گزارش درس روش پژوهش و ارائه

عنوان:

DevOps

نگارش:

امیرمحمد رضوانی نیا

زیر نظر استاد فرزانه کیمیائی


Developer
Deployment Wall
Operator

فهرست مطالب

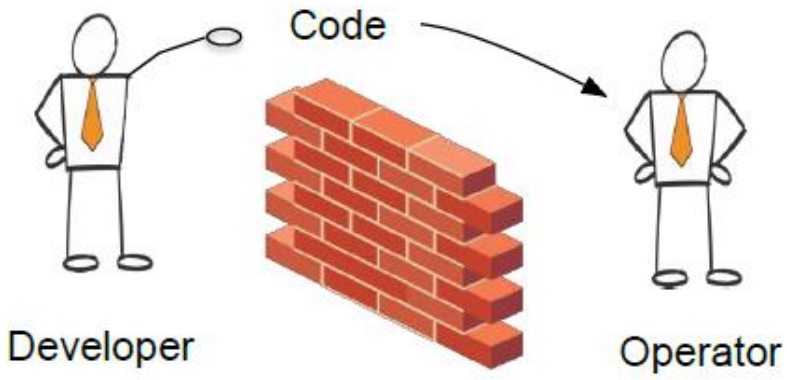
چکیده

برای مدت زمان طولانی ، توسعه و عملیات ماژول های جدا شده بودند. توسعه دهندگان کد می نوشتند و مدیران سیستم مسئول استقرار و ادغام آن بودند. از آنجا که ارتباط محدودی بین این دو گروه وجود داشت ، متخصصان بیشتر به طور جداگانه در یک پروژه کار می کردند.

امروزه ، DevOps یکی از مهمترین رویکردهای توسعه نرم افزار است. این رویکرد در فیس بوک ، نتفلیکس ، آمازون ، اتسی و بسیاری از شرکت های پیشرو دیگر در صنعت اعمال می شود. بنابراین ، اگر بخاطر عملکرد بهتر ، موفقیت در تجارت و رقابت پذیری ، در آغوش گرفتن از DevOps هستید ، اولین قدم را بردارید و یک مهندس DevOps استخدام کنید. اما ابتدا ، بیایید بررسی کنیم که DevOps به چه معناست و چگونه به بهبود تحویل محصول کمک می کند.

فصل اول: مقدمه

در گذشته دو تیم در شرکت های نرم افزاری وجود داشتند یکی تیم توسعه (Dev) و دیگری تیم عملیات (Ops) بود. در سیستم سنتی این دو تیم متضاد هم بودند و ارتباط نزدیکی بین تیم ها وجود نداشت. توسعه دهندگان اساساً دوست داشتند نرم افزار بسازند و سریع چیزها را تغییر دهند، در حالی که تیم عملیات بر پایداری و قابلیت اطمینان تمرکز داشت. (شکل ۱)



شکل ۱- (تیم توسعه و عملیات)

و چون این دو تیم درک و دانش کافی نسبت به کار یکدیگر نداشتند بین این دو تیم یک دیوار وجود داشت و این عدم هماهنگی و تعامل نداشتن افراد دو گروه باعث به وجود آمدن ایده اصلی دواپس شد. در واقع دواپس بر اساس ایجاد فرهنگ همکاری بین این دو تیم پایه گذاری شده است. به عبارتی دیگر دواپس مجموعه ای از فرایندها است که خیلی از شرکت های بزرگ به مرور به این مفهوم رسیدن که ما در پروسه تولید نرم افزار اگر یکسری از فرایند ها را رعایت کنیم هزینه تولید نرم افزار پایین می آید، سرعت تولید نرم افزار بالا می رود و میزان خطایی که ممکن است در تولید نرم افزار پیش بیاید کم می شود.

در این مقاله که به بررسی حوزه دواپس صورت می گیرد، سعی کردیم تا لیستی از کارآمدترین و همچنین مهم ترین ابزارهای دواپس را برای تان فراهم کرده و شما را در مسیر تبدیل شدن به یک کارشناس دواپس موفق، یاری کنیم.

فصل دوم: پیش زمینه

۱. مقدمه

DevOps اصطلاحی جدید است که در درجه اول بر بهبود همکاری، ارتباطات و ادغام توسعه دهندگان نرم افزار و متخصصان عملیات تاکید دارد. این اصطلاح یک فلسفه انعطاف پذیر، نغییر فرهنگی و یا تغییر پارادایم را توصیف می کند.

در کنفرانس چابک سال ۲۰۰۸، اندرو شفر و پاتریک Debois «زیرساخت چابک» را توصیف کردند. اصطلاح DevOps از طریق یک سری رویداد به نام «DevOpsDays» در سال ۲۰۰۹ در بلژیک رایج شد. پس از آن کنفرانس‌هایی با نام DevOpsDays در بسیاری از کشورها در سراسر جهان برگزار شده است . در سال‌های اخیر محبوبیت DevOps رشد کرده است و الهام بخش بسیاری از دیگر جنبش‌های موازی، از جمله OpsDev و WinOps شده است . WinOps هم مظهر همان مجموعه شیوه‌ها و تأکید بر فرهنگ DevOps است؛ اما به‌طور خاص مایکروسافت محور شده است. سایت استک اورفلو نشان میدهد که میزان علاقه به مفهوم DevOps رو به رشد است. (شکل ۲)

شکل ۲-نمودار میزان علاقه به مفهوم DevOps در استک اورفلو

نباید DevOps را به عنوان یک کلید واژه گنگ و مبهم تصور کرد، بلکه باید آن را یک مفهوم مهم با قابلیت بهبود چشمگیر محصولات دانست.

۲. تعریف DevOps

کلمه DevOps ترکیبی از **Dev**elopment به معنای توسعه و **Op**eration**s** به معنای عملیات می‌باشد. DevOps شامل امنیت، روش های همکاری مشترک، تجزیه و تحلیل داده ها و موارد دیگر است.

DevOps روش هایی را برای سرعت بخشیدن به فرایندهایی را توصیف می کند که با استفاده از آن ایده توسعه به استقرار در یک محیط تولید می رسد که می تواند برای کاربر ارزش ایجاد کند. این رویکرد مستلزم آن است که تیم های توسعه و تیم های عملیاتی مرتباً با یکدیگر ارتباط برقرار کنند و با همدلی نسبت به هم تیمی های خود به کار آنها نزدیک شوند.

۳. چرا DevOps مهم است؟

با ایجاد فضاهای ابری، بسیاری از تیم های توسعه نرم افزار به سراغ آنها رفتند. این فضاها امکان توسعه چابک نرم افزارها را به کاربران می دهد و در نتیجه امکان تعامل کاربران با سیستم های مدیریت بسیار بهتر و سریعتر شد. ولی مشکلی که وجود داشت هماهنگی تیم های کیفیت، روابط عمومی و تحقیق و توسعه بود. فضاهای ابری سرعت ها را زیاد و فاصله ها را کم کرد، در نتیجه تعارض هایی که تا قبل به مرور قابل حل بودند، اکنون به شدت رخ می نمودند و مشتریان ناراضی و فشارهای توسعه و عیب یابی سبب شد که مدیران به خلق مفهوم جدیدی به نام دواپس اقدام کنند.

در مفهوم DevOps سعی بر این است که تیم ها به همدیگر نزدیکتر شوند، تعاملات بیشتری با هم داشتن باشند و تا حد بسیار زیادی روال های تکراری حذف شود. با انجام اینکار تحویل ارزش به مشتری با شتاب بیشتری انجام می شد و توقف ها بسیار کمتر می گردید.

٤. فواید پیاده سازی DevOps :

- سرعت

DevOps با استفاده از هنر همگن سازی دو تیم توسعه و عملیات ، این امکان را به متخصصین می دهد تا تمام فرآیندهای چرخه ی بالا از جمله Test ، Release، Deployو build را به کمک پیاده سازی یک چرخه ی اتوماتیک با استفاده از ابزارهای مطرح مانند Jenkins که از جمله ابزارهای CI/CD می باشد سرعت ببخشد. با رویکرد دواپس، شرکت‌های تجاری یک محیط عملیاتی پایدار خواهند داشت، سطحی از خودکارسازی وجود دارد، عیب‌یابی سریع‌تر تشخیص داده می‌شود و انتشار محصول به‌صورت پیوسته انجام می‌گیرد. این‌ها همه مزایایی است که چابکی کسب‌وکار ارائه می‌دهد و نتیجه‌ای است که اغلب در سازمان‌های فعال به آن دست پیدا می‌کنند.

- پایداری سرویس

یکی از مشکلاتی که همواره تیم عملیات یا SysAdmin ها با آن مواجه بوده اند فرآیند آپدیت و به روزرسانی نرم افزار می باشد که ممکن است پس از این فرآیند ، کارکرد سیستم را از جنبه های مختلف مختل کند . DevOpsبا استفاده از پیاده سازی مراحل Continuous Integration و Continuous Delivery و حصول اطمینان از صحت عملکرد سیستم با استفاده از اجرای تست های مختلف قبل از Release و مشاهده ی لاگ نرم افزار در سیستم مانیتورینگ پس از Release کمک میکند تا نرم افزار به صورت پایدار همواره در حال سرویس دهی باشد.

- سیستم بهینه تر

در مدل DevOps به این دلیل که تیم توسعه و تیم عملیات همکاری نزدیک تری با یکدیگر دارند می توانند بسیاری از وظایف را که در مدل های سنتی تر تعریف می شود به صورت مشترک با یکدیگر بررسی کنند. این کار این قابلیت را به تیم توسعه می دهد تا با استفاده از دانش تیم عملیات ، کد خود را به صورت بهینه تر و مناسب تر برای محیط عملیاتی توسعه دهند و به تیم عملیات نیز این امکان را میدهد تا از جزئیات نرم افزار آگاه باشند تا نهایتا بسیاری از ریسک ها و مشکلات کاهش یافته و سیستم بهینه تر کار کند.

- امنیت

از آنجایی که متخصصین امنیت نیز یکی از عناصر DevOps می باشند ، بنابراین در تمام چرخه ی بالا حضور داشته و از ابتدای مراحل Plan تا نهایتا Deploy و Operate ناظر بر اجرای فرآیند ها می باشند.

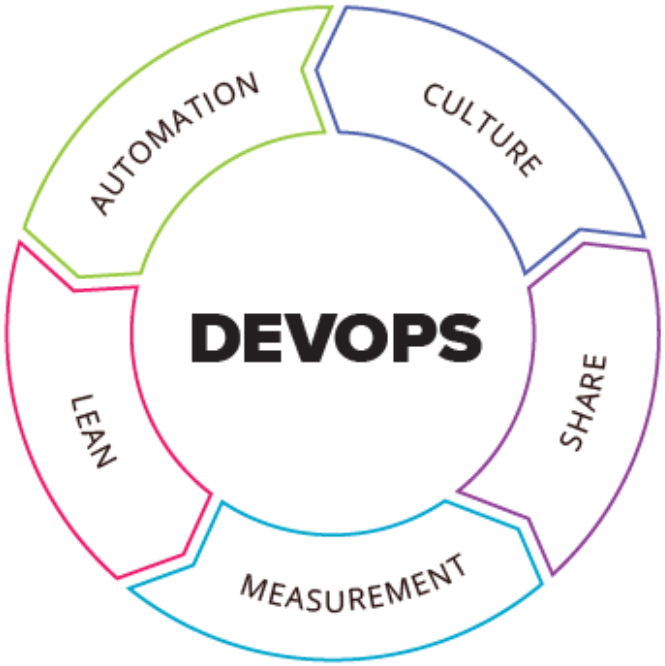
- افزایش ROI

نرخ بازگشت سرمایه یاROI ، شاخصی است که میزان بازگشت سرمایه را در کنار هزینه‌ها نشان می‌دهد. با ارائه سریع‌تر محصول به بازار، ROIبیشتر می‌شود. پیاده‌سازی رویکرد دواپس راه‌حل مناسبی برای شرکت‌های سازمانی است که به دنبال افزایش بازده سرمایه‌گذاری با انتشار مداوم محصول و نیز توزیع محصولات دیجیتالی کاربردی به کاربران نهایی هستند.

۵. اصول DevOps

برای درک راحت‌ترِ چگونگی فرآیند DevOps، از چارچوب CALMS استفاده می‌کنیم. این چارچوب به ۵ لغت زیر اشاره دارد:

Culture – Automation – Lean – Measurement – Sharing (شکل ۳)



شکل ۳- چارچوب CALMS

- فرهنگ (Culture)

پیش‌تر اشاره شد که یکی از تغییرات اساسی که DevOps ایجاد می‌کند، تغییر در فرهنگ تعامل افراد سازمان است. تیم‌ها برای رسیدن به اهداف مشترک باید منسجم و هماهنگ عمل کنند. بنابراین DevOps منجر به تقسیم شدن مسئولیت ها بین آن‌ها خواهد شد.

- اتوماسیون (Automation)

تیم‌ها به دنبال راه هایی برای اتوماسیون‌سازی وظایف هستند. که به این منظور مفاهیم Continuous Delivery، Continuous Integration، Continuous Deployment مطرح شد. پس اگر کارها در یک سازمان به طور دستی انجام می‌شود، نشان‌دهنده اینست که مفهوم DevOps در آن سازمان اجرا نشده است. چون روند کاری را کند کرده و درصد خطا را افزایش می‌دهد.

- بی ارزش یا اندک (Lean)

اکثر مواقعی که اصطلاح Lean در مفاهیم نرم افزاری قرار داشته باشد، این منظور را می‌رساند که عملیات بی ارزشی که فقط باعث هدر رفتن زمان می‌شوند، باید از روند کاری حذف شوند. در DevOps گفته می‌شود به طور مثال گسترش ویژگی‌های نرم‌افزار درحالی که مطابق نیاز کاربر نباشد کاری بیهوده است.

در واقع در مفهوم DevOps اگر اشتباهی رخ دهد، افراد هر واحد به جای این‌که به دنبال پیدا کردن مقصر باشند سعی می‌کنند دلیل آن را بیابند. بنابراین می‌توانند به راحتی کارهای تکراری و غیر ضروری را تشخیص داده و از چرخه‌ی کاری حذف کنند.

- سنجش (Measurement)

یکی از عواملی که نشان می‌دهد نرم‌افزار موفق بوده، وجود قابلیت سنجش بعد از Deploy شدن برنامه است. این قابلیت در DevOps با عنوان Measurment شناخته می‌شود. باید با جمع‌آوری اطلاعات، توانایی های برنامه و همچنین توسعه‌های آینده‌ی آن ارزیابی شود. خوشبختانه، ابزارها و فناوری‌های زیادی برای اندازه گیری کارایی وجود دارد. منظور از سنجش کارایی پاسخ به یک‌سری سؤالات مشخص است، مانند: کاربران در برنامه‌ی شما چقدر وقت می‌گذرانند؟ مطلبی که در وبلاگ نوشته بودید تا چه درصدی بر روی فروش تأثیر داشته؟ لاگ های قبلی چه هشدار هایی داشتند؟

با ملاک‌های خاصی می‌توانیم روال کاری را بسنجیم:

۱. بررسی و مانیتورینگِ زیرساخت
۲. مدیریت لاگ ها
۳. مدیریت کارایی و کاربرد برنامه
٤. بررسی تعداد باگ نسخه‌های قبلی
٥. ارزیابی سرعت میانگین تحویل هر نسخه

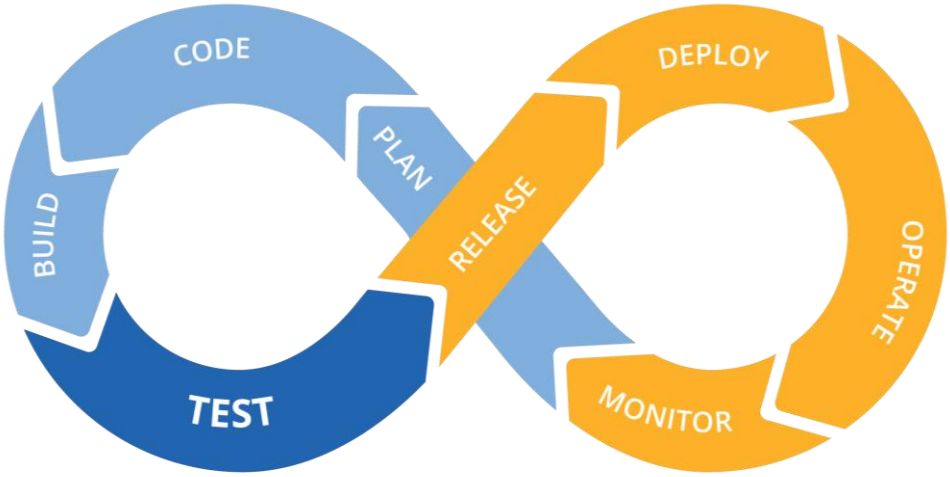
و هر معیار سنجش دیگری که میزان تولید ارزش برنامه را افزایش دهد.

- اشتراک گذاری (Sharing)

این مرحله از DevOps به اشتراک گذاری از همه‌ی جهات (تجربیات، دانش، محیط کاری و غیره) اطلاق می‌شود. همین همکاری و ارتباط نزدیک بین تیم‌ها باعث کاهش اشتباهات تکراری خواهد شد.

۶. چرخه زندگی DevOps

چرخه زندگی DevOps از مراحل مختلفی تشکیل شده است. (شکل ۴)



شکل ۴ – چرخه DevOps

برنامه ریزی: در ابتدا با توجه به نوع برنامه ای که برای توسعه نیاز دارید ، برنامه ریزی کنید. در مورد روند توسعه تصویری ایجاد کنید.

کد: برنامه را طبق نیاز مشتری کدگذاری کنید. با برنامه ، شما در مرحله اولیه ساخته اید.

Build: با اجرای یکپارچه سازی کدهای مختلفی که در مرحله قبل انجام داده اید ، برنامه را بسازید.

تست: این قلب برنامه است. برنامه ای را که تاکنون ساخته اید آزمایش کنید. و در صورت لزوم برنامه را بازسازی کنید.

نسخه ها: اگر در مرحله آزمایش موفق شدید ، زمان انتشار برنامه در Live فرا رسیده است.

استقرار: برای استفاده بیشتر کد را در یک فضای ابری مستقر کنید. این کار به روشی انجام می شود که هرگونه تغییر ایجاد شده نباید بر عملکرد وب سایت پربازدید تأثیر بگذارد.

Operate: در صورت وجود عملیات را روی کد انجام دهید.

Monitor: عملکرد برنامه را طبق نیاز مشتری کنترل کنید. در مورد عملکرد برنامه یادداشت کنید. در صورت وجود برای جلب رضایت مشتری اصلاحاتی را انجام دهید. و اگر تا مارک نرسد برای جلب رضایت مشتری در آن قسمت خاص تغییراتی ایجاد کنید.

و به این ترتیب چرخه حیات DevOps ادامه دارد.

۷. چالش های نبود DevOps در سازمان ها

در سیستم های سنتی تولید و نگهداری نرم افزار همواره چالش های فراوانی میان دو تیم توسعه به عنوان تولید کننده ی نرم افزار و تیم عملیات و پشتیبانی شبکه به عنوان ایجاد کننده و نگهدارنده ی محیط عملیاتی وجود داشته است که از جمله ی این چالش ها می توان به موارد زیر اشاره نمود:

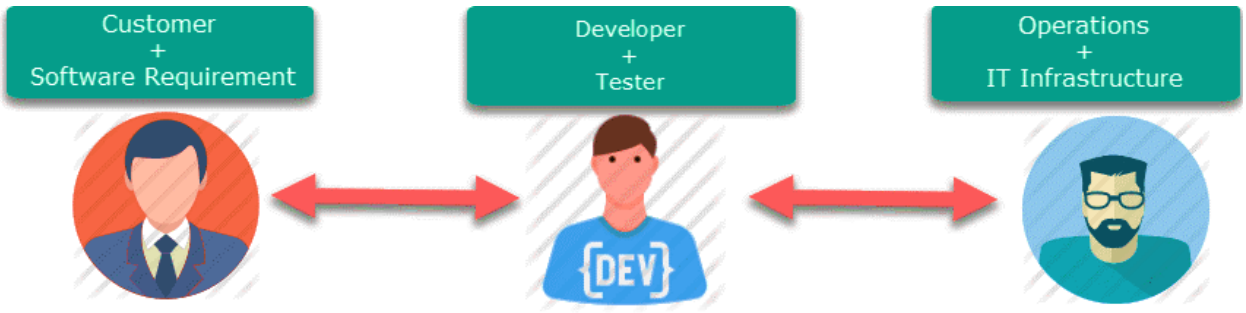
۱. تیم توسعه بدون درک و دانش کامل از محیط عملیاتی ، نرم افزار را توسعه داده و پس از اتمام تمام فرآیندهای توسعه آن را در اختیار تیم عملیات قرار میدهد تا در محیط عملیاتی برای کاربران در دسترس قرار دهد. این عدم وجود درک کافی از ساختار محیط عملیاتی می تواند شامل مواردی چون عدم شناخت ساختار شبکه اعم از Switching و Routing ، نوع Storage مورد استفاده و عملکرد آن ، نوع سیستم عامل ها ، معماری سرورها و Application Server های مورد استفاده ، تجهیزات امنیتی و ... باشد که همگی آنها تاثیر مستقیمی در نحوه ی عملکرد نرم افزار طراحی شده داشته و عدم شناخت کافی آنها می تواند منجر به شکست پروژه گردد.

۲. تیم عملیات نیز بدون درک کافی از ساختار نرم افزار توسعه داده شده و همچنین نبود دانش کافی در حوزه ی برنامه نویسی ، فرآیندهای تولید نرم افزار ، توابع و کلاس های مورد استفاده شروع به عملیاتی نمودن نرم افزار می نماید که این نیز به صورت متقابل نهایتا می تواند پروژه را منجر به شکست کند.

دواپس در واقع تلفیقی است از یک فرهنگ ، ابزار و دانش که با استفاده از آن سازمان ها می توانند اولا روند تولید نرم افزار را چابک تر کنند و ثانیا نرم افزار تولیدی را در محیطی بهینه و پایدار تر پیاده سازی و نگه داری کنند. در این مدل دیگر دو تیم توسعه و عملیات جدای از یکدیگر کار نمی کنند و متخصصین آن در کل چرخه ی تولید ، تست و پیاده سازی آن در محیط عملیاتی نقش خواهند داشت . حتی در برخی از مدل ها که به آن DevSecOps می گویند تیم امنیت نیز از اعضای این تیم بوده و بر تمامی فرآیندها از منظر امنیت نظارت دارد.

۸. تفاوت Agile و DevOps

بسیاری از افراد متدلوژیِ Agile و DevOps را یک مفهوم در نظر می‌گیرند. درحالی که می‌توان گفت DevOps نسخه‌ی پیشرفته‌تر از Agile است. اما این دو مفهوم، تفاوت هایی دارند که تصاویر زیر بیانگر این موارد خواهد بود. (شکل ۵)



شکل ۵ - تفاوت DevOps و Agile

افراد حاضر در یک پروژه فناوری اطلاعات (IT) عادی شامل کاربر، گروه توسعه یا برنامه نویسان و گروه عملیاتی هستند. در این فرآیند تمام این واحد ها باید با یکدیگر تبادل نظر و گفتگو داشته باشند تا برنامه نرم افزاری مطابق نیازمندی های کاربر و البته استاندارد تولید شود.

متد Agile راهکاری برای از بین بردن فاصله‌ی بین مشتری و توسعه دهنده می‌باشد. در واقع ارتباطات بین مشتری و برنامه نویس را برقرار می‌کند. در این متدِ توسعه نرم افزاری، مراحل تولید محصول به بخش های کوچک‌تر قابل انجام تقسیم می‌شود و برای آزمایش نهایی با یکدیگر ادغام و یکپارچه می‌شوند.

اما تمرکز متدولوژی DevOps، همان‌طور که اشاره شد بر روی برقراری ارتباط میان دو تیم توسعه و عملیاتی است. این راهکار به دنبال تولید و ارائه با سرعتی بالاتر و تا حد امکان خودکار می باشد.

از تفاوت های اساسی این دو متد می‌توان به موارد زیر اشاره کرد:

- Task یا وظایف

وظایف افراد در Agile ایجاد تغییرات فوری طبق خواسته‌ی مشتری می‌باشد، درحالی که در DevOps قرار بر تست و Deploy به صورت مداوم است.

- Purpose یا هدف

Agile برای مدیریت پروژه‌های پیچیده استفاده می‌شود؛ که به ارتباط مداوم با مشتری و جمع آوری اطلاعات نیاز دارد. اما ارتباطی که در DevOps صورت می‌گیرد بین متخصصان یک سازمان است تا بتوانند هرچه سریع‌تر فرآیند تولید را پیش ببرند.

- Team Size تعداد اعضای تیم

در Agile هرچه تعداد کمتر و تیم کوچک‌تر باشد، سریع‌تر در مسیر تولید حرکت می‌کنند. در حالی که در DevOps کاملا برعکس است.

- Feedback یا بازخورد

در Agile بازخورد از ابتدای کار از جانب مشتریان است. اما در متد DevOps از آنجایی که محصول در چرخه‌ی تولید دست به دست می‌شود، تیم های داخلی نیز بازخورد خود را ارائه می‌دهند.

۹. کاربرد DevOps کجاست؟

اگر شما یک سرویس یا محصولی تولید می کنید که دائم بر اساس نظرات مشتری یا بازخورد بازار تغییر می کند و ویژگی های جدید به آن اضافه می شود و فکر می کنید مزیت رقابتی شما ارائه سرویس خوب به مشتری است، پس احتمالا باید بدنبال این مفهوم باشید. اما معمولا اگر در سازمان هایی هستید که سرویس هایی با تکنولوژی های خیلی قدیمی وجود دارند و اصولا همه چیز دستی انجام می شود و روال های سازمانی اجازه اتوماتیک شدن به شما را نمی دهند، شاید استفاده از این مفهوم کار بسیار سختی باشد.

در شرایط پرفشار کنونی و در حالی که برنامه‌های زمان‌بندی تا حد امکان فشرده شده‌اند، دیوارهای موجود بین گروه‌های توسعه، تضمین کیفیت و محیط اجرا مانعی بر سر چابکی سازمان هستند و «دواپس» سعی در شکستن این مرزها دارد و مهارت‌های مدیریت گروه را به اندازه مهارت‌های فنی ارزشمند می‌داند. همچنین، تمرکز این رویکرد تنها بر تجربه کاربر از محصول ارائه شده و نحوه اثرگذاری این تجربه بر سازمان است. بنابراین، می‌توان گفت «دواپس» تنها ابزار جدیدی برای انجام همان کارهای قدیمی به گونه‌ای بهتر نیست و معرف یک فرهنگ و فرآیند جدید است. فرهنگی که از همکاری و تعامل گروه توسعه، تضمین کیفیت و محیط عملیاتی برآمده و قرار است تا ضامن خروج سیستمی کارا باشد.

فصل سوم : ابزارهای DevOps

۱. ابزار دواپس Gradle

بدون شک برای انجام بسیاری از تسک‌های مرتبط با دواپس، شما نیاز به ابزاری برای Build کردن دستورات خواهید داشت. سال‌های بسیار زیادی است که ابزارهایی مانند Apache و Maven نام خود را در این حوزه تثبیت کرده‌اند. در کنار این دو، ابزاری با نام Gradle نیز وجود دارد که از سال ۲۰۰۹ تا به امروز توانسته توجهات فراوانی را به سوی خودش جلب کند. این ابزار به شدت محبوب و پرکاربرد، محیطی را برای شما فراهم می‌کند تا بتوانید در زبان‌های مختلف مانند جاوا، C++، پایتون و سایر زبان‌ها برنامه نویسی نمایید. علاوه بر این، Gradle بسیاری از IDE های موجود در بازار همانند Netbeans ، Eclipse و حتی IntelliJ را تحت پشتیبانی دارد. اگر هنوز متقاعد نشدید که از Gradle استفاده کنید، باید این موضوع را هم اشاره کنیم که گوگل این نرم افزار را به عنوان ابزار اصلی Build کد در Android Studio معرفی کرده است. (شکل ۶)

شکل۶- Gradle یکی از ابزار دواپس

بر خلاف Maven و Ant که برای تنظیمات از کدهای XML استفاده می‌کنند، Gradle از یک DSL مخصوص که بر پایه Groovy اجرا می‌شود برای عملیات build استفاده می‌نماید. تیم Gradle در سال ۲۰۱۶ یک DSL مبتنی بر Kotlin را نیز معرفی کرد که این امکان را برای توسعه دهندگان فراهم می‌کند که دستورات Kotlinرا در این فضا اجرا کنند.

در نتیجه باید گفت که یاد گرفتن زیر و بم Gradle اندکی زمانبر است. به همین دلیل داشتن آمادگی ذهنی در زمینه‌هایی از قبیل Groovy ، Kotlin یا سایر زبان‌های JVM ، می‌تواند مسیر یادگیری Gradle را برای شما تسهیل نماید. یکی از نکاتی که برنامه نویس‌ها در زمان کار کردن با ابزارهای جدید نگران آن هستند، حالت‌های مختلف ذخیره‌سازی (Repository Format) است. از آنجایی که Gradle از فرمتی مشابه با maven برای ذخیره اطلاعات استفاده می‌کند، افرادی که سابقه کار با Maven را در رزومه خود دارند، در کار با Gradle مشکل خاصی نخواهند داشت.

استفاده از روش Incremental Build نیز به Gradle کمک می‌کند تا در مقایسه با سایر نرم افزارها، زمان کمتری برای کامپایل کردن کد نیاز داشته باشد. طبق نتایجی که از تست‌های مختلف به دست آمده است، کدها در Gradle چیزی در حدود ۱۰۰ برابر سریع‌تر از Maven کامپایل می‌شوند Gradle .برای بهبود سرعت کامپایل کردن، از کش‌های درون سیستمی و Deamon استفاده می‌کند.

به صورت کلی، Gradle امنیت، سرعت و راحتی را یکجا به توسعه دهندگان ارائه می‌دهد.

۲. نرم افزار مدیریت کد Git

اغراق نیست اگر بگوییم Git یکی از شناخته شده‌ترین و البته پرکاربردترین ابزارهای دواپس در دنیاست . Git در اصل یک ابزار مدیریت سورس کد است که به صورت گسترده توسط تیم‌هایی که به صورت ریموت بر روی یک پروژه کار می‌کنند، استفاده می‌شود. یکی از قابلیت‌های برتر Git امکان بررسی مسیر پیشرفت یک پروژه از ابتدا تا انتهاست. توسعه دهندگان با استفاده از Git می‌توانند به راحتی به نسخه‌های مختلف سورس کد خود دسترسی پیدا کرده و تغییرات مورد نیاز را اعمال نمایند. همچنین Git یک محیط آزمایشگاهی بی‌نظیر هم برای توسعه دهندگان فراهم آورده تا آن‌ها بتوانند با ادغام بخش‌های مختلف، نتایج به دست آمده را بررسی کرده و پیش از انتشار نسخه نهایی، از عملکرد کلی برنامه ساخته شده اطمینان حاصل نمایند. (شکل ۷)

شکل ۷ - Git یکی از ابزار دواپس

برای استفاده از قابلیت‌های Git ، توسعه دهندگان نیاز به یک Repository خواهند داشت تا امکان ذخیره‌سازی پروژه‌ها بر روی آن وجود داشته باشد. وقتی چرخی در فضای اینترنت می‌زنیم، با دو نمونه بسیار پرطرفدار یعنی GitHub و Bitbucket برخورد می‌کنیم. در حالی که GitHub شهرت بیشتری دارد، اما به نظر می‌رسد Bitbucket به دلیل داشتن فضای رایگان و نامحدود ذخیره‌سازی، انتخاب بهتری برای تیم‌های کوچک است. البته کاربران GitHub همچنان می‌توانند به صورت کاملا رایگان به Repository های عمومی دسترسی داشته باشند.

اگر به دنبال یکپارچه سازی ابزار مختلف هستید، باید بدانید که هم GitHub و هم Bitbucket این قابلیت را دارند که با ابزارهایی مانند Slack همگام‌سازی شوند. این امر افزایش سرعت پیشروی کارها را به دنبال خواهد داشت.

۳. ابزار دواپس Jenkins

یکی از ابزارهای محبوب توسعه دهندگان نرم افزار برای اتوماسیون سازی فرآیندهای مرتبط با دواپس، Jenkins است. Jenkins یک سرور CI/CD متن باز است که به تیم‌ها اجازه می‌دهد تا قسمت‌های مختلف از ساخت یک پروژه را به صورت اتوماسیون در بیاورید. پلاگین‌های بی‌شمار موجود در Jenkins یکی از مهم‌ترین دلایل شهرت جهانی این ابزار متن باز است. در حال حاضر بیش از ۱۰۰۰ پلاگین در Jenkins ثبت شده است که امکان پشتیبانی از تمام ابزار دواپس را برای توسعه دهندگان نرم افزار فراهم می‌کند.

استفاده از امکانات Jenkins به توسعه دهندگان این امکان را می‌دهد تا مراحل مختلف CI/CD را طبق نیازهایشان شخصی سازی نمایند.

پشتیبانی تمام سیستم عامل‌های ویندوز، مک و لینوکس از Jenkins سبب شده تا توسعه دهندگان کار سختی برای استفاده از آن نداشته باشند. علاوه بر این، امکان نصب Jenkins از طریق Docker نیز فراهم است. (شکل ۸)

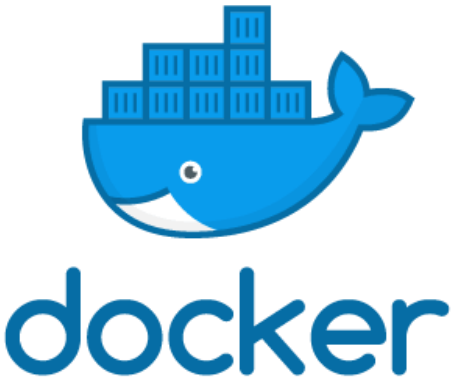
شکل ۸ - Jenkins یکی از ابزار دواپس

راه اندازی و تنظیم سرور Jenkins به سادگی از طریق یک صفحه وب امکان پذیر است. Jenkins برای راحتی بیشتر کاربرانی که خیلی با فضا آشنا نیستند، مجموعه‌ای از تنظیمات را به صورت پیش فرض تعریف کرده است. همچنین افرادی که به صورت حرفه‌ای از این ابزار استفاده می‌کنند، می‌توانند به راحتی در هر مرحله تنظیمات مورد نظر خود را اجرا نمایند.

Jenkins به شما این امکان را می‌دهد تا دستورات توسعه یافته را به سرعت اجرا کرده و عملکرد بخش‌های مختلف آن را مورد بررسی قرار دهید. البته تعدادی از کاربران، محیط Jenkins را بی روح و خسته کننده می‌دانند؛ اما به هر حال هیچ کس نمی‌تواند قابلیت‌های فراوانی که این برنامه در اختیار توسعه دهندگان قرار می‌دهد را انکار کند.

۴. پروژه متن باز Docker

بدون شک Docker از همان روزهای ابتدایی انتشارش در سال ۲۰۱۳، توانست به عنوان یکی از بهترین Container Platform ها شناخته شود. شاید بتوان Docker را مهم‌ترین ابزار مورد نیاز برای مهندسان دواپس دانست. در اصل می‌توان گفت که Docker موانع موجود در عملیات یکپارچه سازی بخش‌های مختلف نرم افزار را کنار می زند. دلیل اصلی این امر، فراهم کردن بستری مناسب برای توسعه گسترده (Distributed Development) و اتوماسیون سازی فرآیند اجرای برنامه است. روش کار در Docker به این صورت است که برنامه به Container های جداگانه تبدیل می‌شود تا اعمال هرگونه تغییری در آن با سرعت بیشتری انجام شده و امنیت کلی نرم افزار تا حد قابل قبولی افزایش پیدا کند. تمام برنامه‌هایی که با استفاده از Docker توسعه می‌یابند، حساسیتی در مقابل سیستم عامل‌های مختلف نداشته و در هر پلتفرمی قابل اجرا هستند. به همین دلیل است که امکان استفاده از کانتینرهای Docker به عنوان ماشین‌های مجازی همانند VirtualBox نیز وجود دارد. (شکل ۹)



شکل ۹ - Docker یکی از ابزار دواپس

نکته جالب توجه در Docker این است که توسعه دهنده نیازی به مدیریت روابط مختلف در بخش‌های گوناگون یک نرم افزار نخواهد داشت. چرا که Docker با دسته‌بندی تمام روابط و قرار دادن آن‌ها در در یونیت‌های جدا از هم، کار مدیریت تمام روابط را بر عهده می‌گیرد. به همین دلیل برنامه‌هایی که با استفاده از Docker توسعه داده می‌شوند، به هیچ پلتفرمی وابسته نیستند.

اگر بخشی از کار شما بر روی پلتفرم‌های Jenkins و Bamboo است، نیازی به نگرانی نیست. چرا که امکان یکپارچه سازی Docker با این پلتفرم‌ها وجود دارد. علاوه بر این، Docker در زمینه رایانش ابری نیز به ابزاری بسیار قدرتمند بدل شده است. طی سالیان گذشته، بسیاری از سرویس‌های رایانش ابری مانند AWS و Google Cloud به صورت مستقیم از Docker پشتیبانی می‌کنند. پس اگر به دنبال پیاده‌سازی زیرساخت‌های خود به صورت Cloud هستید، و یا قصد دارید به عنوان کارشناس رایانش ابری استخدام شوید، حتما به Docker نیاز خواهید داشت.

۵. پلتفرم مدیریت کانتینر Kubernetes

اگر بخواهیم به‌زبان ساده کوبرنتیز را توضیح دهیم باید بگوییم کوبرنتیز اجرا و مدیریت کانتینرهای مختلف را در سرورهای متفاوت که در یک پایگاه داده یا چندین پایگاه قرار گرفته‌اند را بر عهده می‌گیرد. در کوبرنتیز کانتینرهای مختلفی که مشترکاً برنامه کاربردی خاصی را شامل می‌شوند در حالت جداگانه و مستقل تحت عنوان پاد (Pod) دسته‌بندی خواهند شد. این کار فرآیند مدیریت و شناسایی آن‌ها را ساده‌تر می‌کند. (شکل ۱۰)

شکل ۱۰ - Kubernetes یک پلتفرم مدیریت کانتینر

به این ترتیب می‌توان گفت سازمان‌ها و شرکت‌هایی که سرویس‌های مختلف نرم‌افزاری را اجرا می‌کنند ابتدا به کانتینرها و در نهایت به ابزارهایی مانند کوبرنتیز نیاز دارند تا با کمک گرفتن از کانتینرها، برنامه‌ها را در بهترین حالت از یکدیگر جداسازی کنند. این فرآیند تولید و آزمایش اپلیکیشن‌ها و سرویس‌ها را ساده‌تر کرده و امکان اجرای آن‌ها در یک زیرساخت مشترک را فراهم می‌کند.

کوبرنتیز کمک می‌کند تا کانتینرها در گروهی از ماشین‌ها به صورت خودکار و اتوماتیک اجرا شوند، به این ترتیب به زبان ساده‌تر می‌توان گفت کوبرنتیز نقش سیستم‌عاملی را ایفا می‌کند که بر روی چندین سرور در حالت یکپارچه اجرا می‌شود. در نتیجه نیازی به نگرانی برای وضعیت ماشین‌های مختلف وجود ندارد و کاربران در حالی که هیچ تغییری در سرویس‌های اجرا شده مشاهده نمی‌کنند قابل تعامل با اپلیکیشن‌ها و سرویس‌های مورد نظر هستند.

۶. ابزار دواپس Puppet Enterprise

ابزار Puppet Enterprise سرویس زیرساختی برای کد (Infrastructure as Code) را به کسب و کارها ارائه می‌دهد. این ابزار با اتوماسیون سازی فرآیندهای مرتبط با زیرساخت سیستم، این امکان را فراهم می‌کند که تیم توسعه با نگرانی کمتری به کار خود ادامه دهند و محصول در زمان سریع‌تری وارد بازار شود. (شکل ۱۱)

شکل ۱۱ - Puppet Enterprise یکی از ابزار دواپس

به صورت کلی استفاده از Puppet به عنوان یک ابزار متن باز برای پروژه‌های کوچک، گزینه بسیار مناسبی است. اما سازمان‌ها و شرکت‌های بزرگ می‌توانند با استفاده از Puppet Enterprise از قابلیت‌های زیر برخوردار شوند.

- دسترسی به گزارشات در هر لحظه
- امکان اهدا دسترسی‌های مختلف به کاربران
- مدیریت بخش‌های مختلف یک پروژه

به طور کلی، Puppet Enterprise به شما این امکان را می‌دهد که چندین تیم و هزاران زیرساخت را در یک جا مدیریت کنید. با استفاده از علم یادگیری ماشین، این ابزار به مرور زمان میزان دسترسی و نحوه استفاده شما از زیرساخت‌های گوناگون را فرا گرفته و در نتیجه سناریو دقیق‌تری را برای بهبود مدیریت زیرساخت به کار می‌گیرد. همچنین هوش مصنوعی Puppet باعث می‌شود که میزان خطاهای به وجود آمده در سیستم به صورت تدریجی کاهش پیدا کند. یکی از بهترین ویژگی‌های Puppet بهره گیری این سیستم از ۵۰۰۰ ماژول پیش ساخته است Puppet Enterprise .همچنین از تمام ابزارهای دواپس شناخته شده نیز پشتیبانی می‌کند.

۷. Ansible

Ansible یکی ابزار متن باز اتوماسیون تأمین ، مدیریت پیکربندی ، تنظیمات و گسترش برنامه ها بر روی لینوکس و فضای ابری است که برای کنترل سرورها از طریق SSH اقدام نموده و نیازی به Agent بر روی سیستم کلاینت ندارد.

با نصب و کانفیگ این ابزار بر روی سرور اصلی که با نام Control Node شناخته می‌شود ، شما می‌توانید سایر کلاینت را از طریق SSH مدیریت نمایید. (شکل ۱۲)

شکل ۱۲ - ANSIBLE

این ابزار در حال حاضر بر روی توزیع های CentOS، Redhat Enterprise، Debian، Ubuntu و سایر توزیع های استاندارد بر پایه لینوکس‌های ذکر شده قابل نصب و استفاده می‌باشد.

در یک تعریف کلی می‌توان گفت ، Ansible یک ابزار Remote Administration است که این توانایی را به مدیر یک شبکه می‌دهد تا سرورهای لینوکسی دیگر را به صورت جامع در یک پنل کارآمد مدیریت و کنترل نماید.

نتیجه گیری

DevOps آمیزه و مخلوطی از چندین نقش بوده و هدف نهایی آن کنار هم قرار دادن دولوپرها و مهندسان اجرایی است. فرهنگ DevOps ویژگی‌ها و قابلیت‌های جدید محصول را با زیرساخت‌های آن سازگار می‌نماید و سبب می‌شود تا این دو بتوانند در کنار هم به خوبی عمل کنند.

استفاده از سیستم دواپس و مهندسانی که با این ساختار آشنا هستند، می‌تواند تاثیر بسیاری مثبتی بر روی یک برند داشته باشد. با افزایش هماهنگی میان اعضای تیم، می‌توان شاهد بهبود بهره‌وری تیم‌ها و ارتباطات درون سازمانی شد. تمام این موارد می‌تواند در نهایت باعث شود که یک کسب و کار، اعتبار بیشتری به دست آورد و رضایت مشتریانش را جلب کند.

مراجع

1) https://sadr-group.net/
2) http://mimtech.ir/mag/devops/
3) https://www.cheragh.com/blog/
4) https://blog.iranserver.com/kubernetes/
5) https://kaajhost.net/what-is-devops/
6) https://www.datisnetwork.com/what-is-devops.html
7) https://itbaz.net/9612/what-is-devops/
8) https://devops.com/using-calms-to-assess-organizations-devops/
9) https://dzone.com/articles/life-cycle-of-devops
10) https://docs.docker.com/get-started/
11) https://www.git-scm.com/book/en/v2
12) https://azaronline.com/blog/introducing-ansible/
13) https://www.toptal.com/insights/innovation/what-is-devops
14) https://www.redhat.com/en/topics/devops
15) https://fa.wikipedia.org/wiki/%D8%AF%D9%88%D8%A7%D9%BE%D8%B3
16) Introduction to DevOps on AWS *David Chapman December 2014*
17) https://www.altexsoft.com/blog/engineering/devops-principles-practices-and-devops-engineer-role/
18) The DevOps Handbook: How to Create World-Class Agility, Reliability, and Security in Technology Organizations