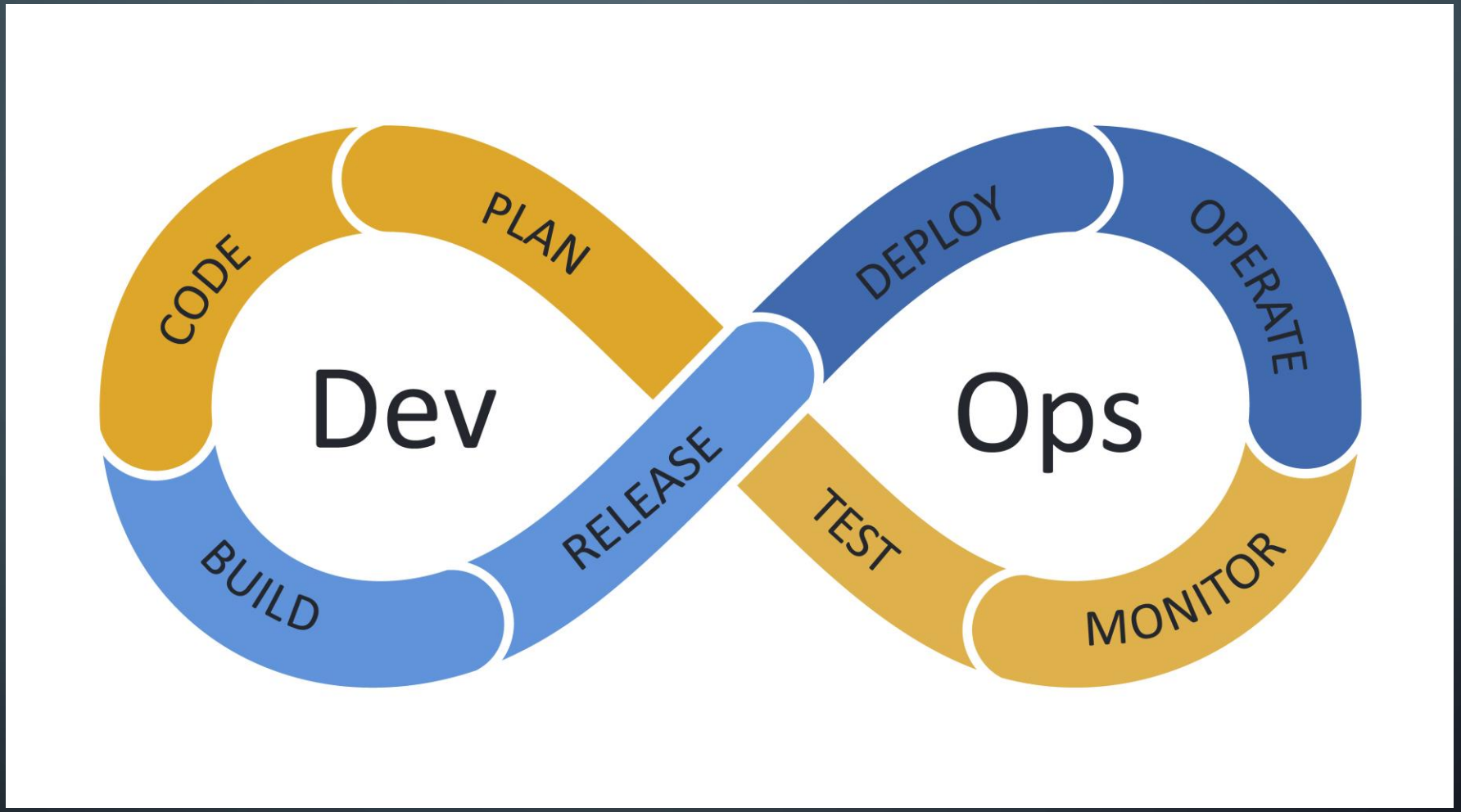
CI/CD PIPELINE

AN AGILE APPROACH TO OUR DEV-OPS BUSINESS PROCESS.

CARL BACCUS

SR SYSTEMS ENGINEER



* IMAGE 'ISTOCK-1288847618.JPG' PURCHASED FROM HTTPS://WWW.ISTOCKPHOTO.COM/HELP/LICENSES

EXISTING MODEL

- Current model consists of our system administrators:
  - Building out various server platforms to house different applications
  - Validating security compliance for each server.
  - Then fixing / patching every time something goes wrong.
  - The team both SA's and developers are very Reactive to issues.
- Then developers utilize the systems built by the system admins
  - Build out code for individual sprints
  - Combine all working parts of the code and system for a release date
  - Then test and fix each individual component (including server fixes with System admins).
  - Finally validate that it is ready, and push to operations. This approach takes more time before product is released.

CI/CD APPROACH

- One DevOps team.
  - System admins build out code to deploy the cloud based servers via ansible or some automated tool.
    - Deployment is tested continuously to validate, and fail fast to fix.
    - Testing and planning allow for a more Proactive approach to issues
    - Faster time for developers to get on working systems.
  - Code is deployed on these cloud instances, and validated with smoke tests while each segment is being written.
  - When fixes are required to either infrastructure or code, we can use forks to test new features without breaking old ones, and then integrate them into the master in a continuous integration.

COMPARISON

Current process's

- Separate teams with separate goals
- Integration errors, with reactionary fixes that take more lead time to coordinate and fix.
- Tedious compliance process that is done after deployment, causing delays often.
- Longer deployment cycles, often taking several sprints before a product is released.

CI/CD

- Task automation reduces errors, and allows team to fail fast so they can remediate fast.
- Automation also allows for consistency of both infrastructure and code which fosters proactive planning for improvements.
- Quicker development lifecycle, as we reduce redundant tasks.
- Quicker compliance checks due to consistency.

SUMMARY

By adopting the CI/CD approach, our team will be able to better deploy both infrastructure and code in a more maintainable fashion, and be able to be more proactive to quickly add on or build new features. This in the end will result in a more agile team, allowing us to put out better products with faster time to market, which in turn will be better for profit and scalability of the entire organization.