RED HAT CONSULTING

redhat.

WHITEPAPER

MODERNIZING APPLICATION DELIVERY WITH CONTAINER PLATFORMS

EXECUTIVE SUMMARY

Business managers want new applications and features faster than ever. Developers need efficient, easy-to-use tools for building and releasing software. Administrators want to use resources effectively, maintain security, and comply with regulations. And every team throughout an organization wants to control costs.

Improved approaches, collaboration, and technology have led to faster software life cycles. Linux® containers and container application platforms, a new category of enterprise software, can help businesses benefit from these advances. But to take advantage of new technologies, IT managers must consider a wide range of activities, teams, and processes that together form the desired but elusive DevOps culture.

An effective adoption program—using practices guided by experience—can lead organizations to successful container-based software delivery infrastructure. This infrastructure helps organizations deliver applications that are thoroughly tested, quickly released, frequently updated, widely replicated, and efficiently maintained and monitored.

THE EVOLUTION OF CONTAINERS

In-house software development has become central to business strategy, increasing demands on internal development teams. In addition to high quality and predictable cost, time to market is now a critical objective. Development teams must not only successfully release new applications, but also update those applications often to respond to constantly evolving markets.

Agile and other iterative methodologies help developers create and adapt software quickly, but the process often slows when it is time for IT operations to release applications into production. Development and operations teams may each work successfully, but the two organizations were created to work separately, rather than as one integrated team. As a result, they often lack trust and communication, as well as shared standards, practices, and tools that make collaboration easier. IT operations is often seen as impeding the rapid release of software, but their concerns—such as security and compliance—are equally valid.

To help IT operations keep up with developers, organizations have evolved to embrace DevOps. With a DevOps approach, teams use automation, as well as continuous integration and continuous delivery (CI/CD) practices, to release software quickly. Still, developing and deploying applications for multiple environments is complex—even more so when web, cloud-ready, and mobile applications are built from highly distributed microservices.

facebook.com/redhatinc
@redhatnews
linkedin.com/company/red-hat

redhat.com

Containers have emerged as a solution. Developers can use containers to bundle an application and all of its runtime dependencies—such as libraries, servers, and system resources—into a well-defined, portable artifact. Containers combine the potential for rapid delivery with the flexibility to define the details of the underlying platform, from the development language to the operating system.

Additionally, with organizations aligned more closely on goals and processes, containers provide a common framework and language that tie development, operations, security, quality assurance (QA), and other teams together. Together, they can use containers to build and deploy applications at a cadence that meets business needs.

CONTAINERS IN THE ENTERPRISE

This alignment of IT functions to rapidly develop and continuously deploy applications is an improvement, but further progress is needed. Complex applications are built from services that run on different hosts. These hosts are potentially organized into separate tiers and might be physical or virtual servers, running in private or public clouds. As a result, there are typically multiple instances of each service to provide greater availability and scalability. Over time, an application may have workload cycles, during which capacity is expected to scale up or down as needed.

Once an application is in use, stakeholders want to know how much it is used and how it is performing, including any issues—preferably through a consolidated report for all components and instances in all locations. Operations needs a way to track resource use by different applications to charge related expenses to the appropriate business unit. And when issues occur, an aggregated view of all activity across all components and instances can be helpful.

In addition to the complexity of a single containerized application, there is also the broader operational need to comply with regulations and provide authentication, authorization, and privacy for use of application components.

THE BENEFITS OF CONTAINER PLATFORMS

Container application platforms help enterprises address the challenges of deploying containers. These platforms let administrators monitor, manage, secure, and scale containers similar to how they treat existing servers and virtual machines. Container application platforms not only provide basic management of containers, but also orchestrate distributed containers. Further, these platforms extend management throughout the application life cycle—beyond development and deployment to ongoing production use.

With container application platforms, administrators can define a set of standard operating platforms to provide developers with the flexibility to choose while encouraging commonality in tools and architecture. This standardization reduces technology isolation. In addition, developers can quickly provision new instances of the standard platforms themselves, without intervention by IT administrators.

In addition, container application platforms make other difficult, time-consuming operational tasks faster and less complex by providing:

- Easy access to persistent storage for stateful applications or application services.
- Simplified load balancing, scheduling, and automatic scaling by offering these critical functions through configuration.
- Streamlined deployment of complex applications to physical or virtual and public, private, or hybrid cloud infrastructures using the same underlying container orchestration and automation system.

Any container execution environment should provide a way to monitor an individual container. A container application platform, such as Red Hat® OpenShift Container Platform, provides a view across the entire infrastructure and lets administrators create groupings of an application's containers. Administrators can then configure networking between these containers and specify their access and security, resource requirements and prioritization, and other properties at the application level. Container application platforms provide an application view across all of the application's containers, regardless of the number of copies or their location.

With these capabilities, a container application platform is appropriate for the most critical applications. Legacy applications also often benefit from redeployment as containerized applications.

MODERN SOFTWARE DELIVERY

The container application platform is a new category of enterprise software that offers more powerful technology for developing and delivering applications. However, just as the first relational databases needed new roles, procedures, architectures, and workflows, container application platforms also require organizations to adapt teams and work processes to ensure the technology is productive and sustainable.

As a reflection of this requirement, DevOps is an attempt to address the disconnection between IT teams through standardized, automated software delivery. With a DevOps approach, roles do not change, but some responsibilities are refined to give team members greater autonomy and reduce challenges. Standardizing and automating processes and technology makes software delivery more efficient and accurate.

Process standardization defines the exact sequence of steps needed to achieve a task, such as releasing an application to a new environment. Processes are standardized regardless of how they are executed. Technology platforms can be standardized while preserving team autonomy, giving developers and operations teams choices based on relevant skills and application suitability.

Standardized processes can then be automated. Automation is key to CI/CD, which are critical for aligning software delivery with business objectives. Automation can be developed to run with carefully controlled checkpoints or with minimal intervention. For example, when a developer checks in new code, it can be automatically integrated and built into an executable format, then automatically tested. Administrators can also develop automation that requires a user to trigger a process. For example, a developer would initiate a request for a new development environment, but the environment is automatically created and provisioned without operator involvement.

Within the high-level activities of the application life cycle—including development, integration, testing, release, monitoring, and maintenance—fine-grained tasks must also be defined. These tasks are not purely technical and may extend beyond built-in system capabilities. For example, to avoid accumulating unused containers that use resources unnecessarily, rules can be established for when these unused containers can be shut down. In addition, most organizations have a method for charging IT costs to the business that must be adapted for a shared, container-based platform.

A highly automated, container-based environment offers considerable benefits to both developers and operations staff. Developers can spend more time on actual development and less on building platforms, and accelerated feedback from product owners lets them quickly build features and fix issues. Automated testing finds problems before release, leading to higher-quality software. And automated platform builds create greater consistency between platforms—such as QA, test, and production—to reduce the time spent discovering configuration differences. Operations staff benefit from built-in availability and scalability that is implemented the same way for all applications. Additionally, the need to support custom platforms decreases. Automated deployment frees operations from deploying urgent application fixes by eliminating human error. Finally, a single, consistent platform is easier to secure and monitor.

CONTAINER ADOPTION

Enterprises need a flexible program of connected projects to assess their current application development and gradually migrate to a modern, container-based approach. These projects should use technical, tactical approaches to demonstrate success quickly and create the broad organizational change needed to dramatically accelerate software delivery. The ultimate focus of the program must be on creating the cultural changes needed for DevOps improvement.

Innovative new technology and processes are not always welcomed by developers or operations staff. This new approach is not harder, but creates a challenge to product owners, developers, and operations staff who are comfortable with their current methods. People often resist changes that are introduced without their participation. An approach designed to get the support of all participants through naturally developed enthusiasm should:

- Include all stakeholders in the earliest planning stages.
- Demonstrate and celebrate early, measurable success.
- Recruit volunteers who have experienced the changes in their organizations to promote them to other teams.

This new approach is implemented through an iterative program that includes well-defined tasks, desired results, and milestones—with the goal of demonstrating business value in each iteration. The program contains phases that are divided into three workstreams:

- **Infrastructure:** The servers and software needed to build the container platform and integrate it with enterprise systems.
- **Release management:** Standardization and automation of processes for deploying and delivering applications.
- **Development:** Includes the processes and tools developers need to build containerized applications, migrate existing applications to containers, or develop microservices that use container orchestration.

DISCOVERY AND DESIGN

In the initial phase, stakeholders from all related teams are involved in initial planning. With emerging technologies such as containers and container platforms, a short, timed pilot implementation ensures that the team understands the technology prior to the first design discussions about long-term implementation. A reference implementation of the container application platform is built, and a small set of applications can be migrated to containers to demonstrate the platform in the actual environment. This implementation precedes the design workshop, helping inform it by giving participants an early view of the platform in action.

IMPLEMENTATION AND AUTOMATION

In the next phase, the program's workstreams can diverge to be completed in parallel. The infrastructure team can build the full container application platform defined in the first design iteration. The production implementation's design will evolve over time to define minimal viability for the platform. At the same time, a release management team can begin to design and build the CI/CD pipeline, including all software development components—such as versioning, artifact repository, and automation for build, testing, and deployment stages. This work can continue in subsequent iterations. Finally, application development teams can be trained on tools and techniques for containerized development and begin to develop applications using this platform.

In the next phase and any additional phases needed, release management continues to build the end-to-end automated pipeline, and processes are expanded to include automatic delivery of containerized applications. Application teams start migrating existing applications to the container application platform. However, not all applications need to be moved, and teams must learn how to analyze an application and assess the potential migration effort. Some existing applications can benefit from a redesign that divides them into microservices, and in these phases, development teams also develop better understanding of container-based microservice approaches.

CONTINUOUS IMPROVEMENT

In subsequent phases, feedback from earlier stages generates continuous improvement. The infrastructure team can work on increasingly sophisticated integration and management, including adding identity management, dynamically provisioning infrastructure based on container platform capacity, and developing operations for patches, upgrades, and disaster recovery. The release management group can create more elaborate automation approaches and develop metrics to optimize the overall pipeline based on empirical data. Application teams can use approaches and lessons learned from initial application migrations to containers to reduce time and costs for successive migrations. And microservices development teams can continue to expand services to create fully featured, flexible distributed systems with comprehensive telemetry that helps identify performance bottlenecks and other areas for improvement.

CONCLUSION

Containers represent a modern era of IT transformation, similar to the virtualization era. Unlike virtualization, however, containers reach beyond IT operations and require detailed input from multiple organizations. As a result, businesses must change how these organizations work together. Successful container adoption depends on people and process as much as technology.

A well-planned but agile program addressing people, process, and technology can help businesses achieve this success. The phased approach described demonstrates early value and gains support as it builds on this success in later iterations. Separate workstreams let each team proceed at their desired pace—according to their ability to accommodate change—while still progressing on the larger plan.

Using technology as a foundation for more substantial organizational change, businesses can achieve the efficient software delivery that has become key to success.

Red Hat Consulting uses the unified approach presented in this paper to help organizations achieve the full value of containers. To learn more, read the datasheet at redhat.com/en/resources/modernize-application-delivery-container-platforms-datasheet.