Release

Engineer (Intern)

Contact Info

Team: Internet Technologies & User Privacy

- **Myself: Michael Liz ~**
  - **Email: Lmic1219@gmail.com or m_liz@apple.com**
  - **Phone number: (917) 686 0399**

- **Supervisor: Mohsin Qureshi**
  - **Email: mohsinq@apple.com**
  - **LinkedIn:** https://www.linkedin.com/in/mohsinqureshi51/

Abstract

Software Release Engineer at Apple

The primary goal of my work as a Software Release Engineer Intern is to manage source code and ensure seamless software operations and efficient release engineering services for teams working on Messages, FaceTime, Calls, Mail, and Sharing. By managing the build and integration of complex cross-functional features such as Contact Posters, Stickers, and Collaboration via Messages, I help streamline the release process across multiple organizations and software projects. My focus is on maintaining accuracy and quality in software releases across all of Apples platforms (iOS, watchOS, macOS, etc.) while enabling flexibility to meet the evolving needs of engineering and quality teams. Additionally, I develop automation workflows and tools using technologies like Apache Airflow, Python, to improve efficiency, reduce manual overhead, and enhance the overall reliability of our software delivery pipelines.

Tools

- Command Line
- Git
- Python3
- GitHub
- API’s
- An IDE
- Apache Airflow

Use Case(s)

1 **Automated Build and Integration Pipeline**

Scenario: A team working on a new feature for Messages, such as Contact Posters, needs to integrate their code across multiple repositories while ensuring compatibility with other projects.

Input: Developers commit code changes to the repository, triggering an automated pipeline. The pipeline pulls the latest changes, runs dependency checks, compiles the code, and executes unit and integration tests.

Expected Output: If the build succeeds, the pipeline generates a software image and integrates it into the broader ecosystem. If issues arise, automated notifications alert the responsible engineers, reducing manual debugging time and ensuring a smoother release process.

---

2 **On-Demand Release Management Tooling**

Scenario: A cross-functional team needs to release a critical fix for Collaboration via Messages outside of the standard release cycle.

Input: Engineers specify release parameters, such as the affected components, target OS versions, and priority level. The system dynamically adjusts release workflows to accommodate the emergency update while maintaining stability.

Expected Output: The update is integrated and shipped efficiently, ensuring minimal disruption for end-users while maintaining adherence to quality and security standards.

Use Case(s)

3 **Dependency and Compatibility Monitoring**

Scenario: A feature update in Mail introduces a new dependency that could potentially conflict with existing system components.

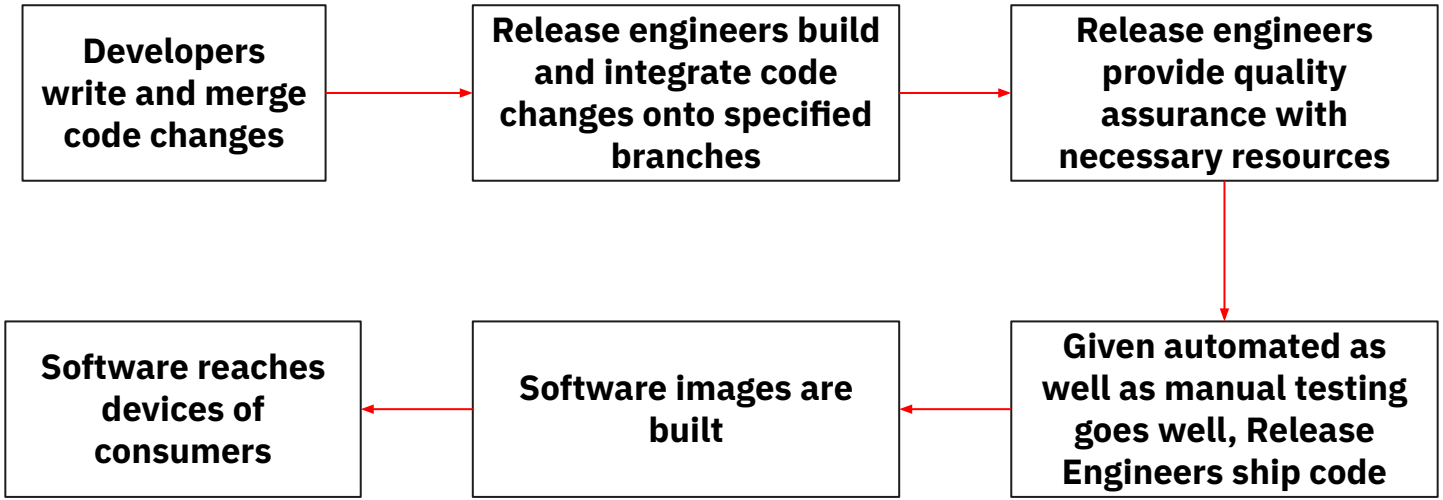
Input: The system continuously monitors software dependencies, scanning for version mismatches, deprecations, or security vulnerabilities in integrated libraries.

Expected Output: If conflicts are detected, engineers receive automated alerts with recommendations for resolution. This proactive approach minimizes integration delays and reduces the risk of broken dependencies in production releases.

Tentative Schedule

- Monday - Friday
- 8+ hrs daily
- Goals
  - Implement Automated Build and Integration Pipelines
  - Develop Automated Test and Quality Assurance Workflow
  - Integration
  - Management of source code from varying teams
  - Build On-Demand Release Management Tooling
- Estimated time: **Varies**

Diagrams


Developers write and merge code changes
Release engineers build and integrate code changes onto specified branches
Release engineers provide quality assurance with necessary resources
Given automated as well as manual testing goes well, Release Engineers ship code
Software images are built
Software reaches devices of consumers

Link to GitHub

https://github.com/shhhmike/4900-Files