KOÇ UNIVERSITY

COMP 350
Selected Topics - Introduction to DevOps

Lecture 1
Introduction

Hakan Ayral

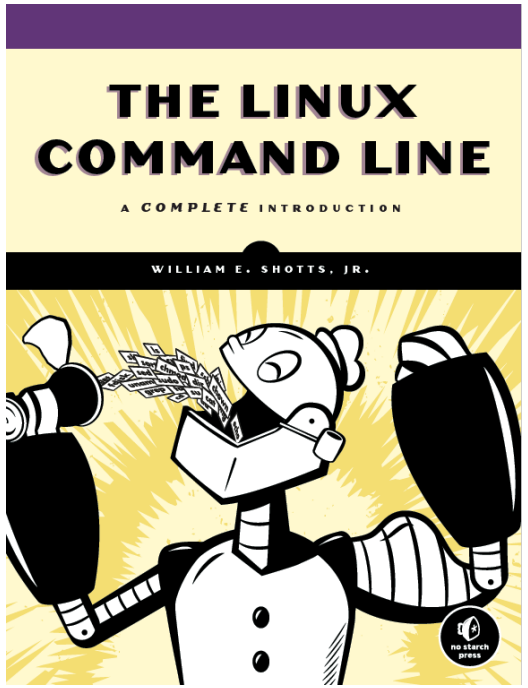
Administrivia

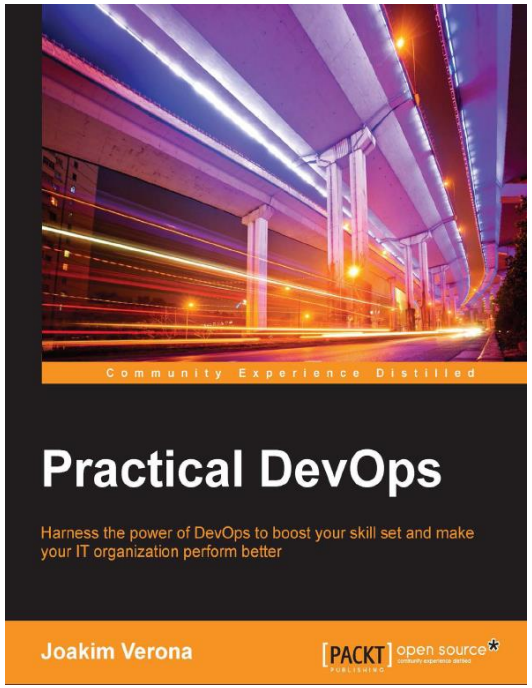
- Textbooks
  - first half of the course is about Linux system administration:
    - The Linux Command Line, William E. Shotts, No Starch Press
  - on the second half we will focus on actual DevOps
    - mainly online references (git, docker,…) and lecture notes
    - but also Practical DevOps, Joakim Verona, Packt Publishing

- Office Hours: Wednesday 14:20-15:20 or any other time by appointment
- Participation and attendance is 10% of course grade

- You are expected to do some reading and practicing

- Grading
  - 2x Projects 30%
  - Midterm Exam 25%
  - Final Exam 35%
  - Participation and attendance 10%





Syllabus

Meeting Times	Subject
1	Linux Operating System, Kernel and kernel modules, Kernel boot parameters, Boot process of Linux. Initial ram disk, systemd/initStorage Management and File Systems on Linux. • Common file systems • Access Control for files and directories, permissions
2	Devices, Block devices, mounting block devices for storage • Create and manage basic volumes, spanning volumes, redundancy, volume management and RAID configurations • Swap partition, virtual memory, Swap vs ZSwap vs zRAM
3	What's a process? How to manage processes in Linux? • Monitoring processes • StdIO, StdErr, I/O Redirection, Pipes, Filtering with grep
4	Shell as a process, CLI interaction and scripting • Terminal multiplexers • Secure shell remote access • User management, access control and System Users • cron • System Logs
5	Package management systems and scripting/programming ecosystems: • Apt, yum, deb, rpm • Python, pip, conda • Nodejs and npm • Ruby and gemsBuild tools: make, cmake, GNU Build System (aka autotools), Apache Ant/Maven, Gradle
6	Code repositories, Versioning, Git and github in depth: repo, branch, commit, clone, fast forward, rebase, cherry picking
7	Container systems, virtualization and scaling: • Quick look at the beginnings: Chroot jail • Virtualization • Containerization • Docker, docker file, docker compose
8	Networking in virtualized/containerized environments • NAT, Port forwarding, Routing, Packet filtering
9	Container Orchestration: Docker swarm, Kubernetes, Amazon Elastic Container Service (ECS) • Load balancing for scaling

Syllabus (cont'd)

10	Microservice Architecture: philosophy, building APIs, REST APIs, SOAP, gRPC, XML, YAML
11	Cloud service providers: Amazon Web Services, Microsoft Azure, Google Cloud PlatformServerless: AWS Lambda, Google Cloud Functions, Azure Serverless
12	Infrastructure as Code (IaC): Terraform, OpenTF, AWS CloudFormation,
13	Continuous Integration (CI)/Continuous Delivery (CD) : • Gitlab CI, Azure pipelines, AWS Code pipeline • Jenkins, Travis, Teamcity, Circle CI
14	Configuration Management: Chef, Puppet, Salt, Ansible, Terraform, CFEngine

What are we interested in?

- Linux Operating System
- System Administration
- Storage Management and file systems
- Inter-process communications

System Admin

- Computer communications and Networking

Network Admin

- Well-structured documents
- Regular expressions
- Computer Languages
- Scripting/Programming platforms and related package management systems

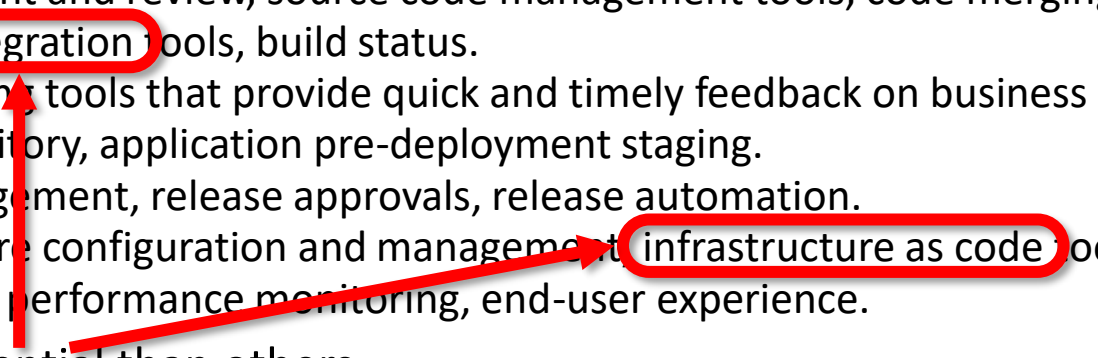
Developer

- Container systems, virtualization and scaling for deployment, and production environment
- Dev Ops best practices
- Code repositories, Versioning, automated build and Continuous Integration (CI)
- Compile arbitrary software directly from source

Dev Ops

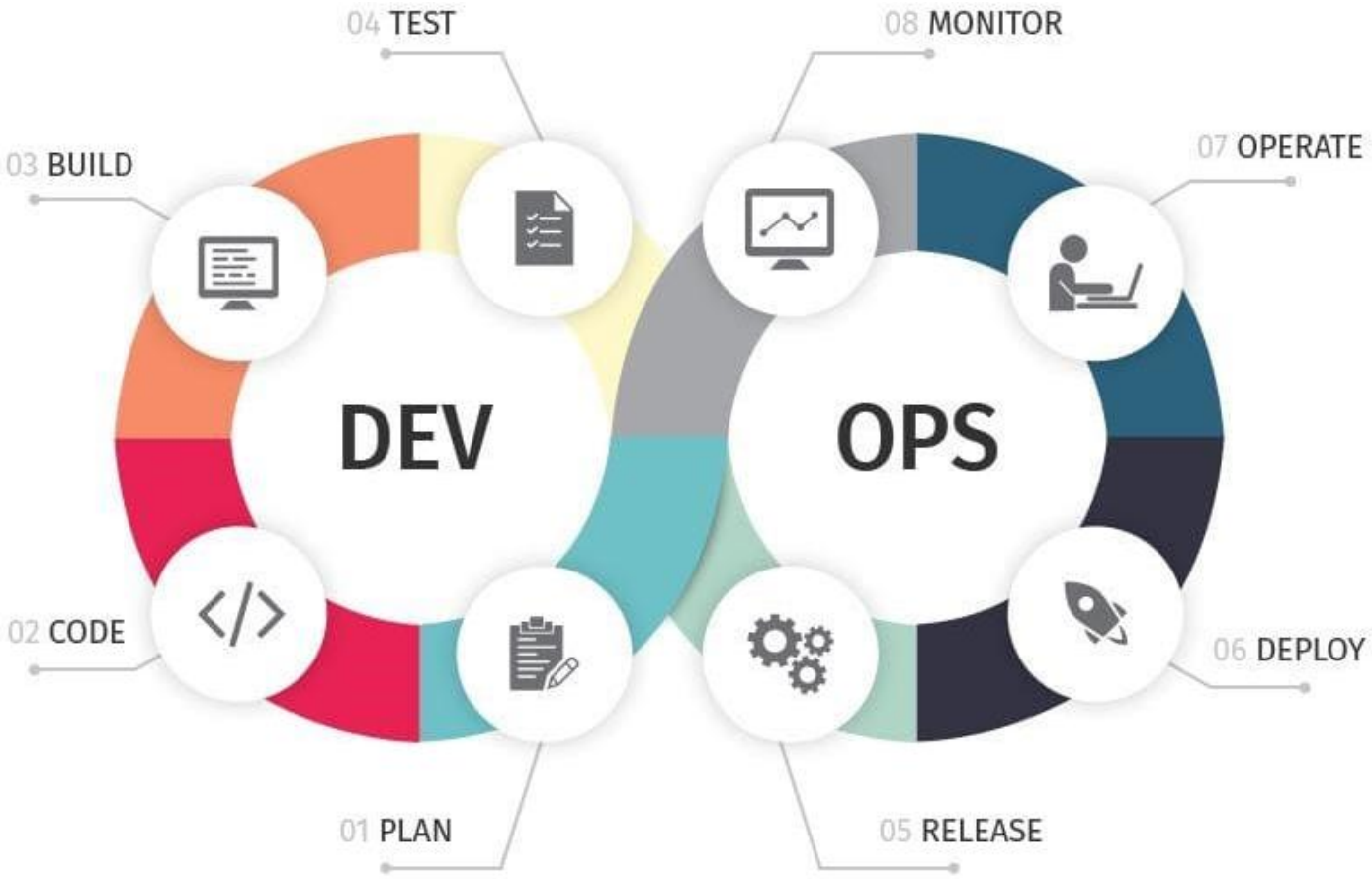
What is DevOps?

- DevOps is a set of practices that combines software development (Dev) and IT operations (Ops).
- It aims to shorten the systems development life cycle and provide continuous delivery with high software quality.
- DevOps is complementary with Agile software development.
- As DevOps is intended to be a cross-functional mode of working, those who practice the methodology use different sets of tools—referred to as "toolchains"—rather than a single one.
- These toolchains are expected to fit into one or more of the following categories, reflective of key aspects of the development and delivery process:
    1. **Coding** – code development and review, source code management tools, code merging.
    2. **Building** – continuous integration tools, build status.
    3. **Testing** – continuous testing tools that provide quick and timely feedback on business risks.
    4. **Packaging** – artifact repository, application pre-deployment staging.
    5. **Releasing** – change management, release approvals, release automation.
    6. **Configuring** – infrastructure configuration and management, infrastructure as code tools.
    7. **Monitoring** – applications performance monitoring, end-user experience.
- Some categories are more essential than others.



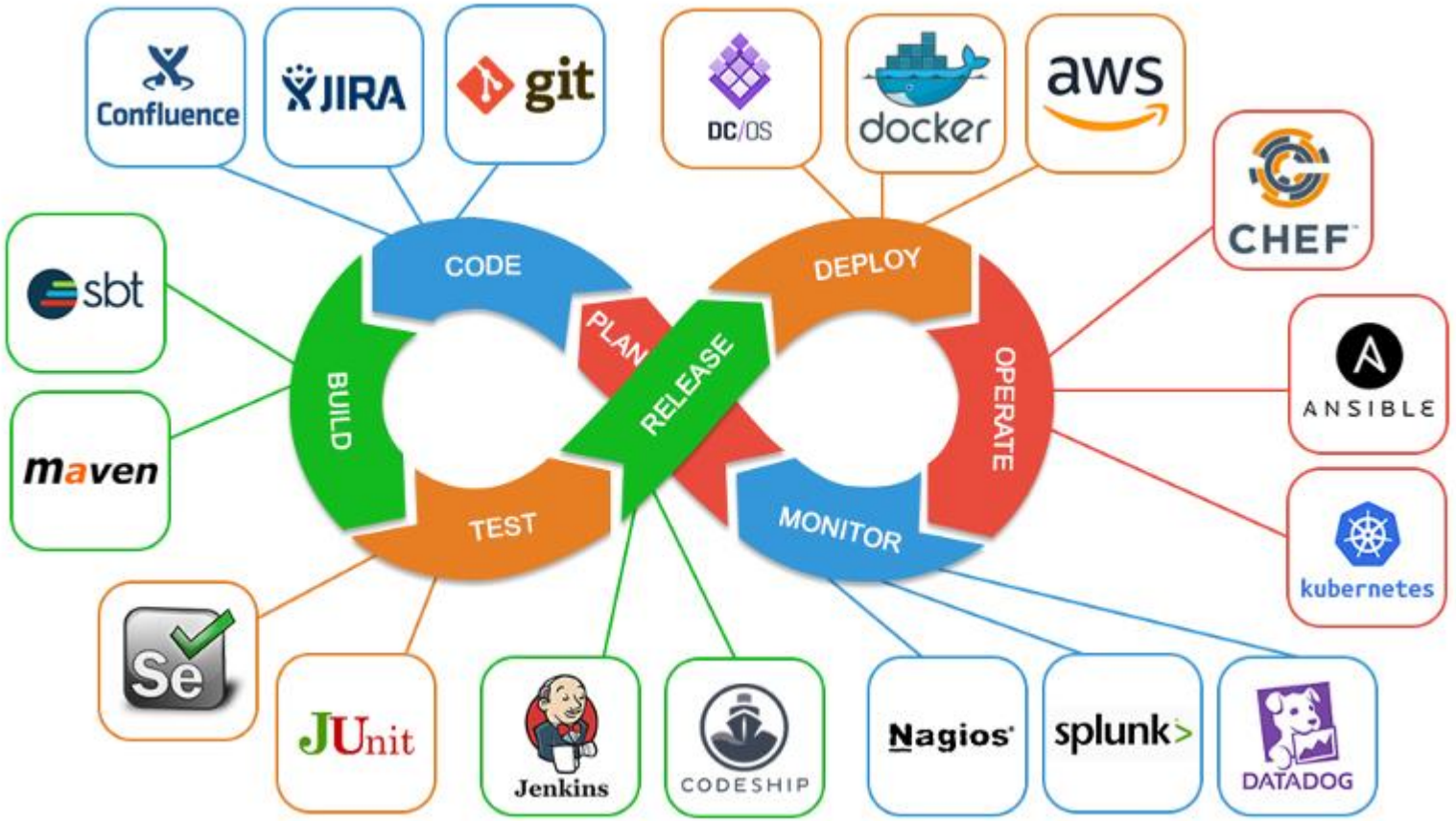
DevOps Cycle

- DevOps automates the **building** of both infrastructure and applications.
- DevOps applies a series of automated **tests** to both infrastructure and applications (unit, integration, performance etc) to prove they meet their functional and non-functional requirements.
- When proven to be fit to release, DevOps automates the **deployment** to end-users.
- DevOps primarily applies to businesses that build or assemble their own software, versus just using pre-built software.
- Forsgren et al. found that IT performance is strongly correlated with DevOps practices like source code management and continuous delivery. (Nicole Forsgren; Gene Kim; Nigel Kersten; Jez Humble (2014). "2014 State of DevOps Report")



Some tools used on different phases of the cycle

- If you are highly proficient on any 3 of the tools below you can start working on a big company tomorrow.



What is the goal of DevOps?

- The goals of DevOps span the entire delivery pipeline. They include:
  - Improved deployment frequency
  - Faster time to market
  - Lower failure rate of new releases
  - Shortened lead time between fixes
  - Faster mean time to recovery (in the event of a new release crashing or otherwise disabling the current system)

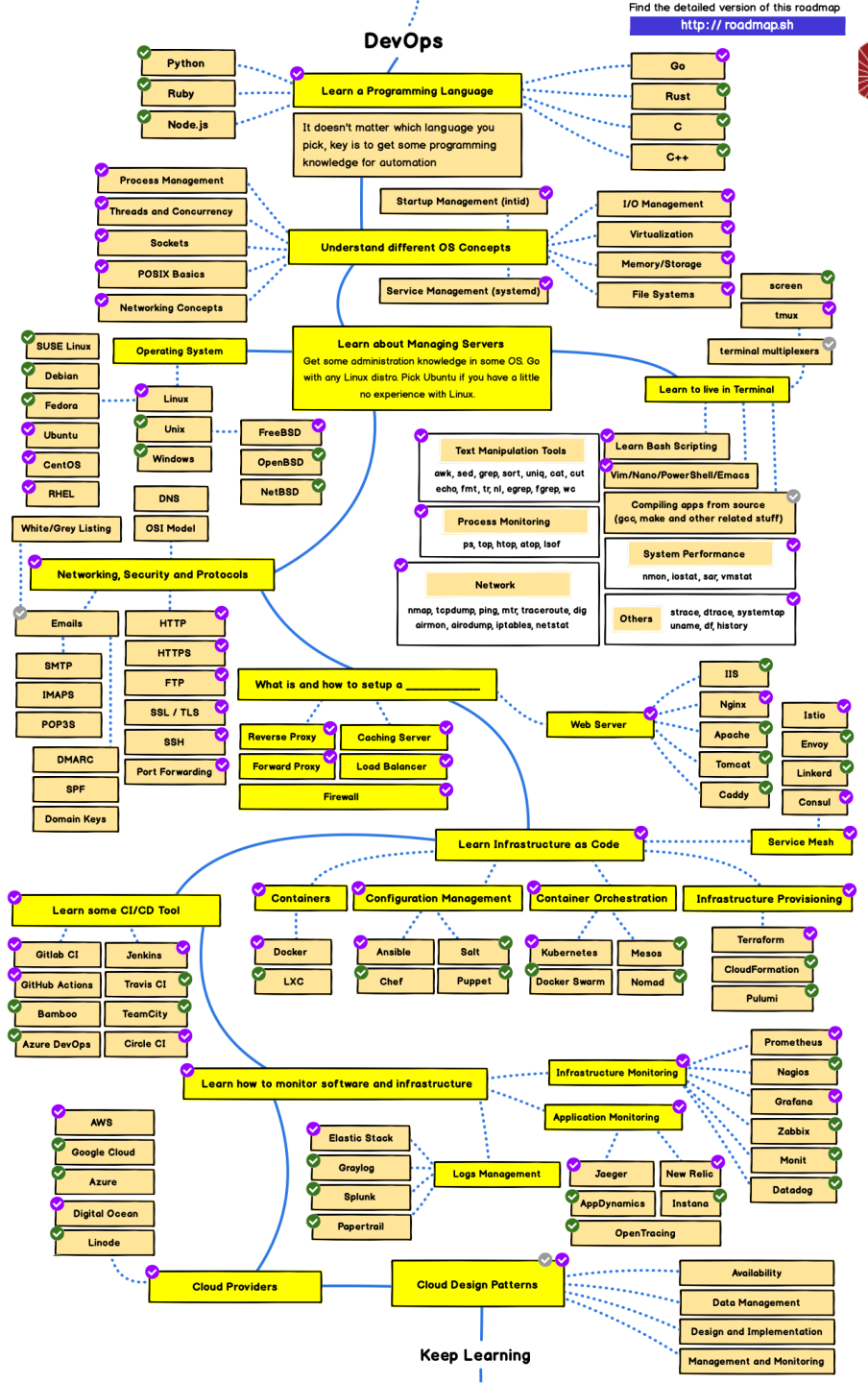
- DevOps as a job title: While DevOps describes an approach to work rather than a distinct role (like system administrator), job advertisements are increasingly using terms like "DevOps Engineer".

Aim of the course

- We will spend the first half of the term for familiarization with fundamental system administration tasks for enterprise environment.
- The purpose of this course is **not** to make a full-time **system admin** out of you.
- The purpose is to introduce you the DevOps tools and techniques needed for a **developer** to manage and orchestrate the IT operations needed during the steps of the DevOps cycle.
  - DevOps Engineer = Developer + some Sys Admin skills + extra skills specific to DevOps
- Obviously the length of a semester is not long enough to cover every tool and technique in detail.
- You are expected to do a lot of reading and hands-on practice.

An Unofficial Roadmap

1. Learn a Programming Language
2. Understand different OS concepts
3. Learn to Live in terminal
4. Networking and Security
5. What is … and how to setup? (i.e. app servers)
6. Learn Infrastructure as code
7. Learn some Continuous Integration and Delivery (CI/CD) tools
8. Learn to monitor software and infrastructure
9. Learn about Cloud Providers



The Linux Command Line

PART 1 - LEARNING THE SHELL

- WHAT IS THE SHELL?
- NAVIGATION
- EXPLORING THE SYSTEM
- MANIPULATING FILES AND DIRECTORIES
- WORKING WITH COMMANDS
- REDIRECTION
- SEEING THE WORLD AS THE SHELL SEES IT
- ADVANCED KEYBOARD TRICKS
- PERMISSIONS
- PROCESSES

PART 2 - CONFIGURATION AND THE ENVIRONMENT

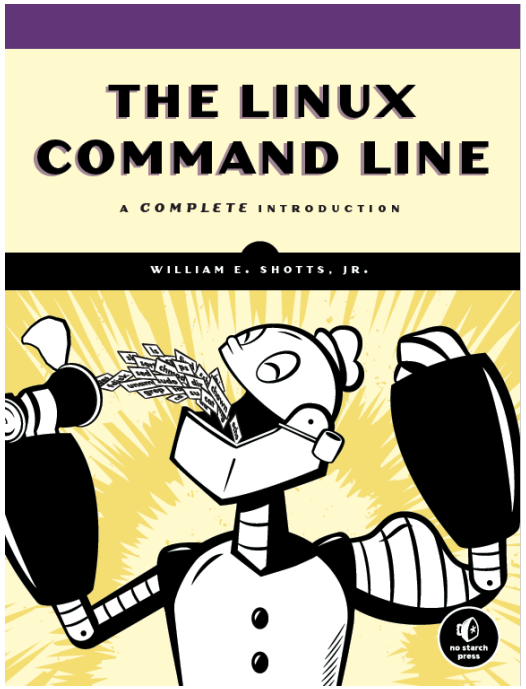
- THE ENVIRONMENT
- A GENTLE INTRODUCTION TO VI
- CUSTOMIZING THE PROMPT

PART 3 - COMMON TASKS AND ESSENTIAL TOOLS

- PACKAGE MANAGEMENT
- STORAGE MEDIA
- NETWORKING
- SEARCHING FOR FILES
- ARCHIVING AND BACKUP
- REGULAR EXPRESSIONS
- TEXT PROCESSING
- FORMATTING OUTPUT
- PRINTING
- COMPILING PROGRAMS

PART 4 - WRITING SHELL SCRIPTS

…



What is Linux?

- A family of open-source Unix-like operating systems.
- It is based on the Linux kernel (an operating system kernel first released on September 17, 1991, by Linus Torvalds)
- Linux is typically packaged in a Linux distribution.
- Distributions include the Linux kernel and supporting system software and libraries.
- Desktop Linux distributions include a windowing system such as X11 or Wayland, and a desktop environment such as GNOME or KDE Plasma.
- Distributions intended for servers may omit graphics altogether, or include a solution stack such as LAMP (=Linux+Apache+MySQL+Php).
- Linux is freely redistributable, therefore anyone may create a distribution for any purpose.
- Without the mastery on OS you can neither be a good developer, nor a good sys admin (not to mention DevOps)

How widespread is Linux? (aka why is this course based on linux?)

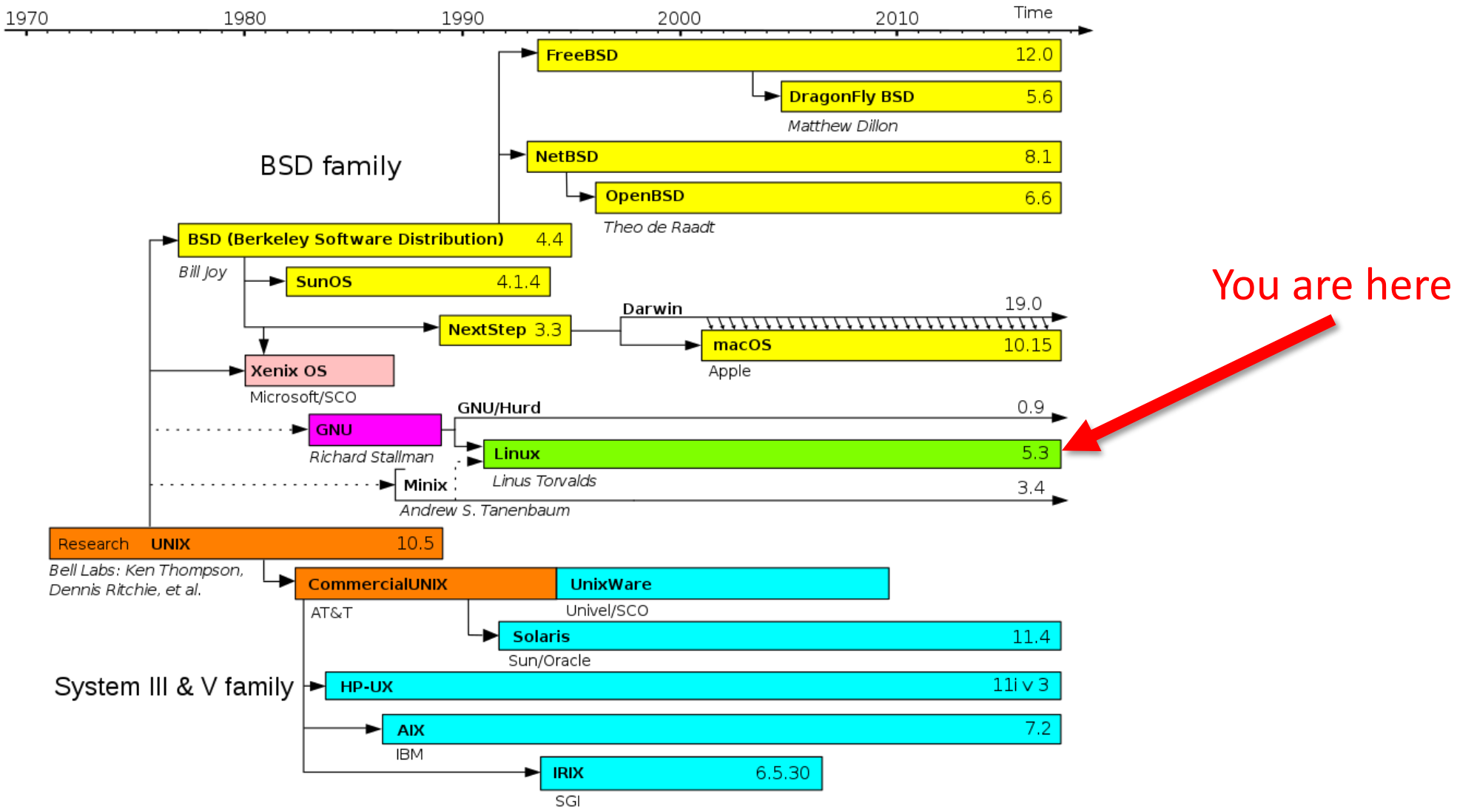
- Because of the dominance of the Linux-based Android on smartphones, Linux has the largest installed base of all general-purpose operating systems.
- But it is used by only around **2.3%** of desktop computers.
- Linux is the leading operating system on servers (over **96.4%** of the top 1 million web servers' OS is Linux)
- It is also the only OS used on TOP500 supercomputers (since November 2017)
- Linux also runs on embedded systems (→ devices whose OS is typically built into the firmware)
  - This includes routers, automation controls, smart home technology (e.g. Google Nest), televisions, automobiles (e.g. Tesla, Audi, Mercedes-Benz, Hyundai, and Toyota), digital video recorders, video game consoles, and smartwatches.
  - The Falcon 9 space launch vehicle's and the Dragon 2 space craft's avionics use a customized version of Linux.
- **90%** of all cloud infrastructure is powered by Linux including supercomputers and cloud providers.
- **74%** of all smartphones in the world are Linux-based. (due to Android)

How “Linux” is pronounced?

- /ˈlɪnʊks/
- Listen at https://upload.wikimedia.org/wikipedia/commons/0/03/Linus-linux.ogg
- or if available click the icon below:

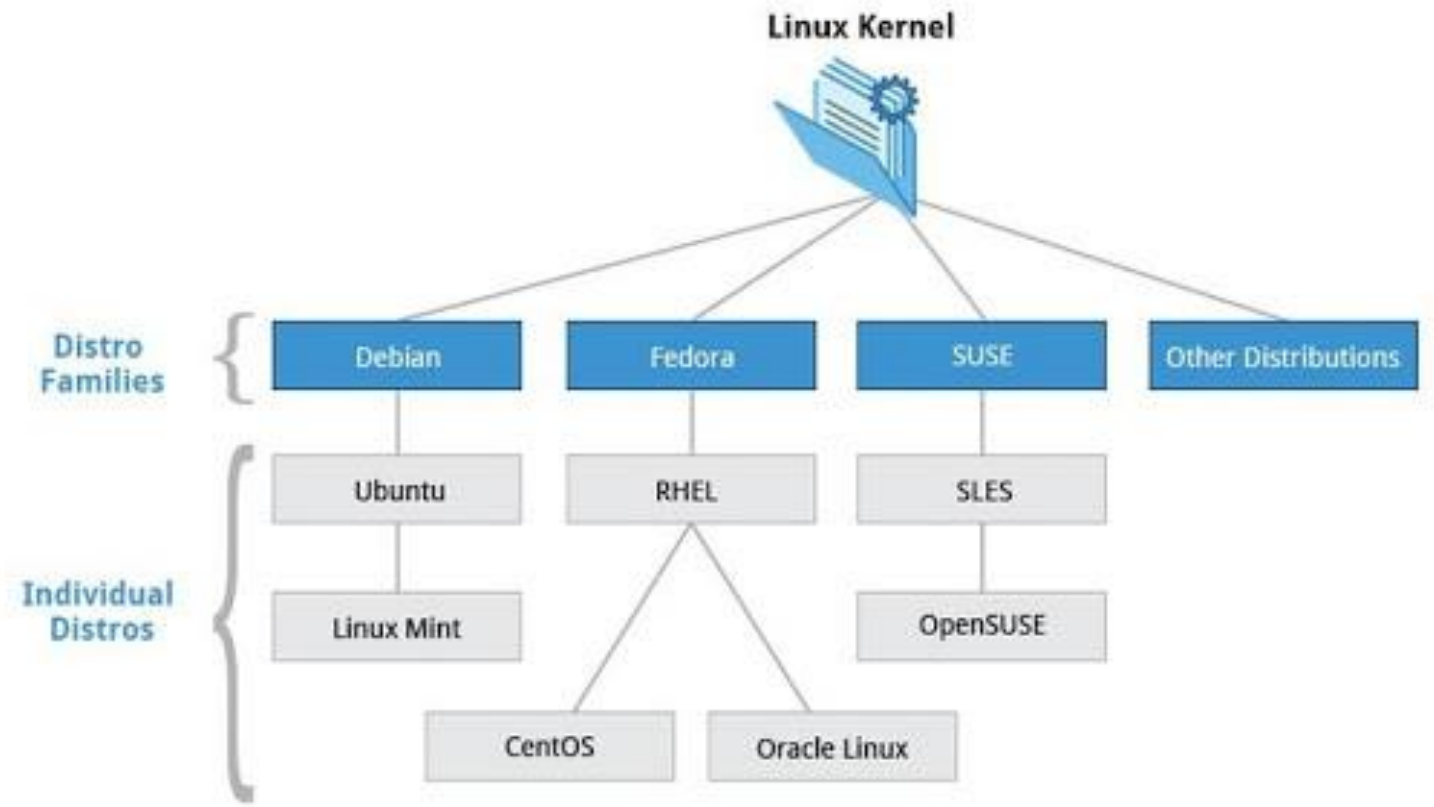
The name "Linux" is also used for a laundry detergent made by Swiss company Rösch.[190]

Before Linux Timeline



You are here

Some popular linux distribution families



Linux Distribution Timeline

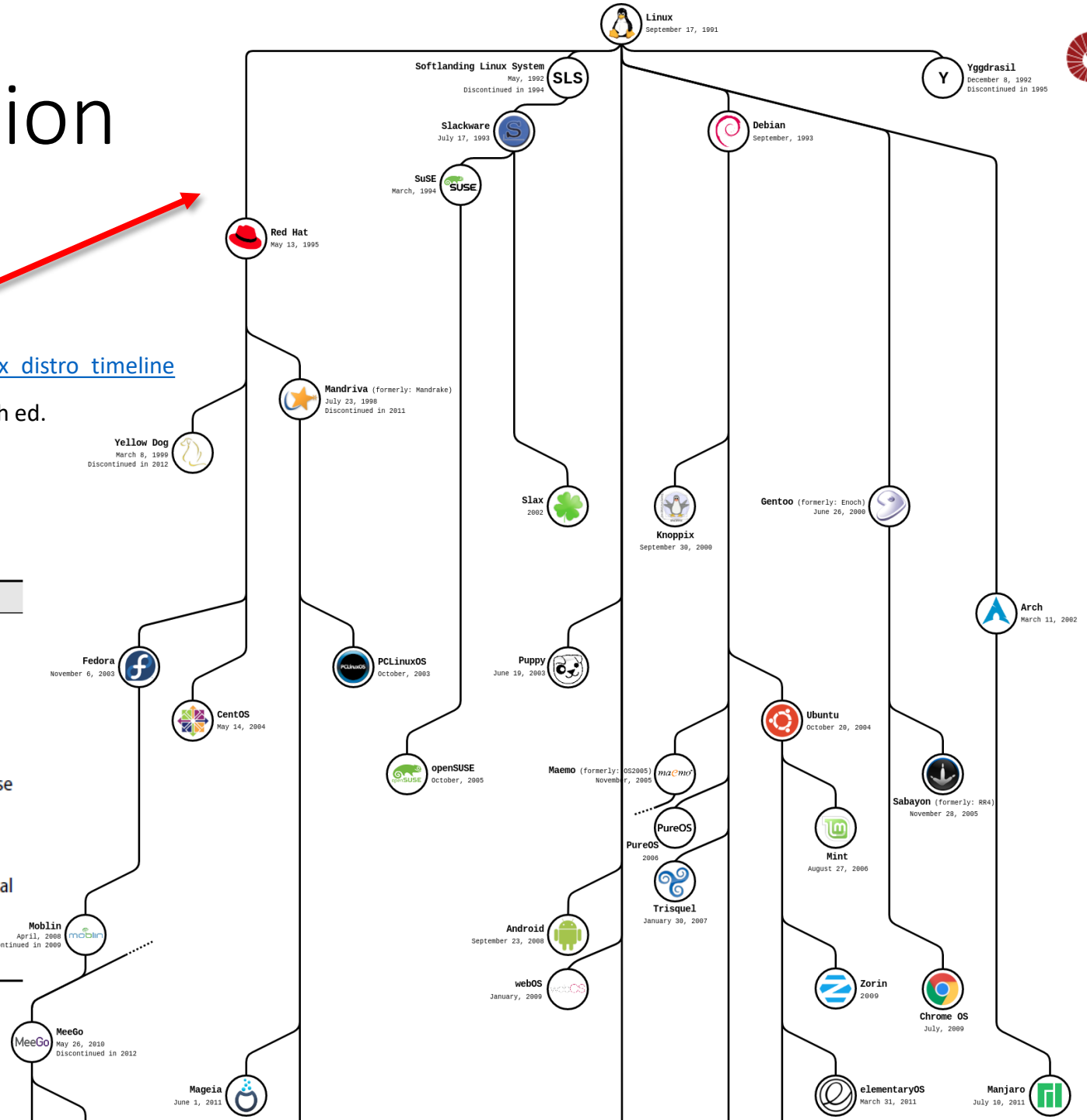
source:

https://commons.wikimedia.org/wiki/Category:Linux_distro_timeline

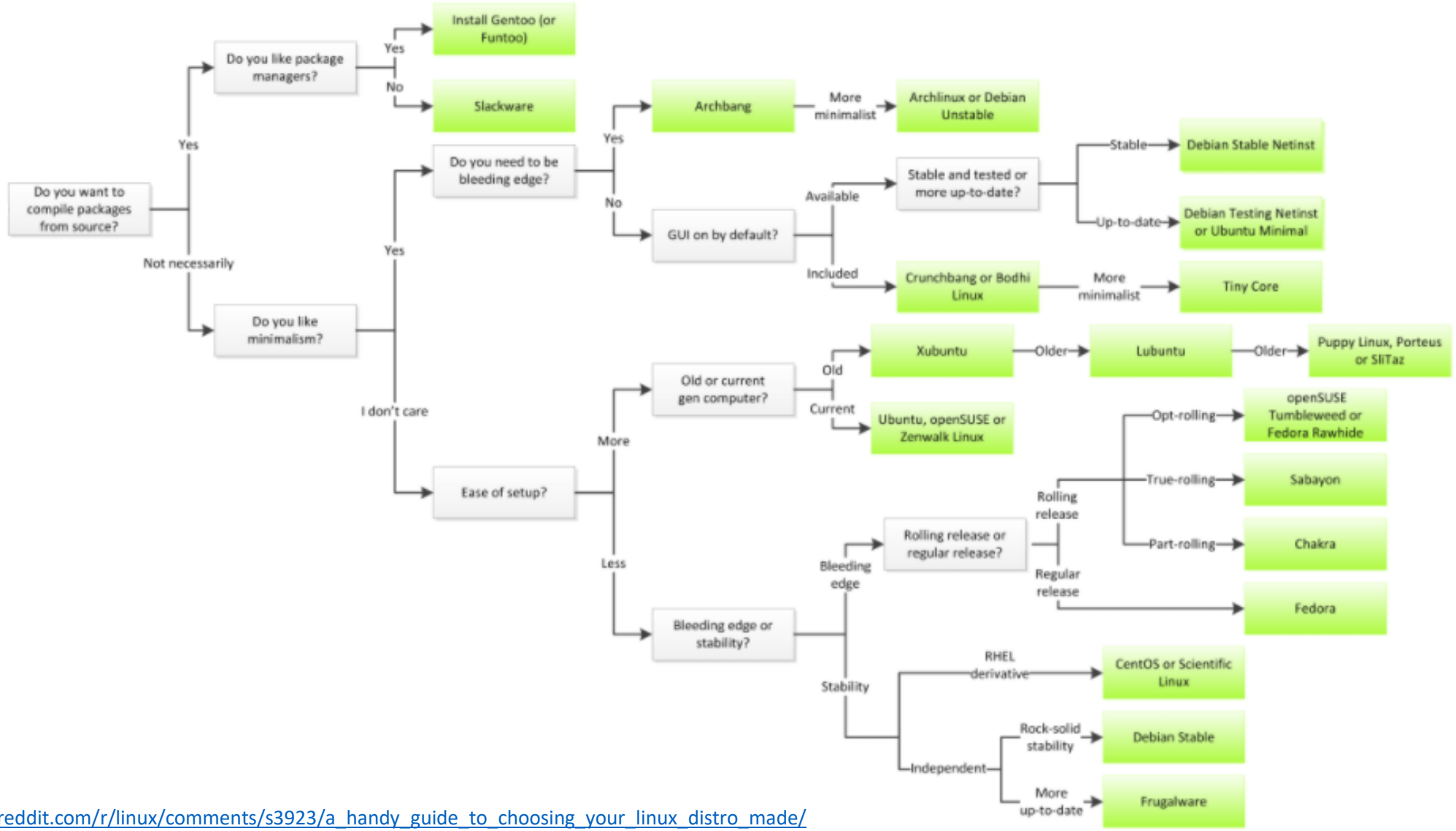
Unix and Linux System Administration Handbook, 4th ed.

Most popular general-purpose Linux distributions

Distribution	Web site	Comments
CentOS	centos.org	Free analog of Red Hat Enterprise
Debian	debian.org	Closest to GNU
Fedora	fedoraproject.org	De-corporatized Red Hat Linux
Gentoo	gentoo.org	Compile-it-yourself, optimized
Linux Mint	linuxmint.com	Ubuntu-based, elegant apps
Mandriva	mandriva.com	Long history, "easy to try"
openSUSE	opensuse.org	Free analog of SUSE Linux Enterprise
Oracle Enterprise Linux	oracle.com	Oracle-supported version of RHEL
PCLinuxOS	pclinuxos.com	Fork of Mandriva, KDE-oriented
Red Flag	redflag-linux.com	Chinese distro, similar to Red Hat
Red Hat Enterprise	redhat.com	Reliable, slow-changing, commercial
Slackware	slackware.com	Grizzled, long-surviving distro
SUSE Linux Enterprise	novell.com/linux	Strong in Europe, multilingual
Ubuntu	ubuntu.com	Cleaned-up version of Debian



Choose a distro, any distro



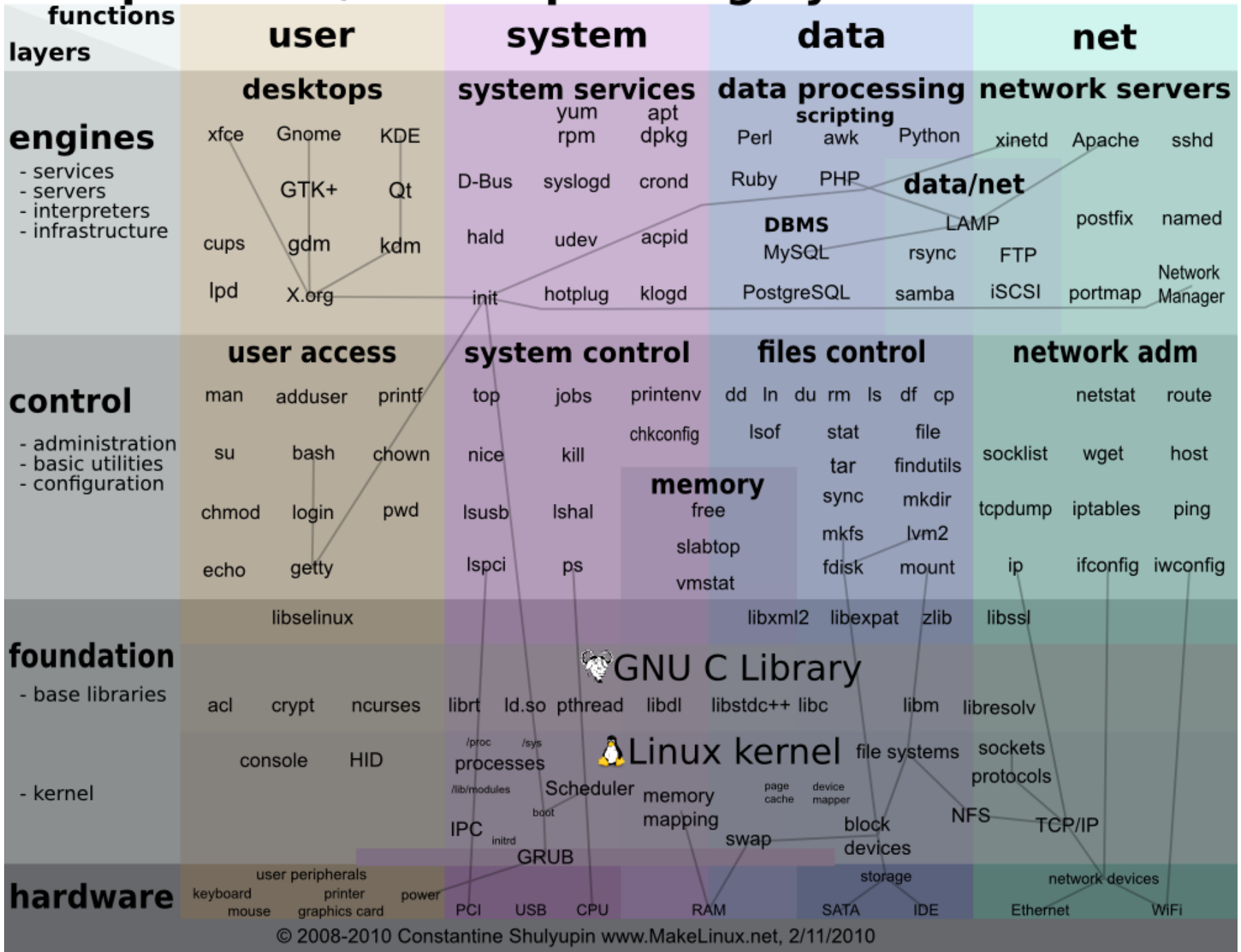
source: https://www.reddit.com/r/linux/comments/s3923/a_handy_guide_to_choosing_your_linux_distro_made/

Top 10 distros (for personal use) for 2020

POSITION	2020
1	MX Linux
2	Manjaro
3	Linux Mint
4	Ubuntu
5	Debian
6	Elementary OS
7	Solus
8	Zorin OS
9	Fedora
10	Deepin

source: Distrowatch

Map of GNU/Linux Operating System Internals



Practice

- Download and Install Oracle VirtualBox
- Download the latest Ubuntu or Linux Mint Distribution

- Create a virtual machine on VirtualBox
- Install the downloaded OS (do not run as live CD, install to virtual disk)
- Boot the OS
- Open a terminal
- Run a few commands like: ls, cd, mkdir, rm (be careful), man, ps, top, nano

note: you'll be using this virtual machine a lot during the rest of the course.