CompSci 401: Cloud Computing

Classic Software Engineering

Prof. Ítalo Cunha

昆山杜克大学
DUKE KUNSHAN UNIVERSITY

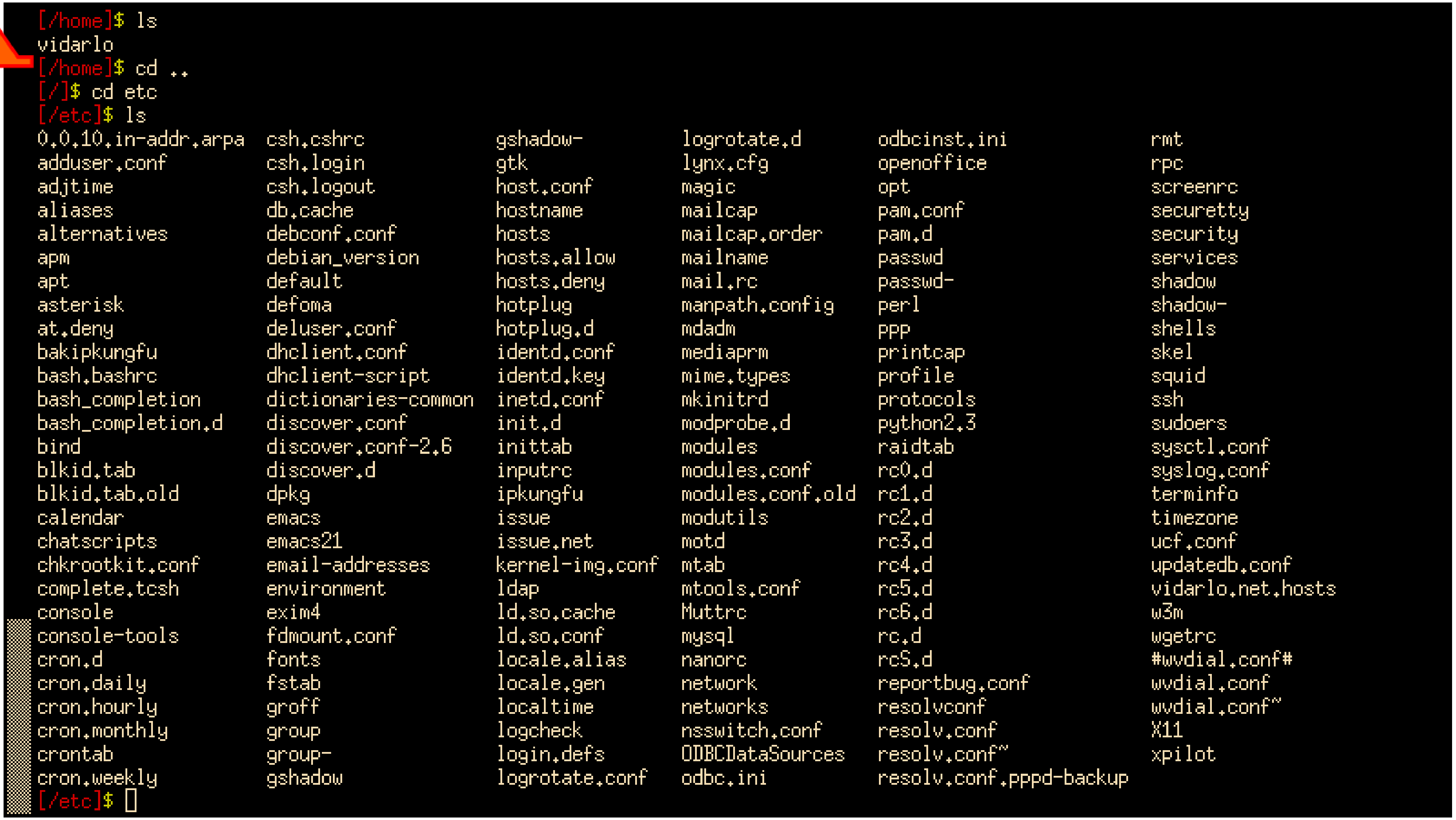
Software development

1. Identify need for a new piece of software
2. Software engineers design, develop, test, and deploy software
3. Users run or invoke the software as needed indefinitely

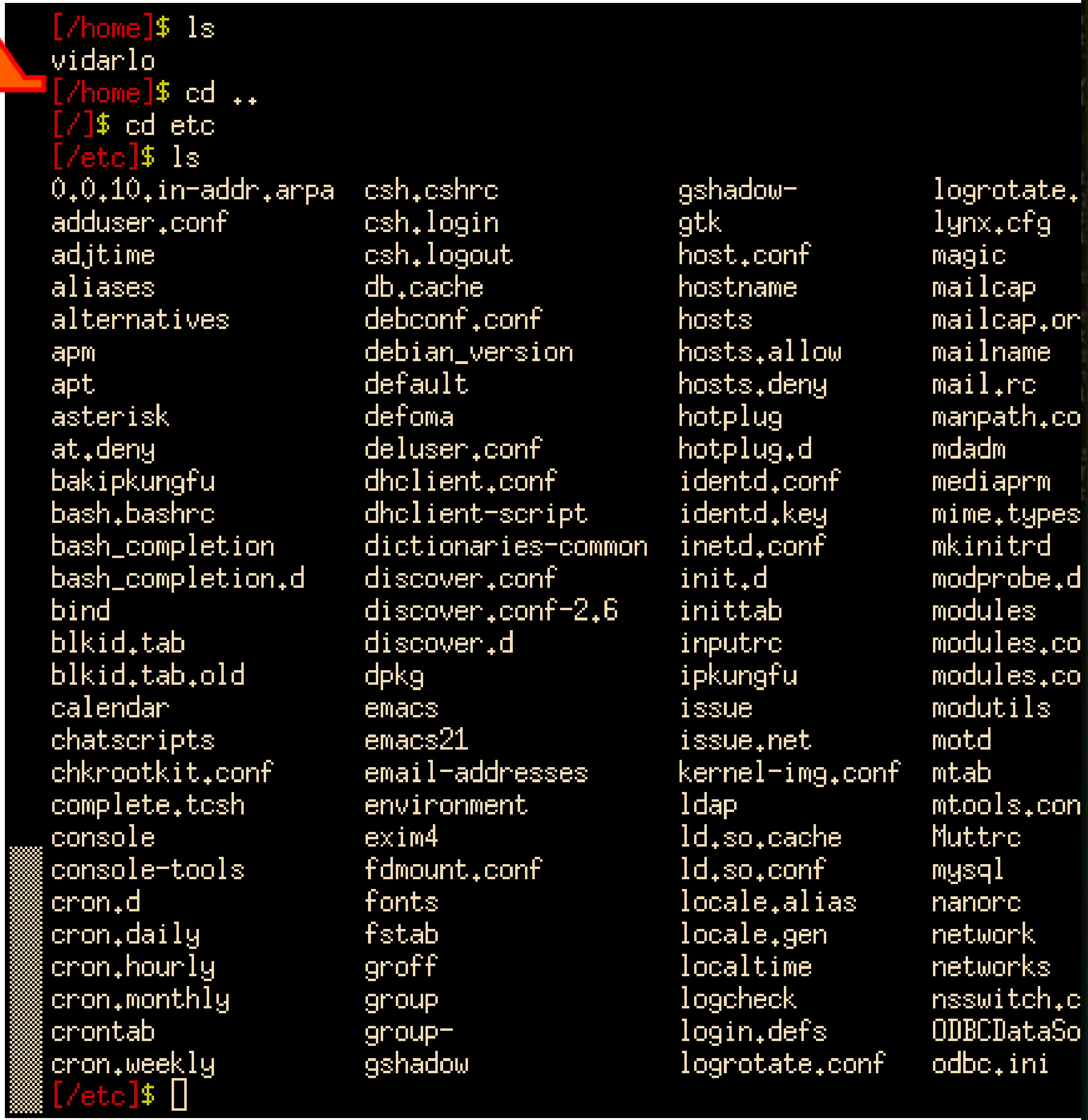
Software development in the real world

1. Identify need for a new piece of software
2. Software engineers design, develop, test, and deploy software
3. Users run or invoke the software as needed ~~indefinitely~~

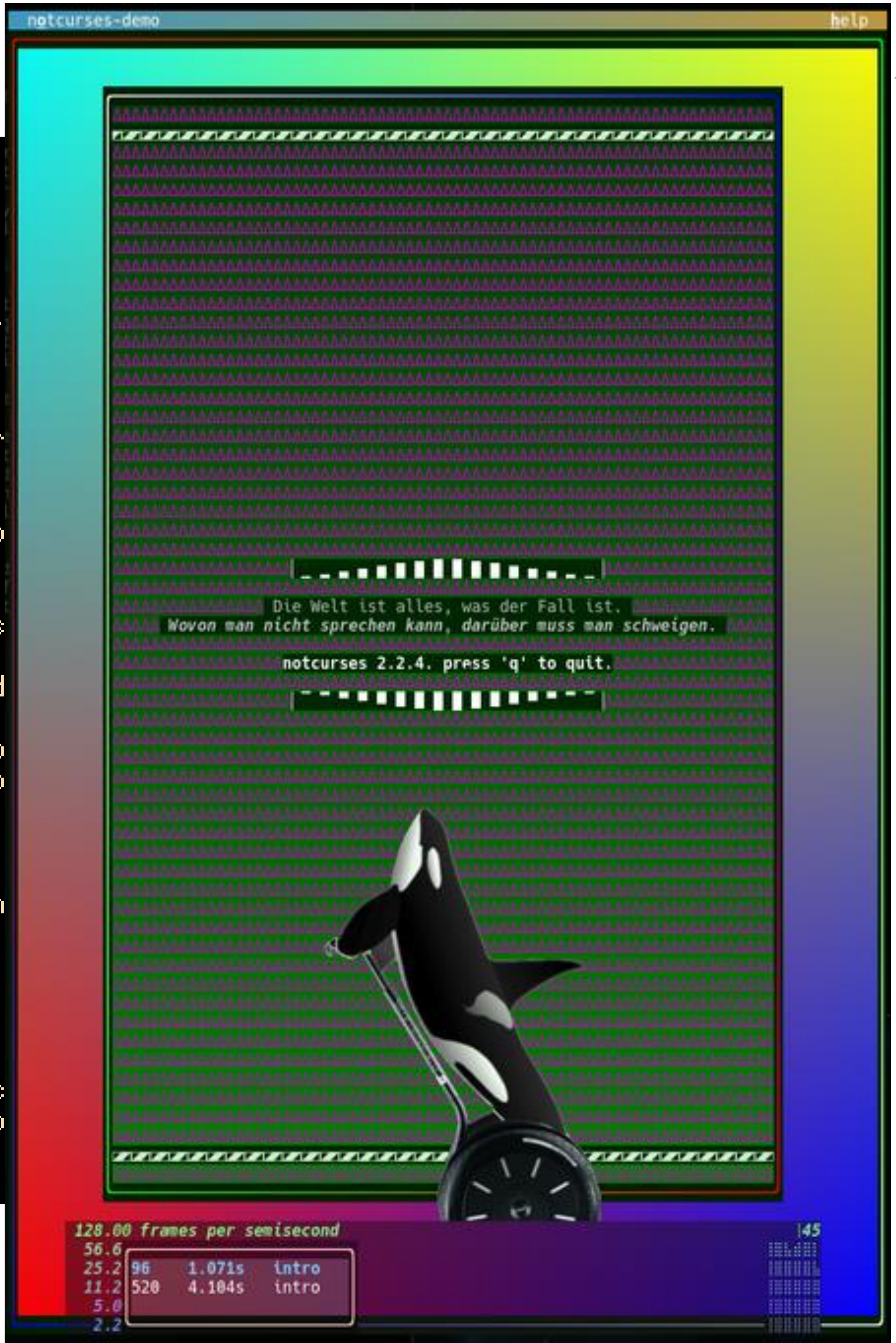
- Operating system and frameworks evolve over time
  - Requires updates to software (e.g., Django 1 → 2 → 3 → 4)
- Software will have errors that require fixing
- Users ask for additional functionality and extensions



```
[/home]$ ls
vidarlo
[/home]$ cd ..
[/]$ cd etc
[/etc]$ ls
0.0.10.in-addr.arpa  csh.cshrc            gshadow-         logrotate.d       odbcinst.ini             rmt
adduser.conf         csh.login            gtk              lynx.cfg          openoffice               rpc
adjtime              csh.logout           host.conf        magic             opt                      screenrc
aliases              db.cache             hostname         mailcap           pam.conf                 securetty
alternatives         debconf.conf         hosts            mailcap.order     pam.d                    security
apm                  debian_version       hosts.allow      mailname          passwd                   services
apt                  default              hosts.deny       mail.rc           passwd-                  shadow
asterisk             defoma               hotplug          manpath.config    perl                     shadow-
at.deny              deluser.conf         hotplug.d        mdadm             ppp                      shells
bakipkungfu          dhclient.conf        identd.conf      mediaprm          printcap                 skel
bash.bashrc          dhclient-script      identd.key       mime.types        profile                  squid
bash_completion      dictionaries-common  inetd.conf       mkinitrd          protocols                ssh
bash_completion.d    discover.conf        init.d           modprobe.d        python2.3                sudoers
bind                 discover.conf-2.6    inittab          modules           raidtab                  sysctl.conf
blkid.tab            discover.d           inputrc          modules.conf      rc0.d                    syslog.conf
blkid.tab.old        dpkg                 ipkungfu         modules.conf.old  rc1.d                    terminfo
calendar             emacs                issue            modutils          rc2.d                    timezone
chatscripts          emacs21              issue.net        motd              rc3.d                    ucf.conf
chkrootkit.conf      email-addresses      kernel-img.conf  mtab              rc4.d                    updatedb.conf
complete.tcsh        environment          ldap             mtools.conf       rc5.d                    vidarlo.net.hosts
console              exim4                ld.so.cache      Muttrc            rc6.d                    w3m
console-tools        fdmount.conf         ld.so.conf       mysql             rc.d                     wgetrc
cron.d               fonts                locale.alias     nanorc            rcS.d                    #wvdial.conf#
cron.daily           fstab                locale.gen       network           reportbug.conf           wvdial.conf
cron.hourly          groff                localtime        networks          resolvconf               wvdial.conf~
cron.monthly         group                logcheck         nsswitch.conf     resolv.conf              X11
crontab              group-               login.defs       ODBCDataSources   resolv.conf~             xpilot
cron.weekly          gshadow              logrotate.conf   odbc.ini          resolv.conf.pppd-backup
[/etc]$ 
```

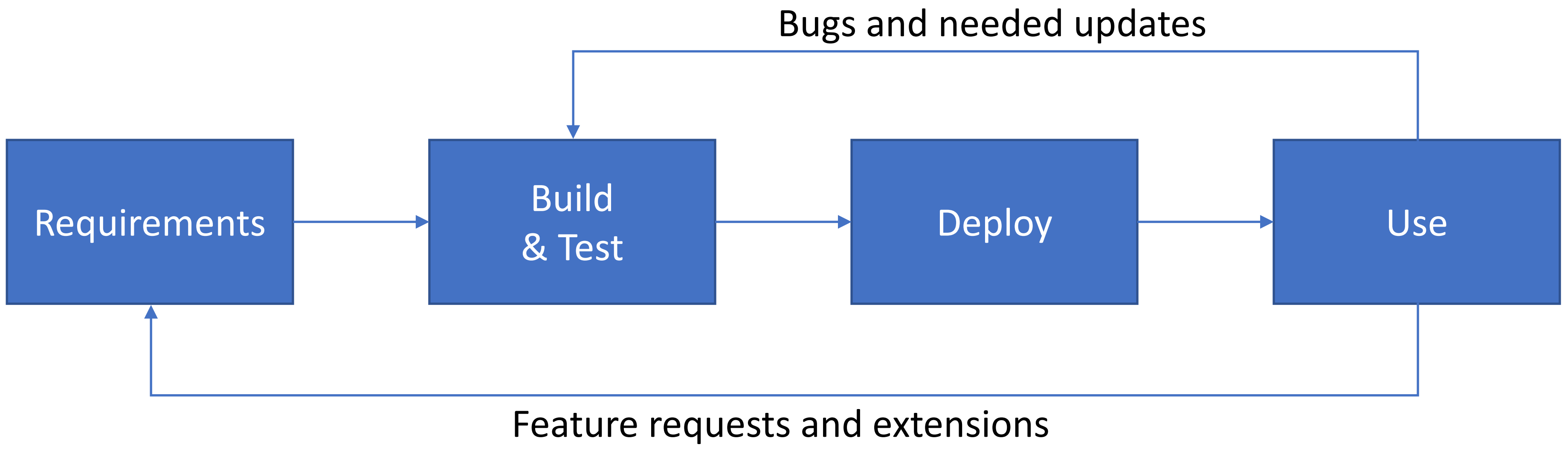


```
[/home]$ ls
vidarlo
[/home]$ cd ..
[/]$ cd etc
[/etc]$ ls
0.0.10.in-addr.arpa  csh.cshrc             gshadow-          logrotate.
adduser.conf         csh.login             gtk               lynx.cfg
adjtime              csh.logout            host.conf         magic
aliases              db.cache              hostname          mailcap
alternatives         debconf.conf          hosts             mailcap.or
apm                  debian_version        hosts.allow       mailname
apt                  default               hosts.deny        mail.rc
asterisk             defoma                hotplug           manpath.co
at.deny              deluser.conf          hotplug.d         mdadm
bakipkungfu          dhclient.conf         identd.conf       mediaprm
bash.bashrc          dhclient-script       identd.key        mime.types
bash_completion      dictionaries-common   inetd.conf        mkinitrd
bash_completion.d    discover.conf         init.d            modprobe.d
bind                 discover.conf-2.6     inittab           modules
blkid.tab            discover.d            inputrc           modules.co
blkid.tab.old        dpkg                  ipkungfu          modules.co
calendar             emacs                 issue             modutils
chatscripts          emacs21               issue.net         motd
chkrootkit.conf      email-addresses       kernel-img.conf   mtab
complete.tcsh        environment           ldap              mtools.con
console              exim4                 ld.so.cache       Muttrc
console-tools        fdmount.conf          ld.so.conf        mysql
cron.d               fonts                 locale.alias      nanorc
cron.daily           fstab                 locale.gen        network
cron.hourly          groff                 localtime         networks
cron.monthly         group                 logcheck          nsswitch.c
crontab              group-                login.defs        ODBCDataSo
cron.weekly          gshadow               logrotate.conf    odbc.ini
[/etc]$ █
```

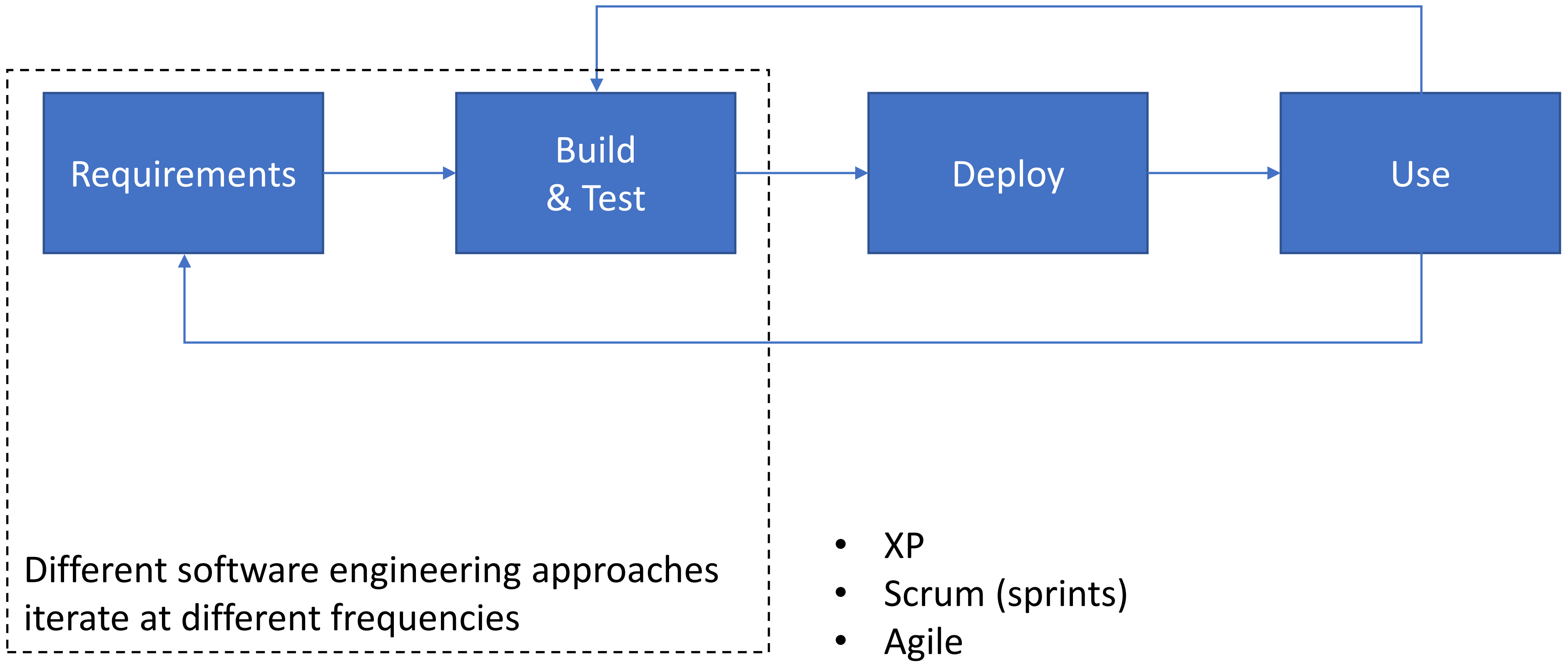


```
Die Welt ist alles, was der Fall ist.
Wovon man nicht sprechen kann, darüber muss man schweigen.

notcurses 2.2.4. press 'q' to quit.

128.00 frames per semisecond
 56.6
 25.2  96     1.071s    intro
 11.2  520    4.104s    intro
  5.0
  2.2
```

Software development cycle



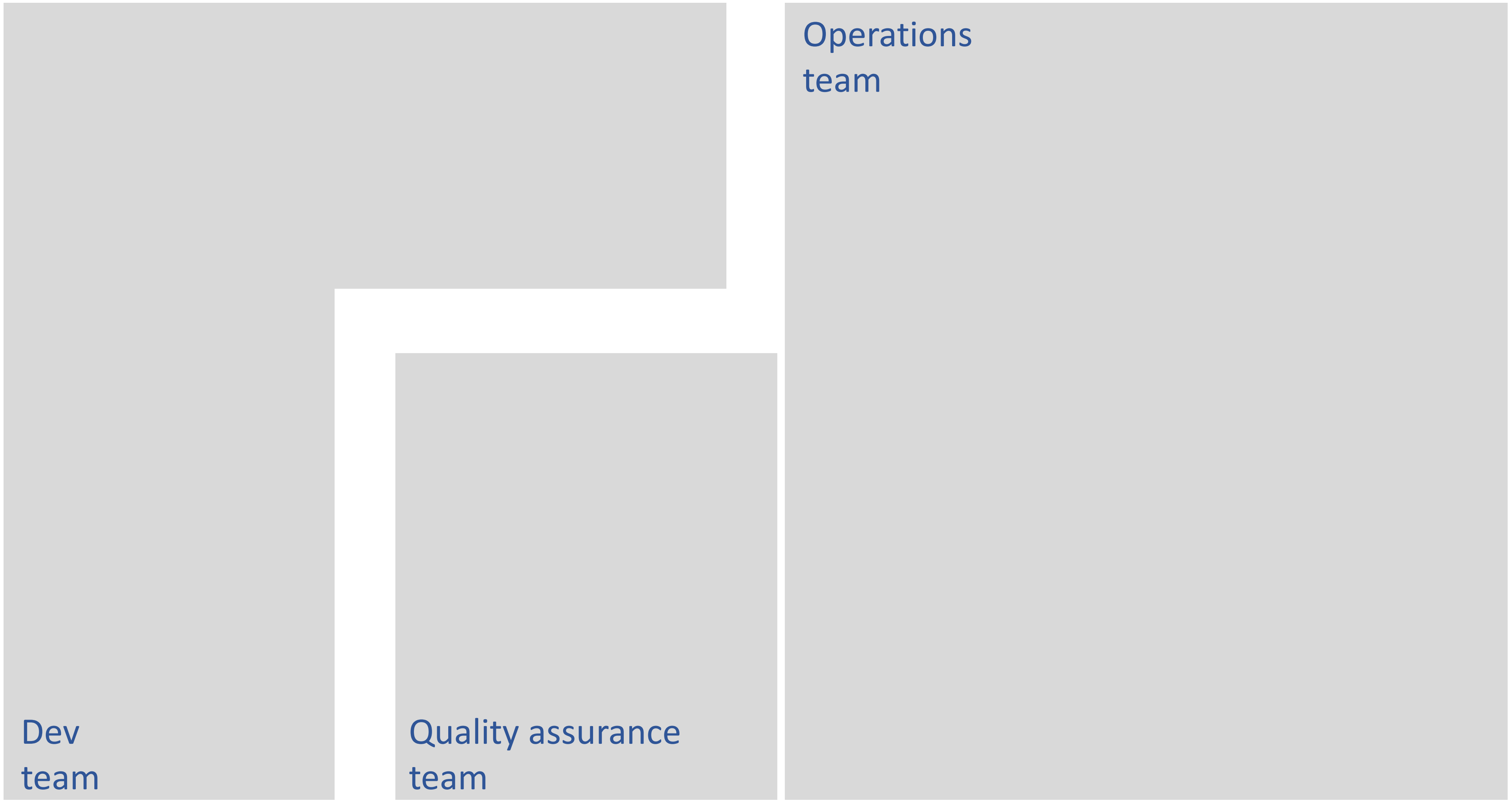
Software development cycle



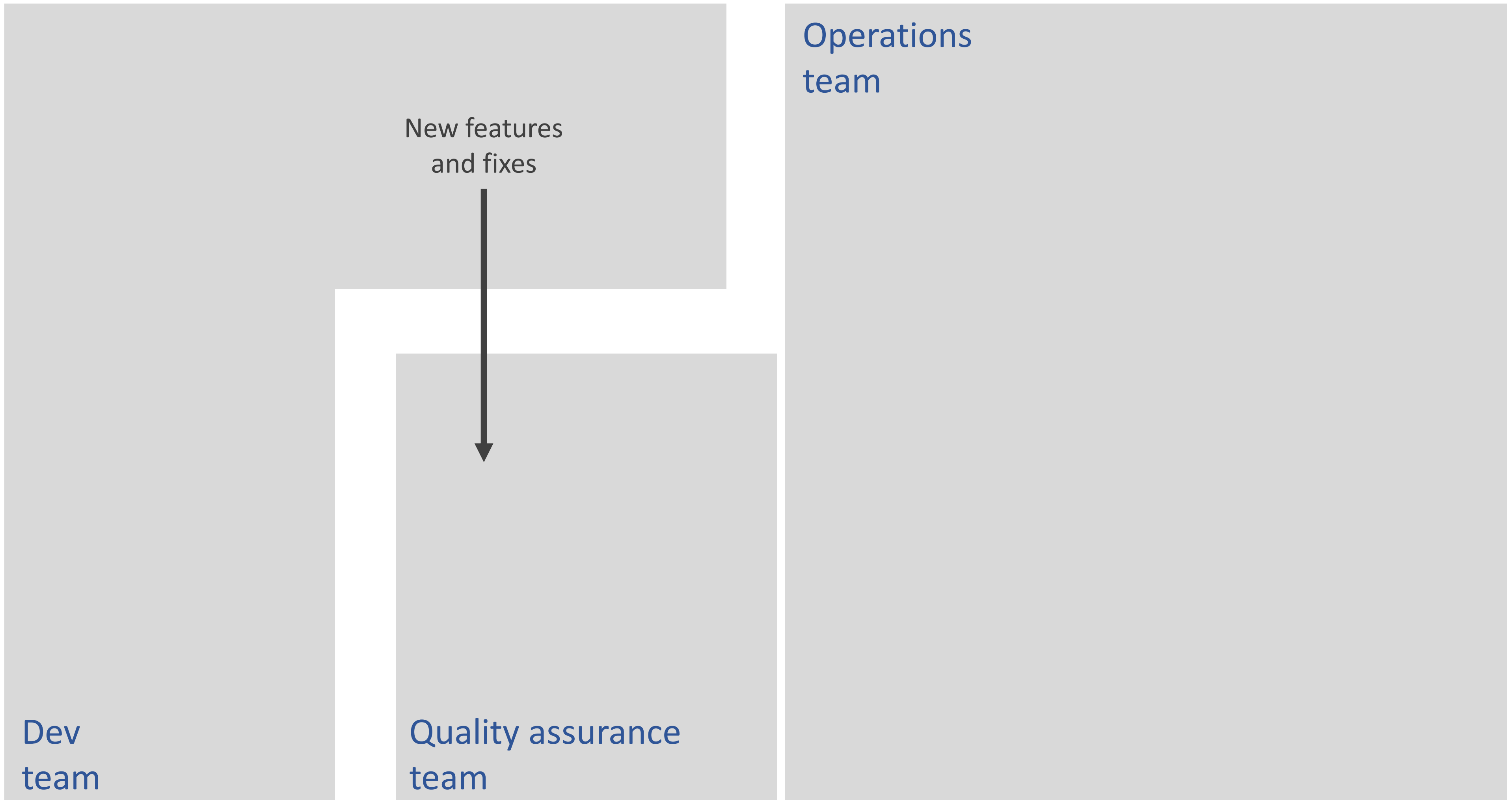
Different software engineering approaches iterate at different frequencies

- XP
- Scrum (sprints)
- Agile

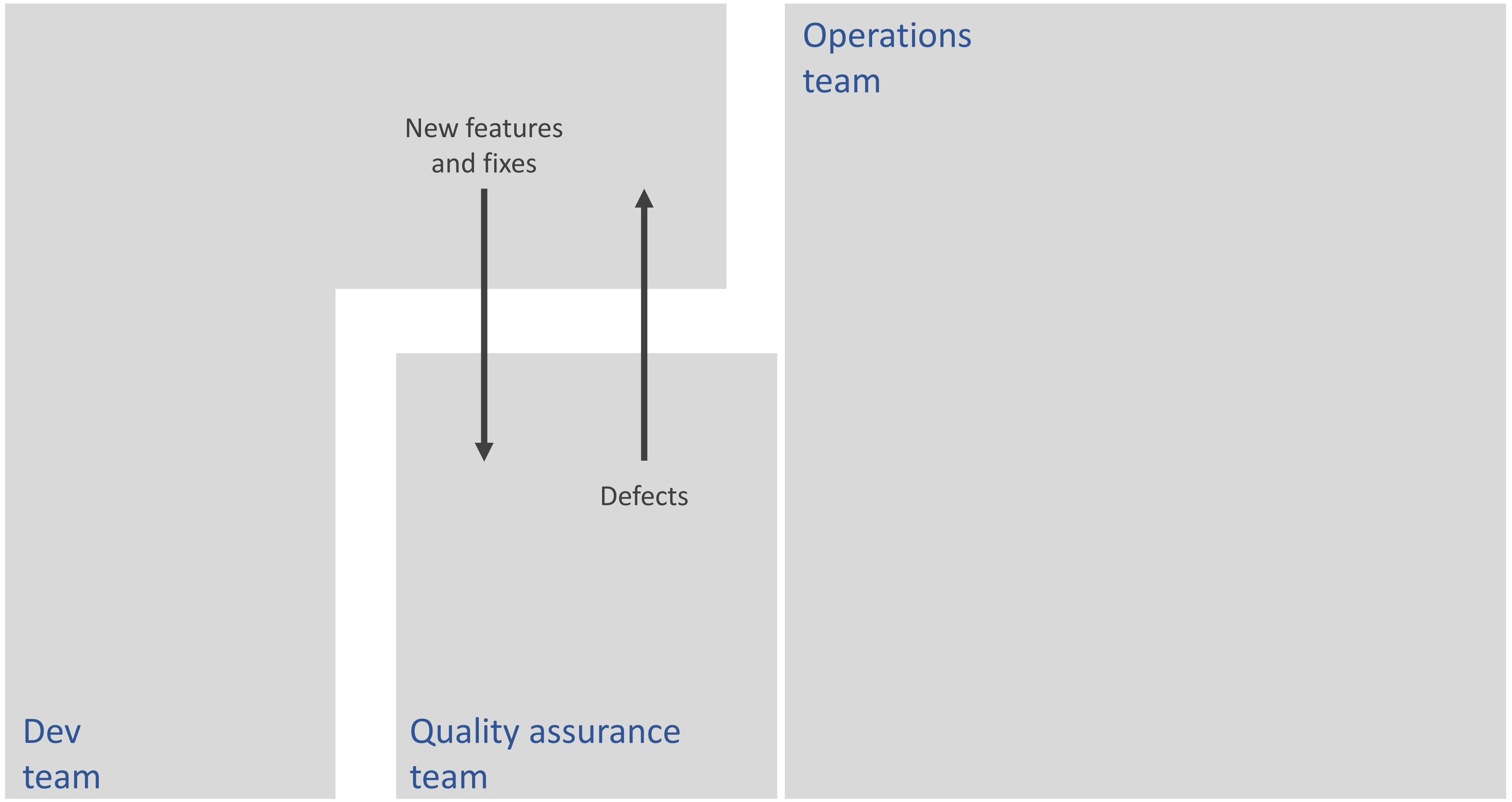
Large projects and teams



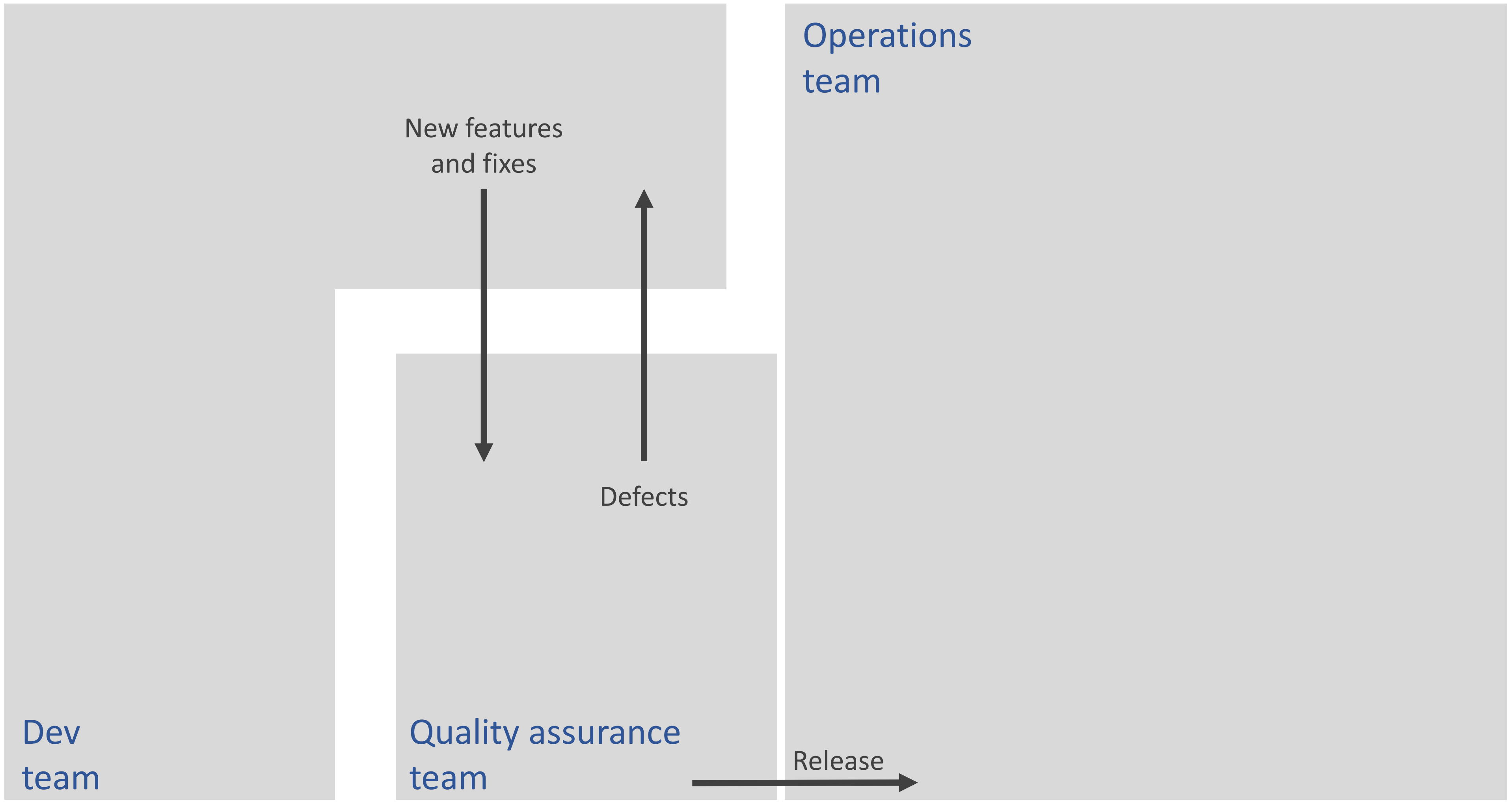
Large projects and teams



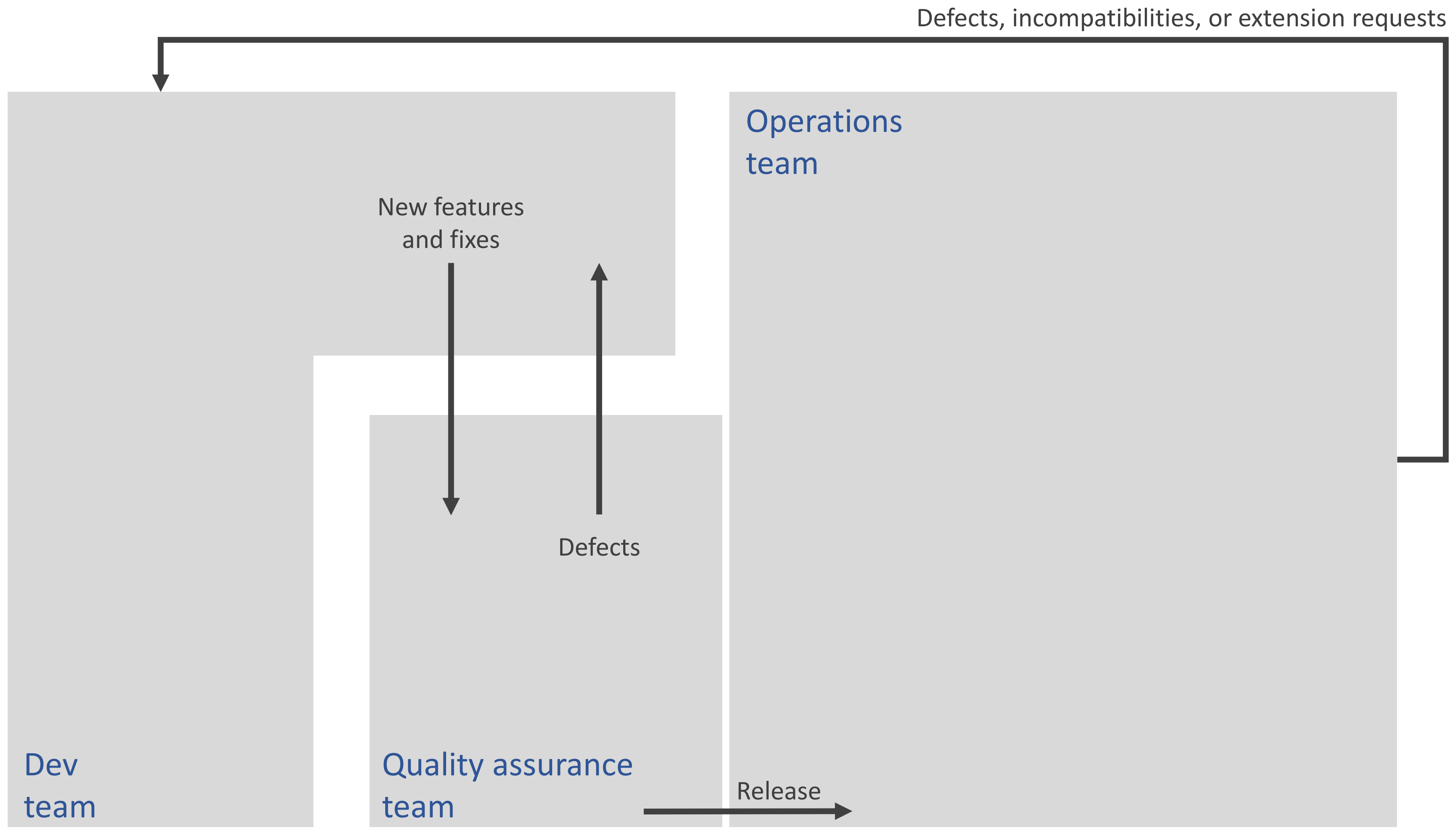
Large projects and teams



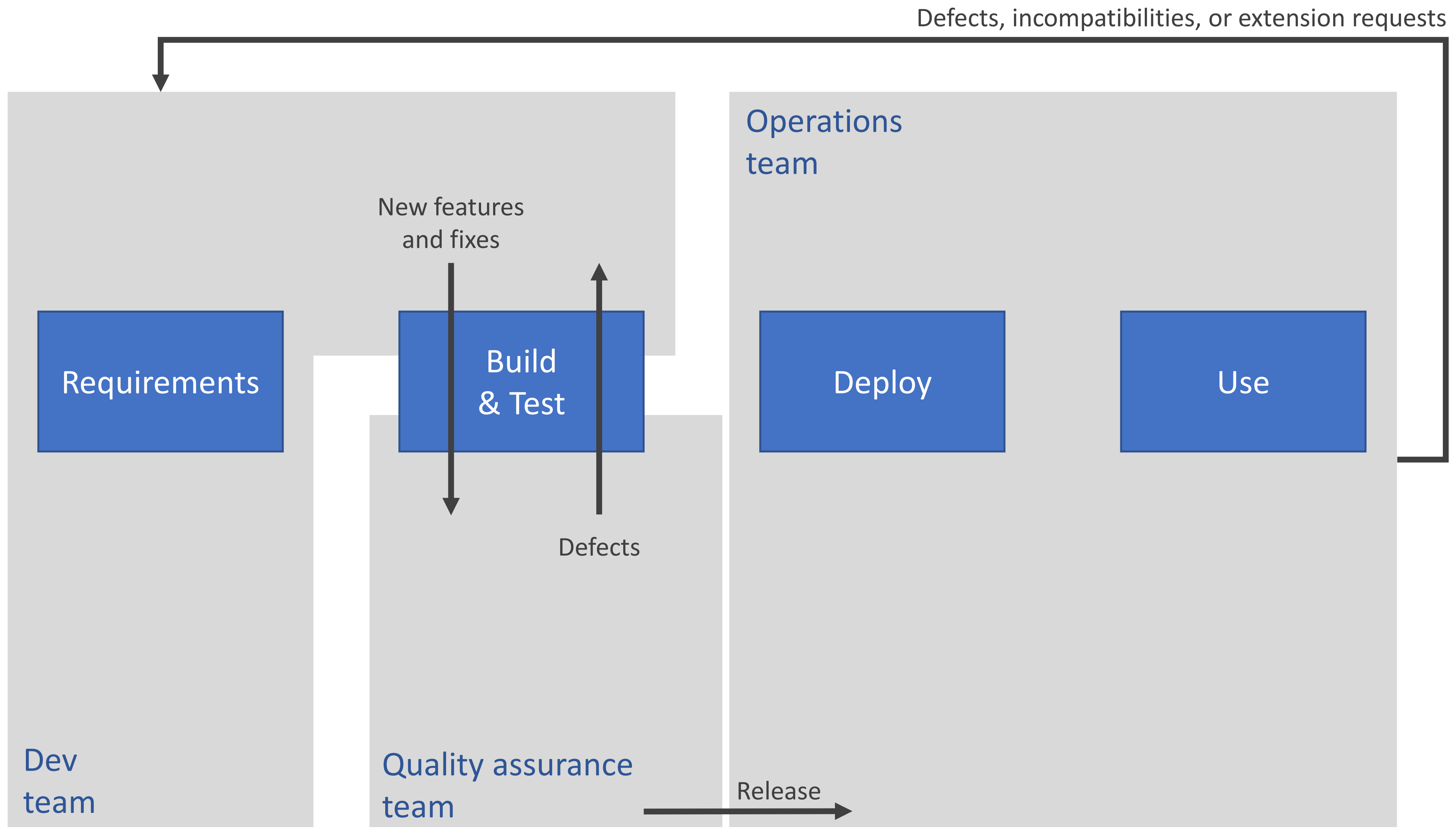
Large projects and teams



Large projects and teams



Large projects and teams



Conflicting interests

- More features
- Ship *yesterday*
- Latest libraries and frameworks
- Unlimited resources

Dev team

Quality assurance team

- Stable libraries and frameworks
- Lightweight code, resource constraints
- Zero errors

Operations team

CompSci 401: Cloud Computing

DevOps

Prof. Ítalo Cunha

昆山杜克大学
DUKE KUNSHAN UNIVERSITY

DevOps paradigm

- A culture/organizational approach to software development
- Join development, QA, and operation teams
- An extension to agile methods
  - Relies heavily on automation and orchestration

DevOps paradigm

- A culture/organizational approach to software development
- Join development, QA, and operation teams
- An extension to agile methods
  - Relies heavily on automation and orchestration

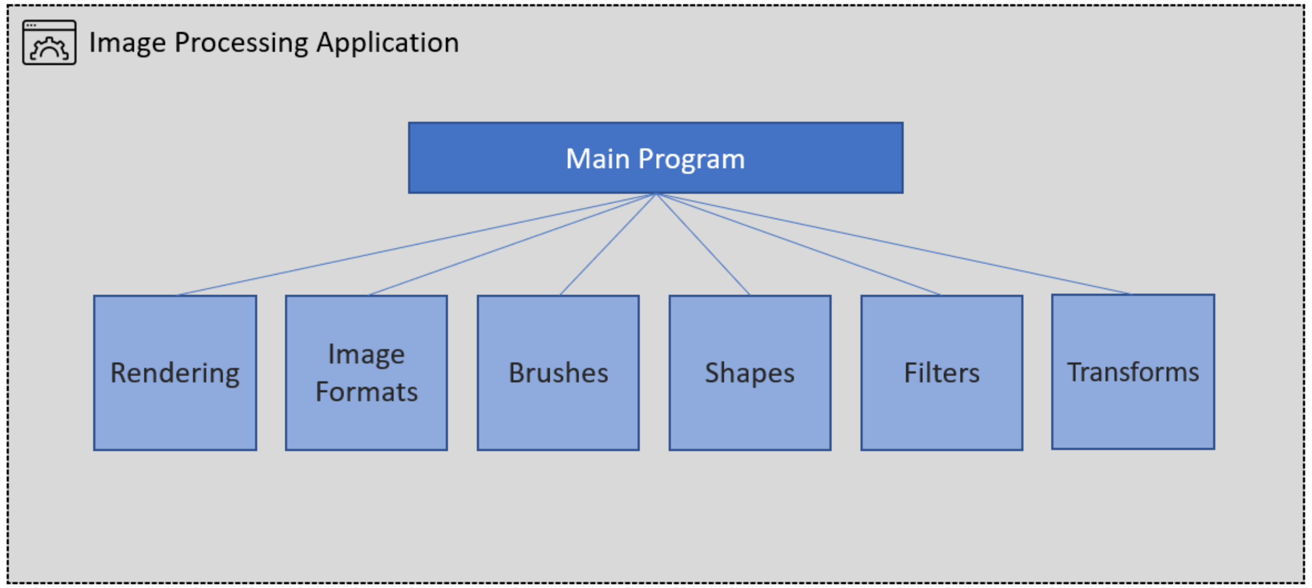
- If operations requires a serverless application can coordinate with developers to build stateless code
- Operations can inform developers a preferred/desirable microservice approach, including how to modularize/size

DevOps requirements

- Automation and orchestration
  - Continuous integration to automate building and testing
    - Version management to separate development code from production code
    - Small software modules to compartmentalize testing
    - Extensive testing to ensure software quality (e.g., test-driven development)

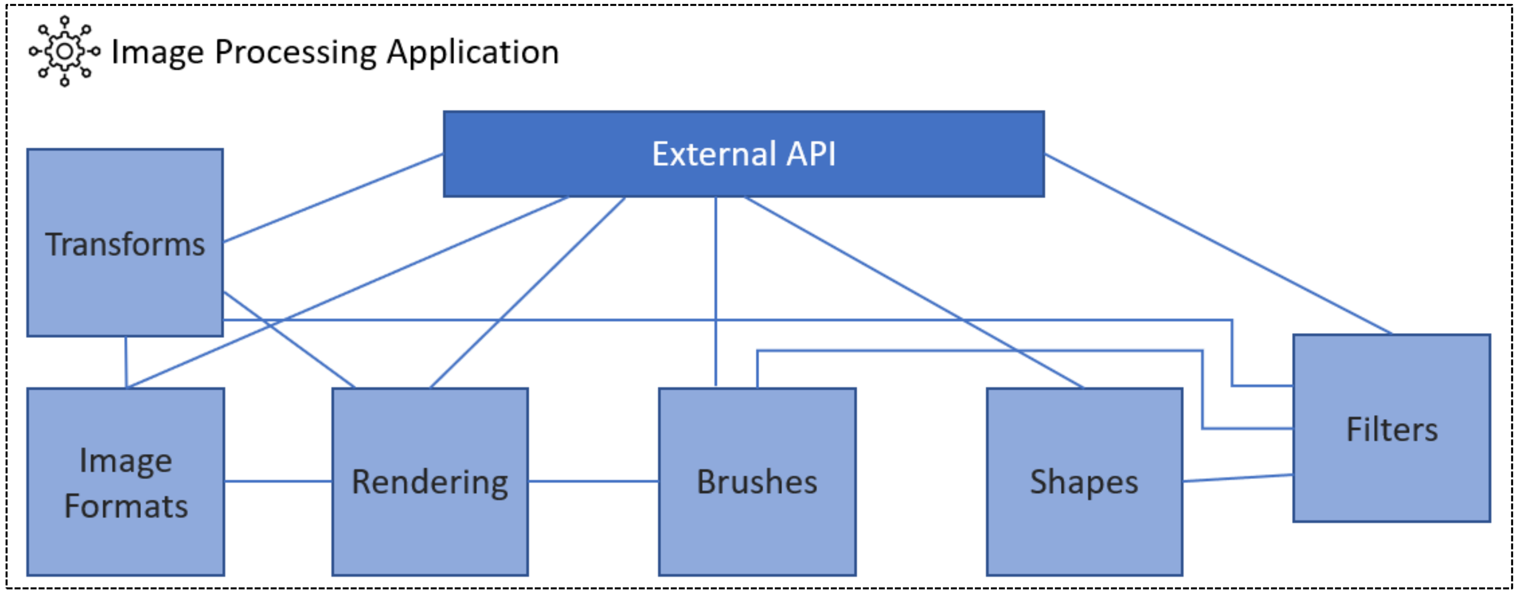
DevOps requirements

- Automation and orchestration
  - Continuous integration to automate building and testing
    - Version management to separate development code from production code
    - Small software modules to compartmentalize testing
    - Extensive testing to ensure software quality (e.g., test-driven development)



DevOps requirements

- Automation and orchestration
  - Continuous integration to automate building and testing
    - Version management to separate development code from production code
    - Small software modules to compartmentalize testing
    - Extensive testing to ensure software quality (e.g., test-driven development)



DevOps requirements

- Automation and orchestration
  - Continuous integration to automate building and testing
    - Version management to separate development code from production code
    - Small software modules to compartmentalize testing
    - Extensive testing to ensure software quality (e.g., test-driven development)

DevOps requirements

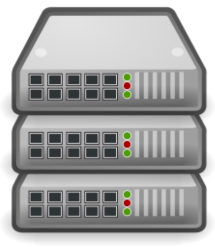
- Automation and orchestration
  - Continuous integration to automate building and testing
    - Version management to separate development code from production code
    - Small software modules to compartmentalize testing
    - Extensive testing to ensure software quality (e.g., test-driven development)
  - Continuous delivery to automate software deployment

Classic software releases and updates

- Package software
- Pick “low risk” time to deploy
  - Maintenance window
  - Warn users of possible unavailability
- Possibly bring the application offline to swap versions

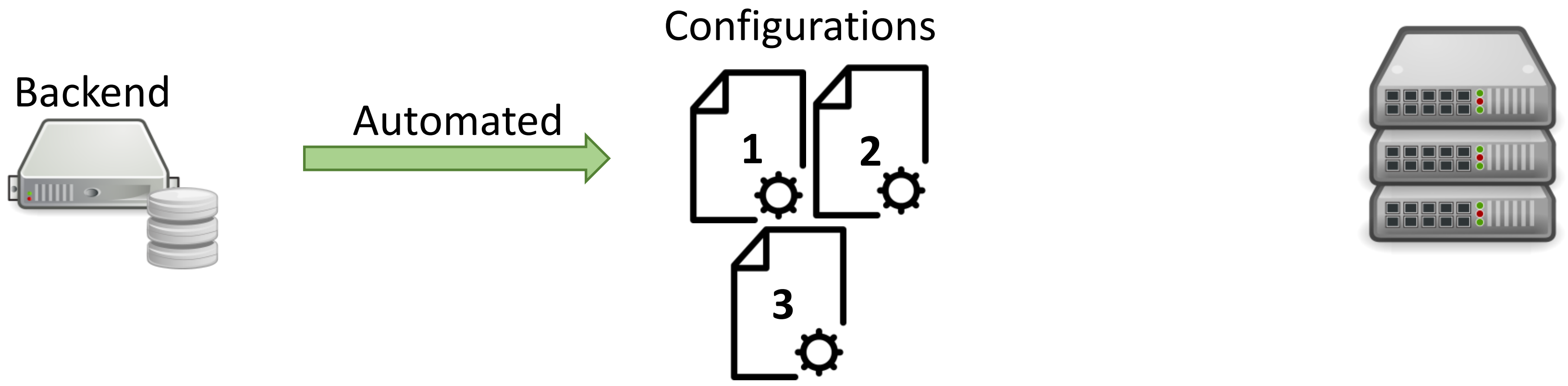
Classic software releases and updates

- Package software
- Pick “low risk” time to deploy
  - Maintenance window
  - Warn users of possible unavailability
- Possibly bring the application offline to swap versions



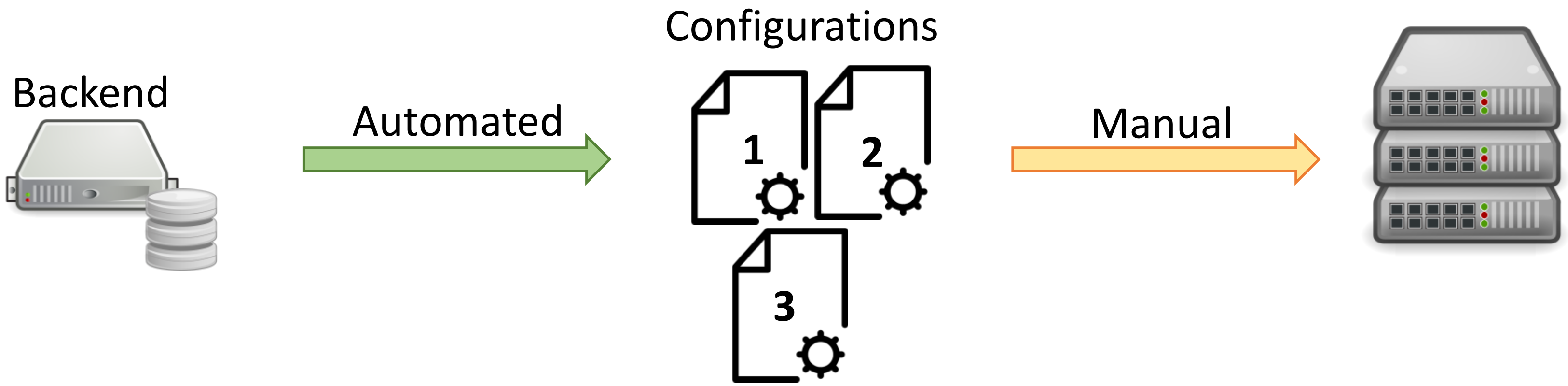
Classic software releases and updates

- Package software
- Pick “low risk” time to deploy
  - Maintenance window
  - Warn users of possible unavailability
- Possibly bring the application offline to swap versions



Classic software releases and updates

- Package software
- Pick “low risk” time to deploy
  - Maintenance window
  - Warn users of possible unavailability
- Possibly bring the application offline to swap versions



Classic software releases and updates

- Package software
- Pick “low risk” time to deploy
    - Maintenance window
    - Warn users of possible unavailability
- Possibly bring the application offline to swap versions

Classic software releases and updates

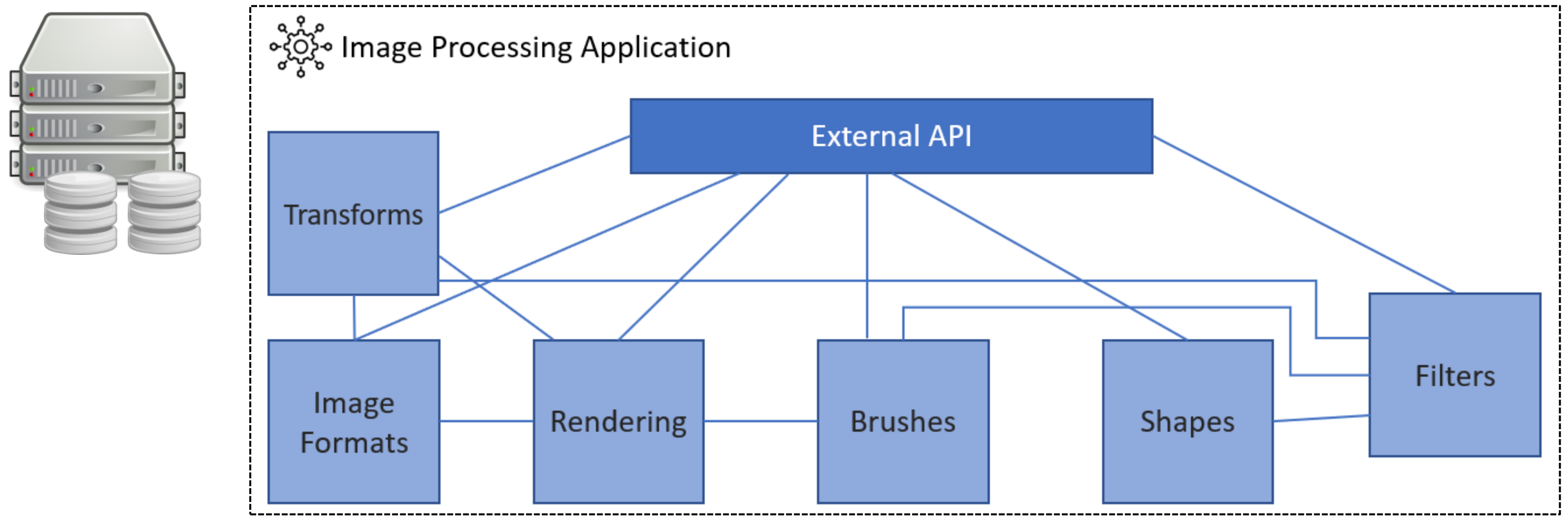
- Package software
- Pick “low risk” time to deploy
  - Maintenance window
  - Warn users of possible unavailability
- Possibly bring the application offline to swap versions

- Not a good fit for cloud environments
  - Large/global user base implies there is no “low risk” time to deploy
  - In a microservice architecture, other services depend on the updated service
  - Stopping a microservice for a release may be impossible

CI/CD to deploy updates rapidly

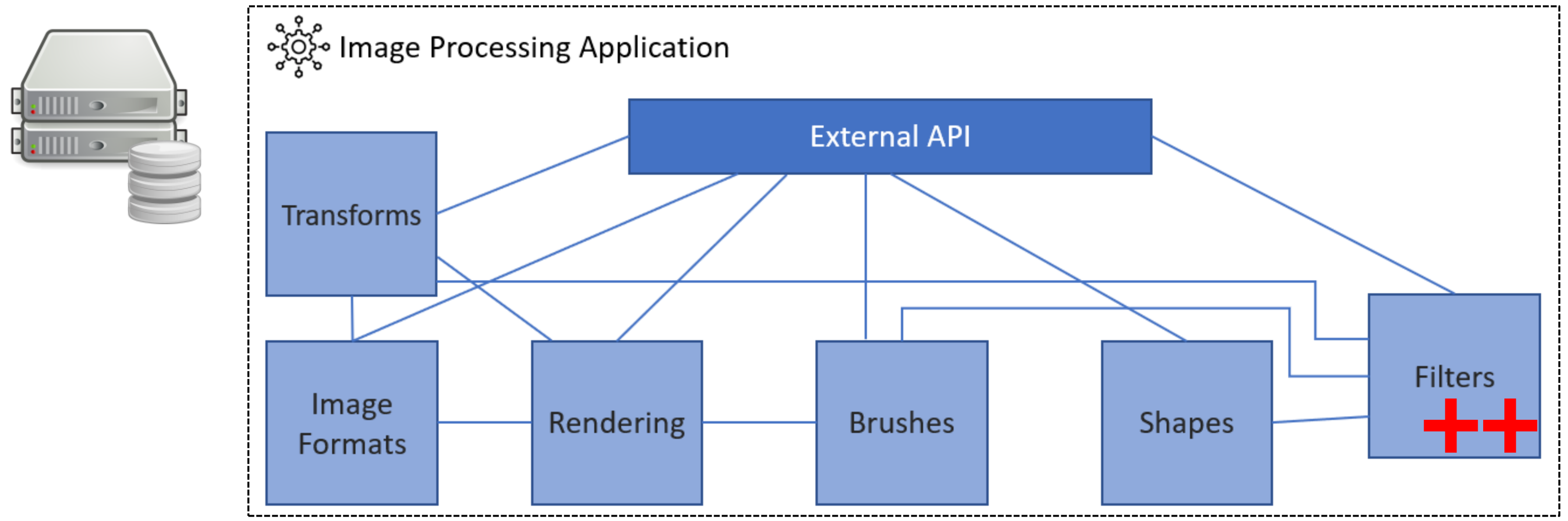
- Smaller modules and extensive testing improve software quality
  - Less errors in production
- Rollout
  - Containers running old version can continue running during upgrade
    - Can finish serving ongoing requests
  - Containers running new version spawn and are monitored closely for errors
    - Eventual migration if no errors occur

Sandbox trial



Production version

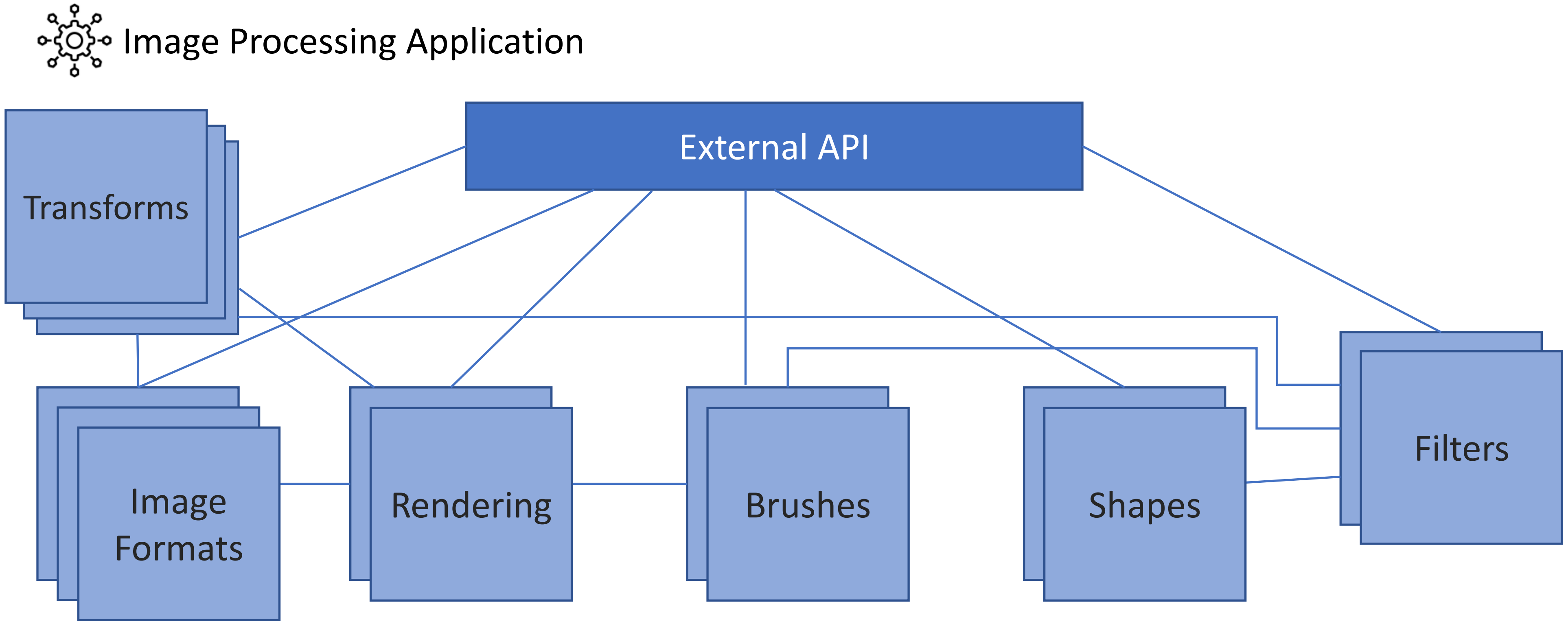
- Unchanged



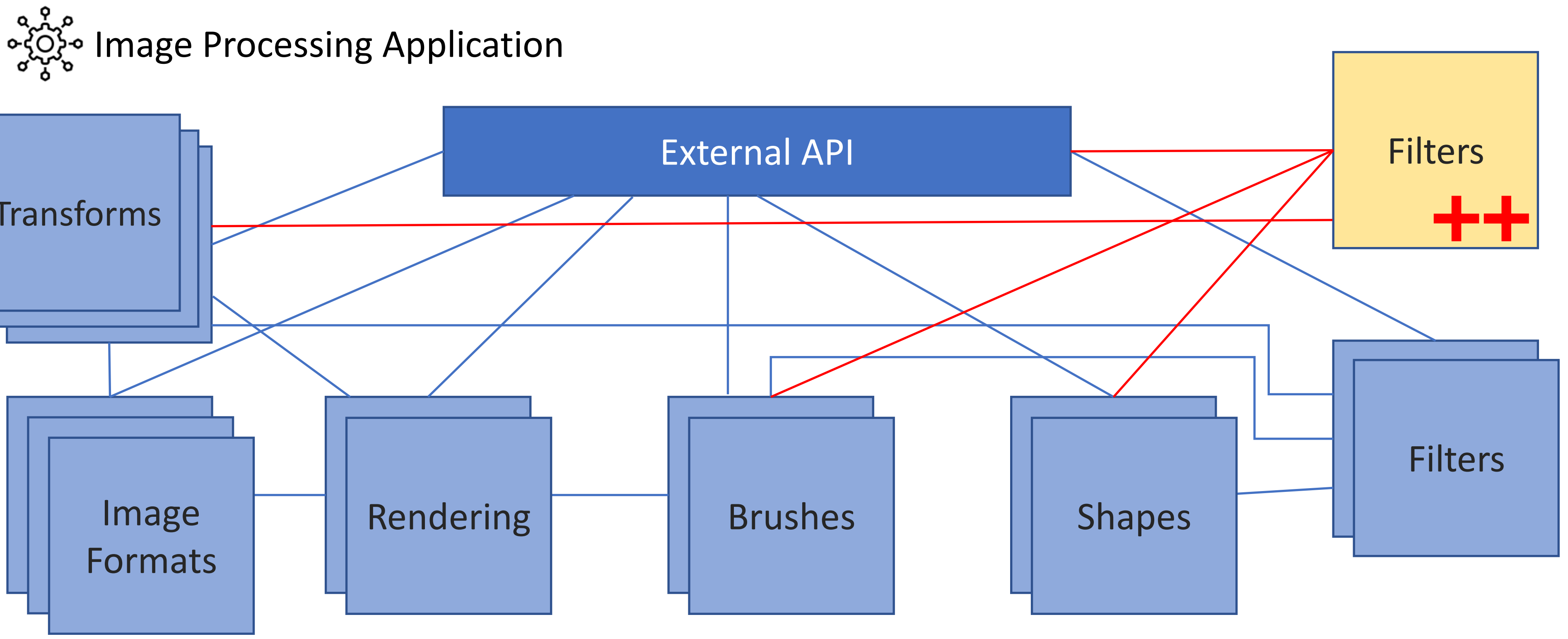
Sandbox trial

- With updates
- Does not impact production version
- May not catch all errors due to isolation
  - Different users and infrastructure
- High cost

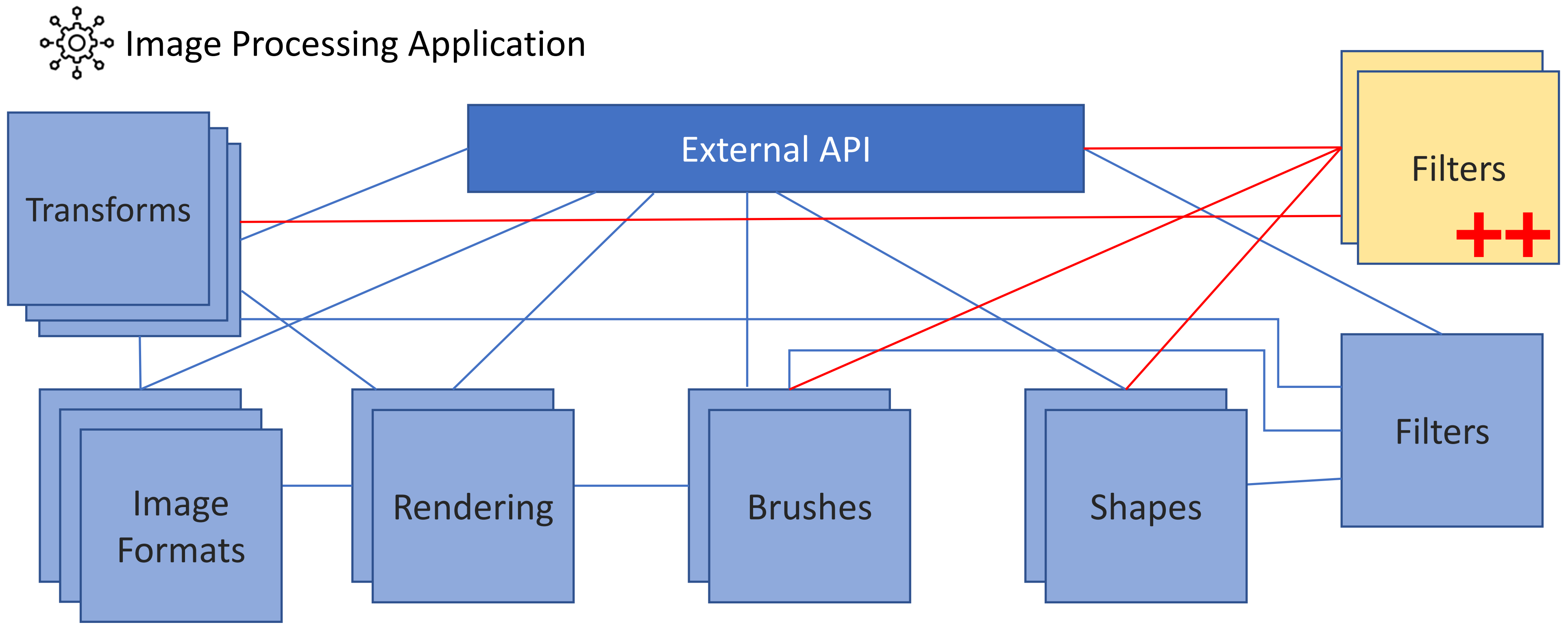
Canary deployment



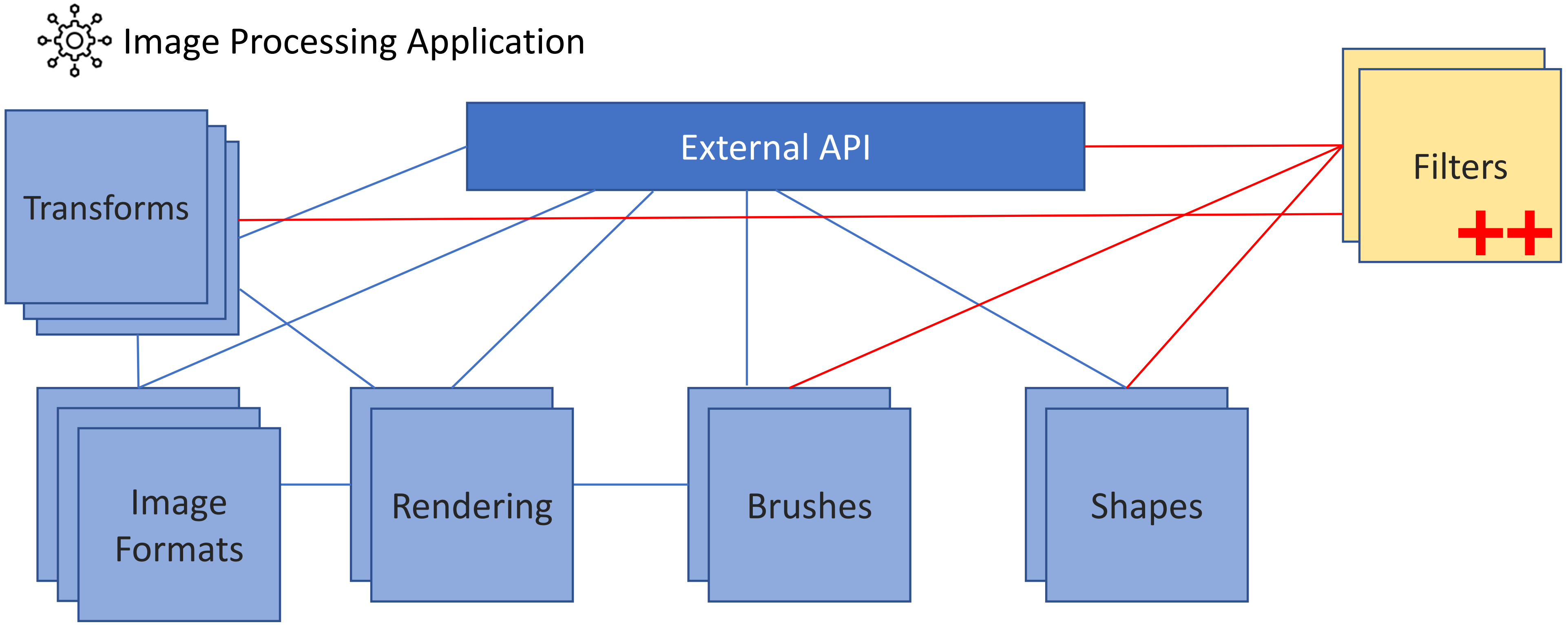
Canary deployment



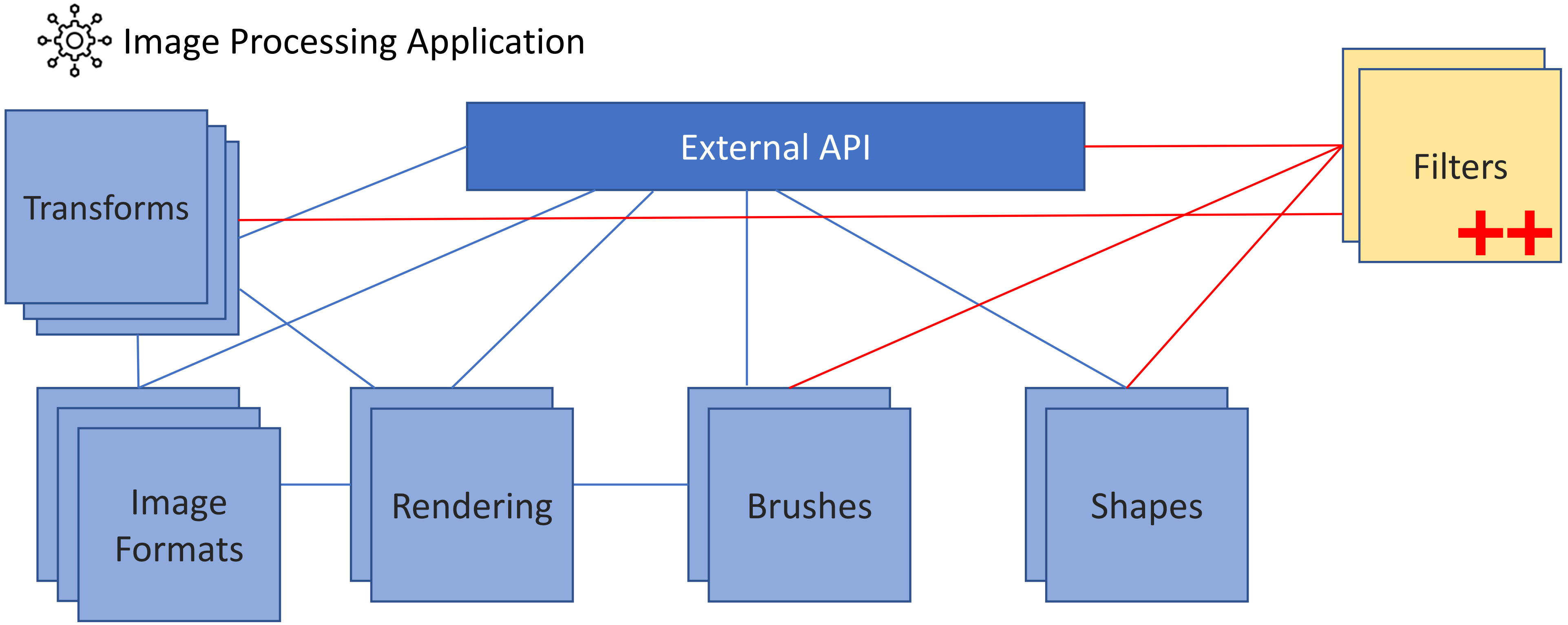
Canary deployment



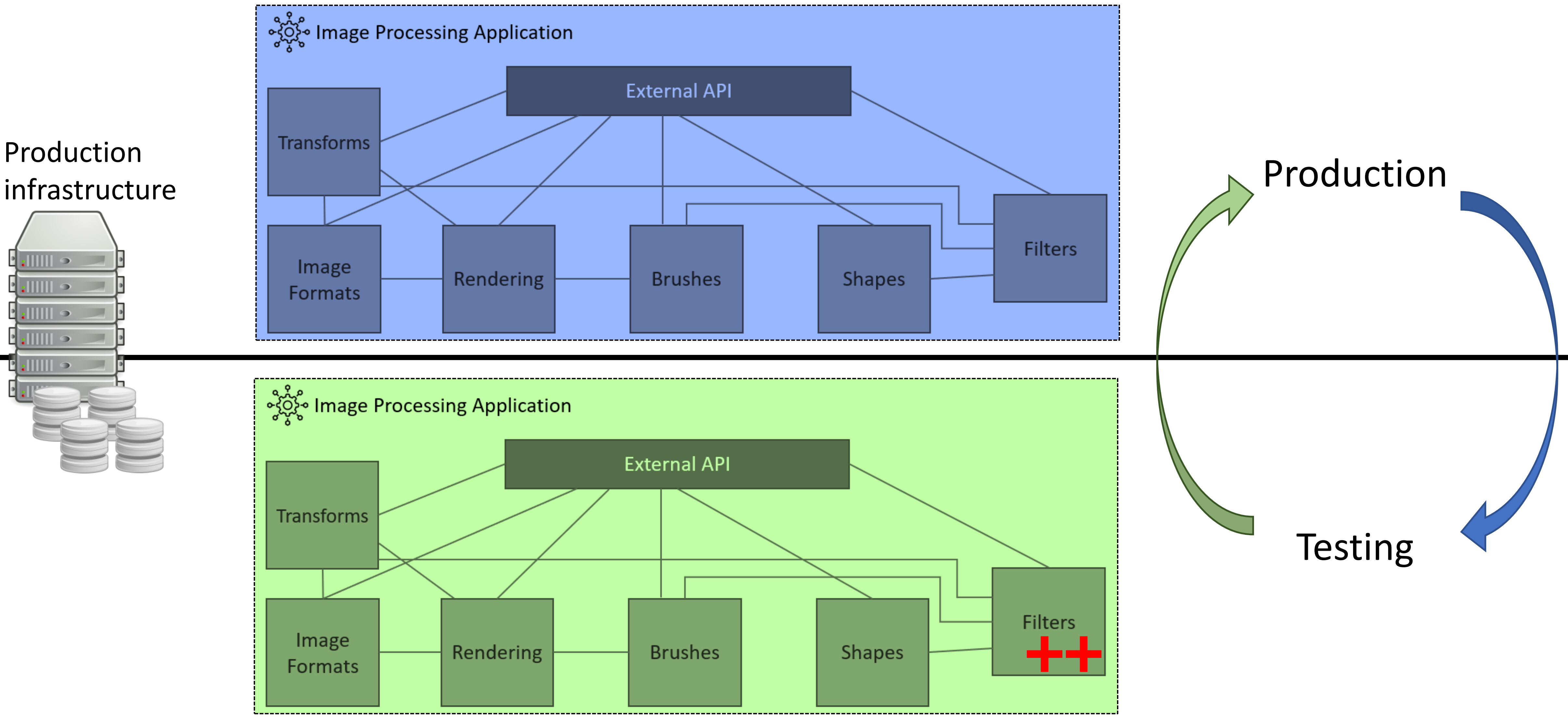
Canary deployment



Canary deployment

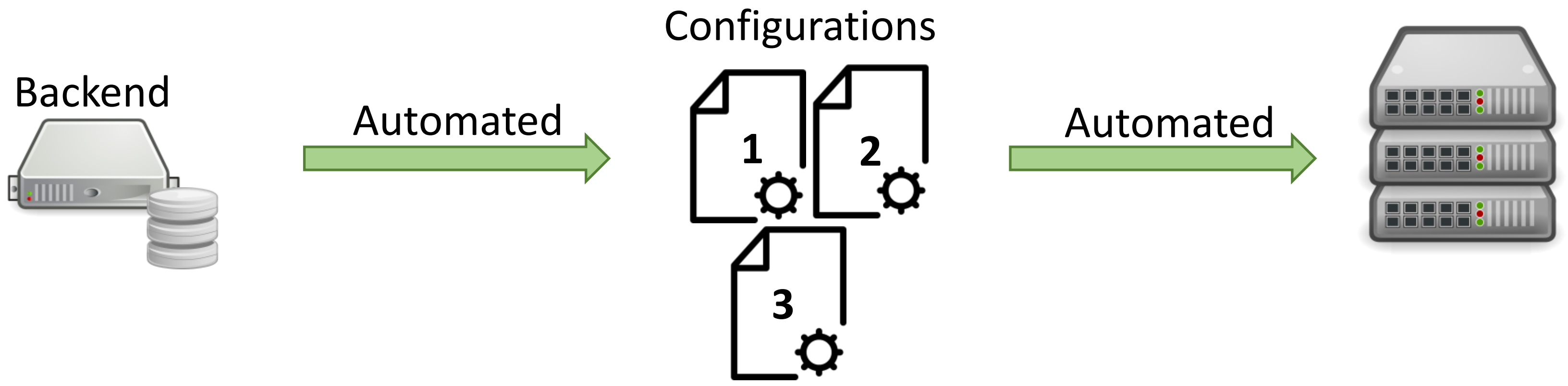


Blue-green deployment



Modern software releases and updates

- Generate configurations
- Have configurations pushed automatically
  - For example, periodically push configurations every night
- Monitor router and traffic behavior
- Automatically roll back if errors occur



Difficult aspects of DevOps

- Culture shift
  - Teams have to collaborate more closely or merge
  - Realignment of incentives and rewards
- Extensive automation and orchestration
  - Initial setup costs, maintenance overhead
- Higher risk from rapid development
  - Despite tests and rollout/rollbacks, failures may occur more often
- System-wide failures instead of module failures
  - DevOps performs extensive tests of small modules,
    but global or cascading failures may occur in production

CompSci 401: Cloud Computing

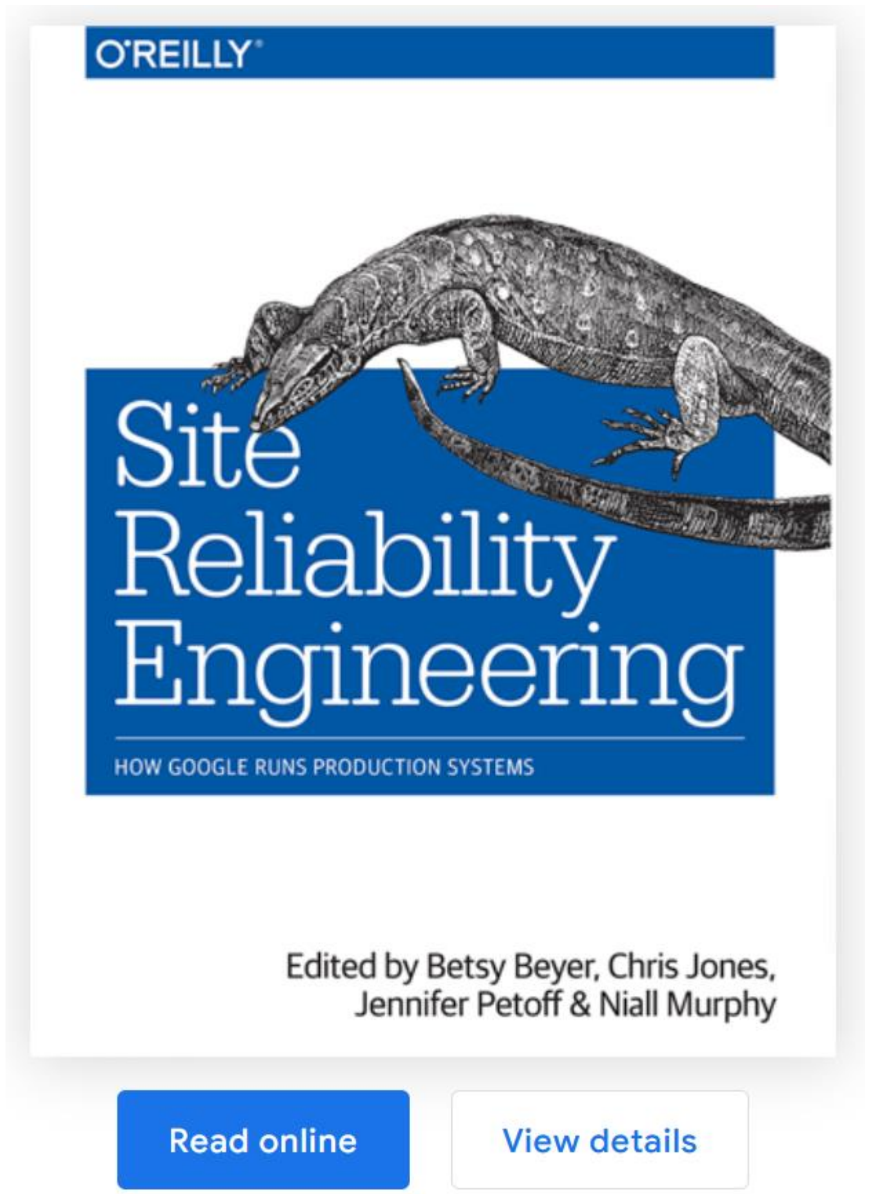
Site Reliability Engineering

Prof. Ítalo Cunha

昆山杜克大学
DUKE KUNSHAN UNIVERSITY

Site Reliability Engineering

- Systems management within Google
- Very similar to DevOps


O'REILLY
Site Reliability Engineering
HOW GOOGLE RUNS PRODUCTION SYSTEMS
Edited by Betsy Beyer, Chris Jones, Jennifer Petoff & Niall Murphy
Read online
View details

SRE at Google

- SRE team members are software developers
- 50% cap on “ops” tasks
- At least 50% on development
- SREs build automation solutions
  - Or “automatic” systems that can run and repair themselves

Continued focus on engineering

- 50% of the time on development work
- Operational load above 50% redirected to development teams
- 1-2 incidents per 8-12h shift
  - Gives engineers time to dig deep and write post-mortem reports
  - Avoids pager fatigue

Error budgets and change velocity

- Most services aim for less than 100% availability
  - 99.99% availability is *much* easier to achieve than 100%
  - And the difference between the two might not be perceptible by users
- Allows teams to implement and deploy changes quickly
  - Provided the service is above the availability target
- Improve robustness if availability is below target

Emergency response

- Humans add latency
- Automate emergency response as much as possible
  - Progressive rollouts and automated rollbacks
- Playbooks for dealing with issues