Group Discussion

Facilitator Notes:

Divide the students in groups and discuss about releasing an application to the production, CD engineering best practices, Continuous Testing and Continuous Feedback.

We've so far discussed about different aspects of releasing an application to the production, the best practices followed by CD engineering , Continuous testing and Continuous Feedback. Let's discuss Compuware DevOps case study and know how they got benefited by adopting DevOps model.

1.3.8 Compuware DevOps Transformation Case Study

Challenges

- Compuware was following Waterfall model for 40 years.
- They are a slow-moving development organization.
- They were trying to compete in the digital economy, which requires you to be fast, (fast beats slow).
- They needed to be innovative, and have lot of new ideas. Ideation is the key to success in the new economy.
- Being a software vendor they need to maintain quality.

Facilitator Notes:

Explain the participants about Compuware DevOps transformation case study.

For Compuware, the solution to above problems were:

Agility: They wanted to be fully agile so that they could have continuous code drops and fulfill business needs in a timely basis.

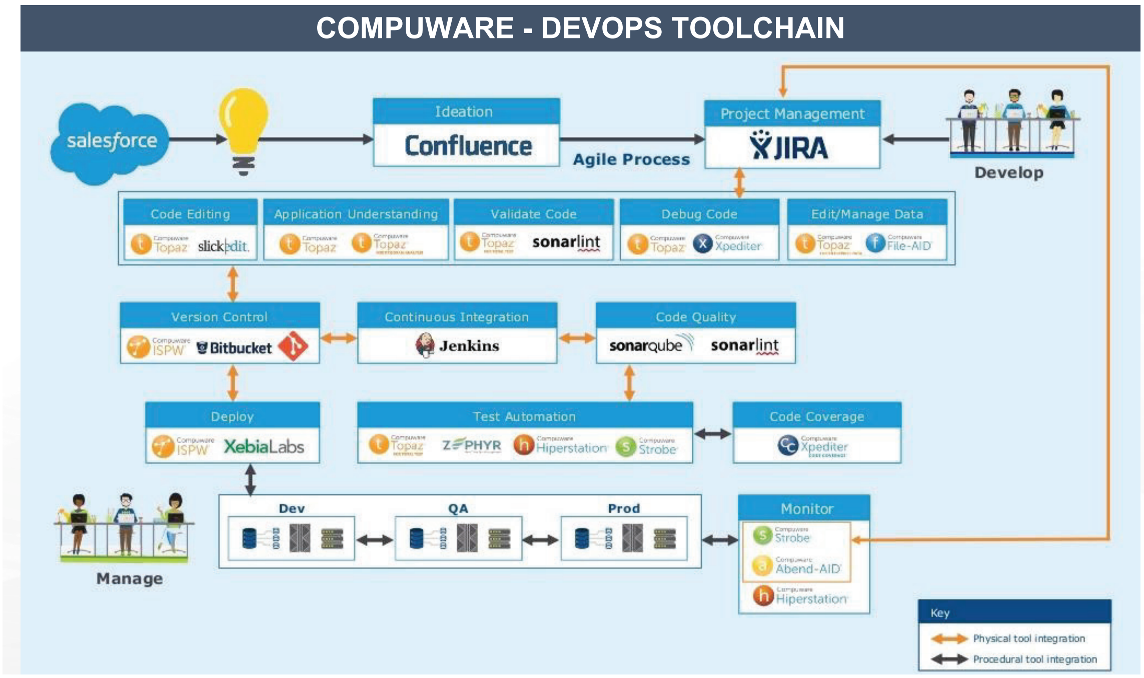
Confidence: They wanted to have the confidence to know that when changes are made, what they were delivering was going to work and was good, so we had to know that they could move at a fast pace to be able to deliver what they wanted.

Efficiency: They had to be efficient, and move across the entire enterprise.

Ease of use: Their tools had to be easy to empower the developers to be able to move at that pace, and to make updates and enhancements that would be valuable, provide business needs, and meet the business objectives.

Integrations: Being in the enterprise companies, the mainframe integrates across the entire enterprise, through the cloud, and all the way back to your mainframe system of record. So they had to make sure that those integrations would be available.

1.3.8.1 Compuware - DevOps Toolchain



Facilitator Notes:

Give a brief to the participants about Compuware DevOps toolchain.

Determining the Right tools:

What they did was to put in a toolchain, below you'll see an example of Compuware DevOps Toolchain that they are using to develop the software.

- First, in the upper left hand side, let's start with Salesforce, which is how they interact with the customers so that they can provide ideas to Compuware, and can communicate using that portal.
- Next tools is Confluence. When they have a new idea that comes in, they put it out in Confluence, and they get a lot of collaboration from the people around the world who are interacting with their customers and using Compuware products.
- Finally, they use JIRA for all of the project management, for all of the task management, for the agile boards, etc. so that they can keep track of everything they are doing.

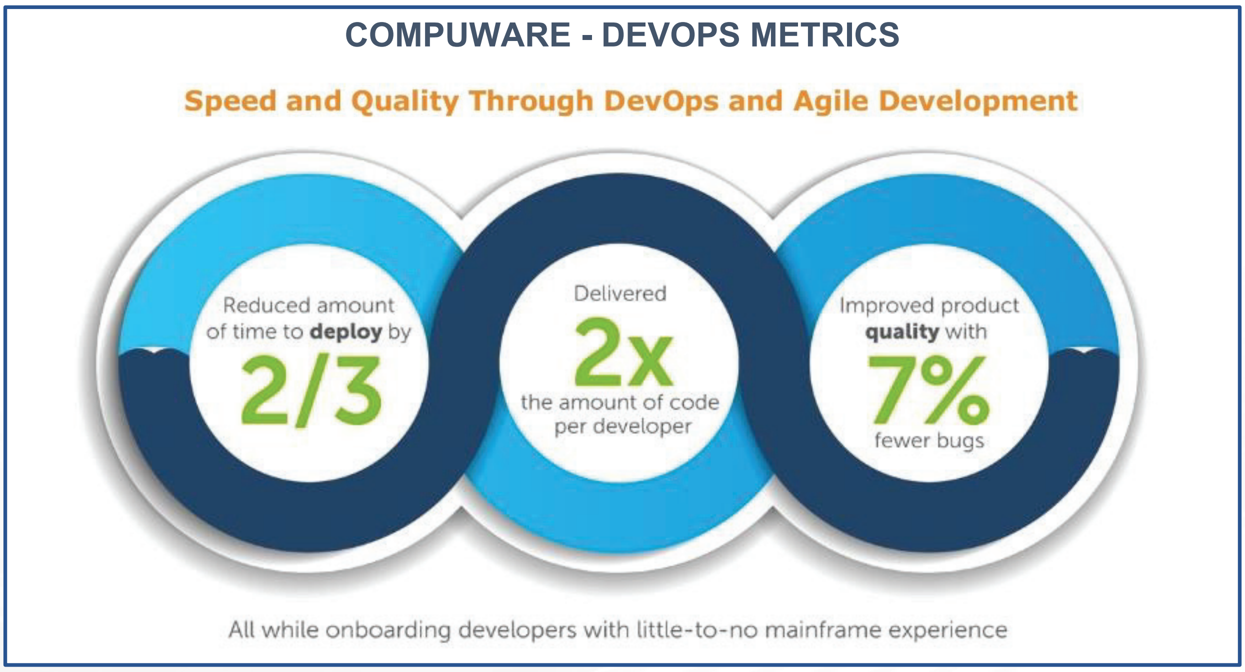
The next phase happens as they move through code editing, application understanding, to validate their code, debug the code, edit, and manage the code, etc.

For this stage, they use the Compuware Topaz base, which is an Eclipse Base ID which interacts with the mainframe tools. It works with the interactive debuggers, and it can integrate with all the different tools.

- They use two version control systems.
- They use ISPW, which is the mainframe source code management system; and they also use Git for the distributed code. They believe that the source code should be managed on the platform where it is going to be built and executed, so mainframe code stays on the mainframe, distributed code stays on distributed platform.
- They use Jenkins for the continuous integration into ISPW and Git to identify when code changes are made. Once those code changes are made, it automatically initiates automated testing; both mainframe, distributed platforms as well as builds.

- Then the results from the testing go into Sonar where they can use it to identify code quality, identify how the changes are made to their products, and how the code is impacting the quality of what they are delivering.
- For test automation, they use not only Topaz products, but also use Zephyr and Hiperstation Strobe for performance. So they are monitoring the continuous performance and continuous viability of all code changes. They like to ensure that they have at least 50% code coverage as they move along.
- And then, finally, they get to where they have to deploy. Again, they use ISPW and XebiaLabs XL Release to do the deployments across different platforms, and that allows them to move and confidently move code changes from development to QA to production.
- Then, once they are in production, they have to monitor if there's faults, errors, or any performance issues. They continuously monitor performance using Strobe, and have an integration back into JIRA. Within the environment, if an issue is found, JIRA issues can be opened and sent back to developers so that they can keep a continuous integration in the true spirit of being fully DevOps enabled.

1.3.8.2 Compuware - DevOps Metrics that matter


COMPUWARE - DEVOPS METRICS
Speed and Quality Through DevOps and Agile Development
Reduced amount of time to deploy by
2/3
Delivered
2x
the amount of code per developer
Improved product quality with
7%
fewer bugs
All while onboarding developers with little-to-no mainframe experience

Facilitator Notes:

Explain the participants about Compuware DevOps metrics that is improved with CI/CD

- Through the DevOps toolchain, Compuware is able to cut their deployment time by two thirds, which means they were able to more rapidly deploy the software that they have.
- As per the estimate they do about two times more code delivery per developer per year, and that's not measured by lines of code.
- They actually measure that by stories, tasks, and enhancements that are delivered. In other words, they are measuring on the value that they are providing to the end users in applications that is being delivered, not just simply lines of code.
- Through automated testing and the use of continuous integration, year after year they have reduced the defects reported by the customers and end users by seven percent.

In a nutshell, we learnt

1. Releasing an application to production
2. CD engineering practices
3. Continuous Development & Integration
4. Continuous testing & promotion of builds
5. Continuous Deployment to successive environments until before Production
6. Continuous monitoring for the delivery pipeline
7. Continuous feedback

Facilitator Notes:

Share the module summary with the audience.

Ask the participants if they have any questions. They can ask their queries by raising their hands.