EXECUTIVE SUMMARY

Organizations are at a strategic crossroads. According to IDC analysts,¹ two-thirds of CEOs will center their corporate strategies around digital transformation initiatives. The goal of digital transformation is only partly to make existing functions more effective. The greater goal is to be able to do new things and use existing data in new, better ways.

Yet, 72% of current IT spending— and the focus of IT resources— is on maintaining existing systems. This creates tension between what needs to be done today and what should be done for the future.

Middle-tier applications, according to IDC,³ are the core of a digital transformation strategy because they provide data integration, messaging, and application programming interface (API) management. Middle-tier applications also provide a platform for application development and management that can handle traditional enterprise applications and cloud-native, distributed applications.

Specifically, a Java™ EE-based application platform can provide support for current technologies and cloud-native applications by:

- Using existing staff and domain knowledge for new technologies.
- Preserving legacy applications and important data.
- Developing new applications in parallel with the existing environment.
- Implementing new processes and architectures.

---

Digital transformation will look different for every organization. As each organization defines its own unique strategy and goals, there are core principles that can help guide those decisions and harness existing IT resources for the greatest effect.

**A LOOK AT THE FUTURE: PLATFORM, PROCESS, ARCHITECTURE**

In a presentation on the current state of software development, IDC analyst Al Hilwa noted that good software is a function of software architecture, developer processes, and developer skills. IT and operations departments are facing significant changes in all three. As cloud and distributed computing have introduced new economies of scale for infrastructure, these three fundamentals—architecture, process, and platform—are evolving to take advantage of cloud environments.

**Platform: Cloud**

Cloud computing will be the core infrastructure of the next wave of IT innovations. Analyst group IDC estimates that up to 70% of total infrastructure expenditures will be for cloud services by 2020. The key to the effectiveness of cloud is its responsiveness to changing environments. Virtualization is a major IT driver because it abstracts the operating environment from the physical environment. Virtualization treats the operating system as entirely separate from the physical system, which allows multiple operating system instances to be installed and run using the same hardware. Cloud breaks down the environment even further, abstracting the running applications from the underlying operating system or physical environment.

Cloud-based services exist across a distributed group of nodes rather than being strictly assigned to a physical system. This provides redundancy (if a node fails, other nodes carry on the operations) and scalability, since new instances can be created and destroyed according to demand for a specific service.

This lightweight infrastructure makes it possible to support highly decentralized architectural patterns (like microservices). Cloud environments also introduce operating efficiency and cost savings by maximizing resource utilization.

---


6 Ibid., Gens.
Process: DevOps and agile

Agile, continuous integration/continuous delivery (CI/CD), and DevOps are related concepts that are essential to how modern IT teams are structured and function.

Agile is an approach to project planning and execution. An agile approach, as defined in the Agile Manifesto,\(^7\) has four core principles:

- Placing the people/team first.
- Collaborating with external groups.
- Responding to changing situations.
- Creating simple-to-understand software.

Agile processes break project functions down into manageable tasks, with the goal of iterating quickly through tasks and delivering functioning software feature by feature. This is in contrast to the confined, sequential approach of waterfall projects. The agile approach is gaining popularity; waterfall projects have dropped to below 50% according to a recent Gartner assessment.\(^8\)

One change agile approaches have introduced to project development is involving other groups in the entire planning and development process, rather than after development is complete. At the end of a sprint, software has to be functional, which means it has to be tested as it is developed. Agile teams include testing within the development cycle. This idea of continuous testing became continuous integration, and the rapid releases became continuous deployment. These are now ongoing processes rather than defined and discrete stages.

---


But the application life cycle does not end at delivery. The application has to be deployed and maintained. That job has typically fallen to operations and IT departments rather than development. If these teams are too separated, the developers may not understand the operating environment, or the operations team may not understand the strategic purpose of the product. This gap in understanding can create an environment where software or infrastructure is performing below expectations. The practice of bringing development and operations teams together is DevOps.

**Architecture: Microservices**

Enterprise software has traditionally been very large. A single application was expected to handle peak loads for an entire enterprise. Any functions required by that application had to be performed by that application. Services were just features of one monolithic application. A good example of a monolithic application is a database, where a single database can service an entire organization.

As enterprise computing demands became more complex, the maintenance requirements for monolithic applications increased. Even minor changes to a monolithic infrastructure can be difficult because changing part requires changing the whole. This led to a new approach to architecture called service-oriented architecture (SOA). Rather than using a single application that did everything, within a service-oriented architecture, some functions can be supplied by different applications that are loosely coupled through an integration pattern, like an enterprise service bus (ESB).

SOA introduces additional system complexity into the overall environment. While certain aspects of the architecture are made easier (such as introducing new components or performing upgrades), SOA also risks cascading changes throughout the environment if component interactions are not clearly understood.

![Figure 3. Service relationships in different architectures](image)

Although SOA is a move in the right direction, a more mature model, called microservices architecture, has evolved. Microservices architectures are similar to SOA patterns in that services are specialized and loosely coupled, but microservices are broken down even further. A microservices architecture defines services in a very clear way:
STILL NUMBER ONE

"Java has a very very bright future. Languages and platforms do not change overnight. There is a lot of exciting technology, most of it in open source, but there's nothing that can unseat Java right now. If you look at some of the work around microservices, a lot of that is Java-based. They are really redefining enterprise Java and recasting it in this modern paradigm with open hybrid cloud computing. There are great things happening, and all of it is happening in open source."

RICH SHARPLES
SENIOR DIRECTOR OF PRODUCT MANAGEMENT FOR MIDDLEWARE, RED HAT

• A single, clear purpose
• Well-defined parameters
• Polyglot implementation

The services within the architecture use a common messaging framework, like REST APIs, which they use to communicate with each other without difficult data conversion transactions or additional integration layers.

This messaging framework allows, and even encourages, faster delivery of new features and updates. Each service is discrete. One service can be replaced, enhanced, or dropped without affecting any other service in the architecture. This lightweight architecture helps optimize distributed or cloud resources and supports dynamic scalability for individual services.

Conway’s Law and the future

Cloud, DevOps, and microservices share a common trait: distributed complexity. In 1967, software developer Melvin Conway\(^9\) observed that software is designed in patterns that match the communication structure of the teams involved in developing it. The more rigid, opaque, or incomplete the communication, the more poorly software was executed.

Wherever your organization is on the path to digital transformation, the end result will always reflect the culture of your IT teams. As such, there are some cultural\(^10\) considerations before addressing technology.

• Establish well-understood communication patterns across teams.
• Minimize barriers between groups.
• Promote flexible infrastructure and bureaucracy.

Microservices, like any complex, distributed architecture, rely on a solid foundation. Technologically, microservices are not possible without a cloud-based infrastructure.\(^11\)

Organizationally, microservices architectures are bound to fail without a solid DevOps or agile working environment—one built on collaborative, highly communicative, cross-functional teams.

THE PAST AND FUTURE OF JAVA EE

If the future holds the productivity—and complexity—of cloud-based, lightweight architectures, how can you move your current IT department into that future?

Since its introduction in 1995, Java has grown to be the most popular programming language in the world.\(^12\) Oracle put the global population of Java developers at over 9 million.\(^13\) This is nearly 82% of the 11 million professional developers worldwide.\(^14\)

---

Sun (and later Oracle) built on the Java programming language by defining a set of standards for common operations, a set of APIs, and a runtime environment for applications. Together this is known as Java Enterprise Edition (Java EE). Servers that implement the Java EE specification are Java application platforms, and these application platforms are the core of many IT development environments. Java’s server-client model of programming made it well adapted to early internet frameworks and later enterprise-wide applications.

In the early 2000s, when application platforms were being developed, most IT architectures were built on monolithic applications. Java EE application platforms were expected to host multiple Java applications in a single, centralized location. In practical applications, due to traffic distribution, network bandwidth and latency, redundancy, and organizational segregation, an application platform often only hosted a single application. Gartner referred to these Java application platforms as “superplatforms,” and they tended to waste IT resources.

One of the reasons Java is so resilient as a language and a platform is its adaptability. Java and Java EE are defined through a community called the Java Community Process. In 2017, Oracle moved Java EE to the Eclipse Foundation under the new name Jakarta EE, which will evolve enterprise Java into the world of cloud, containers, and microservices according to the Eclipse Foundation Specification Process. This Java community is active in Enterprise Java development and in related community projects like MicroProfile, Thorntail, and Node.js.

This community is steering Java and Java platforms in a direction that allows cloud-native development and deployment of lightweight, interconnected applications. Java EE 6 introduced the concept of profiles, with a full profile that matches the traditional, full-scale web server and a web profile for lighter-weight applications. Profiles and modularity were extended in Java EE 7. The chief focus of Java EE 8 is support for HTML5 and the HTTP/2 standard, as well as enhanced simplification and managed bean integration and improved infrastructure for applications in the cloud. Several community projects have pushed it even further.

• MicroProfile is a community dedicated to optimizing the Enterprise Java mission for microservice-based architectures. The goal is to define a microservices application platform that is portable across multiple runtimes.

WHAT’S YOUR PATH TO MODERNIZATION?

• Move into the cloud.
• Use an application platform that performs equally in the cloud or datacenter.
• Use your business strategy to drive your IT strategy.
• Decide what aspects of the infrastructure to modernize and how.
• Build your teams and culture.
• Give your developers the right tools.
• Design an IT ecosystem that supports integration, scalability, and interoperability.

• Thorntail treats Java applications as a container image, with all of the required—and only the required—libraries and dependencies bundled into a single fat jar, a standalone Java archive.

Java, like most technologies, is adapting to cloud infrastructures with a crucial advantage: The core language is a technology already understood and used by millions of developers. The architectures and implementations for the applications are new, but the skills needed to create them are already established.

CHANGE THAT MAKES THE MOST OF YOUR RESOURCES

IDC calls cloud the core of the Third Platform IT environment.\(^\text{16}\) IDC defines three epochs in the evolution of computing:\(^\text{17}\)

1. Mainframes and personal computers
2. Internet-based transactions and client/server architectures
3. App-centric technologies (like mobile, social, Internet of Things, and big data) hosted in a cloud

ACCELERATING PACE OF CHANGE
INCREASING DEMANDS ON IT. AND INFRASTRUCTURE

This third platform relies heavily on the other two, but it is not about doing the same thing more efficiently. It is about building on those layers to do new things.

1. Move to the cloud (with Java EE)

Cloud computing is at the core of any modern IT environment. This is because of cloud computing’s scalability, especially the ability to dynamically add and contract nodes on demand in ways that are not possible with traditional physical or virtual systems.


\(^{17}\) IDC. IDC Predicts the Emergence of “the DX Economy” in a Critical Period of Widespread Digital Transformation and Massive Scale Up of 3rd Platform Technologies in Every Industry. 4 Nov. 2015. Web.
With technology platforms like Java EE, it is critical that the platform is able to span multiple types of environments, such as:

- On-premise.
- Public cloud (such as Amazon Web Services, Microsoft Azure, or Google Cloud Platform).
- Private cloud (such as an OpenStack® private cloud).
- Containers.
- Hosted services.

Applications must be able to run in all environments in the same way. Some application platforms may not work in all required environments or may not have feature parity between those environments. Picking the right Java EE platform ensures the interoperability that is crucial to mixed environments.

Figure 6. Overview of a hybrid cloud environment
Along with the complexity of operating in hybrid environments, there is a financial cost to operating in multiple environments. Gartner has identified software licensing as a potential cost reduction for IT departments, calling it the most overlooked area for cost reductions. Software pricing can be complex, with different licenses for on-premise, virtual, or cloud and additional subscription services for different types of support.

2. Ask existential questions

The way to determine the best option for your environment is to balance the risk of change against your strategic goals. The requirements are different for every organization. Your digital strategy should reflect your business strategy.

- What are your organization’s goals or strategic direction?
- Do you have the skills and resources available to execute that strategic direction?
- Do you have an environment that can sustain those applications?

3. Decide what and how to modernize

The core question of digital transformation, according to Peter Marston at IDC, is how to approach application modernization. Based on their research, about 40% of organizations either have set application modernization as their top IT priority or they will within the next three years.

There are several ways to approach modernization:

- Retrofit or re-architect existing applications into a modern environment.
- Migrate an application from one environment to another.
- Replace existing applications with new applications.
- Create parallel environments.

---

The approach you choose depends on where on the journey to microservices your organization wants to go.

One goal is to decrease IT’s focus on maintenance. The ideal split for IT’s spending, and focus, is evenly divided between new and existing projects. However, nearly three-quarters (72%) of IT spending is for maintenance and operational projects. The majority of CTOs (63%) think that is too high.\textsuperscript{20}

A cloud-based Java EE foundation is helpful for shifting maintenance costs. Both existing and new applications can run on the same foundation, even if one is a monolith and the other is a microservice. More importantly, Java applications can ultimately be migrated between environments.

This portability reduces the transaction costs and risks of migration while providing a pathway to modernizing applications. This allows your organization to change its infrastructure incrementally. First, getting the cost benefits of cloud and, next, the operating efficiencies of containers—without having to commit to microservices or highly distributed architectures unless it fits your strategy.

4. Focus on your teams

One of the major challenges facing CIOs is a perceived lack of IT talent capable of performing in the new third platform. Fifty-nine percent of CIOs in a Harvard Business Review analytics report on the IT talent crisis believe a skills shortage prevents them from keeping up with their IT or strategic challenges.\textsuperscript{21} The areas of concern center around IDC’s Third Platform:

- Big data and analytics (36%)
- Architects, either enterprise (27%) or technical (24%)
- Development (27%)
- Mobile development (24%)
- IT strategy (22%)

Nigel Fenwick of Forrester Research noted that IT organizations often try to accomplish every IT objective with a custom solution, and that is not necessarily the best strategic approach. “[W]e’ve plowed millions of dollars into software customization to support generic capabilities. It has made IT more complex, made interfaces more difficult, reduced IT’s agility, and added cost.”\textsuperscript{22}

Rather than trying to do everything, simplify and focus your IT strategy by doing the following:

- Identify two or three core capabilities that the organization must deliver as a strategic objective.
- Use standards-based solutions rather than custom solutions for all other requirements.
- Select solutions that are easy to maintain.

The biggest skills gaps identified by CIOs were related to processes, not technology. Using open standards can help reduce the learning curve on technologies, as can common technologies like Java. Implementing a Java EE application platform as a development platform allows IT departments to

use their existing knowledge and experience to develop projects focused in new areas. This widens the pool of potential contributors because Java experience is commonplace in the professional development community.

5. Give developers the right tools

Integrated, developer-focused tools in the technology platform help to make the overall development cycle smoother, especially with integrated testing or automation. There are several key areas where developers see direct benefits:

- Integrated testing modules for CI/CD
- Automation for deployments
- Developer tool kits
- Custom class loading
- Performance

Even small performance features can have a significant impact. For example, faster start-up times when deploying applications can potentially save hours per week in developer productivity, simply because a developer may restart an application multiple times per day as part of regular development.

Other features, like integrated testing, are crucial for DevOps or CI/CD. One Australian information technology company improved developer productivity by 15% simply by loading testing modules as part of the deployment process. Integration between development, testing, and operations helps improve code quality and shorten the development life cycle.

6. Build an ecosystem

IDC analyst Maureen Fleming wrote that the strength of digital transformation is integration. The Third Platform has multiple facets that can work together, including:

- Virtualization, public clouds, and private clouds.
- Containers and orchestration.
- Data virtualization.
- Memory caches and storage.
- Multiple messaging protocols.
- Different data formats from different data sources.
- Management and deployment tools.
- Test automation.
- Business process automation.

---


When designing a platform for a digital transformation initiative, the actual platform is only a part of your initiative. There is a much larger ecosystem where that platform, and the applications on top of it, will be functioning.

Peter Martson of IDC recommended looking at the broader ecosystem of an application platform provider as part of the larger strategy for application modernization. A solutions provider that can offer experience and guidance across technologies and skills can help in a variety of ways, from architecture planning to cloud provisioning.

CONCLUSION

Digital transformation is a strategic approach to IT that treats IT infrastructure and data as a potential product for customers. Digital transformation requires shifting perspectives from internally focused applications, serving internal business functions, to treating applications as potential products, looking at new ways to use data and data sources and looking at new ways to engage with customers.

Moving forward with digital transformation requires three things:

- Clear organizational goals and strategy
- Solid, collaborative, and communicative processes across functional teams
- A disciplined and well-understood distributed architecture

Java applications have been a central technology for enterprises for two decades. This wealth of data, functionality, and knowledge are critical to enterprises. With Java-based applications, modern development can build on a platform that enables cloud-native architectures while simultaneously supporting existing applications. This combination of traditional enterprise-wide monoliths and cloud-based application deployment allows organizations to take advantage of existing knowledge and resources while actively moving toward newer application models.

Using a cloud-friendly Java application platform benefits organizations in several ways.

- It mitigates the impact of skills gaps or technology gaps within the IT department.
- It widens potential resource pools.
- It provides a bridge migration strategy for existing workloads while allowing development in new environments.
- It provides an environment for other technologies to be introduced for integration, data management, or other strategic initiatives.

An appropriate Java application platform can do more than manage existing enterprise applications. It can become a method to achieve the digital transformation that executives are planning, while maximizing existing IT resources and continuing critical maintenance projects.

---