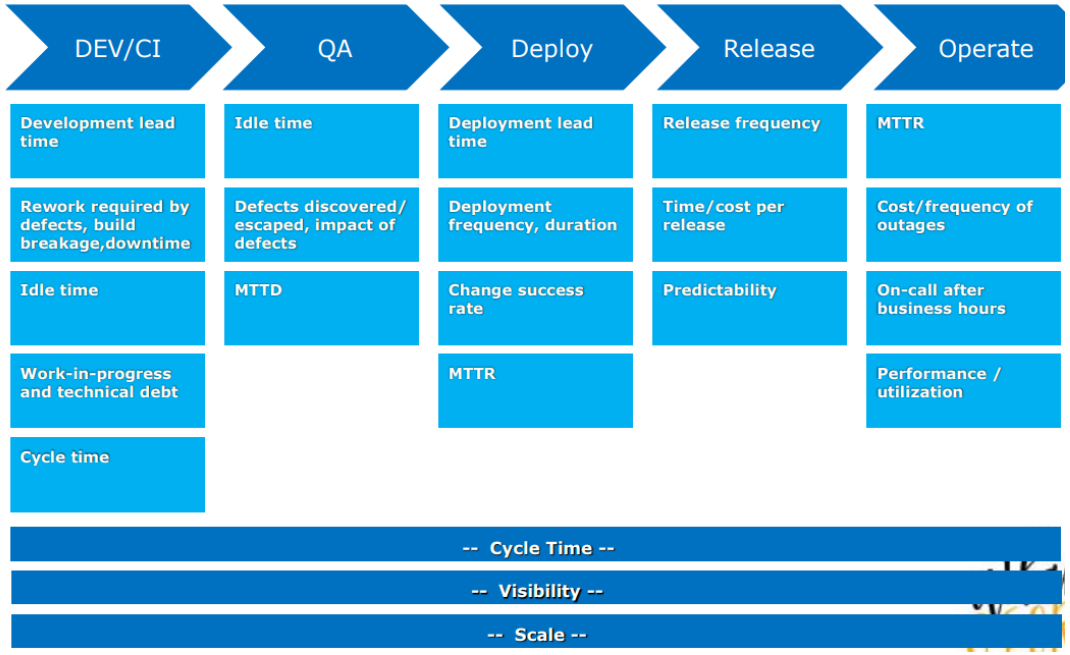
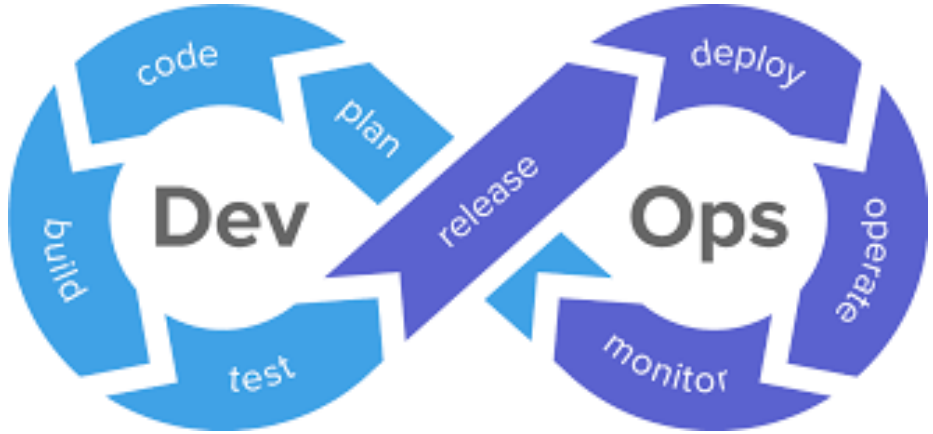
COMP3122 Project

Examples of DevOps Metrics

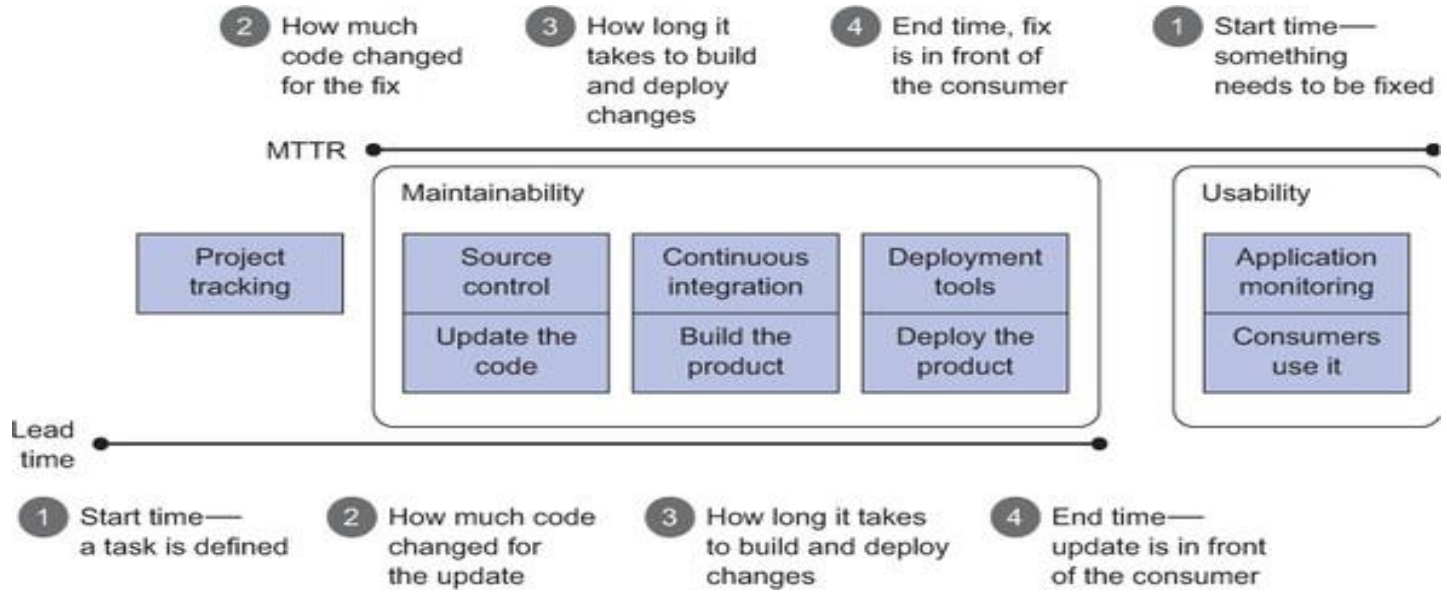
- **Velocity**: lead time, change complexity, deployment frequency, MTTR
- **Quality**: deployment success rate, application error rate, escaped defects, number of support tickets, automated test pass percentage
- **Performance**: availability, scalability, latency, resource utilization
- **Satisfaction**: usability, defect age, subscription renewals, feature usage, business impact, application usage and traffic





https://devopedia.org/devops-metrics

- Mean time to repair (MTTR)
  - Time from when you realize something is wrong in production, the issue is triaged, and a fix is determined and deployed.
- Lead time
  - time between the definition of a new feature and when it gets to the consumer.

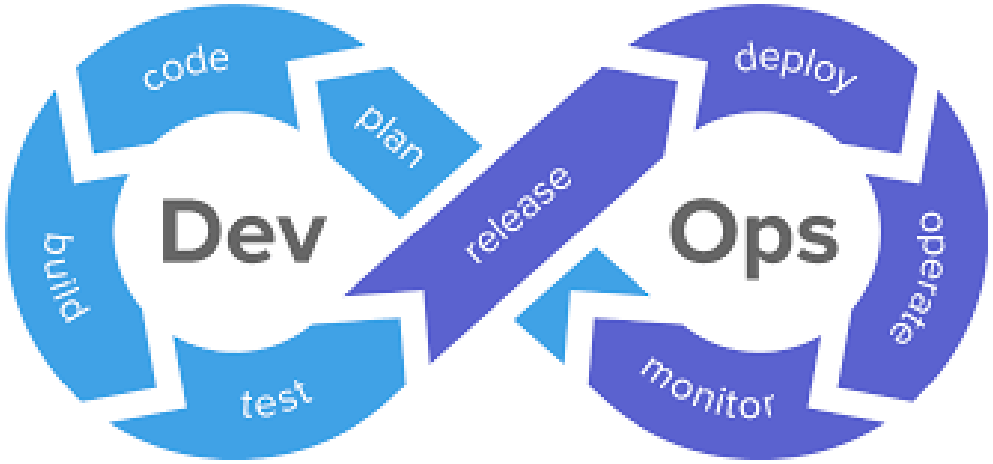


Project overview

- Develop to a platform do simulate a project team who is involved in the development of a vacation rental online marketplace platform
  - E.g. similar to Airbnb
- Your deliverables should demonstrate your understanding of
  - Modern software development and operation practices, pitfalls and challenges
  - DevOps metrics

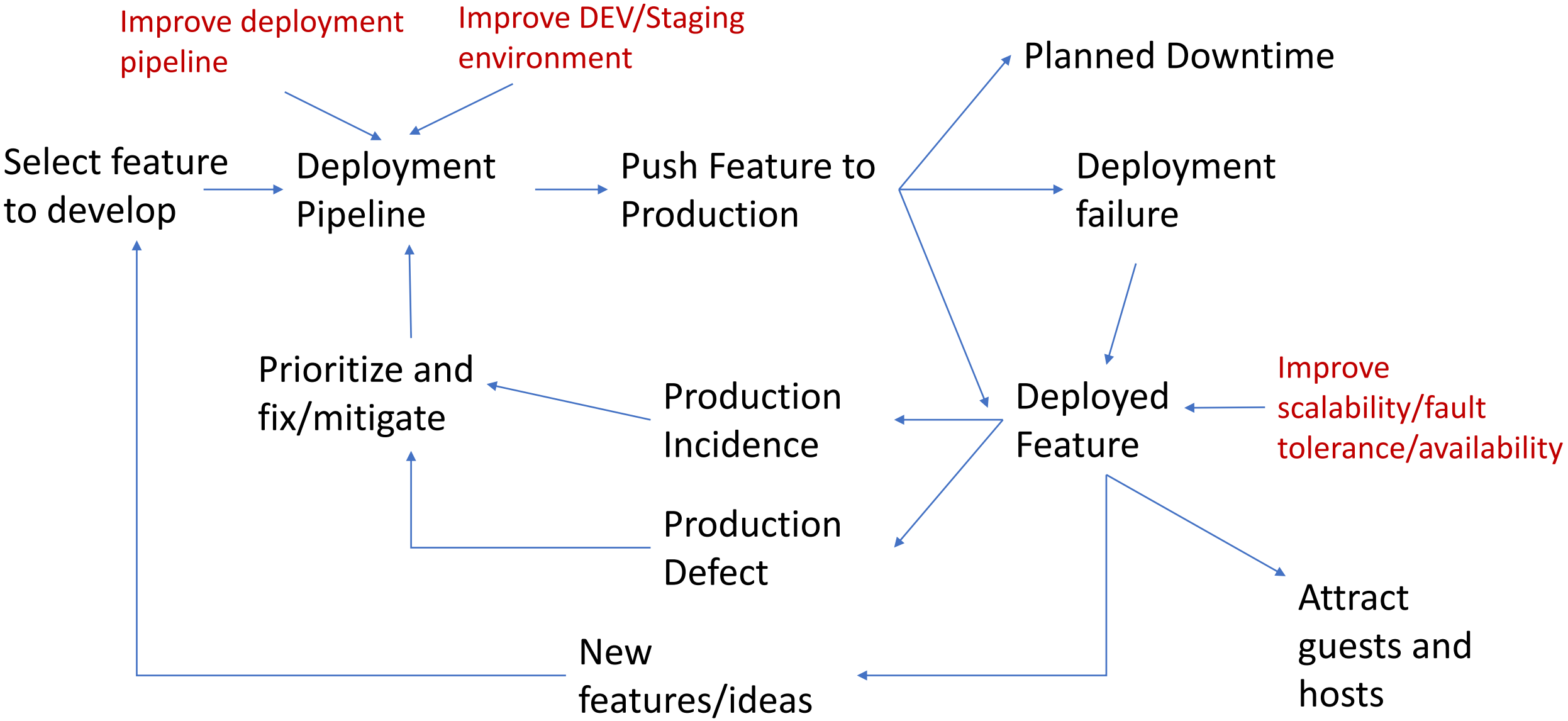
Elements to be included

- Select features to be developed and deployed to customers
  - Receive feedback and learn about customer’s preference
  - Improve time to market
- Outage and downtime when features is pushed to production
  - Balancing feature iteration and service reliability
- Application monitoring and maintenance
  - Fixing production defects
  - Handing production incidence



- Tradeoff
  - Pushing new features to production
  - Fixing defects in production
  - Handling production incidents
  - Improving application availability, fault tolerance, scalability, etc
- Different ways to improve the development/staging environment and operating environment
  - reduce lead time
  - making the release process more reliable
  - reduce downtime in software deployment
  - improving software quality
- Provide metrics and feedback on how good the user in making decisions in different dimensions

Illustration



- Should we improve the DEV environment vs. Prod Environment first?
- Where is the bottleneck in the deployment environment/pipeline?
  - Long time to perform the manual test, High deployment failure rate, …
- Where is the bottleneck in the production environment?
  - Slow application response time?
  - Too many production defects?
  - Scalability? Availability?
    - Is the bottleneck in the application or database?
- What are the improvement options?
  - Which approach is better under a certain context? Why?
  - What are the process for improvement?

Good Design

- Prioritization and resource allocation
- Alternative choices with constraint, tradeoff and dilemma
- Provide context and metrics to support decision making

Final Report/Presentation

- Background research and learning process
  - How did you do your background research? What is your game design process? How do your team collaborate on the project?
- Simulation/Game Flow and Demo
- Discussion and Justification
  - Provide examples of good, average and bad strategy
  - Are there any decisions which are not-so-obvious decisions, or alternative options with trade-off and dilemma?
  - How does the deliverables demonstrate your understanding of devop metrics and modern software development practices/issues/pitfalls/challenges?
- Appendix
  - Work breakdown

Schedule

Week	Deadline	Details
4	28/9/2020 (MON)	Group Formation (4-5 students) Send the student ID and name of your group to richard.lui@polyu.edu.hk. Please let me know if you need help in group formation.
8	26/10/2020 (MON)	Submit a brief summary (max 2 pages) of your progress (one submission per group).
11	16/11/2020 (MON)	Submit final report, slides and source code/executable to blackboard (one submission per group)
11-12		Online Presentation during lecture time

References

- The DevOps Handbook
- Web Scalability for Startup Engineers
- Google SRE
  - https://landing.google.com/sre
- Agile metrics in action : how to measure and improve team performance
- Implementing Lean Software Development: From Concept to Cash

Bad vs. Good Design

Bad

- Meaningless decisions
  - there is a choice to be made, but it has no effect on gameplay.
- Obvious decisions
  - there is clearly one right answer
- Blind decisions
  - have an effect on the game, and the answer is not obvious, but there is now an additional problem: the players do not have sufficient knowledge on which to make the decision

Good

- Risk versus reward
  - One choice is safe. The other choice has a potentially greater payoff, but also a higher risk of failure.
- Choice of actions
  - You have several potential things you can do, but you can't do them all
- Short term versus long term
  - You can have something right now, or something better later on.
  - The player must balance immediate needs against long-term goals.
- Dilemmas
  - You must give up one of several things.
- Resource trades
  - You give one thing up in exchange for another, where both are valuable
  - Test the player's ability to correctly judge or anticipate value

https://gamedesignconcepts.wordpress.com/2009/07/20/level-7-decision-making-and-flow-theory/