Conceptual Document

NetApp and Docker Technology to Enhance and Accelerate Dev/Ops in Hybrid-Cloud Environments

Kumaraswamy Namburu, NetApp

March 2015 | WP-7214 | Version 1.0

Abstract

This conceptual document describes the software development lifecycle and one way to streamline and accelerate the process using Docker and NetApp® storage technologies. Note that the information provided here details concepts that, in theory, should work as described.

TABLE OF CONTENTS

1 Introduction 4

1.1 Scope 4

1.2 Intended Audience 4

2 Microservices and Dev/Ops 5

2.1 Microservices: An Overview 5

2.2 Dev/Ops: Bringing Dev and Ops Even Closer Together 5

2.3 Challenges in the Software Development Process 5

3 Docker: An Overview 6

4 Managing Data in Docker Containers 6

4.1 Managing Docker containers on multiple hosts 6

4.2 Clustered Data ONTAP Docker Volumes 7

4.3 Clustered Data ONTAP and Docker in Dev/Ops and Microservices 9

4.4 Creating and Mounting a Data Volume in a Docker Container 9

5 Software Configuration Management 10

6 Software Builds 10

6.1 Faster Workspace Creation from SCM Systems 10

6.2 Software Build Management 12

6.3 Build Farm Infrastructure as Docker Containers 12

6.4 Prebuild Object Files 12

6.5 Faster Builds with Docker and NetApp Technology 13

7 Continuous Integration with Docker and Clustered Data ONTAP 13

7.1 Continuous Integration 13

7.2 Jenkins 13

7.3 Jenkins-Docker Plug-in 14

7.4 Instant Dev/Test Labs for Continuous-Integration Testing 14

7.5 Autorevert Bad Changes in Continuous Integration 14

8 Hybrid Cloud and NetApp Private Storage 15

9 Conclusion 16

References 17

Version History 17

LIST OF FIGURES

Figure 1) Benefits of a clustered namespace. 7
Figure 2) Volume cloning with FlexClone. 13
Figure 3) Spinning up dev/test environments using NetApp storage and Docker containers. 14
Figure 4) Spinning up dev/test environments using NetApp storage and Docker containers in a hybrid cloud. 16

1 Introduction

Software development has evolved to become so complex that it requires support for a diverse set of compute, network, and storage requirements. It also typically involves collaboration between geographically distributed teams and requires a multitude of development tools and components. Several factors impact how smooth the software development process functions and thus the time to market.

Stability and reliability of software builds are vital for continuous integration[1] in dev/ops[2] both for monolithic- and microservices-based application development models.[3] Network issues, insufficient compute, and storage hardware (for open-source and in-house projects) lead to long wait times for software builds and continuous-integration jobs. Huge time, resource, and productivity challenges result from the need to add capacity because companies have to go through the process of ordering and waiting for deployment of increased capacity—and the right capacity. The lack of caching, the right choice of virtualized technologies, and the ability to share prebuilt artifacts are three shortcomings driving high demand and unpredictable usage of compute and storage resources.

Most companies are adopting continuous integration and continuous delivery—dev/ops—and microservices to address software-development challenges and adopt a streamlined process that improves time to market.

Docker containers are emerging as the new standard for dev/ops workflows in hybrid-cloud architectures. Docker containers minimize the variables that occur between test and production environments and seamlessly help software application packaging.

Data storage is critical for all applications. The NetApp clustered Data ONTAP® storage operating system features unified storage technologies such as NetApp Snapshot® backup and data recovery software and NetApp FlexClone® data replication technology to provide instantaneous, writable copies of data in a matter of seconds.

This paper describes how you can leverage NetApp technologies in conjunction with Docker in each phase of continuous integration and deployment in dev/ops workflows. Doing so can accelerate the software development and test process, provide cost-effective design solutions, and make production software deployments seamless.

1.1 Scope

This conceptual document provides you with a method to consider using to improve software development and test lifecycle workflow and efficiency by leveraging NetApp technology and Docker container technology.

1.2 Intended Audience

This document is intended for use by individuals who are responsible for design, management, and support of any of the phases of the software development lifecycle. It assumes that readers are knowledgeable about dev/ops, microservices, NetApp clustered Data ONTAP storage, and Docker technologies.

---

[1] Continuous integration is a software engineering practice in which isolated changes added to code base are immediately tested and rapidly corrected, if necessary, to prevent defects in the code base.
[2] Dev/Ops is a software development and operations quality assurance method stressing communication, collaboration, integration, automation, and measurement among cooperating developers and IT staff.
[3] Monolithic architecture puts all functionality in a single process, replicated across multiple servers; microservices architecture puts different functionality into different processes, distributed across many servers.

2 Microservices and Dev/Ops

2.1 Microservices: An Overview

In enterprise hybrid-cloud application development, the monolithic application development method and architecture are difficult to manage at scale. The monolithic method, therefore, is being replaced by microservices application architectures. Enterprises are decomposing large applications—with all the functionality built in—into smaller, purpose-driven services that communicate with each other through common Representational State Transfer application programming interfaces (APIs).

Monolithic Application Characteristics

- Large code base
- Many components, no clear ownership
- Long deployment cycles
- As code base increases:
  - Tends to increase "tight coupling" between components
  - All components have to be coded in the same language

Microservices Characteristics

- Many smaller (fine-grained), clearly scoped services
  - Single responsibility principle
  - Domain-driven design
  - Bounded context
  - Independently managed
- Clear ownership for each service
  - Typically need/adopt the dev/ops model

2.2 Dev/Ops: Bringing Dev and Ops Even Closer Together

Dev/Ops is about the quest to deliver products faster with reduced risk.

Dev/Ops

- Is a cultural movement rooted in the agile software development movement
- Embraces automation and automated testing
- Allows pushbutton deployments to any environment
- Leverages continuous-integration/delivery concepts
- Permits quick reverting of bad changes

2.3 Challenges in the Software Development Process

Below are a few challenges inherent in the software development process.

- **Software build.** A software build is a process of converting the source code into standalone artifacts that can run on a computer. The major components of a software build are code compilations, which involve converting source files into direct executable and intermediate-object files in preparation for testing and then release. The challenges in software builds are long build times, reproducibility, and compute and storage requirements.
- **Distributed development.** In distributed development, the engineering and IT teams work on the same project but in different locations around the world in multiple business organizations. Although the project scope and requirements may remain the same across the different organizations, working in geodiverse locations makes it difficult to collaborate.

- **Lack of infrastructure scalability.** During the development phase, source files are copied from the central repository to the developer workspaces, which are processed on the build servers (known as the "build farm"). As the number of engineers within an organization grows, so does the need to provide resources for them. Because the servers cannot scale seamlessly, the development process slows down while additional resources are provisioned and deployed; this, in turn, impacts the overall product's time to market. The lack of elasticity requires resources to be allocated to handle peak workloads.

3 Docker: An Overview

Docker is an open platform for distributed applications for developers and system administrators. This technology is becoming widely used to build and deploy applications. It is based on the idea of homogeneous application *containers*. Docker consists of containers based on preexisting technologies such as Linux® containers and *Union file systems* (UnionFS). The containers themselves are only half the story. The other half is Docker *images*. Docker images are indexed snapshots of the entire file system that the container is meant to run. Every time a change is made to the file system, a new version of the image is automatically created and assigned a hash ID.

Docker Advantages

1. **Deployment repeatability.** All prior production deployments can be rebuilt.
2. **Version-neutral application management.** Applications that rely on different versions of the same package or image can be managed equally.
3. **Orthogonal application isolation.** Unrelated applications running on the same machine do not negatively affect one another.
4. **Virtual environment distributed management.** A GitHub-like repository manages organization and deployment of application versions.
5. **Reduced overhead.** Lighter-weight Linux container solutions; no hypervisor requirement.

4 Managing Data in Docker Containers

A *volume* is a specially designated directory within one or more Docker containers that bypasses the UnionFS to provide several useful features for persistent or shared data. Such features include the following:

- Volumes are initialized when a container is created.
- Data volumes can be shared and reused between containers.
- Changes to a data volume are made directly.
- Changes to a data volume are not included when you update an image.

4.1 Managing Docker containers on multiple hosts

Kubernetes is an open-source container cluster manager. It can be used for creating container replicas across a group of node instances. A master instance provides the Kubernetes API, through which tasks are defined. Kubernetes spawns containers on nodes to handle the defined tasks used in your workflows. It was started by Google and now it is supported by others as well.

Kubernetes serves two purposes. Once you are using Docker containers, the next question is how to scale and start containers across multiple Docker compute hosts, balancing the containers across them. It provides a higher-level API to define how containers are logically grouped, allowing you to define pools of containers for application load balancing.

Docker also has announced orchestration tools for managing the multi-container distributed apps for dev/test workflows.

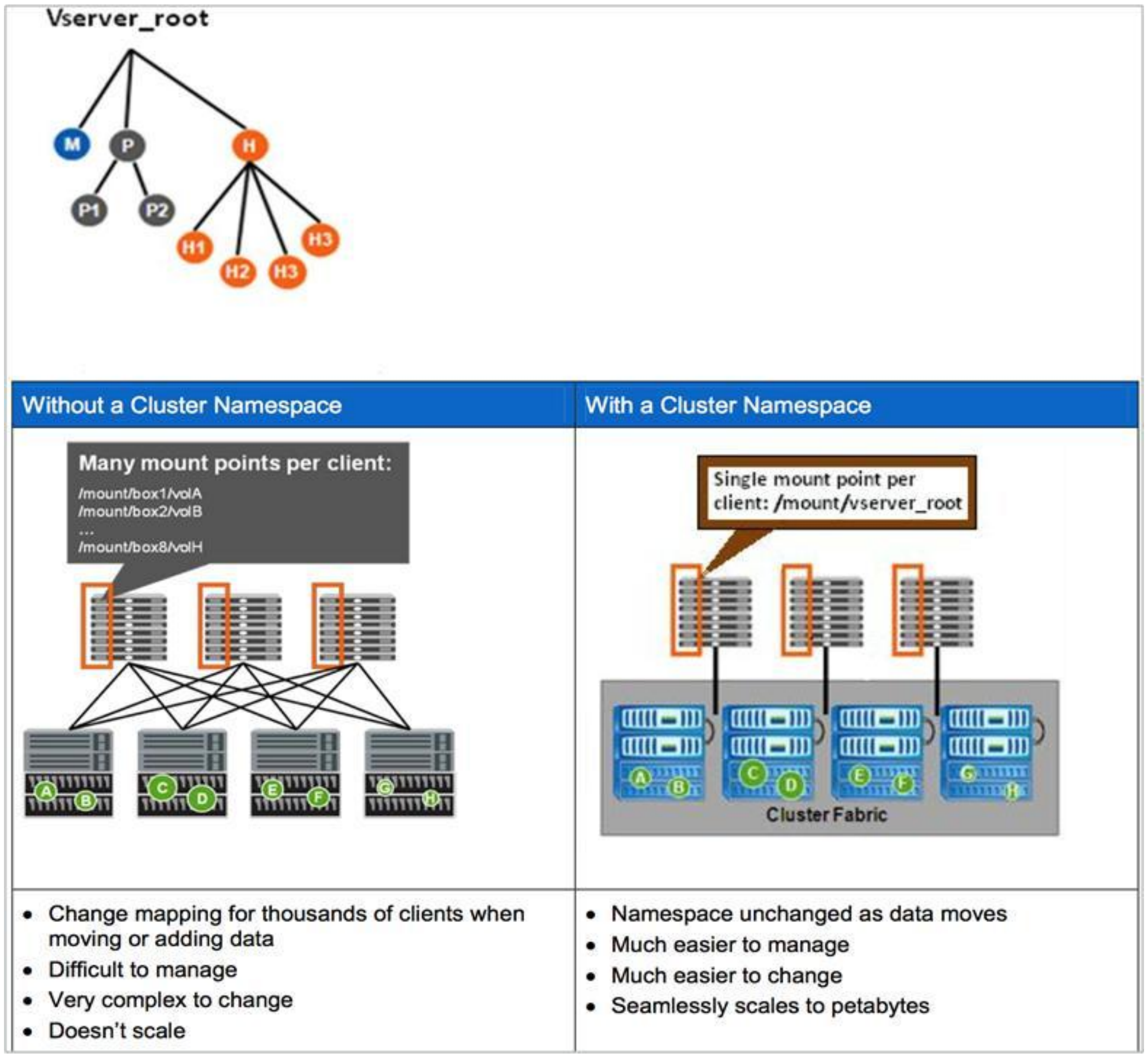
Docker containers can be managed by one of above-noted orchestration tools, or you can pick your tool of choice.

4.2 Clustered Data ONTAP Docker Volumes

Based on our knowledge and experience with NetApp® storage, we developed a conceptual design that combines the value and features of clustered Data ONTAP® volumes with Docker containers to provide a scalable infrastructure to meet a variety of new application requirements. The following are NetApp clustered Data ONTAP® storage virtualization features that you can use to make Docker data volumes fully virtualized and share the data containers in Docker seamlessly.

- **Storage Virtual Machine (SVM).** An SVM is a logical file system namespace capable of spanning beyond the boundaries of physical nodes in a cluster. Each SVM has a root volume under which additional volumes are mounted, extending the namespace. An SVM can span several physical nodes in a cluster. It is associated with one or more logical interfaces; clients access the data on the virtual server through the logical interfaces that can live on any node in the cluster.
- **Logical Interface (LIF).** An LIF is essentially an IP address with associated characteristics, such as a home port, a list of ports for failover, a firewall policy, a routing group, and so on. Client network data access is through LIFs dedicated to the SVM. An SVM can have more than one LIF. You can have many Linux/Network File System (NFS) clients mounting one LIF or one client mounting several LIFs. This implies that IP addresses are no longer tied to a single physical interface.
- **Aggregates.** An aggregate is a RAID-level collection of disks; it can contain more than one RAID group. Aggregates serve as resources for SVMs and are shared by all SVMs.
- **Cluster Namespaces and Junction Paths.** A cluster namespace is a collection of file systems hosted from different nodes in the cluster. Each SVM has a file namespace that consists of a single root volume. The SVM namespace consists of one or more volumes linked by means of junctions that connect from a named junction inode in one volume to the root directory of another volume. A cluster can have more than one SVM. All the volumes belonging to the SVM are linked into the global namespace in that cluster. The cluster namespace is mounted at a single point in the cluster. The top directory of the cluster namespace within a cluster is a synthetic directory containing entries for the root directory of each SVM namespace in the cluster. There is a global namespace per virtual server. In clustered Data ONTAP, a volume can have a designated junction path that is different from the volume name. This applies to both NFSv3 and NFSv4. Refer to the following link for more information: Clustered Data ONTAP NFS Best Practice and Implementation Guide.

Figure 1) Benefits of a clustered namespace.



Without a Cluster Namespace	With a Cluster Namespace
• Change mapping for thousands of clients when moving or adding data • Difficult to manage • Very complex to change • Doesn't scale	• Namespace unchanged as data moves • Much easier to manage • Much easier to change • Seamlessly scales to petabytes

- **Flexible Volumes (FlexVol® Volumes).** A volume is a logical unit of storage in clustered Data ONTAP. The disk space that a volume occupies is provided by an aggregate. Each volume is associated with one individual aggregate and therefore with one physical node. In clustered Data ONTAP, data volumes are owned by an SVM. This provides more flexibility to move volumes within a single namespace to address issues such as capacity management and load balancing. Volumes can be moved from aggregate to aggregate with NetApp DataMotion™ for Volumes data migration software, without loss of access to the client. This capability provides more flexibility to move volumes within a single namespace to address issues such as capacity management and load balancing for Docker containers.
- **NetApp Snapshot® Copies.** A NetApp Snapshot copy is a read-only image of a traditional or FlexVol volume or of an aggregate that captures the state of the file system at a point in time. By default, every volume contains a directory named "snapshot" through which users can access old versions of files in that directory. Users can gain access to NetApp Snapshot copies depending on the file-sharing protocol used—NFS or Common Internet File System (CIFS). NetApp Snapshot point-in-time copy software technology uniquely protects data with no performance impact on performance and uses minimal storage space. NetApp Snapshot technology advantages include:
  - Make instant data copies while your applications run.
  - Create Snapshot copies in under a second, for any volume size.

    – Make up to 255 Snapshot copies per volume for online backup and recovery.

- **NetApp FlexClone.** NetApp FlexClone technology instantly replicates data volumes and datasets as transparent, virtual copies—true clones—without compromising performance or demanding additional storage space. NetApp FlexClone technology lets you create dataset replicas of entire NetApp Data ONTAP volumes as well as individual file or LUN replicas. FlexClone efficiency and virtualization enable you to afford to create multiple clones needed to improve productivity, develop and test applications faster, and attach clones to the containers as Docker volumes.

4.3 Clustered Data ONTAP and Docker in Dev/Ops and Microservices

Docker is gaining popularity for dev/ops and microservices, for which the trend is to decompose applications into decoupled, minimalist, and specialized containers that are designed to do one thing really well. The fully encapsulated Docker container enables efficiency in dev/ops by creating a highly efficient distribution model for microservices applications. These changes are being adopted in development practices by putting larger-scale architectures such as those used at large enterprises within the reach of smaller development teams. Developers can work inside the containers, and ops engineers can work outside the containers in parallel.

NetApp FlexClone volumes provide great flexibility and scalability as well as create instantaneous copies of the data. Managing persistent and shared data across multiple Docker containers becomes seamless in a variety of application scenarios.

4.4 Creating and Mounting a Data Volume in a Docker Container

Several applications require persistent data to be shared between Docker containers. NetApp volumes can be exported through the junction path and mapped to Docker data volumes to share data seamlessly between the containers.

Here are the high-level steps that you can automate through a simple script.

1. Provision the storage on the NetApp clustered Data ONTAP cluster and create the junction path.
2. Create the volumes with required storage and mount them under the junction paths.
3. Mount the volume on Linux Docker host (data persists in the appliance-mounted volume on the junction paths).
4. Create a Docker ambassador instance something like “Docker run -v /mnt/dockerpoc:/data/db --name dockerpocdb.”
5. Start up a Docker app client linked to the app instance:
    a. docker run -it --rm --link dockerpocdb:dockerpocdb.
6. Take the point-in-time copies from the original volumes as required using Snapshot create.
7. Create clones from the Snapshot copy as required.
8. Mount the clone on the same junction paths under the new directory.
9. Create a Docker ambassador instance and spin up a new Docker container. Point the application to the newly created directory accessible through the same junction path, something like “docker run -v /mnt/dockerpoc2:/data/db --name dockerpocdb2.”

Refer to the following links for the correct syntax for Docker commands:

- https://docs.docker.com/userguide/dockerlinks/
- https://docs.docker.com/userguide/dockervolumes/

The Benefits

- Provides one mount point to access several NFS volumes (for example, logs, data)
- Provides point-in-time instantaneous copies for attaching multiple Docker containers to run the tests on independent copies and validate “what-if” scenarios
- Helps with immutable infrastructure
- If required, allows seamless data movement without changing Docker container configuration files
- Enables data management with leading storage software

5 Software Configuration Management

After the code has been designed and developed, a software configuration management (SCM) tool is typically used by engineering teams to manage and track revisions of software components as they are developed. Effective utilization of SCM is required to coordinate hundreds of engineers working across teams and spread across the globe. SCM improves productivity by reducing the overhead of versioning files, labels, and organizing code as they change during the development process. The key features of SCM are:

- **Version Control.** Also known as source control, version control is the management of changes to the source code as it is developed. Changes are usually identified with a change number, timestamp, and association with the engineer making the change. Version control systems are essential for any form of distributed, collaborative application development.
- **Parallel Development.** Software is being developed by teams that work in parallel on the same source files, which are either colocated or distributed across different locations. SCM is the backbone of parallel development because it allows each engineer to work on his or her own copy of resources, usually a source file, without waiting for others to finish a task. Each engineer is unaffected by the changes made by other engineers until he or she chooses to merge or commit the changes back to the project. SCM automatically flags any conflicts and merges the latest changes to the project.

6 Software Builds

A software build is a process of converting source code files into standalone artifacts that run on a computer in preparation for testing and then release. During builds, the compiler tool chain is used to create intermediate object files, executables, and libraries. For a single program, the process consists of compiling a single file; for a complex program, source code may consist of many object files that can depend on one another.

The object file is a compiled source, also known as machine code, that depends on the target machine and the compiler used.

Building software involves many steps that can be time consuming and difficult to repeat consistently. Build systems are a common point of delay in software development.

6.1 Faster Workspace Creation from SCM Systems

Developers work with large volumes of data such as code base, image files, text files, and so on. It can be challenging to get a quick copy of a source and prebuilt workspaces for source code systems. NetApp software build experience has proven that workspaces can be created in minutes, instead of hours, with NetApp® FlexClone® technology.

Conceptually, there are two methods for populating developer workspaces.

1. A developer can create a client from source code, populate this client workspace with source code, and build the binaries. It takes an inordinately long time, from a few minutes to several hours, to download the source code and from several minutes to hours to build the initial workspace.
2. Alternatively, the developer can use a fully populated prebuilt workspace, using hard links back to the existing prebuilt workspace. This approach avoids the build step because it provides prebuilt objects, but it is slow and error prone because files are shared among developers (using hard links) as well as with the build daemon. Any user can trash a shared file, and then all the other users (including the build daemon) suffer.

NetApp FlexClone technology provides the ability to create instant and space-efficient thin-provisioned copies, that is, clones, of the production or the test code base. FlexClone technology is based on NetApp Snapshot® technology, which creates a writable point-in-time copy of the volume. The cloned volume consists of only the data that changes between the source and the clone, thus minimizing the storage requirements. A clone of a volume can be created in a few seconds because it creates a copy of the metadata that is very small compared to the actual data. The FlexClone volume can be separated from the parent volume such that both volumes can work independently. However, new disk space is allocated only when there is an addition to the parent or the cloned volume, saving additional storage.

The following sections tie workspace creation all together and describe the tools and techniques to accelerate the software build process using NetApp FlexClone with Docker containers and SCM.

Procedure to Create Workspaces Using FlexClone

FlexClone technology offers an opportunity to provide a developer with a reliable, prebuilt workspace in a matter of minutes because these files are not shared (there are no hard links) with other workspaces. The generic steps to clone a workspace are as follows.

- Create a NetApp Snapshot copy of a clustered Data ONTAP® FlexVol volume that contains the prebuilt workspace on the NetApp storage system.
- Create a FlexClone volume from the NetApp Snapshot copy.
- Export the cloned volume on the NetApp storage system and then mount the clone onto the host through the junction path in the cluster namespace.
- Change ownership/permissions (that is, change the ownership of every file/directory in the tree to be owned by the user to which you want to give the volume). You can do this efficiently by creating a script that takes a user ID (UID) as a numeric value and a list of files that then calls the known system call for each file, setting the owner to the passed UID. If the number of files or list of files is large, you can use “xargs” to split the list into multiple invocations of the script.
- If the makefiles do not have relative paths, update them to use the cloned source directory instead.
- Define an SCM client to let SCM know about the existence of the new workspace, modifying fields such as Root Directory and Owner with user-specific information. Perforce®, GIT, CVS, and SCM have built-in commands to support this feature. Development teams might be able to find the equivalent commands in their SCM systems as well.
- Allow the SCM to make the workspace think it has the file content already. (If you are not cloning at the head revision of the data, you can use the template workspace as a revision identifier.)

Note: If a clone is used only for continuous-integration tests in which no edits are needed, then SCM operations such as creating clients and flushing the workspace are not required.

Using conventional approaches to create SCM workspaces can be time consuming, especially when the workspaces become large. It takes only minutes for each user workspace to be created from the central NetApp Snapshot copy of the built-out workspace. This is possible because cloning using FlexClone technology is instantaneous. It also has the added advantage that new clones do not take additional space until changes are made to the workspace. For more information on NetApp FlexClone technology, refer to A Thorough Introduction to FlexClone and Back to Basics on FlexClone. For more on NetApp and

Perforce, see Streamlining Engineering Operations with Perforce SCM Software, Perforce Software on NetApp Clustered Data ONTAP®, and Backup and Recovery of Perforce on NetApp Storage Solutions**.**

Considerations for Using FlexClone to Create Faster Workspace

Following are a few considerations to keep in mind when using FlexClone for faster workspace creation:

- Cleanup of cloned workspaces avoids accumulation of abandoned workspaces.
- FlexClone copies are limited to 32,767 per volume in clustered Data ONTAP 8.2.
- There are 12,000 NAS volumes in a clustered Data ONTAP cluster in clustered Data ONTAP 8.2.
- There are 1,000 volumes per node in a cluster on NetApp FAS6280 storage.

6.2 Software Build Management

When source code files, compilers, and other components needed to create an application are ready, developers make an application “build.” A developer should be able to reconstruct the build environment before editing and submitting any changes made to the source files. The SCM carries out activities such as workspace creation that create an environment for the build process to run in. They also capture metadata about the inputs and outputs of the build process for repeatability and reliability.

6.3 Build Farm Infrastructure as Docker Containers

The build farm infrastructure is used for:

- Software compiles and linking steps for distributed builds for a variety of builds
- Creating prebuilt objects for developers
- Automating builds used for continuous integration and deployments.

Compared to traditional solutions, Docker container technology helps to create lighter, more portable, self-sufficient containers from any application. It is fast, reliable, and fit for creating compile build hosts simultaneously to spin off multiple hosts based on the workload. Distcc is an open-source utility widely used to distribute builds of C, C++, Objective C, or Objective C++ code across several machines on a network. Distcc should always generate the same results as a local build, is simple to install and use, and usually is much faster than a local compile. To take full advantage of distcc, run it on multiple build hosts spun off as Docker containers and specify all their names in the DISTCC_HOSTS environment variable on the target. Then use, for example, "make -j 10" to run multiple compiles in parallel, each of which will get farmed out to different build hosts. Several Docker containers for distcc are already published to the Docker Hub registry.

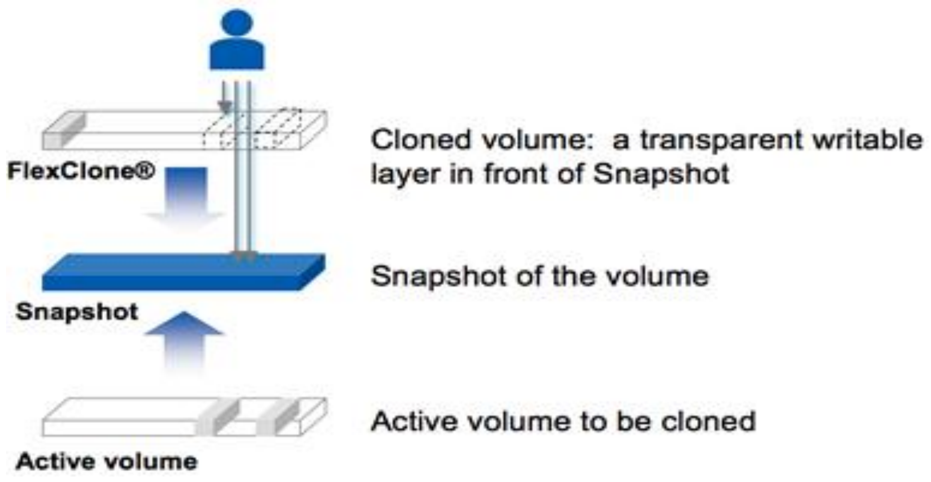
6.4 Prebuilt Object Files

NetApp Snapshot and FlexClone technologies can be leveraged to share the prebuilt object files that run periodically to check whether any new code has been checked in. If it has, the new code from source code systems like P4 and GIT is synced and runs a build. If a build fails, the continuous-integration process can be automated to file a bug and notify the developers as to who has checked in the code since the last successful build. If a build succeeds, it updates all the object files and closes the bug if one was previously filed. The creating build daemon process (prebuilt object files) involves the steps listed below.

- Create a volume on the NetApp storage system.
- Make the volume accessible using export and mount on the host’s server through the junction path using a single namespace.
- Run SCM commands on this volume to populate the template workspace.
- Run the build process in the template workspace to generate build artifacts.
- Run the build process periodically through continuous-integration jobs using open-source tools such as Jenkins to sync the new changes (there is more on Jenkins later in this document).

- Create a Snapshot copy at the end of the build process.
- The developer can create cloned volumes from the latest published Snapshot copies.

Figure 2) Volume cloning with FlexClone.


FlexClone®
Cloned volume: a transparent writable layer in front of Snapshot
Snapshot
Snapshot of the volume
Active volume
Active volume to be cloned

6.5 Faster Builds with Docker and NetApp Technology

By combining NetApp FlexClone to share prebuilt objects with Docker containers as the build hosts for distributed compiles, and using open-source tools such as distcc for C, C++, and Maven for Java, you get the following benefits:

1. You avoid redundant compilations.
2. Required compilations are distributed to a faster build host.
3. Build times are faster, speeding up the development cycle, creating work efficiency, and reducing required compute resources by leveraging the prebuilt objects files.
4. Storage and compute requirements for large compiles are significantly reduced.

7 Continuous Integration with Docker and Clustered Data ONTAP

7.1 Continuous Integration

Continuous integration is a software engineering process that applies quality controls continuously over a relatively small set of content rather than using the more traditional method of applying those controls *after* development is complete. The process aims to verify that the product, at a particular code change at any point in time, achieves the basic level of functional stability that the testing team requires to continue making forward progress. The goal is to find regressions as early as possible, take immediate action when they are discovered, then improve the process so that a similar regression is caught even earlier in the future.

7.2 Jenkins

Jenkins® is an open-source continuous-integration tool written in Java®. Jenkins provides continuous-integration services for software development. It is highly customizable and provides the ability to automate development and to push the code/artifacts/binaries to the test and production environment, eliminating the need to deploy manually. You can configure Jenkins to sync the source code from SCM systems and configure Jenkins jobs to run builds and tests every time new code is pushed.

7.3 Jenkins-Docker Plug-in

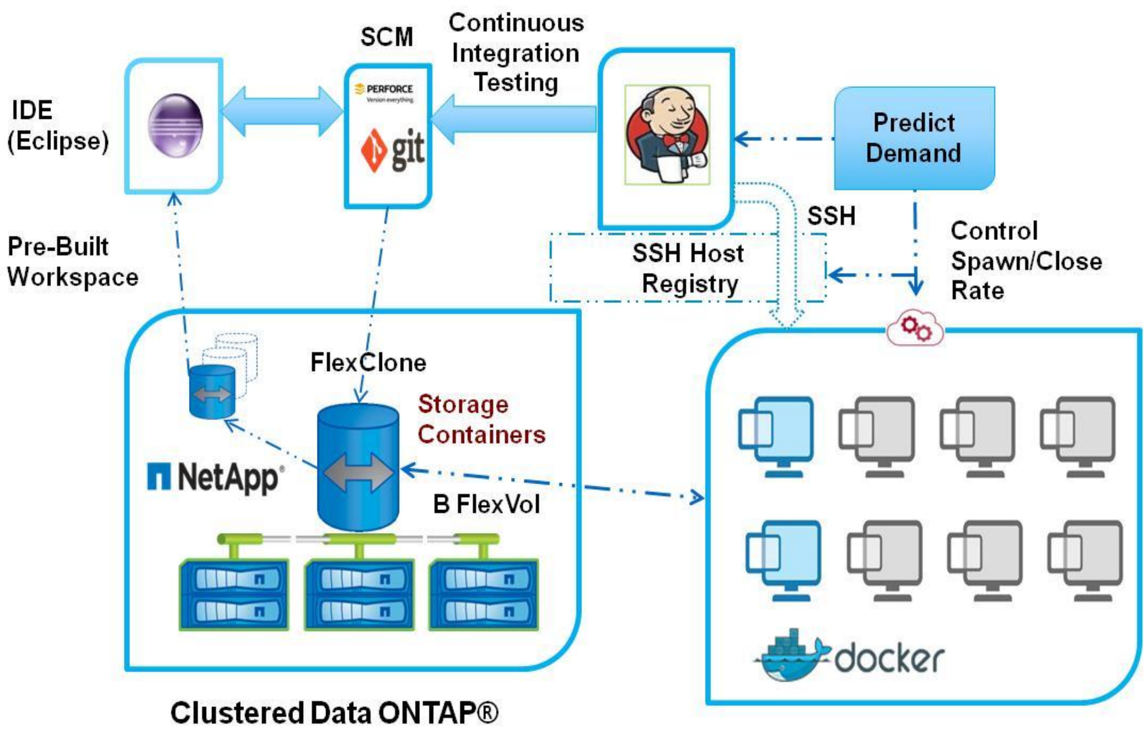
This Jenkins-Docker plug-in lets you use Docker to dynamically provision a Jenkins build slave, run a build or test, and then tear down that slave, optionally saving it for future use.

7.4 Instant Dev/Test Labs for Continuous-Integration Testing

Using NetApp® clustered Data ONTAP® and Docker to spin up multicontainer environments that are synchronized with the latest code and production topology in seconds enables deployment of instant dev/test labs. This combination of technology tremendously helps developers test every code change against a live topology. It also eliminates the "works on my machine" category of bugs and reduces compute/storage footprint by containerizing labs. In addition to this, development teams can also spin off clones to test "what-if" scenarios and reproduce problems.

The following figure conceptually illustrates how Jenkins, Docker, and NetApp technologies can help bring elasticity to dev/ops and create instant dev/test labs.

Figure 3) Spinning up dev/test environments using NetApp storage and Docker containers.



7.5 Autorevert Bad Changes in Continuous Integration

Below is an outline of continuous-integration processes—bisect and autorevert—and the way you can leverage NetApp and Docker technologies together to achieve development goals.

- **Bisect.** The bisect process is a form of automated binary search used to find the precise change that caused a test (or build) or continuous integration to fail. One of the key aspects of continuous integration is to identify all bugs and errors as early as possible and take immediate action against them. The bisect process is enabled as follows.
  - Take the first change that a test was run against that caused it to fail.

  - Take the last change that passed the same test.
  - Choose a predefined number of changes in between those two changes.
  - Generate an SCM client at those changes.
  - Build the client (or clients) and run the test against it by spinning up a Docker container and attaching cloned storage to it.
  - Subdivide the changes and run additional tests until the change that caused that test to fail is identified.
  - Once the process is complete, the exact change number that caused the problem is identified.
  - Unmount the clone, offline the cloned volume, and delete it.
- **Autorevert.** Once the bisect process identifies the change that caused a test (or build) to fail, autorevert can automatically rip that change out of the code line and notify the developer who made that change.

By using a combination of Docker and NetApp technologies (Snapshot® copies and FlexClone® volumes) to accelerate the instantaneous virtual copy process of compute and storage at every development submittal, you can use continuous integration for exceptional help in pinpointing regressions introduced into development code lines. The way to apply continuous integration to virtual copying is as follows.

- Create a Snapshot copy of the daemon volume.
- If a build or test fails, the bisect process is initiated to find the change that caused the failure.
- Create Docker containers for compute hosts and attach the data packaged in FlexClone to run the build and tests.

8 Hybrid Cloud and NetApp Private Storage

Automation is the key to accelerating software development and deployment. Provisioning a server in three weeks is no longer acceptable—organizations must be able to provision a server in a few minutes. A hybrid cloud is a cloud-computing environment in which an organization integrates its private-cloud environment with the scalability and flexibility of a public-cloud environment. Hybrid-cloud architecture allows an enterprise to accelerate dev/ops by breaking down organizational barriers and eliminating silos. It does this by providing elastic compute and storage resources to cater to fluctuating demands. Software development and testing can be performed in a production-like environment, and applications can be deployed and scaled in hours instead of weeks. Combining dev/ops with hybrid-cloud architectures gives teams not just the principles, but also the technology necessary to achieve development goals.

By combining a hybrid cloud and dev/ops:

- IT departments maintain control, visibility, and security.
- Dev/test teams remain agile and collaborative.
- Organizational barriers are broken down.
- Innovation and automation can thrive.

The Benefits of NetApp Private Storage in Development Environments

Enterprise customers demand choice and control—especially in development environments. That's why NetApp® Private Storage (NPS) for Cloud solutions are designed to let you control your data. With NetApp NPS, you can use multiple industry-leading public clouds while maintaining complete control over your data on dedicated, private storage. Choose solutions from an expanding NetApp network of cloud service providers, including:

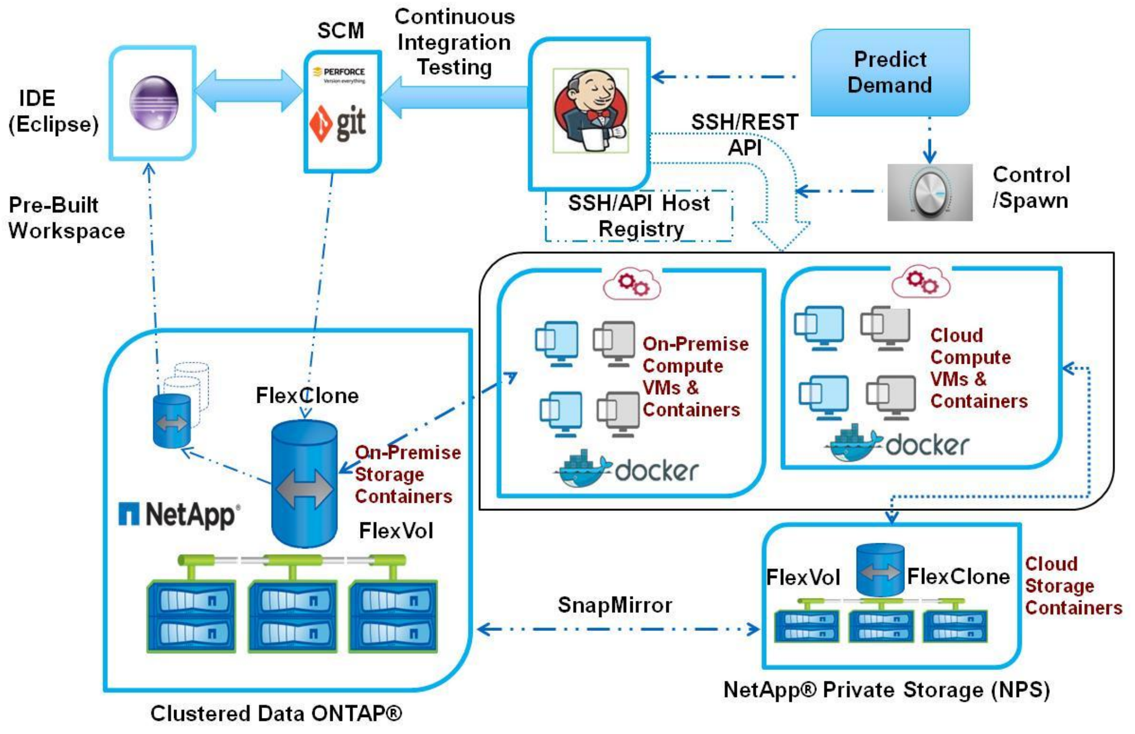
- NPS for SoftLayer®
- NPS for Microsoft Azure®

- [NPS for Amazon Web Services®]

With an NPS approach, you put your dev/ops data "next to" multiple clouds in tier 1 data centers, such as those of Equinix. Once you make your first cloud connection, you can add or switch to any or all of the clouds in minutes using, for example, the Equinix Cloud Exchange. Own and control your data. Meet your data compliance and sovereignty requirements because you know where your data is located at all times.

The following figure conceptually illustrates how Jenkins®, Docker, and NetApp technologies can help bring elasticity and create instant dev/test labs in a hybrid cloud.

Figure 4) Spinning up dev/test environments using NetApp storage and Docker containers in a hybrid cloud.



9 Conclusion

This conceptual document highlights the benefits of combining Docker containers (OS-level virtualization) and NetApp clustered Data ONTAP® (storage-level virtualization) to create multiple isolated and secure environments used in dev/ops.

These concepts can be leveraged in small, medium, and enterprise companies distributed globally and working in parallel on a variety of software development projects. The document describes how NetApp® technology can be used in each phase of software development to address challenges of the software development lifecycle. Based on the concepts provided herein, a benefits recap is follows.

Conceived NetApp and Docker Benefits for Dev/Ops in a Hybrid-Cloud

- **Optimization.** Compute usage can be optimized using Docker and container orchestration tools. Clustered Data ONTAP significantly optimizes the storage footprint and improves the speed of operations. The combination of faster compute and storage is key to a successful dev/ops model.
- **Manageability.** NetApp technology addresses software development data management challenges by delivering the performance, high availability, and manageability needed.

- **Speed.** In the software-build phase, NetApp Snapshot® and FlexClone® technologies help create faster workspaces, in minutes as compared to hours. FlexClone technology also accelerates the continuous-integration process, bestowing even greater efficiencies and faster time to market. NetApp technology used in conjunction with Docker container technology helps the organization spin up virtual dev/test environments in minutes and keeps the development code line stable.
- **Flexibility.** NetApp technology works on the premises as well as in private-, public-, and hybrid-cloud environments. This feature enables NetApp storage-based benefits and flexibility, regardless of where you operate the environment.

References

- Docker
  - https://www.docker.com
- Clustered Data ONTAP
  - http://www.cptech.com/wp-content/uploads/2014/06/Clustered-Data-on-Tap.pdf
- A Thorough Introduction to FlexClone
  - http://www.netapp.com/us/media/tr-3347.pdf
- Back to Basics on FlexClone
  - http://community.netapp.com/t5/Tech-OnTap-Articles/Back-to-Basics-FlexClone/ta-p/84874
- Case Study: Streamlining Engineering Operations with Perforce SCM Software
  - http://www.netapp.com/us/system/pdf-reader.aspx?pdfuri=tcm:10-60745-16&m=wp-7094.pdf
- Deployment and Implementation Guide: Perforce Software on NetApp Clustered Data ONTAP
  - http://www.netapp.com/us/media/tr-4164.pdf
- Backup and Recovery of Perforce on NetApp Storage Solutions
  - http://www.netapp.com/us/media/tr-4142.pdf

Version History

Version	Date	Document Version History
Version 1.0	February 2015	Kumaraswamy Namburu, NetApp

Copyright Information

Trademark Information