Introduction to DevOps - part 1

8th Sept 2021

Chinmaya Dehury

chinmaya.dehury@ut.ee
OUTLINE

Part 1
• Software Development Life Cycle
• Waterfall model
• Agile model
• DevOps Motivation

Part 2
• DevOps history & stats
• ....many more on DevOps
Know audience

Quick Survey

https://www.menti.com
Code: 8162 1643
DevOps Experience:

- 33% 0 Month
- 42% 1-6 months
- 13% 7-12 months
- 4% 1-2 years
- 4% 2-4 years
- 4% > 4 years
Which tools/concepts are you familiar with?

- GitHub: 19%
- GitLab: 13%
- Bitbucket: 9%
- Docker: 16%
- Kubernetes: 5%
- HashiCorp: 1%
- HashiCorp: Vault: 1%
- HashiCorp: Consul: 0%
- Chef: 4%
- Puppet: 4%
- Ansible: 5%
- Jenkins: 7%
- Cloud computing: 9%
- Container-Orch: 5%
- TOSCA: 2%
- Terragraf: 4%
Software Development Life Cycle (SDLC)

• A lifecycle covers all the stages of software from its inception with requirements definition through to fielding and maintenance [3].
Software Development Life Cycle (SDLC)

• A lifecycle covers all the stages of software from its inception with requirements definition through to fielding and maintenance [3].
Software Development Life Cycle (SDLC)

• A lifecycle covers all the stages of software from its inception with requirements definition through to fielding and maintenance [3].

Anything missing?
Software Development Life Cycle (SDLC)

- A lifecycle covers all the stages of software from its inception with requirements definition through to fielding and maintenance [3].
Software Development Life Cycle (SDLC)

• A lifecycle covers all the stages of software from its inception with requirements definition through to fielding and maintenance [3].

Can you arrange them?
Software Development Life Cycle (SDLC)

SDLC models

- Waterfall Model
- Spiral Model
- Agile Model
- Incremental model
Software Development Life Cycle (SDLC)

Waterfall Models

• Most basic stages [Davis et al.]
  • System requirements
  • Software requirements
  • Preliminary Design
  • Detailed Design
  • Code and Debug
  • Test and pre-operation
  • Operations and maintenance
Software Development Life Cycle (SDLC)

Waterfall Models

- **Client**
  - **Requirements**
    - Gather the requirements and defining them (users, login, dashboard, purchase cart, payment options)
    - Language, DB, dev platform, other technical requirements...

- **Planning/Designing**
  - Writing the code, detecting and removing of existing and potential errors
  - report, monitor, resolve and retest software components until they reach the quality standards...

- **Coding & Debugging**
  - Deliver and deploy the software, ready for production. (deliver the car...😊)

- **Testing**
  - Make sure that the customized car is running smoothly....😊

- **Operation**

- **Maintenance**
Software Development Life Cycle (SDLC)

Waterfall Models - **Advantages**

- Simple and easy understanding and implementation
- Very structured organization
  - Define -> Design -> Code
- Phases are very clear
- Easy to manage, arrange tasks
- Focus on one at a time
- Easy and well documentation
- Very straightforward steps
Software Development Life Cycle (SDLC)

Waterfall Models

Client

Programmer

Planning/Designing

Coding & Debugging

Testing

Operation

Maintenance

New Requirements

No client is involved
Software Development Life Cycle (SDLC)

Waterfall Model - **Disadvantages**

- Slow delivery speed
- Very expensive to reflect the new requirements
- No concurrency (just think about a computer with single-thread computation)
- Risky and uncertainty, as no client is involved in other phases
- No room for requirement modification in later stage
Software Development Life Cycle (SDLC)

Agile Model

Client → Requirements → Programmer → Planning/Designing → Ask for feedback

Requirements

Planning/Designing

Ask for feedback

Coding & Debugging

Testing

Operation

Maintenance

Client

Client's feedback
Software Development Life Cycle (SDLC)

Agile Model - **Advantages**

- Persistent Software delivery
- Frequent interaction
- Frequent changes at any stage
- Inspect and Adapt
- Increased client’s satisfaction
Software Development Life Cycle (SDLC)

Agile Model

Agile : Iteration 1
- Planning
- Designing
- Develop
- Testing
- Deploy
- Feedback

Aprox. 2 weeks

Agile : Iteration 2
- Planning
- Designing
- Develop
- Testing
- Deploy
- Feedback

Aprox. 2 weeks

Agile : Iteration 3
- Planning
- Designing
- Develop
- Testing
- Deploy
- Feedback

Aprox. 2 weeks

...
What makes a development method an agile one? [6]
• Breakdown the whole application to different sub-services
• **Incremental**: small software releases, with rapid cycles
• **cooperative**: customer and developers working constantly together with close communication
• **Straightforward**: the method itself is easy to learn and to modify, well documented
• **adaptive**: able to make last moment changes
• No monolithic application development
Agile is a team-based alternative to some of the traditional project management techniques that we are accustomed to using.

We prefer to start executing earlier, deliver frequently, repeatedly evaluate objectives and repeatedly confirm satisfaction.
Software Development Life Cycle (SDLC)

Agile Model – **Advantages**

Advantages of Agile development model

1. Persistent software delivery
2. Quicker release through fixed, reliable and short-term schedules
3. Quick adoption of new features
4. Increased client’s satisfaction
5. Frequent interaction
6. Continuous improvement
7. Increased transparency through tighter collaboration
8. Improve product quality and predictability
9. Reduced risks
Software Development Life Cycle (SDLC)

Agile Model - Values

- **Values of Agile**: (value is all about what you derived out of this model)
  - Focus on the client’s requirement
  - Working on software over comprehensive documentation
  - Customer collaboration (over rigid contract)
  - Responding to changes rather than following a specific plan
Software Development Life Cycle (SDLC)

Agile Model – Principles

**Principles** of Agile development model [5]

1. Client’s satisfaction becomes higher priority
2. Welcome changing requirements
3. Frequent interaction with clients
4. Frequent working software delivery
5. Provide motivated environment and support to individuals
6. Face-to-face conversation (now probably on virtual mode)
7. Working software becomes the primary measure of progress
8. Sustainable development
9. Continuous attention to technical excellence and good design enhances agility
10. Simplicity – Maximize the amount of work not done
11. Frequently improve the effectiveness rather just focusing on fine-tuning and adjusting
12. The best architectures, requirements, and designs emerge from self-organizing teams
Software Development Life Cycle (SDLC)

Agile Model – existing methods

Agile Methods [6]

1. Extreme Programming
2. Scrum
3. Crystal family of methodologies
4. Feature Driven Development
5. The Rational Unified Process
6. Dynamic Systems Development Method
7. Adaptive Software Development
8. Open Source Software development
Software Development Life Cycle (SDLC)

Seems this is a perfect model for software development.

then why we need DevOps?
Software Development Life Cycle (SDLC)

Agile Model:

Disadvantages:

1. Developer computer is used mainly for testing of those features.
2. Test and production environment configuration mismatch.
3. Usually not tested in production environment.
4. Developers team and operations team are in silos.
5. Continuous involvement of all the stakeholders.
Software Development Life Cycle (SDLC)

Agile Model:

Disadvantages:

Developers team → Developed product → Operations team → Product deployment → Production environment

Send product back to developer → Failed Product
Software Development Life Cycle (SDLC)

**Agile Model:**

Disadvantages:

Agile addresses gaps in Customer and Developer communications

DevOps addresses gaps in Developer and IT Operations communications
DevOps
Development + Operations

A culture
Development + Operations

Foster the culture
Focus on Outcomes
DevOps History

• Concept of DevOps emerged out of a discussion between Andrew Clay and Patrick Debois in 2008.
  Read more at: https://www.appknox.com/blog/history-of-devops

• Devopsdays event:
  • a worldwide series of technical conferences
  • Topics: software development, IT infrastructure operations, and the intersection between them.
  • Started 2009
  • A series of event in this month (https://devopsdays.org/events)

* this is not an advertisement
End of Part-1
Lab Sessions

- About the ETAIS marketplace
- Get familiar with creating, deleting and managing VMs
- Invitation to ETAIS DevOps course organization
References


Any Question?

THANK YOU