1. How do you define DevOps?

DevOps is a clipped compound term which combines "Development" and "Operations" practices of IT software development, having originated in the mid 2000s among IT professionals looking for efficient and innovative ways to automate and speed up the process of software delivery. As a change agent, DevOps promotes a culture of collaboration and information sharing across the organization, a radical departure from the 'silos' of the past.

Nowadays, the DevOps cultural movement has spread far and wide among the technical community and can no longer be confined to software development alone. Its scope of adoption has pervaded product engineering services, various devices of 'Internet of Things' and Cloud-enabled services, with CloudOps as a resulting metonym.

2. Why is there a need for DevOps or why does DevOps matter?

While the Developer community performs software development which is all about coding; writing the code, implementing, testing and re-writing the code, operations team looks after the systems that run that code. They work on things like how much processing power the software will need to run, how to make the software secure, how to make it run efficiently, and how to keep it running.

DevOps is needed when these two teams work for the same system, but operate with relatively different thought processes. DevOps helps these developer and operations communities to learn how to work in a new way that facilitates the complete cooperation with each other.

With DevOps, operations staff uses many of the techniques used by developers in their systems. DevOps provide a way for operations and development people with common methodologies and processes to work together in a mutual cooperation.

3. How DevOps in IT differs from DevOps in Internet of Things (IoT)?

The biggest difference is that compared to pure play IT where you have to deal with software codes alone, when applying DevOps to IoT developments, you also need to take physical components, devices and existing legacy systems into account which together build in an extra layer of complexity in the overall automation process.

Then, there are specific challenges during each and every stage of the project, including a fragmented development pipeline comprising very small teams, automated Build-Verification-Test (BVT) plans, greater usage of legacy systems, non-unified release cadence and conversion of the entire production environment to a singular code ("environment as a code").

For more information on this subject, we highly recommend that you check out this white paper written by eInfochips subject matter experts, titled "DevOps in the Age of Connected Devices".

4. Why is it that DevOps can be so easily extended to the world of product engineering services?

The electronic and industrial devices and intelligent gadgets commonly seen in IoT era operate based on the software running inside them producing advanced business intelligence (BI). This software has to be managed, tested and updated frequently, and in real time. Also, there are physically devices connected to each other, which may include GPS, cameras, accelerometers, energy meters, medical devices, machine sensors and so much more. DevOps is the only methodology available right now which can help prepare for real time failure scenarios with the use of simulators, virtual machines and remote monitoring and updates of firmware.

5. In which industries can you find DevOps organizations?

DevOps has been a great disruptor in practically every industry that depends on software delivery, and other application delivery endpoints including diverse devices, web, and mobile services.

Some of the industries where eInfochips has direct, proven experience in enabling DevOps services include:

- Home automation
- Industrial automation
- Medical devices
- Video surveillance
- Networking

6. Are Agile and DevOps same?

No, agile is not same as DevOps. However, agile can be used as a part of DevOps. Here are the ways in which they are different.

Agile is a software development methodology. Once the software is developed and released, the agile team doesn't care about how the software is doing, instead, they move to next sprint.

DevOps is all about developing software, making it ready for release and deploying it in the safest, most reliable way. In DevOps, the software development doesn't need to be using agile discipline. It may use waterfall development process as well.

7. Can DevOps be helpful in Remote Device Management (RDM)?

Indeed it is. DevOps is the most efficient way to realize the vision of IoT because as a methodology, it supports bulk operations on many devices simultaneously.

8. What are the issues facing development teams in DevOps?

Development teams adopting DevOps have to overcome challenges mainly due to their existing business environments comprising organizational silos which are a major impediment to the success of DevOps. The biggest problem lies in prioritizing the importance of the products, projects and applications for which monitoring and deployment has to be performed at multiple ends. In order to tackle these issues, DevOps streamlines automation processes to achieve business agility. This helps in delivering a product with total commitment and achieve better quality standards.

9. How to measure DevOps?

DevOps can be measured according to the following mentioned categories:

- Deployment frequency: Direct and indirect measure of response time, team efficiency and capabilities, and DevOps tools evaluation and effectiveness.
- Mean time to recover (MTTR): It is a metric which stands for time to recover from a given failure. It measures both team capability and the rate of failures.
- Change lead time: The time elapsed from first code sent to operational teams to its deployment at customer end which defines the complexity of the code and the team capabilities of developers.
- Change failure rate: The rate of frequent deployments across multiple endpoints on everyday basis to a benchmarked value.

10. How to achieve Continuous Delivery (CD) in DevOps without downtime?

It is possible to achieve continuous delivery with zero downtime for DevOps using any of the following techniques:

- A/B switch
- Software load balancers
- Delaying the port binding

11. What is TOSCA in DevOps?

TOSCA stands for Topology and Orchestration Specification for Cloud Applications. It deals with a missing piece in continuous testing hosted on cloud applications and services. The aims of TOSCA include a reduction in escaped bugs, reducing the cost of rework and gaining faster feedback cycle.

12. What is DevOps Scrum methodology ?

DevOps scrum methodology is a method of scrum which uses standard DevOps techniques to improve overall agility in a given business. It's a more thoughtful approach which focuses on monitoring operational teams, QA and product teams in a cycle. It's an agile development framework which includes multiple scrum features like product owner, web, mobile and QA which forming a scrum of scrum to deliver a product feature to customer.

13. What are the top 5 challenges of DevOps implementation in a product environment?

Ans: Some of the key challenges while adopting DevOps for Product Development include

Complex and fragmented development pipeline: Unlike pure-play IT, it is very challenging to streamline the team composition and workflow across multiple application delivery end points, including web, mobile, and devices.

Treating the "Environment" as a code: Product Development would require virtualization of multiple associated devices along with server infrastructure. The concept of 'Infrastructure as code' in IT needs to be extended towards offering complete 'Environment as code'.

Ensuring product pipeline: With variety of devices and applications, it is very challenging to ensure product delivery to multiple customers (customization & enhancement) and varied market segments (Low, mid, & high). It requires multiple custom production environment that is difficult to reproduce and maintain at development stage.

Support to Legacy devices and solutions: Product along with cloud infrastructure brings in requirement for managing, updating, and maintaining existing devices on field, along with newly added devices, resulting in increased variations and complexity for Devops.

Release Cadence: Developing a unified release plan becomes a challenge with multiple solution components, including firmware, web app, mobile app, and pc app.

14. What are the various components in DevOps implementation?

A typical DevOps workflow will include:

- **Continuous integration:**
    - Integration of multiple pipelines (Device, Web and mobile) and prepare main and customized builds.
    - Configuration and Automation of the environment setup
    - Auto triggering of alerts and reports
- **Continuous testing:**
    - Virtualization/Simulation
    - Developing and triggering test script automation with simulation and physical devices
- **Continuous delivery:**
    - Auto Build Deployment on devices and sensors
    - Rollback management on the live environment, and generating automated alerts and reports on failure scenarios and performance issues.
- **Continuous monitoring:**
    - It includes monitoring and automated troubleshooting of production and test environment, including device health.
    - An automated alert mechanism

15. What are the benefits that an organization can get with Devops transformation?

Key benefits for an organization moving to DevOps are:

1. Reduced time-to-value / Faster time-to-market: Improved agility in product development
2. Collaboration: Overcoming from Silos culture and opting for more collaboration and efficiency with different working departments.
3. Customer Delight: Faster releases, improved agility & quality, and customized builds enables quicker resolution of customer requests.
4. Operational Efficiency: The ability to scale faster helps in Opex/Capex optimization
5. Cost Savings: High-scale automation enables to control remotely, reduce the outages, and eliminates product recalls.

16. How is build verification testing process implemented in DevOps?

Automated Build Verification Test (BVT) – BVT is focused on delivering the highest quality products, on time. BVT in agile-DevOps methodologies plays an important role in which QA team runs a set of pre-defined test cases for each new build to ensure that they are a stable build for further testing. These pre-defined test cases are for core functionality so that the product's core functionality is not broken, with further eligibility to release this build to QA team for a thorough testing. You can read in detail about the automated BVT process here: https://goo.gl/zdKC5K

17. What are the benefits of automation testing?

Following are the advantages of automation testing:

1. Supports execution of repeated test cases
2. Aids in testing a large test matrix
3. Enables parallel execution
4. Encourages unattended execution
5. Improves accuracy by reducing human errors

Read a detailed overview of Testing automation which helps in improving the performance of systems and bring products to the market faster

18. How does continuous delivery fit in product development methodologies?

Continuous delivery means that when a developer makes any changes, the build and code fixing will happen simultaneously. It comes under the environment where automatic testing has to be achieved. It includes a development environment, a test environment, and a staging environment. These environments have three different teams of people performing product deployment and testing, also known as deployment pipeline.

19. How are fixes delivered for a CD release?

Continuous integration is an s/w development practice which helps reduce fixes wherein a code is checked very frequently and delivered for a Continuous delivery (CD) release. CI is a very useful component in which making CD process successful.

20. What are the tools being used in deployment pipeline?

Following are the examples of software tools being used in deployment pipeline:

1. Go
2. AntHill Pro

21. What are the benefits of CI?

The list below outlines continuous integration benefits:

1. Fixing the bugs early
2. Reducing risks greatly
3. Reducing manual tests for successful continuous delivery
4. Increasing transparency between QA and developer teams

22. What is the role of QA in continuous delivery?

The key element introduced in CD of QA team is test automation. The role of QA in CD includes the following steps:

1. Applying analytical reasoning and carrying out different tests .
2. A testing strategy in which code passes through various layers of testing like unit test, automated test, QA test, User acceptance test, monitoring, and regular feedback.
3. Improving the process in order to check all testing and tools are being properly used with proper feedback and communication.
4. Upgrading QA technical skills which will be helpful in successful CD pipeline.

23. What is meant by CI?

CI stands for continuous integration. It's a software engineering practice of integrating changes from different developers in a frequent interval of time which will be done several times per day. This process helps developers to make sure the code of individual developer working on will not divert other team members and maintain the time consuming process throughout.

24. What are the benefits of continuous integration?

There are several benefits associated with CI:

1. Improvement in developer productivity.
2. Risk reduction.

3. Ease in implementation with the help of DevOps tools i.e. Jenkins.
4. Frequent testing leading to bugs removal in less time. Real time updates.

25. What are the benefits of Continuous Delivery (CD)?

Following are the benefits of CD:

1. Eliminate DIY from continuous delivery process.
2. Integrate team and respective process in a unified pipeline.
3. Connecting DevOps tools and technologies into each workflows.
4. Fast and frequent deployments of product without compromising the security.

26. What are the top continuous integration tools?

The top 6 open source CI server tools:

1. Jenkins
2. Buildbot
3. Travis CI
4. Strider
5. Go
6. Integrity

Read more about above tools here: https://goo.gl/G8iVNw

27. What are continuous deployment tools?

Get the ultimate list of automated deployment tools which makes your deployment process less error prone, ease in access, and frequent releases.
Visit: https://goo.gl/4e96Bg

In order to understand in detail, read about role of QA in DevOps.

28. How does continuous deployment differ from continuous delivery?

With continuous delivery (CI) in DevOps, every code change is built, tested, and then pushed to a staging environment. It may go through multiple, parallel test stages before passing it to a production deployment.

The difference between continuous deployment and continuous delivery (CD) is the need for a manual approval to update to production. With continuous deployment, production happens automatically as it doesn't need approval from the developer.

29. What is the use of Jenkins in DevOps?

Jenkins is a continuous integration tool for software development. It has become the open source standard in the development side of DevOps methodology from source code management to deliver the code to the operations team. It helps in making development faster and improves the quality at the same time. For more information, click here.

30. What is the use of Chef in DevOps?

Chef is configuration management tool for handling physical servers and virtual machines in the cloud. Chef solves the query by treating infrastructure as a code. This gives rise to continuous delivery and automation testing. 'Know about environment as a code in DevOps methodology.

31. What is the Role of Docker in DevOps?

Docker is a software container management platform / environment that works with the same capacity as real domain without adding overheads of virtual machines. It is essential for continuous integration and continuous deployment workflows since it streamlines the entire Software Development Lifecycle (SDLC) process.

Docker in DevOps helps by providing Dev and Ops team facilities to run and manage code in an isolated container, imparting collaborative capabilities and role-based access to codes and apps. The Docker CaaS (Container as a Service) platform helps to secure and isolate software from the external dependencies as well as transfer content to the another environment without delays, creating transparency in between Dev and Ops team.

With Docker, Dev team can automate repetitive tasks like configuring development environment keeping the codes efficient and light-weighed in an isolated container. This further helps Ops team to build, test, debug, deploy and deliver the codes securely without hassles or downtime.

eInfochips helped a leading Diagnostic solution provider implement DevOps for Product Development in Medical Devices with Dockerization. Get a copy of case-study, Applying DevOps for Product Development in Medical Devices.

32. What are Microservices?

Microservices is a type of SOA (service-oriented architecture) that decompose applications in smaller parts and artifacts, and structure them as loosely-coupled services. This way, it makes applications lightweight and easy to develop, test and deploy.

In the software development and DevOps methodology, Microservices help develop and deploy individual applications into its own container as assigned to respective Operation team. This accelerates continuous delivery and deployments and also enables autonomous scaling of small services without dependencies. In that, if any Microservices fail, it will not affect other services or processes which will make troubleshooting process easier.

33. How DevOps increase system security in an organization?

Here are the top 5 ways in which DevOps increase system security in any organization:

1. DevOps maximize communications and thereby maximizes team's visibility into the software lifecycle. This enables the team to identify security flaws and errors before the code is released into production.
2. DevOps maximize task automation in product development which drives consistency, predictability, resulting in lesser human errors during a manual process which otherwise could impede security.
3. DevOps enable faster development and software delivery, which means faster debugging and fixing of security bugs.
4. DevOps enable utilization of DevOps helps in adopting any tool that is more secure and reliable according to the software lifecycle processes without the need to be locked into outdated and non-secure tools and frameworks. Know more about Top DevOps Tools and their respective features.
5. DevOps favors agile development with Microservices and Containers, which help in isolating and securing applications from external attacks, human errors. Since these technologies break applications in smaller components, it doesn't compromise with the entire workflow or application in case of security attacks or errors.

To know more about security practices, visit the link

Pic Credit: DevOps.Com

34. What is DevSecOps?

The DevSecOps requires everyone in the software development life cycle to be responsible for security. It aims to bring operations and development together, ensuring security during all the stages of the development process.

It tries to automate the core security tasks by incorporating security controls and processes early in the DevOps workflow rather than attaching it in the end.

DevSecOps brings automation from the start of the development cycle and thus, reduces the chances of misadministration and mistakes, which may lead to downtime or attacks. Automation also reduces manual configuration of security consoles.

35. How does DevOps work in AWS?

AWS enables organizations to build and deliver products rapidly and reliably using AWS and DevOps practices with built-in flexible services. These services are designed to simplify provisioning and management of infrastructure, deployment of application code, automation of software release processes, as well as monitoring of application and infrastructure performance.

AWS Developer tools automate manual tasks, help teams manage complex environments at scale, and keep engineers in control of the high velocity that is enabled by DevOps. It also helps organizations in storing and versioning of application's source code securely and automatically build, test, and deploy the application to AWS or the on-premises environment.

AWS CodePipeline is useful to build a continuous integration or continuous delivery workflow that uses AWS CodeBuild, AWS CodeDeploy, and other tools, or use each service separately.

36. How eInfochips helps in DevOps transformations?

eInfochips working in the new era like DevOps Services and CloudOps services for IoT empower Consumer IoT and Industrial IoT companies with continuous testing and deployment capabilities. Let's add value to your business by the following offerings:

- DevOps Consulting: Assessment framework and process consulting
- DevOps Implementation: Build verification testing, containerization microservices, Deployment automation, test automation
- DevOps Toolchain: Evaluation and execution; Migration, customization and enhancement

- CloudOps managed services: Infrastructure automation and application monitoring, collaboration automation, Bot automation

Error Troubleshooting

A. Version Control (Git)

- **Error:** Recovering from accidental deletions.
    - **Troubleshooting:**
        - Utilize git reflog to view a history of recent actions.
        - Use git checkout <commit_hash> to restore a previous state.
        - Consider using a Git garbage collector to reclaim disk space after recovering files.

B. Continuous Integration/Continuous Delivery (CI/CD)

- **Error:** Slow pipeline execution times.
    - **Troubleshooting:**
        - Optimize build and test stages for performance.
        - Utilize caching mechanisms to reduce redundant operations.
        - Parallelize tasks whenever possible.
        - Analyze pipeline logs to identify bottlenecks.

C. Containerization (Docker)

- **Error:** Image size bloat.
    - **Troubleshooting:**
        - Minimize the number of layers in the Dockerfile.
        - Use multi-stage builds to create smaller final images.
        - Leverage a base image that is specifically tailored to your application's needs.

D. Orchestration (Kubernetes)

- **Error:** Resource exhaustion (CPU, memory).
    - **Troubleshooting:**

        - Monitor resource usage within the Kubernetes cluster.
        - Adjust resource requests and limits for pods.
        - Implement horizontal pod autoscaling to automatically scale pods based on demand.

E. Infrastructure as Code (IaC) - Terraform/Ansible

- **Error:** State file corruption.
    - **Troubleshooting:**
        - Use the terraform state list command to verify the state.
        - If necessary, re-create the state file from scratch.
        - Consider using a remote backend for state storage (e.g., AWS S3, Azure Blob Storage).

F. Monitoring and Logging

- **Error:** High alert fatigue.
    - **Troubleshooting:**
        - Refine alerting rules to reduce noise.
        - Implement intelligent alerting based on severity and context.
        - Use machine learning to identify and suppress irrelevant alerts.