INVENTORYMANAGEMENTSYSTEMFORRETAILERS

Domain:CloudApplicationDevelopment

TeamID	PNT2022TMID31378
ProjectName	InventoryManagementSystemforRetailers

Containerizetheappincloud

Toexplorethehistoryofcontainerizationtechnology,thebenefitsandadvantagesofutilizingthetechnology,andhowitisrelatedtovirtualization.

Containerization has become a major trend in software development as analternative or companion to virtualization. It involves encapsulating or packaging upsoftwarecodeandallitsdependenciessothatitcanrununiformlyandconsistentlyon any infrastructure. The technology is quickly maturing, resulting in measurablebenefits for developers and operations teams as well as overall softwareinfrastructure.

Containerization allows developers to create and deployapplicationsfasterandmoresecurely.Withtraditionalmethods,codeisdevelopedinaspecificcomputingenvironmentwhich,whentransferred toanewlocation,oftenresultsinbugsanderrors.Forexample,whenadevelopertransferscodefromadesktopcomputertoavirtualmachine(VM)orfrom aLinuxtoaWindowsoperatingsystem.
Containerization eliminates this problem by bundling the application code togetherwith the related configuration files, libraries, and dependencies required for it torun.Thissinglepackageofsoftwareor"container"isabstractedawayfromthehostoperating system, and hence, it stands alone and becomes portable—able to runacrossanyplatformorcloud,freeofissues.

The concept of containerization and process isolation is decades old, but theemergence of the open source Docker Engine in 2013, an industry standard forcontainers with simple developer tools and a universal packaging approach,accelerated the adoption of this technology. Research firm Gartner projects thatmore than 50% of companies will use container technology by 2020. And resultsfromalate2017surveyconductedbyIBMsuggestthatadoptionishappeningevenfaster, revealing that 59% of adopters improved application quality and reduceddefectsasaresult.

Containersareoftenreferredtoas"lightweight,"meaningtheysharethemachine'soperatingsystemkernelanddonot require theoverhead ofassociatinganoperating system within each application. Containers are inherently smaller incapacitythanaVMandrequirelessstart-uptime,allowingfarmorecontainersto run onthesamecomputecapacityasasingleVM.Thisdriveshigherserverefficienciesand,in turn,reducesserverandlicensingcosts.

Put simply, containerization allows applications to be "written once and runanywhere." This portability is important in terms of the development process andvendor compatibility. It also offers other notable benefits,like fault isolation, easeofmanagementandsecurity,tonameafew.Clickheretolearnmoreabout thebenefitsofcontainerization.

Applicationcontainerization

Containers is used to encapsulate an application as a single executable package ofsoftwarethatbundlesapplicationcodetogetherwithalloftherelatedconfigurationfiles,libraries, and dependencies required for it to run. Containerized applications are"isolated"inthattheydonotbundle inacopyoftheoperatingsystem.Instead,an open sourceruntimeengine(suchastheDockerruntimeengine)isinstalledonthehost's operating system and becomes the conduit for containers to share anoperating systemwithothercontainersonthesamecomputingsystem.

Othercontainerlayers,likecommonbinsandlibraries,canalsobesharedamongmultiple containers. This eliminates the overhead of running an operating systemwithin each application and makes containers smaller in capacity and faster to startup,drivinghigherserverefficiencies.Theisolationofapplicationsascontainersalsoreducesthechancethatmaliciouscodepresentinonecontainer willimpact othercontainersorinvadethehostsystem.

The abstraction from the host operating system makes containerized applicationsportableandabletorununiformlyandconsistentlyacrossanyplatformorcloud.Containerscanbeeasilytransportedfromadesktopcomputertoavirtualmachine(VM) orfromaLinuxtoa Windowsoperatingsystem,andtheywillrunconsistentlyon virtualized infrastructures or on traditional "bare metal" servers, either on-premiseorinthecloud.Thisensuresthatsoftwaredeveloperscancontinueusingthetoolsandprocessestheyaremostcomfortablewith.

One can see why enterprises are rapidly adopting containerization as a superiorapproachto applicationdevelopmentandmanagement. Containerizationallowsdeveloperstocreateanddeployapplicationsfasterandmoresecurely,whethertheapplication is a traditional monolith (a single-tiered software application) or amodular microservice (a collection of loosely coupled services). New cloud-basedapplications can be built from the ground up as containerized microservices,breaking a complex application into a series of smaller specialized and manageableservices. Existing applications can be repackaged into containers (or containerizedmicroservices)thatusecomputeresourcesmoreefficiently.

Benefits

Containerization offers significant benefits to developers and development teams.Amongthesearethefollowing:

- **Portability:** A container creates an executable package of software that isabstractedawayfrom(nottiedtoordependentupon)thehostoperatingsystem,and hence, is portable and able to run uniformly and consistently across anyplatformorcloud.
- **Agility:** TheopensourceDockerEngineforrunningcontainersstartedtheindustrystandard for containers with simple developer tools and a universalpackagingapproachthatworksonbothLinuxandWindowsoperatingsystems.Thecontainerecosystemhasshiftedtoenginesmanagedbythe OpenContainerInitiative(OCI).Software developers can continue using agile or DevOps tools and processes forrapidapplicationdevelopmentandenhancement.
- **Speed:** Containers are often referred to as "lightweight," meaning they share themachine's operatingsystem (OS)kernel and arenot boggeddownwith this extraoverhead.Notonlydoesthisdrivehigherserverefficiencies,italsoreducesserverand licensingcosts whilespeedingupstart-times as there is no operatingsystemtoboot.
- **Fault isolation:** Each containerized application is isolated and operatesindependentlyofothers.Thefailureofonecontainerdoesnotaffectthecontinuedoperation of any other containers. Development teams can identify and correctany technical issues within one container without any downtime in othercontainers. Also, the container engine can leverage any OS security isolationtechniques—suchasSELinuxaccesscontrol—toisolatefaultswithincontainers.
- **Efficiency:** Softwarerunningincontainerizedenvironmentssharesthemachine'sOS kernel, and application layers within a container can be shared acrosscontainers. Thus, containers are inherently smaller in capacity than a VM andrequire less start-up time, allowing far more containers to run on the samecomputecapacityasasingleVM.Thisdriveshigherserverefficiencies,reducingserverandlicensingcosts.
- **Ease of management:** A container orchestration platform automates theinstallation, scaling, and management of containerized workloads and services.Container orchestration platforms can ease management tasks such as scalingcontainerized apps, rolling out new versions of apps, and providingmonitoring,logging and debugging, among other functions. Kubernetes,perhaps the mostpopular container orchestration system available, is an open source technology(originally open-sourced by Google, basedon theirinternalprojectcalledBorg)that automates Linux container functions originally. Kubernetes works with manycontainer engines, such as Docker, but it also works with any container systemthatconformstotheOpenContainerInitiative(OCI)standardsforcontainerimageformatsandruntimes.
- **Security:** The isolation of applications as containers inherently prevents theinvasion of malicious code from affecting other containers or the host system.Additionally,securitypermissionscanbedefinedtoautomaticallyblockunwantedcomponents from entering containers or limit communications with unnecessaryresources.

Types

Therapidgrowthininterestandusageofcontainer-basedsolutionshasledtotheneed for standards around container technology and the approach to packagingsoftware code. The Open Container Initiative (OCI), established in June 2015 byDockerandotherindustryleaders,ispromotingcommon,minimal,openstandardsand specifications around container technology. Because of this,theOCI ishelpingto broaden the choices for open source engines. Users will not be locked into aparticular vendor's technology, but rather they will be able to take advantage ofOCI-certified technologiesthat allow themtobuild containerized applicationsusingadiversesetofDevOps toolsandruntheseconsistentlyontheinfrastructure(s)oftheirchoosing.

Today, Docker is one of the most well-known and highly used container enginetechnologies, but it is not the only option available. The ecosystem is standardizingon containerd and other alternatives like CoreOS rkt, Mesos Containerizer, LXCLinux Containers, OpenVZ, and crio-d. Features and defaults may differ, butadopting and leveraging OCI specifications as these evolve will ensure thatsolutionsarevendor-neutral,certifiedtorunonmultipleoperatingsystemsandusableinmultipleenvironments.

Microservicesandcontainerization

Software companies large and small are embracing microservices as a superiorapproach to application development and management, compared to the earliermonolithic model that combines a software application with the associated userinterface andunderlyingdatabase into a singleuniton a single serverplatform.Withmicroservices,acomplexapplicationisbroken upintoaseriesofsmaller,morespecializedservices,eachwithitsowndatabaseanditsownbusinesslogic.Microservices thencommunicate witheachother across common interfaces(likeAPIs) and REST interfaces (like HTTP). Using microservices, development teamscan focus on updating specific areas of an application without impacting it as awhole,resultinginfasterdevelopment,testing,anddeployment.

The concepts behind microservices and containerization are similar as both aresoftware development practices that essentially transform applications intocollectionsofsmallerservicesorcomponentswhichareportable,scalable,efficientandeasiertomanage.

Moreover, microservices and containerization work well when used together.Containers provide a lightweight encapsulation of any application, whether it is atraditionalmonolithoramodularmicroservice.Amicroservice,developedwithinacontainer,thengainsalloftheinherentbenefitsofcontainerization—portabilityintermsofthedevelopmentprocessandvendorcompatibility(novendorlock-in),as well as developer agility, fault isolation, server efficiencies, automation ofinstallation,scalingandmanagement,andlayersofsecurity,amongothers.

Today's communications are rapidly moving to the cloud where users can developapplications quickly and efficiently. Cloud-based applications and data areaccessible from any internet-connected device, allowing team members to workremotely and on-the-go. Cloud service providers (CSPs) manage the underlyinginfrastructure, which saves organizations the cost of servers and other equipmentand also provides automated network backups for additional reliability. Cloudinfrastructuresscaleondemandandcandynamicallyadjustcomputingresources,capacity, and infrastructure as load requirements change. On top of that, CSPsregularly update offerings, giving users continued access to the latest innovativetechnology.

Containers, microservices, and cloud computing are working together to bringapplication development and delivery to new levels not possible with traditionalmethodologies and environments. These next-generation approaches add agility,efficiency, reliability, and security to the software development lifecycle—all ofwhichleadstofasterdeliveryofapplicationsandenhancementstoendusersandthemarket.

For a deeper dive into microservices, check out "Microservices: A Complete Guide"andwatchthefollowing"WhatareMicroservices?"video(6:38):

Security

Containerized applications inherently have a level of security since they can run asisolated processes and can operate independently of other containers. Trulyisolated,thiscouldpreventanymaliciouscodefromaffectingothercontainersorinvadingthehostsystem.However,applicationlayerswithinacontainerareoftensharedacrosscontainers.Intermsofresourceefficiency,thisisaplus,butitalsoopens the door to interference andsecurity breaches acrosscontainers.The samecould be said of the shared Operating System since multiple containers can beassociatedwith thesamehostOperatingSystem.SecuritythreatstothecommonOperating System can impact all of the associated containers, and conversely, acontainerbreachcanpotentiallyinvadethehostOperatingSystem.

But,whataboutthecontainerimageitself?Howcantheapplicationsandopensource components packaged within a container improve security? Containertechnology providers, such as Docker, continue to actively address containersecurity challenges. Containerization has taken a "secure-by-default" approach,believingthatsecurityshouldbeinherentintheplatform andnotaseparatelydeployed and configured solution. To this end, the container engine supports all ofthedefaultisolationpropertiesinherentintheunderlyingoperatingsystem.

Securitypermissionscanbedefinedtoautomaticallyblockunwantedcomponentsfromenteringcontainersortolimitcommunicationswithunnecessaryr esources.

For example, Linux Namespaces helps to provide an isolated view of the system toeachcontainer;thisincludesnetworking,mountpoints,processIDs,userIDs,inter-process communication, and hostname settings. Namespaces can be used tolimitaccesstoanyofthoseresourcesthroughprocesseswithineachcontainer. Typically,subsystemswhichdonothaveNamespacesupportarenotaccessiblefrom within a container. Administrators can easily create and manage these "isolation constraints" on each containerized application through a simple userinterface.

ResearchersareworkingtofurtherstrengthenLinuxcontainersecurity,andawiderangeofsecuritysolutionsareavailabletoautomatethreatdetectionan dresponseacross an enterprise,tomonitor and enforcecompliancetomeet industrystandardsand securitypolicies,toensurethesecureflowof data throughapplicationsandendpoints,and muchmore.

Learnaboutastrategyforscalingcontainersecurityacrossorganizationsofanysize.

Virtualizationvs.containerization

Containers areoftencomparedtovirtual machines(VMs) becausebothtechnologies enable significant compute efficiencies by allowing multiple types ofsoftware (Linux- or Windows-based) to be run in a single environment. However,container technology is proving to deliver significantbenefits overand above thoseofvirtualizationandisquicklybecomingthetechnologyfavoredbyITprofessionals.

Virtualization technologyallows multipleoperatingsystems andsoftwareapplications to run simultaneously and share the resources of a single physicalcomputer. For example, an IT organization can run both Windows and Linux ormultipleversionsofanoperatingsystem,alongwithmultipleapplicationsonthesame server. Each application and its related files, libraries, and dependencies,includingacopyoftheoperatingsystem(OS),are packagedtogetherasaVM.Withmultiple VMs running on a single physical machine, it's possible to achievesignificantsavingsincapital,operational,andenergycosts.

Formoreofanoverviewonvirtualization,checkoutthe"Virtualizationin2019"videoand"Virtualization:ACompleteGuide."

Containerization, on the other hand,usescompute resources even more efficiently.A container creates a single executable package of software that bundlesapplication code together with all of the related configuration files, libraries, anddependenciesrequiredforittorun.UnlikeVMs,however,containersdo*not*bundle inacopyoftheOS.Instead,thecontainerruntimeengineisinstalledon thehostsystem's operating system, becoming the conduit through which all containers onthecomputingsystemsharethesameOS.

As noted, containers are often referred to as "lightweight"—they share themachine'sOSkernelanddonotrequirethe overheadofassociatinganOSwithineachapplication(asisthecasewithaVM).Othercontainerlayers(commonbinsand libraries) can also be shared among multiple containers, making containersinherentlysmallerincapacitythanaVMandfastertostartup.Multiplecontainerscanthenrunonthesamecomputecapacityasasingle VM,drivingevenhigherserverefficiencies,furtherreducingserverandlicensingcosts.

ContainerizationandIBM

Inanutshell,virtualizationeliminatestheneedforanentireserverforoneapplication.Containerizationeliminatestheneedforanentire OSforeachapplication.

Portability,agility,faultisolation,easeofmanagement,andsecurityareamongtheadvantagesofutilizingcontainerizationtechnology.

Clickheretolearnaboutthesecurityandcontainerorchestrationcapabilitiesof theIBMCloudKubernetesService.

Foran overviewofhowmanagedKubernetescanhelpyouinyourcloudjourney,watchourvideo,"AdvantagesofManagedKubernetes"(3:15):

To learn more aboutbest practices to enableand expedite container deploymentinproduction environments, see thereport "BestPracticesforRunningContainersandKubernetesinProduction."

SignupforanIBMCloudaccountandstartprovisioningyourfirstclusterforfree.

Featuredproducts

- RedHatOpenShiftonIBMCloud
- IBMCloudSatellite