1. Summarize the 6 phases of release engineering discussed in the article and explain the role each phase plays in the overall release process. (1-2 sentences per phase is sufficient)

(1) Integration: Branching and Merging

This phase involves merging code changes from individual development branches into the team's branch and eventually into the project's master branch, ensuring smooth collaboration and code synchronization.

(2) Continuous Integration: Building and Testing

Here, code changes are continuously integrated into the mainline codebase. This process involves automatically building and testing new commits or merges to identify any regressions promptly.

(3) Build System

The build system encompasses the specifications used to generate project deliverables such as binaries or packages from the source code. It's crucial for ensuring consistency and reliability in the build process.

(4) Infrastructure-as-Code

This phase involves defining and managing infrastructure using code, allowing for automated provisioning and configuration of environments. Tools like Puppet, Chef, or Ansible are commonly employed for this purpose.

(5) Deployment

During deployment, the tested deliverables are staged and prepared for release. This phase ensures that the release is ready for deployment into production or other environments.

(6) Release

Finally, the release phase involves making the deployed releases accessible to system users, effectively making the new features or updates available for use.

2. Discuss some of the challenges involved in Continuous Integration, and how some of the potential research areas discussed in Section III-B could address those challenges.

Challenges: Continuous Integration (CI) faces several hurdles, including the need for CI servers to promptly trigger build and test jobs for every code alteration committed to a

project's Version Control System (VCS), across various branches and configurations. The demand for faster CI processes presents a significant challenge for practitioners.

Strategies: (1) Anticipating whether a code modification will cause a build failure. In this case, some models are built to predict whether some modifications developers made will bring build failures or not. If some vital problems are detected by these models, some informative prompts will be presented to developers to help them correct faults before stepping into the next step which might consume many resources. (2) Grouping code changes into chunks and conducting compilation and testing once for each chunk can optimize CI workflows. However, according to the paper, we still require some effective approaches that are applicable to generic situations on how to group those codes smartly. (3) Enhancing the infrastructure with more robust servers, network switches, and storage devices can enhance the efficiency of handling heavier build loads. However, this strategy needs a statistical or data mining model to predict how much resource the project needs in the future because the build load will get heavier while the features get more and more. We can build a model to predict the future demand for resources based on past build history.

3. Explain the unique challenges of Continuous Delivery on current mobile platforms.

Continuous Delivery faces distinct challenges on contemporary mobile platforms, where apps must align with the rapid evolution of web counterparts, ideally synchronizing feature releases across all platforms simultaneously. Current mobile app vendors take control of the release of the app, although numerous web and desktop app companies seek to regulate the deployment and release of their new app versions by either self-hosting their web app or integrating an automated update mechanism into their desktop product, they do not have much right to do the timely synchronization work on their mobile apps.

Challenges: (1)Inversion of Deployment Control: While many web and desktop app companies exercise control over deploying new versions by hosting their web app or integrating automated update mechanisms into their desktop products, current mobile platform vendors restrict such control. (2) Untimely Delivery of Bug Fixes: Addressing bugs in a mobile app involves passing fixes through the app store vetting process, often resulting in delays. Despite a special queue for expedited fixes, deployment speed hinges on external factors. During this period, negative reviews or ratings may accumulate on the app store page, dissuading potential users from downloading the app.

4. Choose one of the 6 phases, and discuss how the modern release engineering practices discussed in the paper enable higher Software Engineering throughput and/or stability.

Chosen phase: Deployment.

In the deployment phase, modern release engineering practices play a crucial role in enhancing Software Engineering throughput and stability by implementing:

Modern release engineering practices: (1) Enhanced Quality Assurance Tools: These tools are essential for preventing critical bugs from slipping into the release, thereby avoiding the need for redeployment and subsequent vetting. They encompass various functionalities such as defect prediction (based on either file- or commit-level analysis), improved update mechanisms, tools for optimizing code review processes, automated test generation, efficient crash report analysis, and effective defect prioritization and triaging.

(2)Streamlined Review Processes by Platform Owners: Mobile platform owners must swiftly assess modifications to a mobile app release, especially in urgent situations requiring emergency fixes. By implementing streamlined review processes, platform owners can significantly reduce the time from bug detection to the deployment of a fixed version, thereby ensuring faster response times and enhanced stability.

(3) According to “When app stores listen to the crowd to fight bugs in the wild,” in Proc. of the 37th Intl. Conf. on Software Engineering (ICSE)", app stores can generate temporary fixes for certain kinds of defects, while the app developers are working on a permanent fix. This measure can reduce the effect the defects bring to the largest extent.