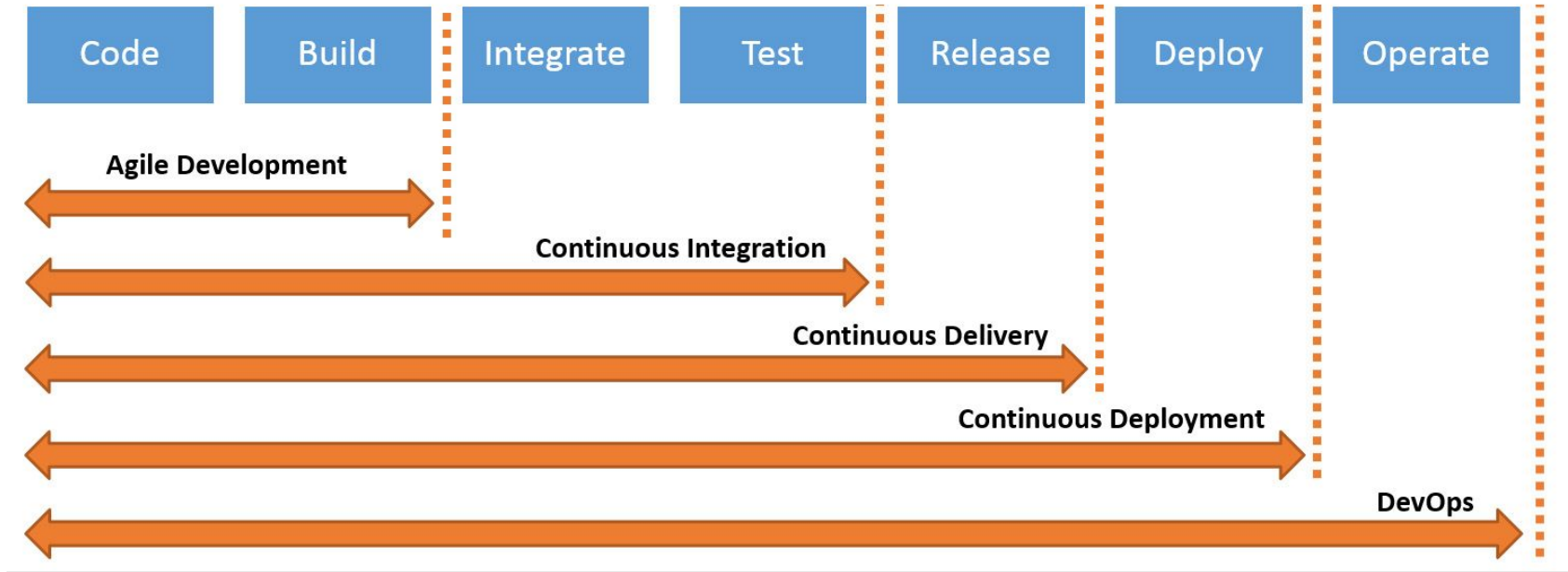


DevOps
CI/CD





CI/CD/CD



Continuous Integration

все изменения, вносимые в код, объединяются в центральном репозитории. Слияние происходит несколько раз в день, и после каждого слияния в конкретном проекте срабатывает автоматическая сборка и тестирование.

Основные цели continuous integration – поиск и устранение потенциальных проблем как можно быстрее, улучшение качества ПО и сокращение время для выпуска обновлений.

Рассмотрим классический процесс CI

Берем задачу из списка

Создаем новую ветку в git и открываем пул реквест

Пишем код

Лично или с помощью коллеги выполняем код-ревью (code review — обзор/проверку кода)

Запускаем тесты

Сливаем ветку в master

Выполняем сборку проекта

Публикуем новую сборку

Continuous Delivery

CD - это **серия практик**, направленных на то, чтобы обновления программного обеспечения происходили практически постоянно. Данные методы гарантируют быстрое развёртывание на продакшене не меняя существующий функционал. Continuous delivery осуществима благодаря различным оптимизациям на ранних этапах процесса разработки.

Continuous Delivery предоставляет бизнесу каждый функционал постепенно. Это позволяет получить сразу отклик от клиента и, при необходимости, сделать некоторые изменения.

Continuous Delivery

В 1998-м году в одной швейцарской страховой компании Кент Бек построил процесс разработки, по которому каждый день после ночных тестов приложение разворачивалось в промышленную эксплуатацию.

Через 12 лет Джек Хамбл и Дейв Фарли публикуют книгу «Continuous Delivery», в основу которой лёг многолетний опыт их работы в компании Thoughtworks, до сих пор являющейся одним из мировых лидеров в области Agile-разработки.

Преимущества Continuous Delivery

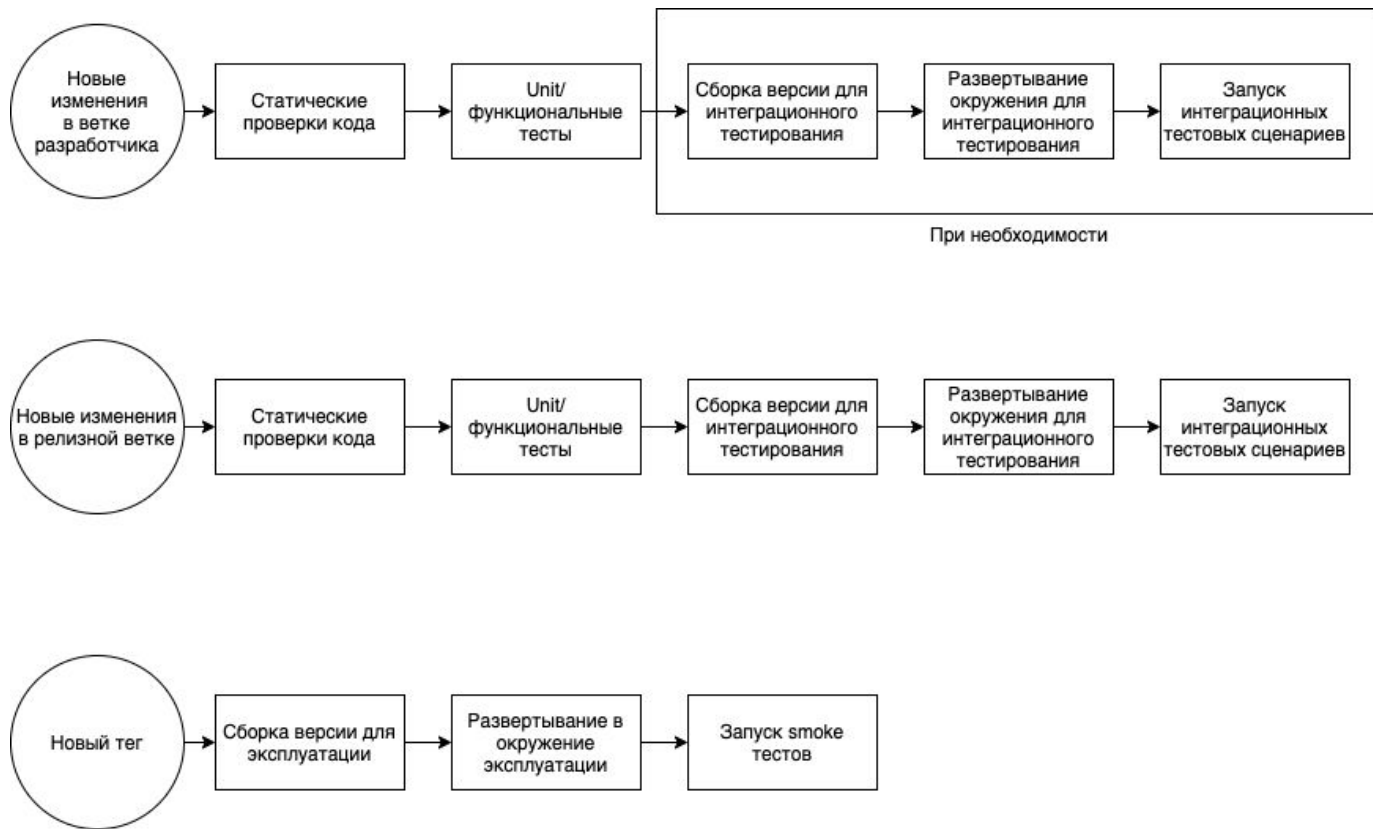
- Внесение нового функционала для проверки совместимости с системой;
- Быстрое реагирование на потребности рынка;
- Возможность подстраивания под изменение бизнес-стратегии;
- Низкое количество потенциальных ошибок.

Continuous Deployment





Continuous Deployment отвечает за то, чтобы **весь новый функционал** после тестирования **сразу же попал** в основную программу **без ручного вмешательства** инженеров DevOps.

Тот же Docker создан для Continuous Deployment. DevOps инженеры могут обновлять контейнеры и разворачивать их сразу на продакшене в автоматическом режиме, весь процесс может занять всего лишь несколько минут.


примерный вид процессов CI/CD в команде



DORA metrics

Software delivery performance metric	Elite	High	Medium	Low
 Deployment frequency For the primary application or service you work on, how often does your organization deploy code to production or release it to end users?	On-demand (multiple deploys per day)	Between once per week and once per month	Between once per month and once every 6 months	Fewer than once per six months
 Lead time for changes For the primary application or service you work on, what is your lead time for changes (i.e., how long does it take to go from code committed to code successfully running in production)?	Less than one hour	Between one day and one week	Between one month and six months	More than six months
 Time to restore service For the primary application or service you work on, how long does it generally take to restore service when a service incident or a defect that impacts users occurs (e.g., unplanned outage or service impairment)?	Less than one hour	Less than one day	Between one day and one week	More than six months
 Change failure rate For the primary application or service you work on, what percentage of changes to production or released to users result in degraded service (e.g., lead to service impairment or service outage) and subsequently require remediation (e.g., require a hotfix, rollback, fix forward, patch)?	0%-15%	16%-30%	16%-30%	16%-30%

Tools - Team City

 Projects Changes Agents 1515 Queue 2405 Architect ?

Jump to...

Matrix

Sign-Code-RunnerTemplate

Notify slack duty assignee

Trunk

Compile

BuildDist +16

BuildDist (tar.gz) +16

BuildServer merge +16

BuildServer scheduled merge +100

MasterBuild (BuildDist + All Tests)

All Tests

Open API Diff

Publish Zion Plugin to repository

Publish nightly +100

Publish Open API Release

Sample plugin build +4

BuildDist (docker) +100

Publish Integration Tests Artifacts +4

Tag Sources +100

Integration Tests

DSL Converters Tests

DSL UI Tests

IntegrationBuild (HSQLDB Incre... 1

IntegrationBuild (Security)

IntegrationBuild (MS SQL) +13

IntegrationBuild (MySQL/... 1

IntegrationBuild (MySQL/Docke... 3

IntegrationBuild (MariaDB/Docker) +2

IntegrationBuild (Oracle/Docker) +2

Hide

Matrix / Trunk

Integration Tests ★

<Active branches> 10 Info items

Overview More

Builds Trends

DSL Converters Tests ★

#11881 master Tests passed: 14, ignor... Trinity:1 teamcity-linu... 0-506 an hour ago 10m:06s Run ...

DSL UI Tests ★

#10811 master Tests passed: 111 Trinity:1 teamcity-linu... 0-498 40 minutes ago 35m:14s Run ...

IntegrationBuild (HSQLDB Incremental) ★

#28192 master Tests passed: 746, igno... Oracle:1 teamcity-win... 42-87 2h:15m left Run ...

#28191 master Tests passed: 6430, ign... Architect:1 teamcity-win... 0-176 1h:30m left Run ...

#28190 master Tests failed: 1 (1 new), ... Trinity:1 teamcity-win... 0-170 44m:47s left Run ...

#28188 master Tests failed: 3 (3 new), ... Oracle:1 teamcity-win... 42-89 16m:26s left Run ...

#28187 master Tests failed: 1 (1 new), ... Architect:1 teamcity-win... 0-172 4m:24s left Run ...

#28185 master Tests failed: 3 (3 new), ... Architect:2 teamcity-win... 0-174 a few seconds ago 2h:53m Run ...

#28151 user_id_sequence Tests failed: 29 (27 ne... Neo:1 teamcity-win... 0-175 4 days ago 2h:41m Run ...

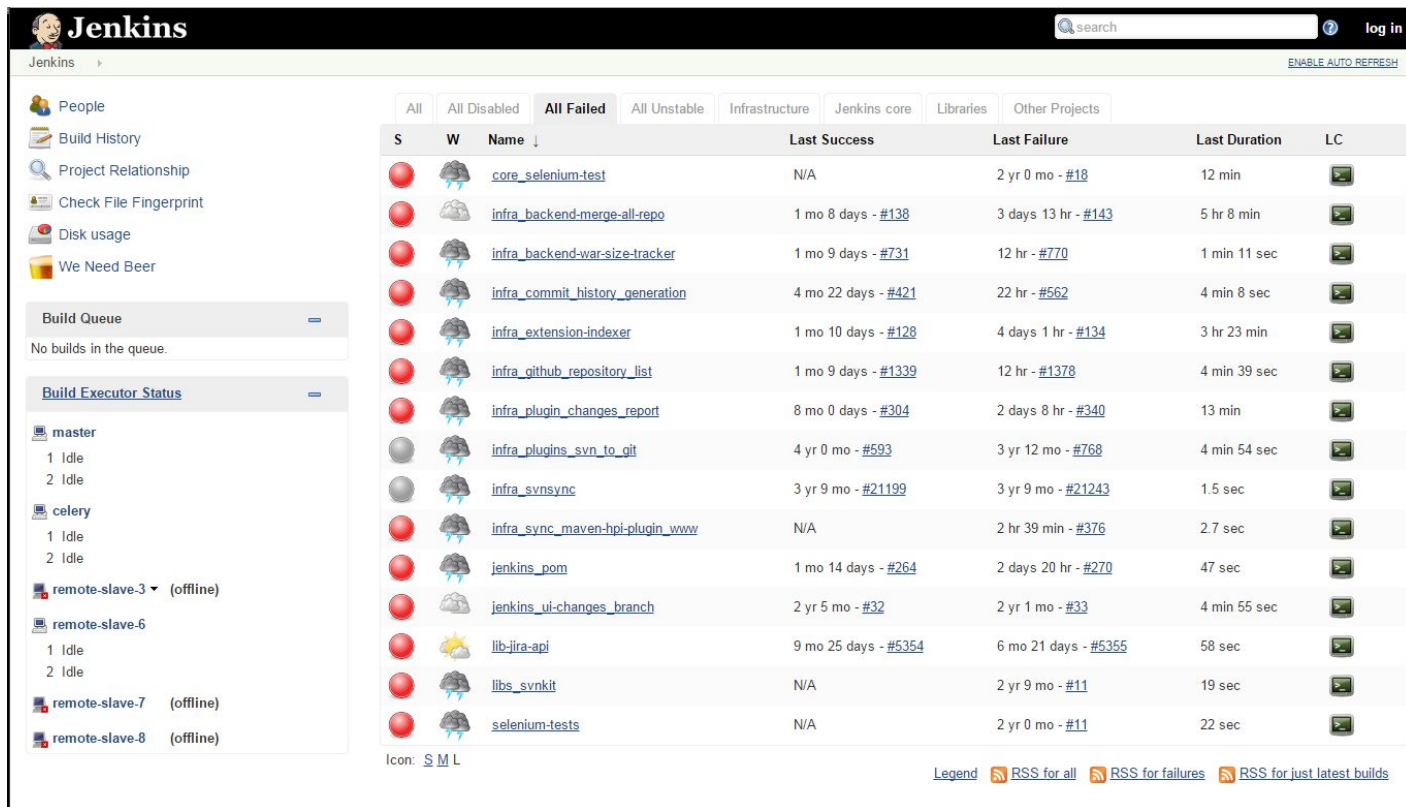
#28137 whiterabbit/master Tests failed: 1 (1 new), ... Smith:3 teamcity-win... 0-165 4 days ago 3h:21m Run ...

IntegrationBuild (Security) ★

#21053 master Tests passed: 331 Trinity:1 teamcity-win... 42-87 an hour ago 33m:38s Run ...

#21022 user_id_sequence Tests passed: 331 Neo:1 teamcity-linu... 0-503 4 days ago 23m:46s Run ...

Tools - Jenkins



The screenshot displays the Jenkins web interface. The top navigation bar includes the Jenkins logo, a search bar, and a 'log in' link. Below the navigation bar, the left sidebar contains links for 'People', 'Build History', 'Project Relationship', 'Check File Fingerprint', 'Disk usage', and 'We Need Beer'. The main content area shows a table of build results, filtered by 'All Failed'. The table columns are 'S' (Status), 'W' (Weather icon), 'Name', 'Last Success', 'Last Failure', 'Last Duration', and 'LC' (Link icon). The table lists 18 builds, all of which are failed. The builds are categorized by project: 'core_selenium-test', 'infra_backend-merge-all-repo', 'infra_backend-war-size-tracker', 'infra_commit_history_generation', 'infra_extension-indexer', 'infra_github_repository_list', 'infra_plugin_changes_report', 'infra_plugins_svn_to_git', 'infra_svnsync', 'infra_sync_maven-hpi-plugin_www', 'jenkins_pom', 'jenkins_ui-changes_branch', 'lib-jira-api', 'libs_svnkit', and 'selenium-tests'. The bottom of the page features a legend for the status icons and RSS feeds for all builds, failures, and latest builds.

Jenkins

search log in

Jenkins

ENABLE AUTO REFRESH

All All Disabled **All Failed** All Unstable Infrastructure Jenkins core Libraries Other Projects

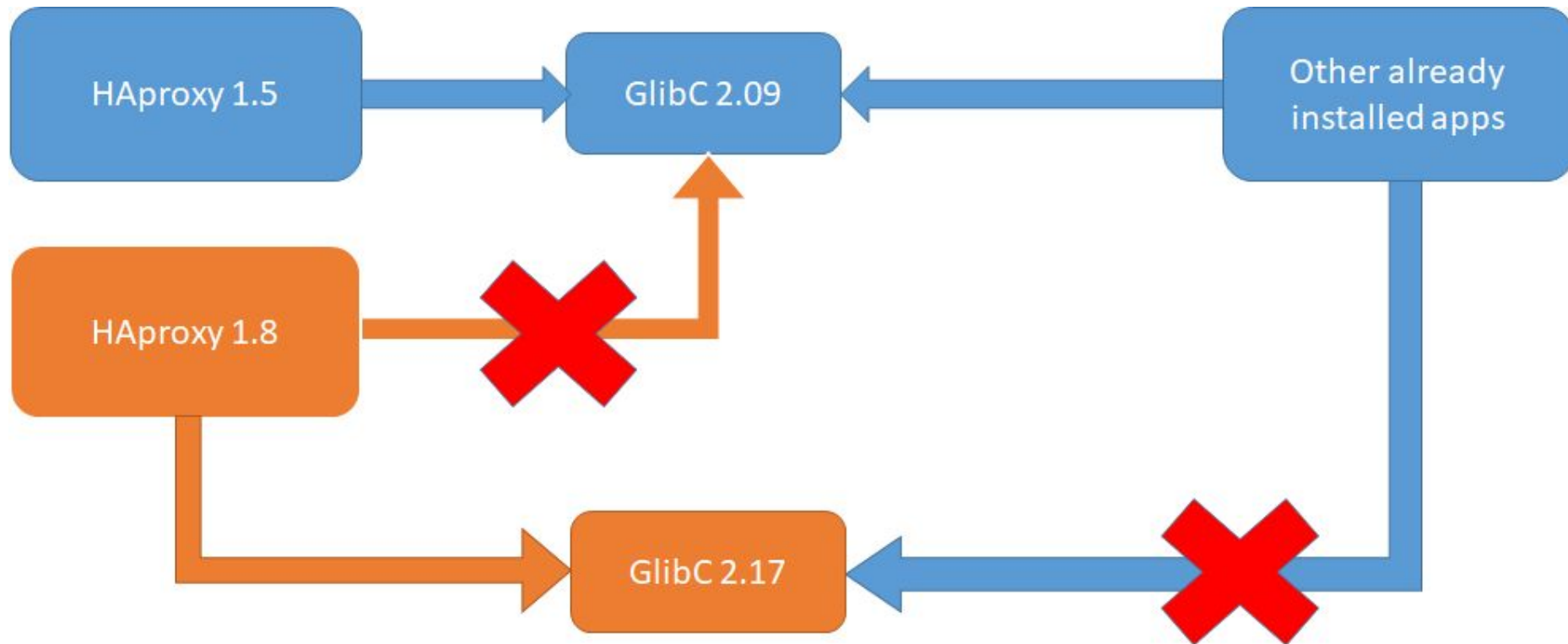
S	W	Name	Last Success	Last Failure	Last Duration	LC
		core_selenium-test	N/A	2 yr 0 mo - #18	12 min	
		infra_backend-merge-all-repo	1 mo 8 days - #138	3 days 13 hr - #143	5 hr 8 min	
		infra_backend-war-size-tracker	1 mo 9 days - #731	12 hr - #770	1 min 11 sec	
		infra_commit_history_generation	4 mo 22 days - #421	22 hr - #562	4 min 8 sec	
		infra_extension-indexer	1 mo 10 days - #128	4 days 1 hr - #134	3 hr 23 min	
		infra_github_repository_list	1 mo 9 days - #1339	12 hr - #1378	4 min 39 sec	
		infra_plugin_changes_report	8 mo 0 days - #304	2 days 8 hr - #340	13 min	
		infra_plugins_svn_to_git	4 yr 0 mo - #593	3 yr 12 mo - #768	4 min 54 sec	
		infra_svnsync	3 yr 9 mo - #21199	3 yr 9 mo - #21243	1.5 sec	
		infra_sync_maven-hpi-plugin_www	N/A	2 hr 39 min - #376	2.7 sec	
		jenkins_pom	1 mo 14 days - #264	2 days 20 hr - #270	47 sec	
		jenkins_ui-changes_branch	2 yr 5 mo - #32	2 yr 1 mo - #33	4 min 55 sec	
		lib-jira-api	9 mo 25 days - #5354	6 mo 21 days - #5355	58 sec	
		libs_svnkit	N/A	2 yr 9 mo - #11	19 sec	
		selenium-tests	N/A	2 yr 0 mo - #11	22 sec	

Icon: [S](#) [M](#) [L](#)

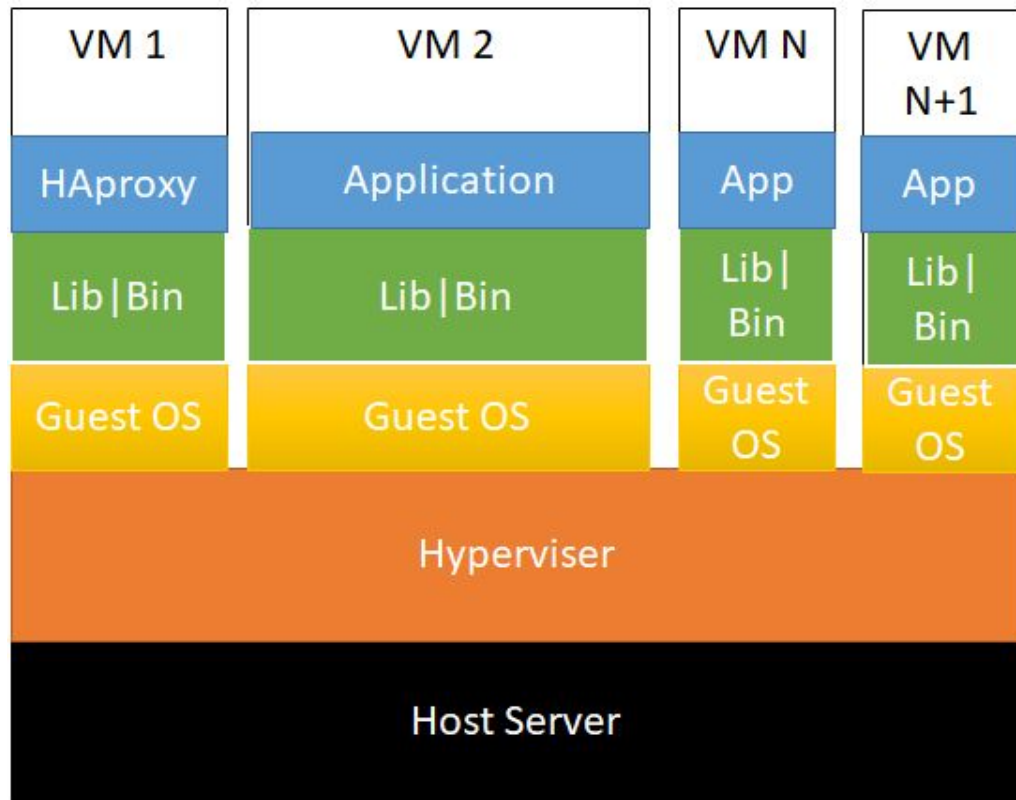
Legend RSS for all RSS for failures RSS for just latest builds

Docker

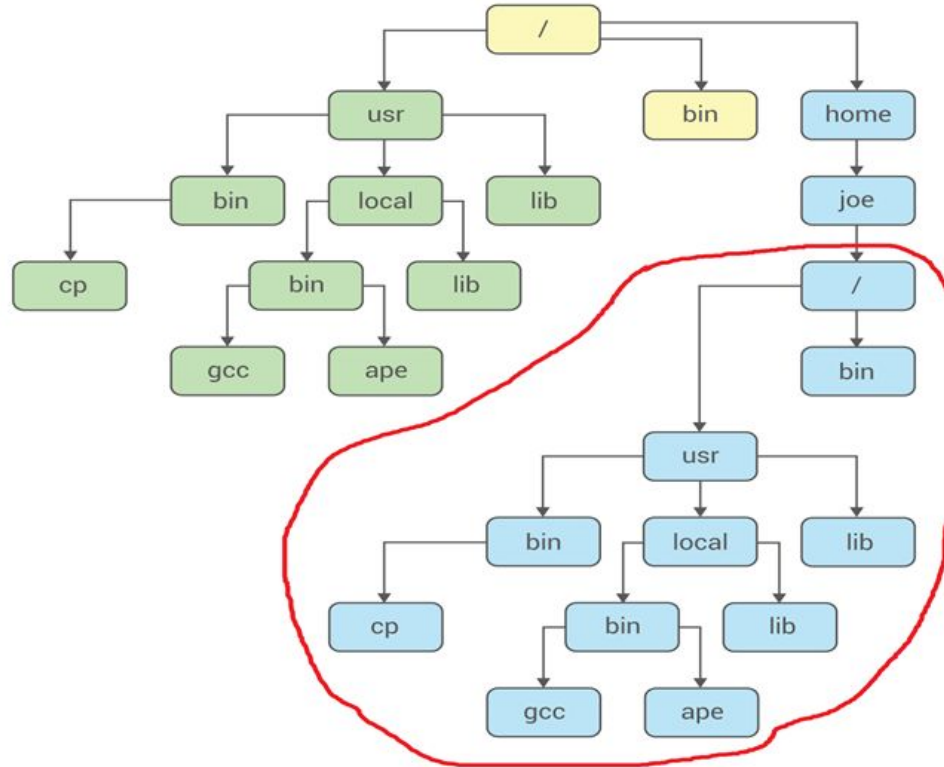
Предпосылки к появлению



Виртуализация

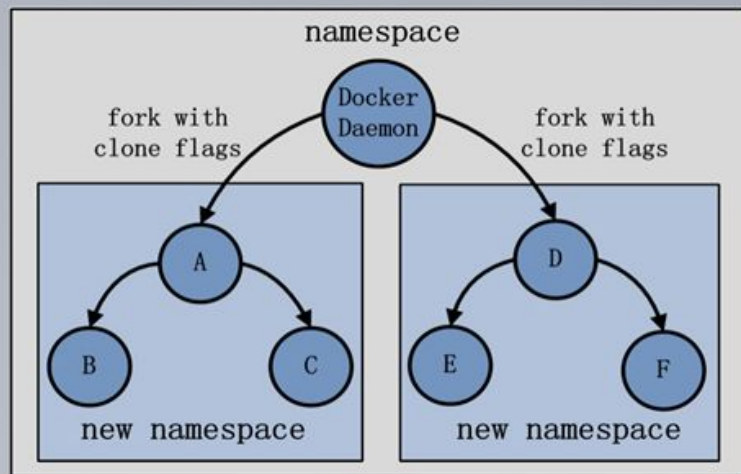


Processes limitation| chroot

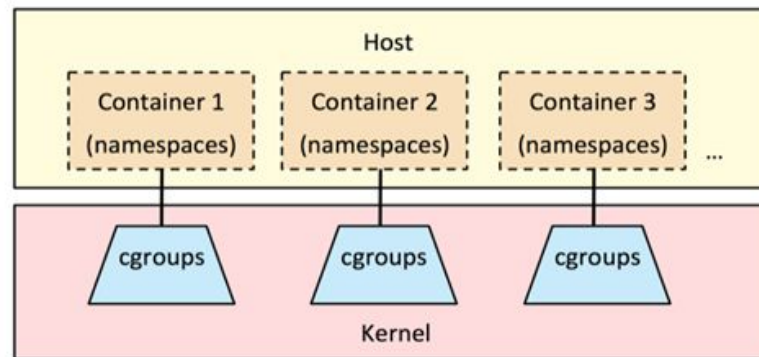


Container =

namespace+Cgroup



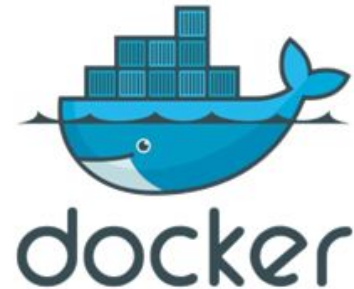
Container = combination of namespaces & cgroups



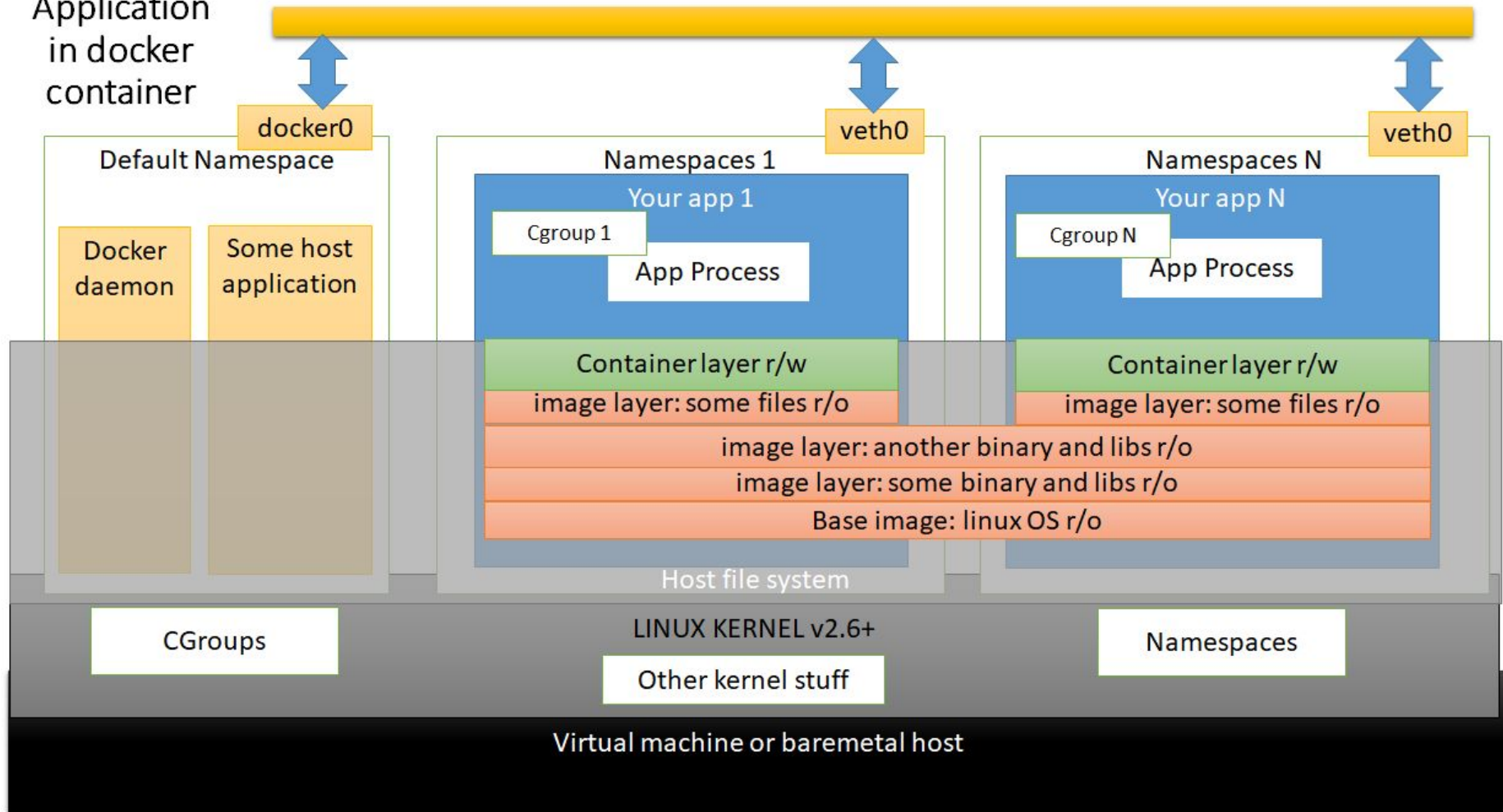
UTS、IPC、Mount、PID、NetWork、User

Почему все-таки Docker?

1. Simplicity
2. Docker paradigm “one container – all inclusive for one application”
3. Layered filesystem
4. Manage containers infrastructure



Application
in docker
container



Containers management systems



kubernetes



AWS ECS



**Google Container
Engine**

FIN