SYSTEMS LABORATORY

Cursul #10

Automatizarea sarcinilor

If it's worth doing, it's worth automating.

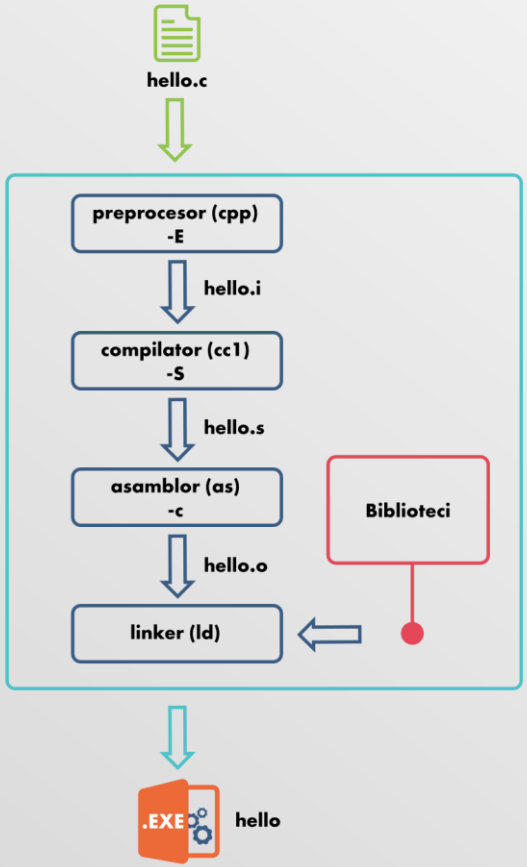
Suport de curs

- Capitolul 13 – Automatizarea sarcinilor
  - https://github.com/systems-cs-pub-ro/carte-uso/releases

Scenariu de automatizare

Fie următorul scenariu

Să ne reamintim



Situație

- multe fișiere cod sursă
- modificări mici și frecvente

- Ce ne interesează?
  - Să scriem cod, să fim creativi
  - Să meargă repede procesul de compilare / build
  - Să testăm ușor modificările

Ce facem?

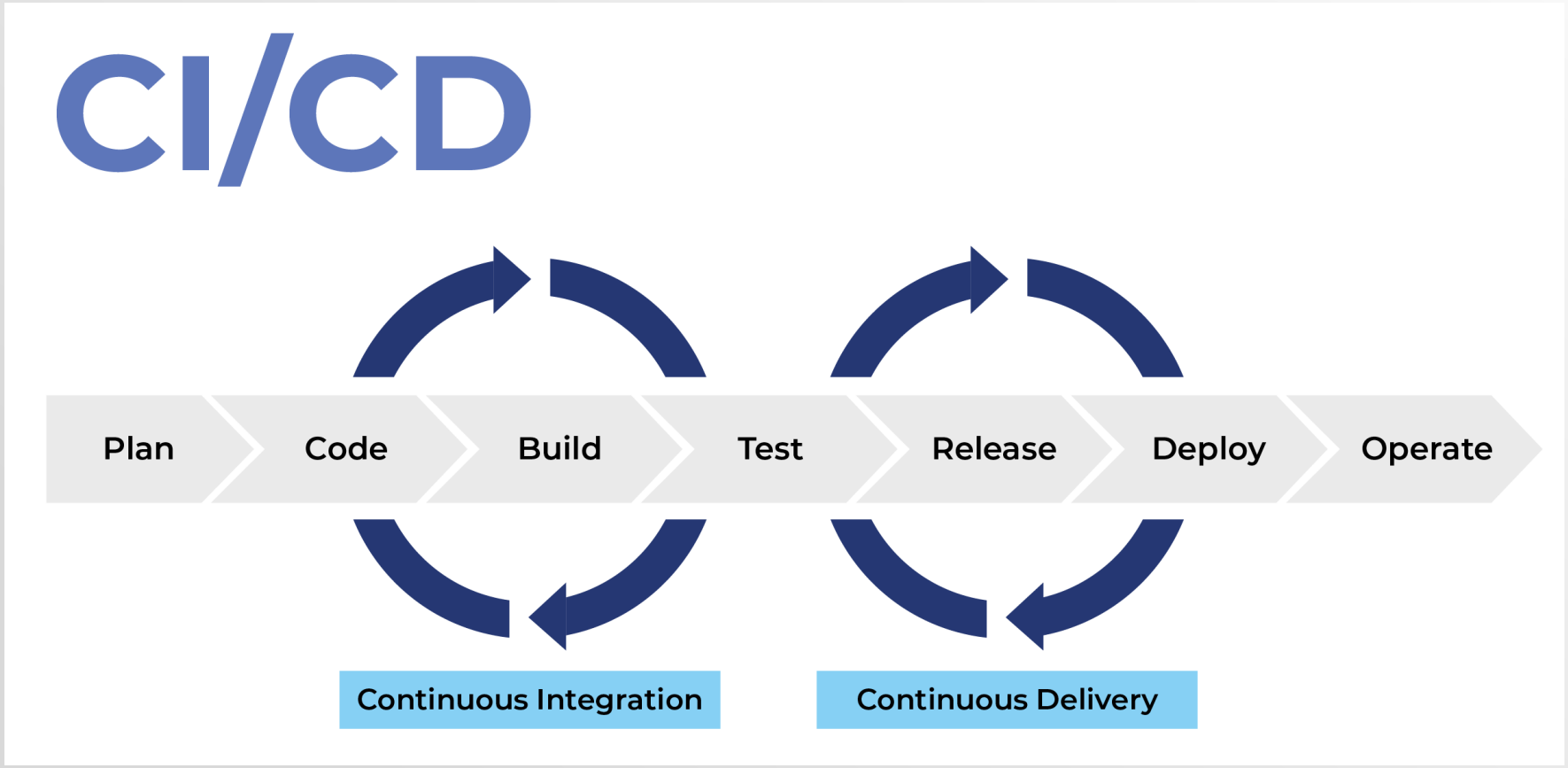
- Ne concentrăm pe scris cod, pe algoritmi, pe soluții inventive, creative

- Folosim soluții de build **automation**
  - Make, Scons, Cmake, Maven, Gradle
- Folosim soluții de setup **automat** al mediului de lucru
  - Docker, Python virtualenv
- **Automatizăm** testarea
  - scripturi care rulează **automat** teste (unit tests, UI testing, regression testing)
- **Automatizăm** deploymentul
  - Scripturi / soluții care creează arhive, medii, împachetează aplicații, publică pachete

Putem mai mult?

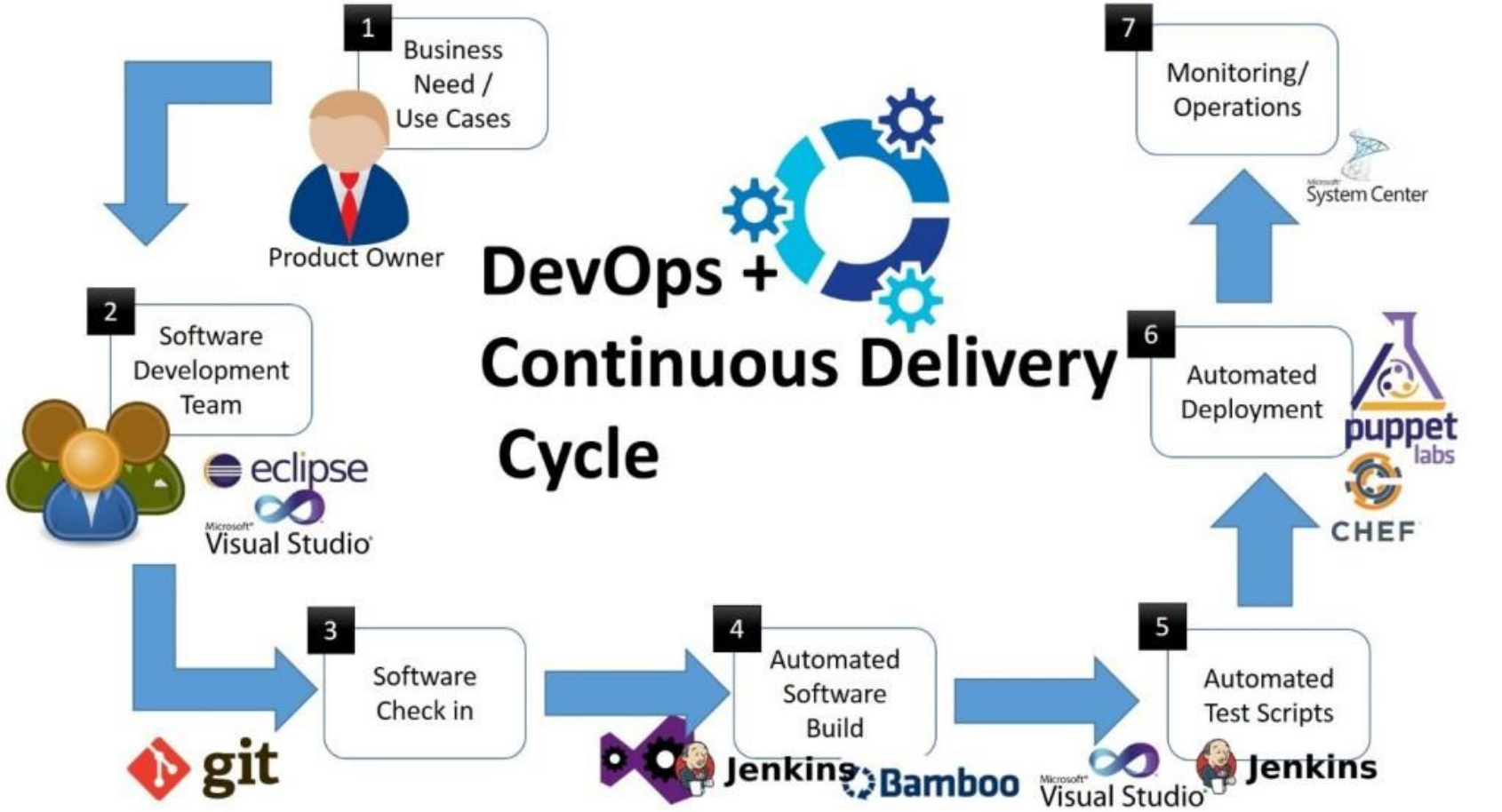
- Rulăm scripturile / soluții de automatizare noi
- **Automatizăm** inclusiv rularea acestor soluții de automatizare
  - nightly builds
  - CI/CD (Continuous Integration / Continuous Delivery/Deployment)
    - Jenkings, TravisCI, GitLabCI, Bamboo, CircleCI
    - Fiecare actualizare a codului duce la build, test, deploy

CI/CD


CI/CD
Plan
Code
Build
Test
Release
Deploy
Operate
Continuous Integration
Continuous Delivery

Automatizare pe pâine

Automatizare pe pâine (2)



Automatizare

Automatizare

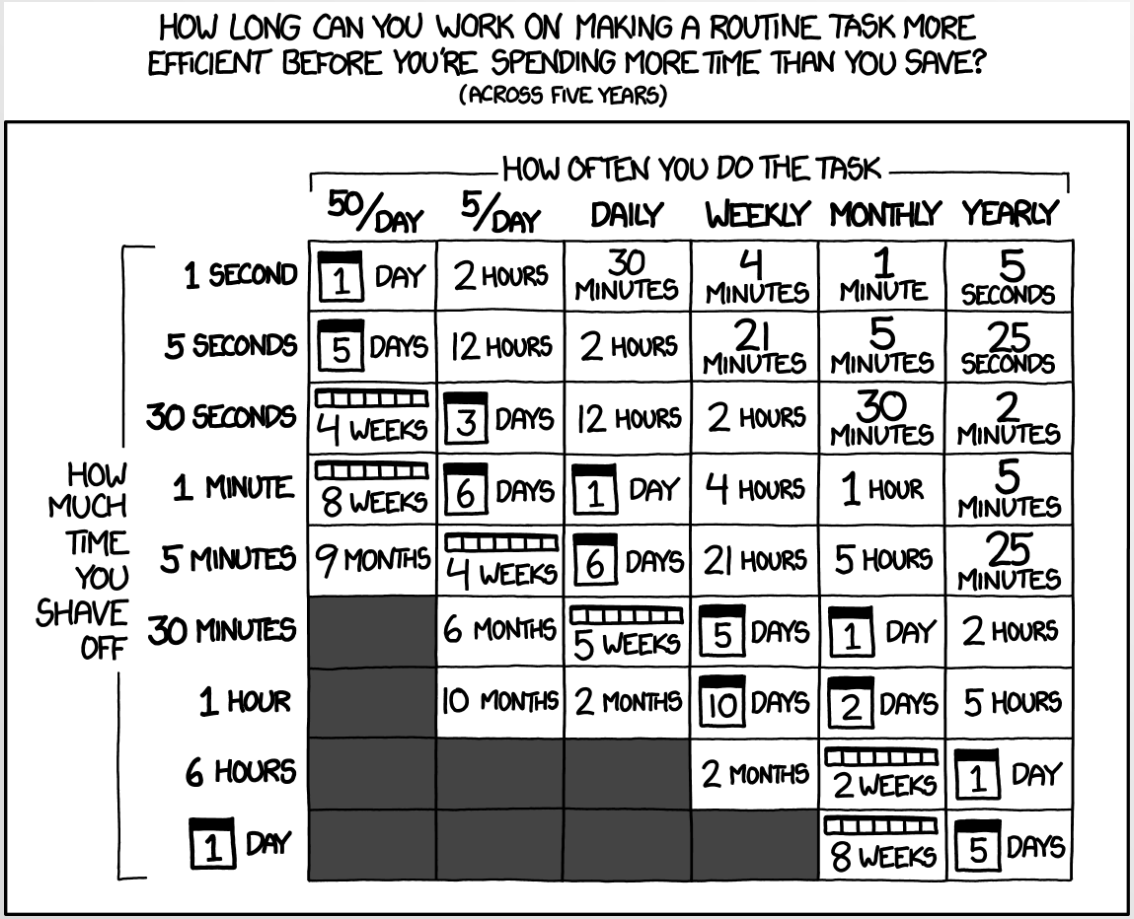
- acțiuni ale utilizatorului sunt preluate de sistemul de calcul
- interacțiunea utilizatorului cu sistemul de calcul trebuie să fie
  - absentă sau
  - minimă
  - operații neinteractive sau minimal interactive
- utilizatorul este preocupat de acțiuni creative

Ce automatizăm?

- acțiuni standardizate, repetitive, necreative

- acțiuni periodice
  - executate la anumite momente
    - build automation
  - generate de utilizator sau un eveniment
    - când o modificare ajunge în repository
- acțiuni scalate
  - aceeași acțiune rulată pe mai multe targeturi
    - prelucrarea datelor financiare a mai multor companii
    - crearea de pachete software pentru mai multe aplicații

Is It Worth the Time?



https://xkcd.com/1205/
https://www.explainxkcd.com/wiki/index.php/1205:_Is_It_Worth_the_Time%3F

Cum automatizăm?

- eficientizare (nu chiar automatizare)
  - keyboard shortcuts
  - aliasuri
- one linere
  - pentru acțiuni combinate, execuții pe mai multe targeturi
- scripturi
  - pentru automatizare (de acțiuni existente)
- programe
  - mai mult pentru noi funcționalități, nu neapărat automatizarea unor acțiuni existente

Programe vs scripturi

- nu există o demarcație clară între scripturi și programe
- în general spunem că scripturile sunt intepretate și programele compilate
- spunem că avem un script Python când facem automatizare
- spunem că avem un program Python când adăugăm o funcționalitate

Shell scripting

- cel mai rapid mod de automatizare
- execuție lentă
  - nu este potrivit pentru acțiuni ce necesită viteză
- folosește comenzi existente
  - interacțiune între comenzi (pipe-uri, command expansion)
- integrat cu shellul / sistemul de operare
- Curs 06: Automatizarea sarcinilor. Shell scripting

Exemplu / scenariu

- Prelucrare de date
- Acțiune multi-target

- Prezentăm scenariul în pași din ce în ce mai complecși
- https://ocw.cs.pub.ro/courses/uso/cursuri/curs-10#prelucrare_note

Utilitare pentru automatizare

Pornirea serviciilor

- scripturi de startup
- pornite de init
- tradițional în /etc/init.d/
- în /lib/systemd/system/ pentru systemd
- operații de pornire, oprire, repornire
  - sudo systemctl stop ssh
  - sudo systemctl start ssh
  - sudo systemctl restart ssh

systemd

-> /lib/systemd/system/my.service

```
[Unit]
Description=My service

[Service]
ExecStart=/var/lib/custom/my.sh

[Install]
WantedBy=multi-user.target
```

```
# sudo systemctl enable my
```

supervisor

- gestiunea de servicii personalizate
- la alegere cu systemd
- permite gestiunea proceselor în numele unui utilizator
- comanda supervisorctl
  - start, stop, restart, add, remove
- configurat în /etc/supervisor/conf.d/

Pornirea periodică

- la momente indicate
- la intervale periodice
- servicii sau comenzi
- at: rulare la un moment dat
  - echo "sh backup.sh" | at 9:00 AM
- cron: rulare periodică

cron

- rulare la intervale periodice
- granularitate de minut
- configurat în /etc/crontab, /etc/cron.*/ și la nivelul utilizatorului
- utilizatorul rulează crontab pentru editare și crontab –l pentru listare

Rulare neinteractivă

- comenzi interactive necesită input de la utilizator
- automatizarea necesită neinteractivitate
- înlocuirea lor cu comenzi neinteractive
  - passwd vs chpasswd
- folosirea de apeluri / comenzi care transmit input automatizat

expect

- furnizare neinteractivă de input
- poate fi folosit în scripturi shell
- pyexpect: în Python

```
spawn ftp myftp
 expect "username:"
 send "hi\r"
 expect "password:"
 send "mom\r"
 expect "ftp>"
 send "prompt\r"
 expect "ftp>"
 send "cd getme\r"
```

screen, tmux

- creare terminal virtual de rulare comenzi
- detașare de la terminal (comenzile rulează în continuare)
  - reatașare la nevoie
- util pentru rularea de comenzi (interactive) la distanță: deconectare și apoi reconectare
- util pentru mai multe instanțe locale: un terminal pentru dezvoltare, unul pentru administrare etc.

Automatizarea testării

Automatizarea testării

- generarea și rularea automată de teste
- baterii de teste transmise automat programului / dispozitivului
  - program under test
  - device under test
- se obțin rapoarte de defecte
- regression testing: teste vechi sunt refolosite

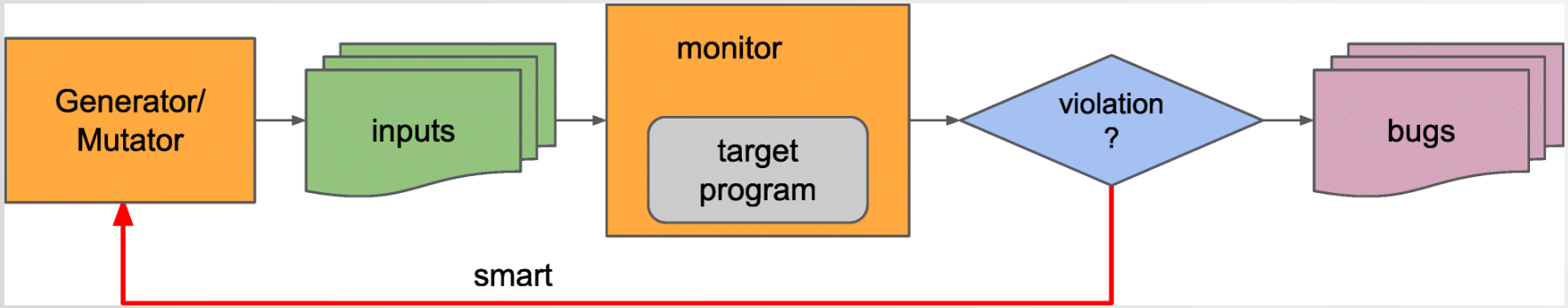
CI/CD

- Continuous Integration / Continuous Delivery/Deployment
- rularea automată a build-ului și testelor la adăugarea de funcționalități noi
- validare rapidă de funcționalități
- Travis CI, Jenkins, GitLab CI, CircleCI, Bamboo

Fuzzing

- generare de input inițial
- testarea unui program
- actualizarea input-ului ținând cont de inputul programului
- reluarea testării
- execuție automată
- durează ore / zile

Fuzzing (2)



https://link.springer.com/article/10.1186/s42400-018-0002-y

UI Automation

- GUI testing
- folosirea automată a componentelor grafice
  - butoane, meniuri
- Web UI automation
  - Selenium
- UI Exerciser (stress testing)
  - Monkey (Android)

Robotic Process Automation

- software bots
- urmărirea utilizatorului
- generarea automată de acțiuni pe baza acțiunilor utilizatorilor
- poate integra componente de inteligență artificială și învățăre automată
- UIPath

Sumar

- delegare către sistemul de calcul
- concentrare pe acțiuni creative
- automatatizare acțiuni standardizate, periodice și acțiuni scalate (multi-target)
- interacțiune minimă (de dorit deloc)
- cazuri de utilizare
  - build automatizare
  - automatizarea testării
  - prelucrarea datelor

Cuvinte cheie

- automatizare
- neinteractivitate
- scripting
- shell scripting
- Makefile, Scons
- Maven, Gradle
- systemd, systemctl
- supervisor, supervisorctl
- at
- cron
- screen, tmux
- expect, pyexpect
- build automation
- CI/CD
- automatizarea testării
- Fuzzing
- UI automation
- RPA