Technical white paper

Bridge Development and Operations for faster delivery of applications

HP Continuous Delivery Automation software

Table of contents

Application lifecycle in the current business scenario

Businesses across varied markets continue to manage the software development life cycle (SDLC) for business applications and processes to deliver and support their products and services. In the current SDLC model, the key factors that drive planning, development, release, and operations toward faster time-to-value are:

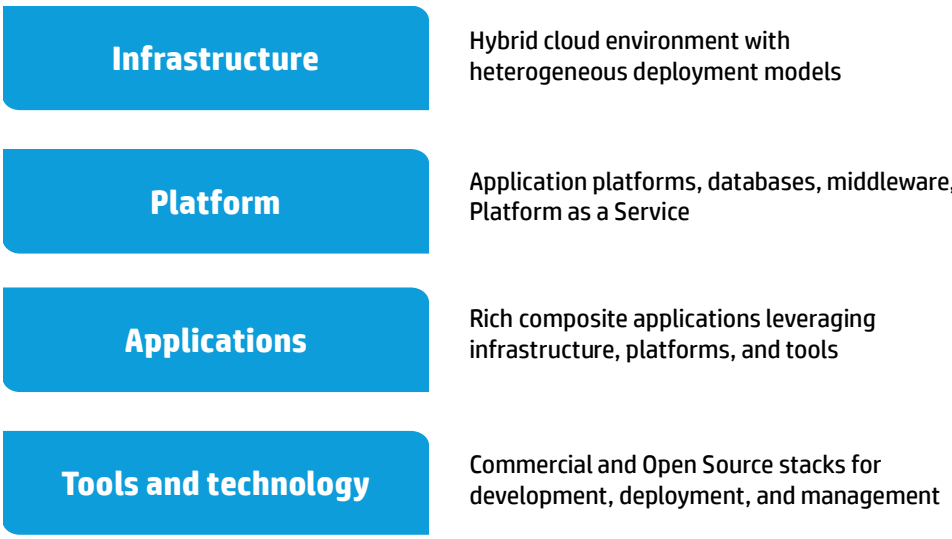
- Agile development
- Continuous integration and delivery
- Lab automation

Combined with disruptive technologies such as cloud computing, virtualization, and consumerization of IT, organizations are now required to manage the end-to-end lifecycle of modern composite hybrid applications. These applications leverage the following entities to set up and manage the composite application ecosystem:

- Heterogeneous infrastructure (physical, virtual, cloud)
- Diverse platforms (DBs, middleware, application servers, and message buses)
- Application software using commercial and Open Source fulfillment and assurance tools

In this end-to-end delivery lifecycle, aspects such as agile development and continuous integration have matured and evolved through generations of tools and techniques. Organizations, however, face new challenges when extending continuous integration into continuous delivery. Challenges include consistently deploying composite applications through development to production environments while considering the differences in the environments (infrastructure as well as tools used) and negotiating the gaps across organization silos. Figure 1 illustrates the different tiers of an ecosystem in the current business scenario.

Figure 1. Tiers of an IT ecosystem



DevOps—An overview

The DevOps movement works toward bridging the gaps between development and operations environments using a set of principles, methods, and practices enabling better collaboration, automation, and governance. The goal is to extend continuous build/assembly integration into repeatable and consistent application deployment across heterogeneous environments.

DevOps provides key values to organizations such as:

- Enabling application developers to use IT resources directly, according to centralized policy.
- Standardizing application and release versions, as well as lifecycle promotion
- Modeled artifacts that normalize releasing application into production and include performance monitoring, workload management, and security binding.

HP Continuous Delivery Automation—Enabling the DevOps environment

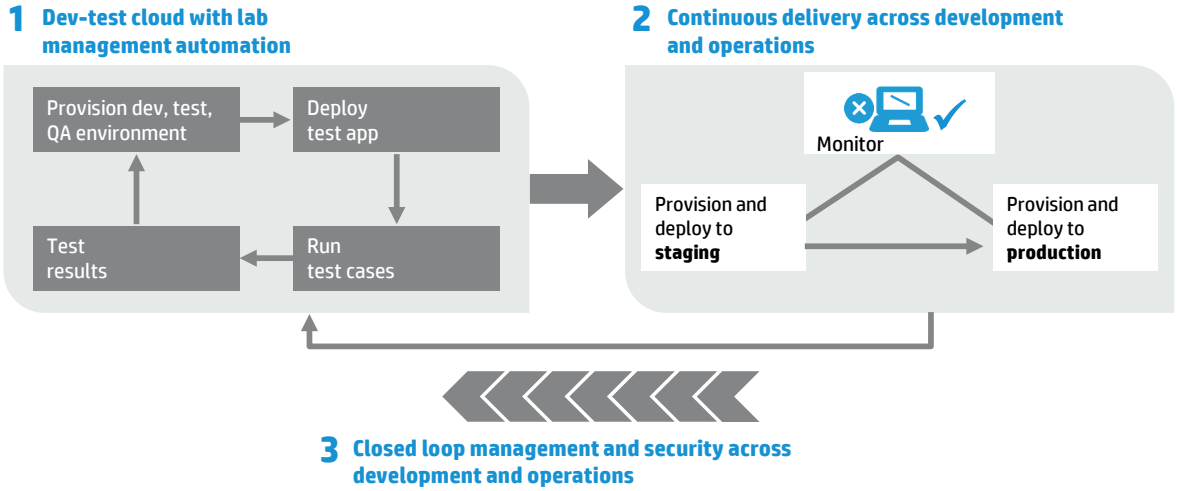
The new Continuous Delivery Automation (CDA) product from HP provides a model-driven approach to DevOps collaboration, automation of application deployment, and monitoring. This enables customers to focus on their core applications and in driving business values while reducing costs, risks, and time to perform provisioning and deployment tasks. The core features provided include:

- Modeling the application and infrastructure configuration to deliver infrastructure as code for deployment and provisioning
- Full artifact version control, role-based access, lifecycle management, and definitive software library (DSL)
- Application deployment management using configurable tool options
- Infrastructure provisioning management across hybrid environments
- Embedded monitoring deployment in conjunction with application deployment

The application release process

To bridge the gap between development and operations environments, a new level of organizational coordination and communication is needed. The requirement is for a defined pipeline of release stages and application candidates moving regularly from planning to development and from release to operations. This ensures that development and operations engineers can work in cooperation to release new versions.

Figure 2. Enabling DevOps: automation, collaboration and governance across development and operations teams



Continuous delivery results in a dynamic set of data and actions that automate release processes. Release process may currently be contained in documents, checklists. and Wikis, but the use of this data is susceptible to human error or interpretation. There are many enterprise tools and products available to enact the process—from provisioning systems to deploying applications and monitoring, but without automation, standardization of tools and environments becomes hard to control. Delivery into a hybrid environment can magnify these inconsistencies when tools and processes must be managed both inside and outside the organization. CDA creates a centralized structure for implementing a DevOps agenda and standardizes the integration of tools best suited to drive the continuous delivery processes. CDA delivers this capability through the following methods:

- Model-driven approach
- Extensible, pluggable architecture

Model-driven approach

When approaching the goal of continuous delivery to enable DevOps, HP CDA provides a data model that represents the artifacts and relationships required to define how composite applications are delivered to a dynamic set of target environments. This model-driven approach allows CDA to centrally store and version artifacts in a definitive software library and then orchestrate the requisite operations that leverage the models for automating tasks such as platform provisioning and application deployment. CDA defines several key model aspects and relationships:

- Platform—The compute, storage, network, and additional software capability required to fully define the target environment for an application.
- Application—The build artifacts, scripts, and commands per layer required to deploy the application, including the basic operations (start, stop, and so on) that can be performed on the application.
- Policy—The monitoring providers and monitoring templates along with the values and parameters to automate the configuration of system and application monitoring during deployment.
- Topology—The dynamic relationship between composite application layers and compute resources within a platform. This defines a unique deployment mapping.

This model-driven approach provides flexibility and portability of applications across platforms. It also allows for dynamic reuse of artifacts to meet the delivery requirements. This translates into value as follows:

- Improved time-to-delivery (TTD) of application ecosystems and associated reduction in costs, errors, rework, and risks by having model-driven automation.
- Improved collaboration between organization teams (development, test, operations) by centralizing information about setup, operations, and monitoring across the release pipeline. The CDA model becomes a collaborative library, which reduces rework of processes and reduces errors.
- Reduced vendor lock-in by separating the model that defines the delivery artifacts from the tools that operate on the model to realize the automation.

Extensible, pluggable architecture

As there is no one delivery environment that fits all businesses, HP CDA provides the flexibility to design continuous delivery around the tools and environments prescribed by an organization. This allows you to rapidly adopt a DevOps paradigm, as you do not need to re-tool the organization and at the same time you can maintain the organizational knowledge. By allowing tools such as software deployers and application monitors to be configured and changed dynamically, CDA provides a future proof method for continuous delivery processes that you require.

The architecture of CDA provides configuration of delivery providers in several key areas:

- Software deployment:
  - The CDA models for applications can be translated into sequences of steps for leading products used for software deployment such as HP Server Automation and Chef. This allows the same application model to be deployed to target platforms through different deployers depending on the user configuration.
  - Many leading software deployment technologies have evolved significant libraries of off-the-shelf content either within organizations or in the public communities. CDA can leverage existing deployment content directly as extensions to both application and platform models.

- Monitoring:
  - After the infrastructure is in place, and applications are deployed, monitoring takes center stage. In this competitive landscape, customers may have standardized on monitoring tools such as HP SiteScope, HP Diagnostics, or Nagios. The HP CDA monitoring policy models abstract the vendor specifics, but provide the plug-in layer to configure specific tools based in the model.
  - By normalizing and centralizing monitoring policy definitions in a central library, monitoring is no longer an activity confined to production. The portability of policies across the release pipeline environments provides a well-defined lifecycle of performance management from development through production.

- Hybrid infrastructure
  - IT organizations have an array of private and public cloud technology and providers as well as traditional physical infrastructure. With this variety comes the demand to make applications easily portable across technologies. This may be dictated by organizational boundaries or investment decisions.
  - HP CDA provides an extensible infrastructure layer for provisioning the right systems in the right environments such as VMware or Amazon Elastic Compute Cloud.

Building the models

CDA provides designer interfaces for building the models required to define a continuous delivery system. These designers allow CDA models to be built by users whose expertise matches the design area:

- Platform designer—used by infrastructure and platform architects and specialists in the organization
- Application designer—used by application architects and specialist in the organization
- Policy designer—used by system architects and performance specialists in the organization

Platform

The foundation of the CDA platform is the infrastructure template. The CDA product delivers an infrastructure service layer, which also provides a graphical designer for defining the compute, storage, and network characteristics of infrastructure services. These services can represent compute capacity that is physical (HP Matrix hardware based) or virtual (public cloud, private cloud, or hybrid). The infrastructure as a service (IaaS) definitions are synchronized with CDA and stored in CDA as infrastructure template models.

A CDA platform model is based on an IaaS template model. This defines the core capability of the platform as well as the location. After a new platform design has been associated with a specific IaaS template, the designer then tags the platform computer resources with the built-in capabilities. This includes any included software (database, middleware, and so on) as well as OS and hardware details. This tagging creates a complete profile that will be referenced by application designers.

The final design step is to optionally model additional software to be installed during provisioning time. Platform designers understand the underlying IaaS templates and the compute attributes. They are also aware of the unique requirements arising from the application teams. When the MOE services do not provide the complete platform environment for applications, the platform designers can use the CDA platform software library to create models of software packages. This library can then be leveraged to model additional platform capabilities to add to the IaaS service. This ensures that the provisioned platform meets the organizational requirements and standards.

Application

The CDA application model conceptually is positioned between the continuous integration (build) environment and the target deployment environment. This model defines the set of build artifacts, scripts, and commands required to deploy and manage the layers of a composite application. An application model defines several default operations to perform functions such as deploy, un-deploy, start, and stop.

CDA provides a hierarchy for managing application models. An application is comprised of versions. A version is then defined by one or more models. A model is a specific definition that defines the layers of the application along with the recipe for managing the deployment of the application. An application version may define more than one model if, for example, the application can be deployed on different operating systems or middleware containers.

Within the application model, the designer defines the layers of the composite applications. For each layer, the designer can define specific endpoint services that the layer exposes to other layers. A layer can also define external services that the later requires. The model defines its build configuration (what external build artifacts to access) along with the location of the build artifacts. The model can define multiple build configurations depending on how the artifacts are managed (either by iterations or other project milestones). After the layers and build configurations are modeled, the designer models all the sequences per layer to manage the deployment. A sequence step defines the following:

- Placed file—based on the model's build configuration model the locations of application artifact versions created by the continuous integration system, and define location, permission and ownership on the target
- Run script—define any custom scripting required to perform the application actions (such as deploy and start)
- Service command—execute a local OS service command on the target
- External component—execute a remote sequence automation from an external service (such as HP Operations Orchestration flow, Chef cookbook)

Topology

When designers use CDA to model applications and platforms, a topology relationship dynamically maps the layers of a specific application model to the compute resources of a platform. This is a modeled relationship which enables an application version model to participate in many topologies with different platforms. This relationship also ensures that a given platform model can be used as the target for many application models.

After applications, versions, and models are created in CDA, the application designers create topology relationships as required to support the release pipeline. A given topology may be deployed multiple times across several provisioned platforms.

Policy

CDA models not only the application, but also the monitoring to accompany the application release. CDA models the specific monitor characteristics from integrated providers (such as HP SiteScope, HP Diagnostics), including parameter values. These monitors can then be modeled dynamically into policies. A given policy can contain multiple monitor definitions across multiple providers.

CDA policies are modeled separately from the application so that the policies can be reused both across applications and versions as well as within a release pipeline. Policies are related in the topology model.

Once a policy has been deployed, CDA provides the capability to integrate with the monitor providers to receive alerts based on the deployed model. This allows the CDA topology deployments to provide status data based on the monitored environments. The CDA user can also update a policy for a deployment to add a new monitor definition and then re-deploy the policy to dynamically update monitoring on the target platform instance.

Managing the DevOps lifecycle

Continuous delivery in today's composite applications to hybrid infrastructure requires the ability to view application versions by DevOps stages, move applications dynamically from one target platform to another (with desired monitoring), define target environments across different providers, and use different vendor tools along the release pipeline.

- The CDA model provides a comprehensive picture of applications and target platforms along with topology relationships and monitoring policies. CDA adds the additional dimension of labeling models and artifacts for specific release lifecycle stages, thus providing DevOps visibility for continuous delivery.
- The dynamic topology relationship in CDA allows application version models to be in multiple platform relationships to facilitate the assignment of versions from one release stage to the next. Within a specific application version model you can have multiple deployment packages of application artifacts that can be uniquely tracked across deployments.
- With CDA and MOE, infrastructure template definitions can support a wide range of public and private cloud environments as well as specific physical provisioning. This means you can have a development platform based on an MOE template defined to use a private cloud (i.e. VMware) and then a QA platform for the same application release on a template defined to use a public cloud (such as Amazon Elastic Compute Cloud).
- CDA models are not vendor specific, but instead provide a plug-in architecture for managing external tools. Platform models define a deployment tool, but as applications can dynamically map to multiple platforms, this gives the flexibility of using multiple vendors through the release process. Similarly with monitoring, monitor definitions are

specific to a tool, but policies can provide definitions from multiple vendors, and different deployments can utilize different policies.

Summary

To summarize, HP CDA enables organizations to employ the tenets of DevOps to join continuous integration and continuous operations together with continuous delivery. CDA provides the following benefits across the application delivery lifecycle:

- Improved collaboration among dispersed teams for a reduction in cost, errors, and time
- Support for heterogeneous infrastructure environments (both public and private)
- Consistent composite application deployment, monitoring, and management across dynamic heterogeneous IT environments
- Model-driven application ecosystem along with development of blue prints with application-aware configuration parameters
- Correct deployment of the automation and testing tools the first time
- Reduced vendor lock-in through open and extensible architecture for provisioning, deployment, and monitoring
- Support for different user personas and specific access controls for different user personas based on the role they play in the application lifecycle.
- Application and associated data, configuration, and artifact portability through the entire application lifecycle.
- Application models that utilize the existing deployment content. This model provides capabilities such as action sequence, parameter passing, and other actions necessary to manage the lifecycle of the application.
- Platform model that utilizes existing templates and content in an organization along with the ability to provide standardized platform provisioning services. The platform model also maintains a library of external software required to be deployed.

For more information

HP CDA Publications

The following publications are available on the HP Software Product Manuals website at **http://support.openview.hp.com/selfsolve/manual**), which requires an HP Passport sign-in:

- HP CDA documentation list—lists all other HP CDA publications, shows where they are located, and indicates when they are updated.
- HP CDA concepts guide—this publication introduces the HP CSA solution.
- HP CDA solution and software support matrix—provides information about platform support requirements for HP CDA core function with links to requirements for component products.
- HP CDA release notes—contains product release notes; please read before installation.
- HP CDA configuration guide—provides instructions for HP CDA installation and configuration.
- HP CDA help (printable PDF)—provides step-by-step information about how to design and deploy HP CDA applications.
- HP CDA troubleshooting guide—contains solutions and workarounds to known problems.

Deploy the most complex multi-tier applications quickly and accurately, visit **hp.com/go/cda**.

4AA4-1499ENW, Created August 2012; Updated October 2012, Rev. 1