user interfaces has become a de facto standard for products irrespective of the industry or the function that the product serves. The high-resolution displays and high-fidelity sound sported by even the most common products have driven demand for increasingly sophisticated user interfaces. A highly aesthetic user interface shouldn't come at the cost of unresponsive or slower rendering of the audio/visual experience. Delivering that requires newer technologies, such as various frameworks of Java Script, that enable a high-class user interface with great responsiveness.

How will you balance ease of use and functionality?

Ease of use is a measure of how easy it is for the intended user to use the product. A common software ease of use metric is the number of clicks a user needs to go through to achieve a certain result. The easiest to use products are intuitive; they don't require user manuals or user handholding.

During the product design phase, there is often a tug-of-war between the desire to deliver both functionality and ease of use within the cost constraints. Adding functionality tends to make a product more complicated, and thus less easy to use. Sometimes, to achieve desired ease of use, some functionality must be sacrificed.

What's your plan to deliver a consistent user experience across all devices?

In the last decade we have seen an explosion in the number and types intelligent devices – computers, smart phones, tablets, smart watches, etc. Among these devices there are multiple display, audio, and touch capabilities; different manufacturers; and different form factors. And there is the distinct possibility that at any given time a user will access your product through any one of these devices. As such, it is essential to ensure a consistent user experience across all devices. Achieving that involves detailed planning and execution – especially since the release cycles of different devices by different manufacturers are getting shorter every day.

What does an elegant user experience mean in the context of your product? Are you set up to deliver it?

Delivering an elegant user experience does not necessarily mean having lots of bells and whistles. It is about providing the user with an intuitive, seamless and consistent experience across features, entry points, and paths. Is the navigation easy to travel through? Can users get "home" when they desire from any point in their journey? Is the experience intuitive? Is the font size clear enough to read? Is it clear what users should do next? Is the product still usable when the user has lower network bandwidth than usual? These are all important questions to ask – and answer – whether you're conceiving a new product or building a new UI for an existing product.

> ### UX & accessibility in action
>
> "We can easily count the benefits directly attributed to Xoriant's improvements. They helped us reduce clicks by 30% for any task, reduce the user learning curve and improve productivity."
>
> – Leader at Machine Learning Solution Provider for Enterprise Operations

ARCHITECTURE TRANSFORMATION

Primary driver: Modern business

Achieving true long-term agility, scalability and flexibility to adapt as technologies change can often necessitate architecture transformation for your software product. Modern software products need to be able to deal with modern environments characterized by:

- Massive volumes of multiple forms of data
- Massive numbers of users – including both humans and programs
- The need to leverage the data into measurable intelligence
- The need to support multiple types of devices and accesses
- The need to deal with immense loads on pieces of functionality
- The need to change rapidly and speed up versions to customers

Individual capabilities may be achieved without major architectural transformation as described in the other elements of product modernization – for example, by upgrading technology, enabling cloud, automating existing processes, enabling modern analytics, and adding new user interfaces.
But to be a truly modern product, it is likely that the software will need to be reactive, fully automated (CI/CD throughout), distributable (leveraging microservices), able to process multiple forms of data, support or even produce advanced analytics, and be accessible across multiple forms of interfaces, devices and channels.

Achieving that level of modernization will likely involve architecture transformation. The good news is that architecture transformation incorporates all other elements of modernization and – ideally – will position your product to thrive in a world where the only constant is growth and change. At the end of the day, your customers are the most critical drivers of your decisions about how, when and what to modernize. If they are upgrading their systems, you will need to do so as well.

What architecture transformation might entail

These are just a few of the elements that a modern architecture might include:

Reactive architecture

Major advancements in computers, systems and standards – e.g. IPv6, hyperconverged infrastructure, etc. – have changed what's possible to do with computers and given rise to the cloud and IoT. With that explosion has come dramatic load variability, as the same programs now have to handle huge numbers of requests at once (for example, millions of customers now wanting to access their bank accounts at the same time via their mobile devices).

In this kind of environment, reactively scalable architectures have arisen, allowing for scale out and scale back based on need. Software built this way can leverage the additional infrastructure available as needed via clouds (private and public). It also has the ability to scale just relevant portions of the software product as the need arises.

Microservices

Microservices architecture allows load isolation, fault isolation, and reduces dependency on single technologies. Breaking up a monolithic application into multiple microservices, if done right, can isolate functionality that gets

exercised during high load periods, and thus only the relevant portions of the application need to scale (reactively). Failure is also isolated to the microservice when it occurs and does not bring down the entire application. By building microservices with common platform architectures, components built with different technologies can co-exist.

New technologies like microservices and containerization also enable growth by allowing for build out or improvement in specific areas without affecting the entire application stack. They enable you to move quickly, and thrive, in a world characterized by constant change.

Polyglot persistence (i.e., multiple forms of persistent data)

Enterprise use of unstructured data is growing rapidly as organizations attempt to derive maximum value from their data. And because relational databases are typically not well suited to huge volumes of unstructured data, enterprise use of non-relational (e.g., NoSQL) databases is growing as well. So your customers are now operating with a hybrid of relational and non-relational databases. Which means polyglot persistence – the use of different data storage technologies depending on the data – is an essential characteristic of a modern product.

The benefits of polyglot persistence can be significant. For example, migrating one client from a traditional master data management system to a NoSQL database (MongoDB) we reduced time to import records by 70% and dramatically improved productivity by allowing for asynchronous execution.

Key considerations for architecture transformation

These are just a few of the key issues you might consider as you think about architecture transformation:

Are you transforming incrementally or all at once?

Architecture transformation can be done incrementally, or it can be an all-at-once radical change. Both paths have advantages and disadvantages. In the case of incremental change, it's important to consider whether the changes are going to make it into production, and if so, how much each incremental step delays the need to address a larger correction with customers. In the case of an all-at-once transformation, it's important to consider whether you'll have parallel teams working on the change and what risks exist in choosing a new architecture. Questions like this abound and must be dealt with before beginning. A large-scale architecture transformation impacts your overall processes and requires an ability to be agile and quickly pivot.

What are the new sources of complexity and potential points of failure?

Architecture transformation increases complexity and introduces new interdependencies. With more granular services interacting with each other, there are more possible points of failure. Security is just one example of a new source of complexity. With microservices, management becomes more difficult because there are more small pieces. An essential part of the solution is automation at every stage. Performance considerations exist as well. The ideal approach is holistic and involves all parts of the overall business architecture as well as the technology architecture.

Does your team have the skillsets to implement a new architecture with the least disruption?

It is likely or possible that existing teams have skillsets well suited for old technologies and architectures but not modern ones. Furthermore, engineering talent focused on maintaining current architecture may not have the skills to build new architecture, which can involve entirely new sets of skills. You need to decide whether it makes more sense to take

your existing team away from what they are doing today and training them to do the work of modernization or hire a separate team for architecture transformation and other modernization related work.

The CTO of a leading software firm realized pretty quickly that he wasn't in a position to hire all the people needed to get the job done in the time span needed. He needed a partner with expertise in microservices, event-driven architecture and more. Xoriant was the right partner for his project.

Architecture transformation in action

"Xoriant re-architected our legacy desktop application to create a SOA and microservices-based SaaS platform, which accelerated mortgage processing speed while reducing costs by improving collaboration and visibility across various stakeholders such as lenders, service providers, and ISVs."

– Leader at On-demand Mortgage Origination Solution Provider

Are you ready to begin your Product Modernization journey?
If this e-book has sparked your interest, we can connect you with one of our experts to further explore the process and how we can help.

Click here if you're ready now for a free Product Modernization Readiness Assessment

If you'd like to explore around a bit more, check out the Product Modernization Hub, where you can access the interactive map and get other tools to help you navigate your product modernization journey.

WHAT'S NEXT?

Now that you've taken the journey through all of the elements of Product Modernization let's talk about your needs. Are you looking to "go bold" with an all-at-once architecture transformation, or take smaller steps by incrementally modernizing, perhaps starting with technology upgrade or automation?

Whatever the case, Xoriant's team of engineers can help you assess your needs, identify opportunities to make significant improvements quickly – and help you visualize the big picture.

Having spent over 25 years in Silicon Valley, building technology products, we have extensive experience in technology ecosystems. We have the experience to identify the right technology and proven technical expertise for every element of the modernization journey to build, integrate and scale it to deliver an end-to-end solution to your product modernization challenges.

We have a firm belief in the value of prototyping and work alongside our clients to ideate and build next-generation solutions based on flexible platforms that we have developed. We integrate those solutions within our clients' existing ecosystems to maximize technology investments already made.

We invest in the next-gen technologies and skillsets that are the foundation of product modernization. In addition to expertise in the latest protocols, Xoriant has proven partnerships and experience providing next-gen IT services and software products.

Last but not the least, while the companies engage their own resources on the modernization journey, someone needs to manage the current product with respect to simple upgrades, support and other customer related issues. Xoriant teams can help you there too.

Bottom line: We have the people, technology know-how and experience proven to help you achieve the benefits of modernization while mitigating the disruption, or avoiding the risks.

SEE XORIANT IN ACTION

Cloud & Mobile Enablement through Partial Refactoring & Re-architecture of Hosted Compliance Solution Increases Customer Base by 20%

Re-architecture of Monolithic Legacy Product to Microservices- based Platform Ensures Data Security, Consistent UI and Cost Savings

Migration from Relational Database to Polyglot Persistence (NoSQL) Improves Speed by 70%

User Interface Modernization with Multi-language Voice Channel Enablement Raises Productivity by 30%