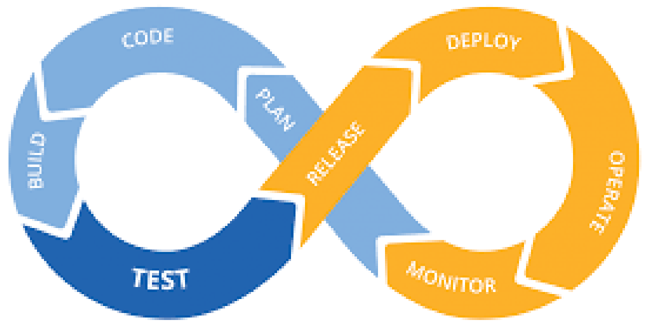
Embracing CI/CD for UdaPeople: Business benefits

BY CHRISTODOULOS KARAKANNAS

What is CI/CD?

- CI/CD consists of two parts
  - CI – Continuous Integration
  - CD – Continuous Delivery
- CI is related to integrating new code changes & product features
- CD is related to deploying new features & delivering value to our customers, quick and efficiently!
- The philosophy behind CI/CD is **automation** & bringing developers & business teams **together**:
  - Build
  - Test
  - Deploy
  - Verify
  - Promote
  - Value



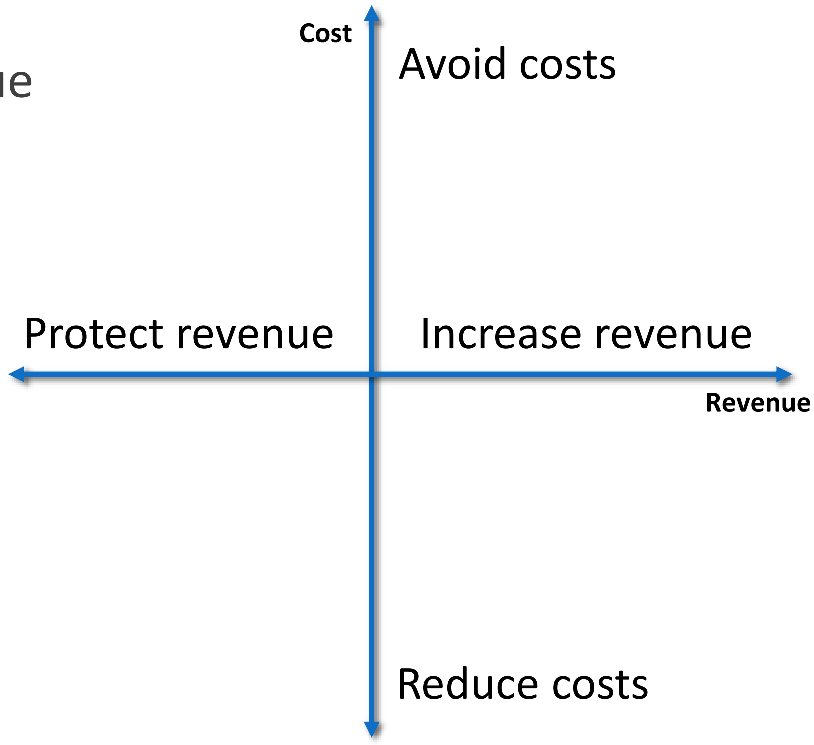
Why invest in CI/CD?

- We invest too much time in configuring and deploying changes
- Our release cycles are currently too slow
  - Value for the customers is not delivered promptly
- Laborious & manual releases increase our costs
- Inconsistent code & too many bugs that go undetected into production
- Too many resources devoted for manual testing, often prone to human errors
- Technical & business teams are not working together efficiently

Benefits of CI/CD for our business

- CI/CD will bring tangible benefits for our business in the long run
- We can measure these benefits across two axis – Costs and Revenue
- Main benefits for us:

Result	Benefit
Automated, faster & consistent deployments	Reduce costs, Increase revenue
Faster release cycles, more value for our customers	Increase revenue
Better quality features with less human errors	Protect revenue
Automated testing, better quality product, less bugs into production	Avoid costs
Better synergy between developers and Ops	Reduce costs



CI/CD is NOT a silver bullet

- CI/CD will generate us huge savings; as long as we collectively embrace a level of discipline
- We have to show willingness & enthusiasm for a paradigm shift into the way of working
- Requires high-level of discipline and organisation
- Requires high-level of communication between Technical and Business Teams
- Every individual has to take responsibility for their part & beyond
- Things to avoid:
  - Broken, neglected tests
  - Broken code and features that leaves us with technical debt
  - Manual deployments and taking shortcuts
  - Making changes without communicating them across the bussiness