

Programming Env

Importance of Programming Environment
Management

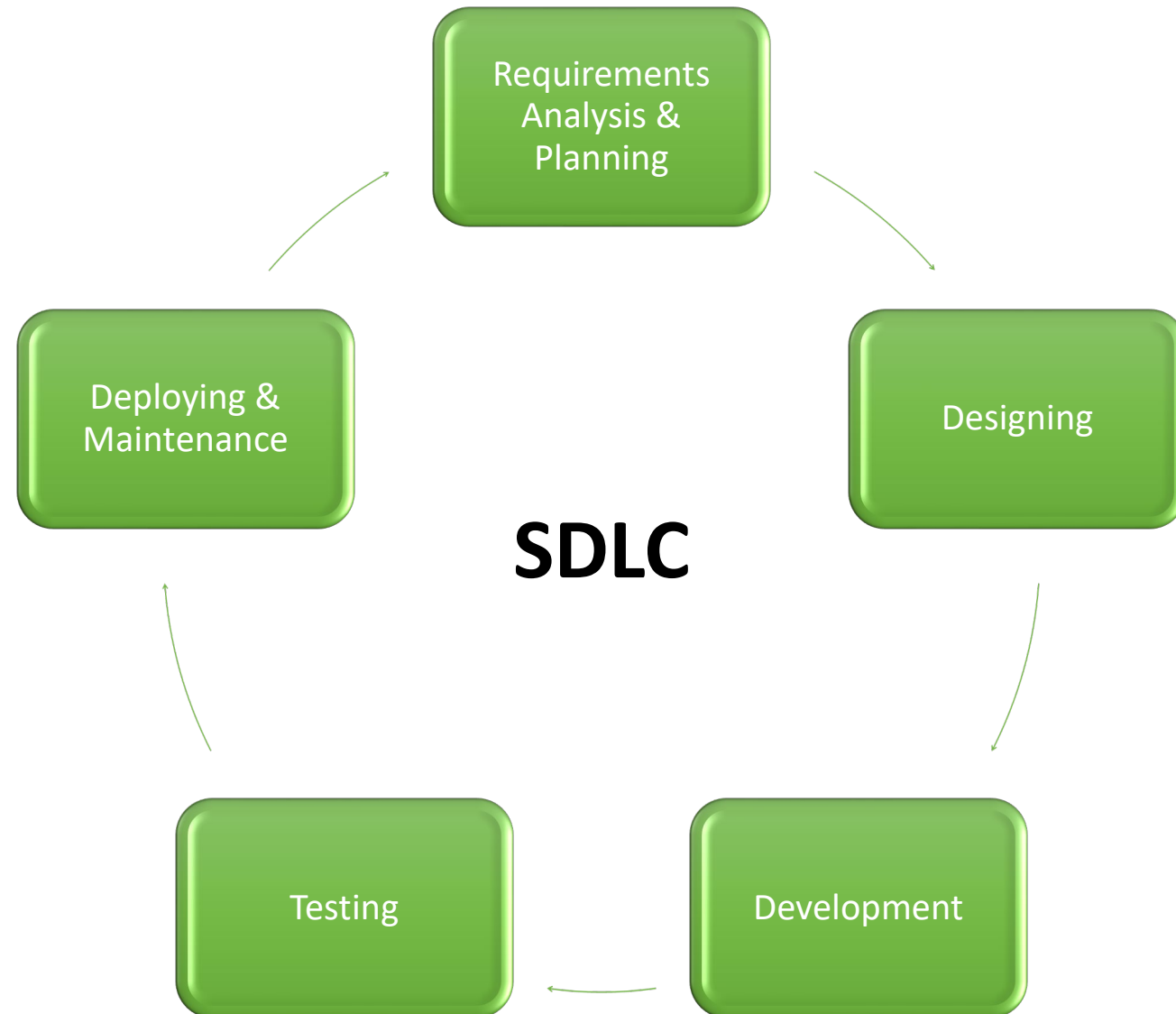
Outlines

- Introduction
- Software development life cycle
- Development process
- Version Control System (VCS)
- Git

I. Introduction

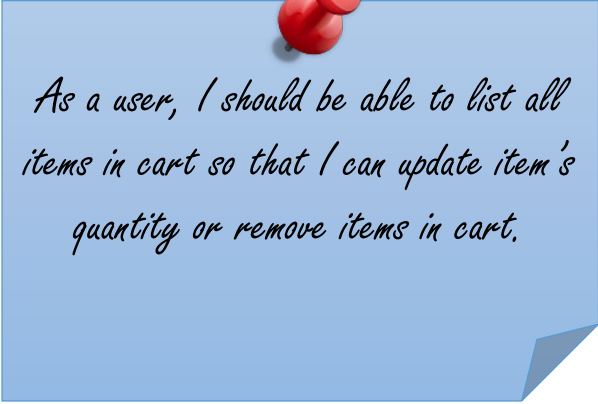
- **Programming:** the **mental process** of thinking up instructions to give to a machine (like a computer).
- **Environment:** the **circumstances, objects, or conditions** by which one is surrounded.
- ➔ **Programming Environment:** the **circumstances, objects, or conditions** by which the **programming** is surrounded.

II. Software Development Life Cycle (SDLC)



Requirements Analysis & Planning

- Statement of needs from clients or customers. It is problem that the outcome software can solve. Example:



As a user, I should be able to list all items in cart so that I can update item's quantity or remove items in cart.

- Analysis of the requirement is to find the way to solve it in digital way. Example:
 - Item in cart should have item name, size, unit price, quantity
 - Should display as rows and in each row should have option to change quantity and to remove the item from the cart
 - Should have link to see the cart content (list items in cart)

Requirements Analysis & Planning

- Planning focuses on listing activities in order and assign who will do it, when it start and when it end. Example:

No.	Task	Assignee	Duration	Start date	End date
1	Create menu item link to see cart content and return	Sok	1 h	02/Mar	02/Mar
2	Design cart content (list items in cart)	Sambath	2 h	02/Mar	02/Mar
3	Apply actions change quantity and remove item	Rath	2 h	03/Mar	03/Mar

Designing

- Database, GUI Layout and drawings are tasks in designing process. Example:

--- Mini Store ---
Choose one of the options below:
1. List products
2. View product details
3. Add product to cart
4. View items in cart
5. Exit

Link to cart view
(next slide)

Designing (Console mode)

- Cart view

--- Cart view – List items ---

No.	Name	Unit	Qty	Subtotal
1.	Shoes	\$ 25	1	\$ 25
2.	Books	\$ 0.5	5	\$ 2.5
3.	Bear Doll	\$ 2	2	\$ 4

Total: \$ 31.5

--- Cart view ---

Choose one of the options below:

1. List items
2. Change quantity of an item
3. Remove an item
4. Back to main menu

--- Cart view – Change quantity ---

Choose an item no.: 2

Change to [5]: 8

No.	Name	Unit	Qty	Subtotal
1.	Shoes	\$ 25	1	\$ 25
2.	Books	\$ 0.5	8	\$ 4
3.	Bear Doll	\$ 2	2	\$ 4

Total: \$ 33

PRESS ANY KEY TO CONTINUE...

--- Cart view – Remove an item ---

Choose an item no.: 3

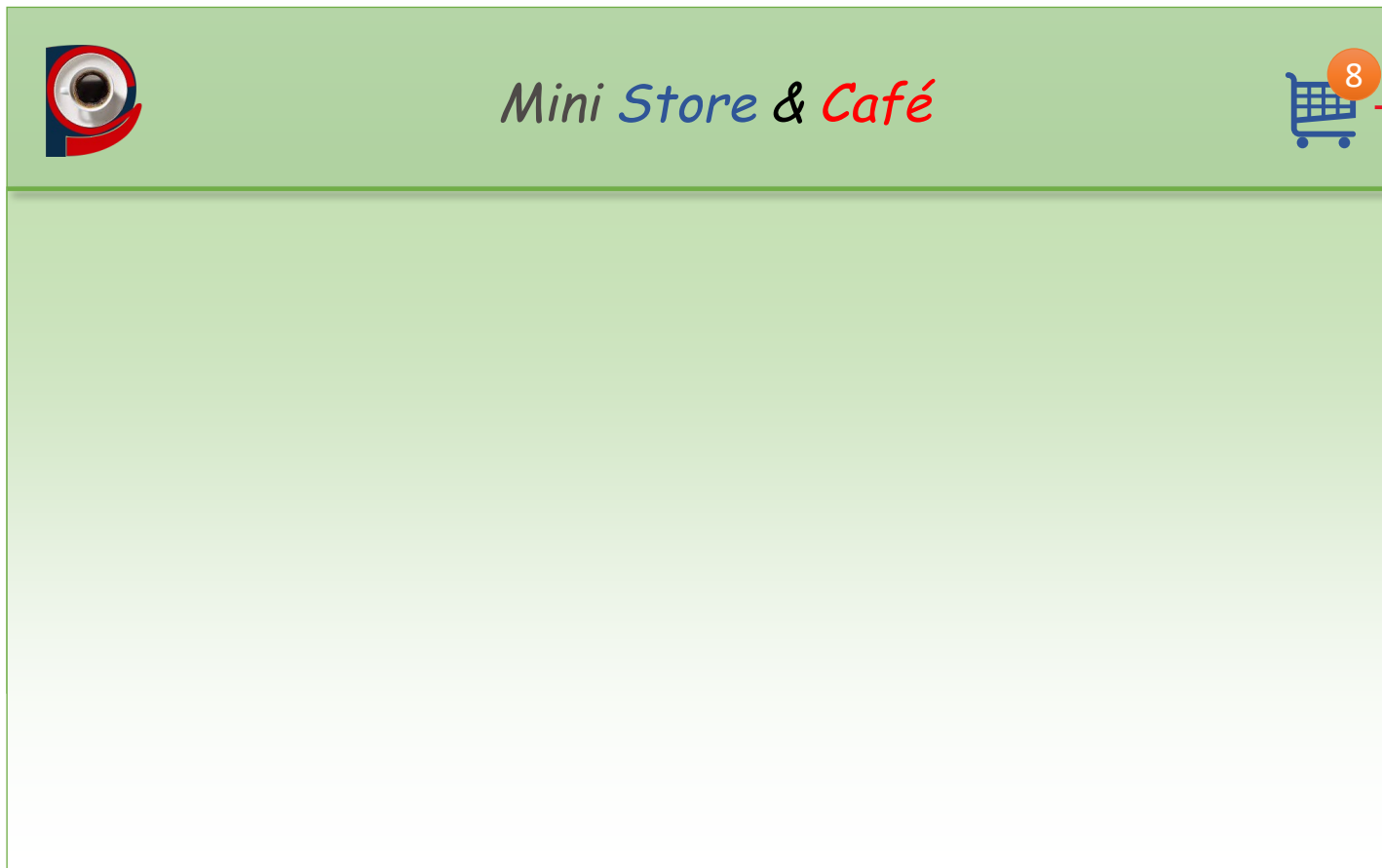
No.	Name	Unit	Qty	Subtotal
1.	Shoes	\$ 25	1	\$ 25
2.	Books	\$ 0.5	8	\$ 4

Total: \$ 29

PRESS ANY KEY TO CONTINUE...

Designing for web platform

- The web page is colorful and rich contents that includes text, image, animation, link, and video. Example:



Badge: There are currently 8 items in cart.

When click it popups a **Cart View** window (in next slide)

Designing for web platform

- The web page is colorful and rich contents that includes text, image, animation, link, and video. Example:

Number input to change the Quantity (minimum value is 1) (Subtotal, Total, and badge will be auto updated)



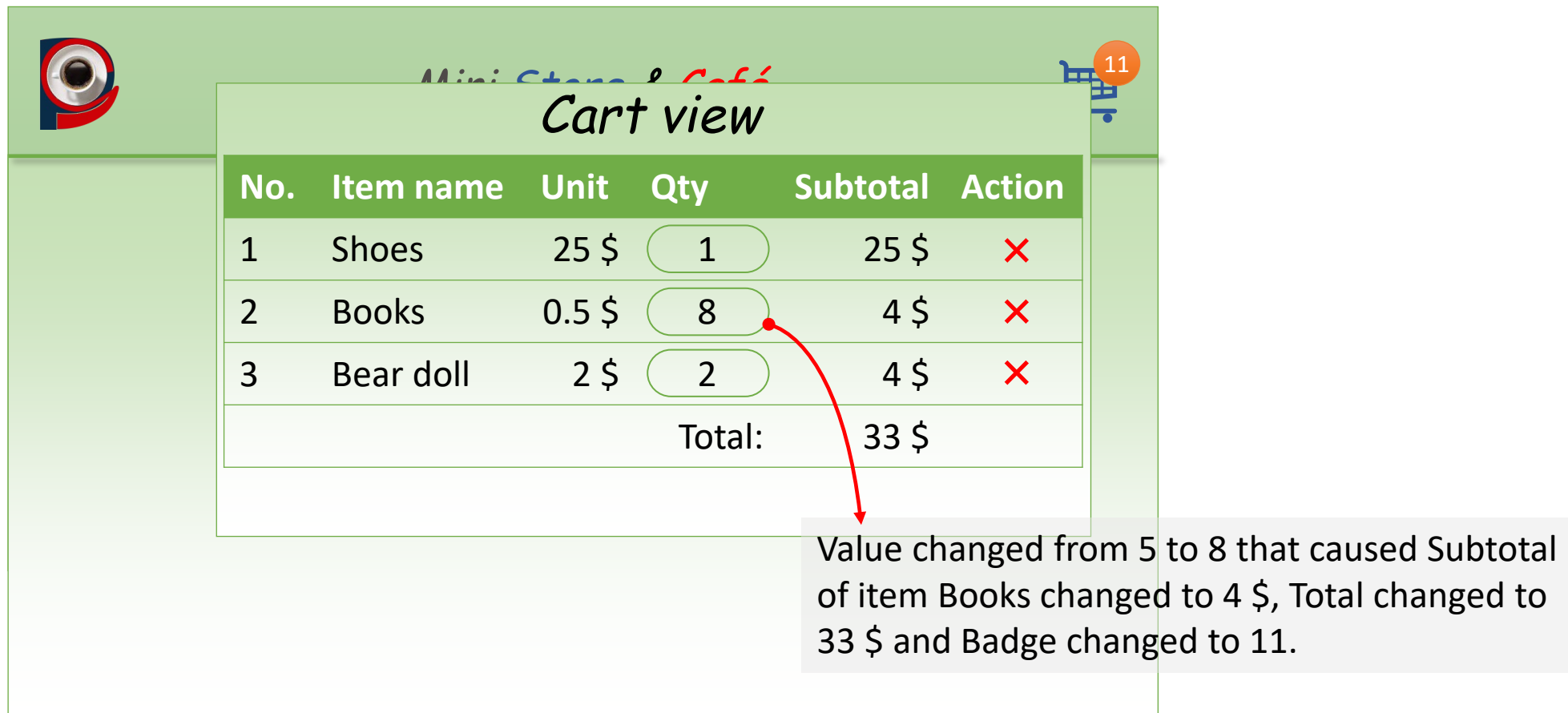
Cart view

No.	Item name	Unit	Qty	Subtotal	Action
1	Shoes	25 \$	<input type="text" value="1"/>	25 \$	✗
2	Books	0.5 \$	<input type="text" value="5"/>	2.5 \$	✗
3	Bear doll	2 \$	<input type="text" value="2"/>	4 \$	✗
Total:				31.5 \$	

Remove an item in selected row when clicking this icon. (Total and badge will be auto updated)

Designing for web platform

- The web page is colorful and rich contents that includes text, image, animation, link, and video. Example:



Mini Store & Café

Cart view

No.	Item name	Unit	Qty	Subtotal	Action
1	Shoes	25 \$	1	25 \$	×
2	Books	0.5 \$	8	4 \$	×
3	Bear doll	2 \$	2	4 \$	×
Total:				33 \$	

Value changed from 5 to 8 that caused Subtotal of item Books changed to 4 \$, Total changed to 33 \$ and Badge changed to 11.

Designing for web platform

- The web page is colorful and rich contents that includes text, image, animation, link, and video. Example:



Mini Store & Café

Cart view

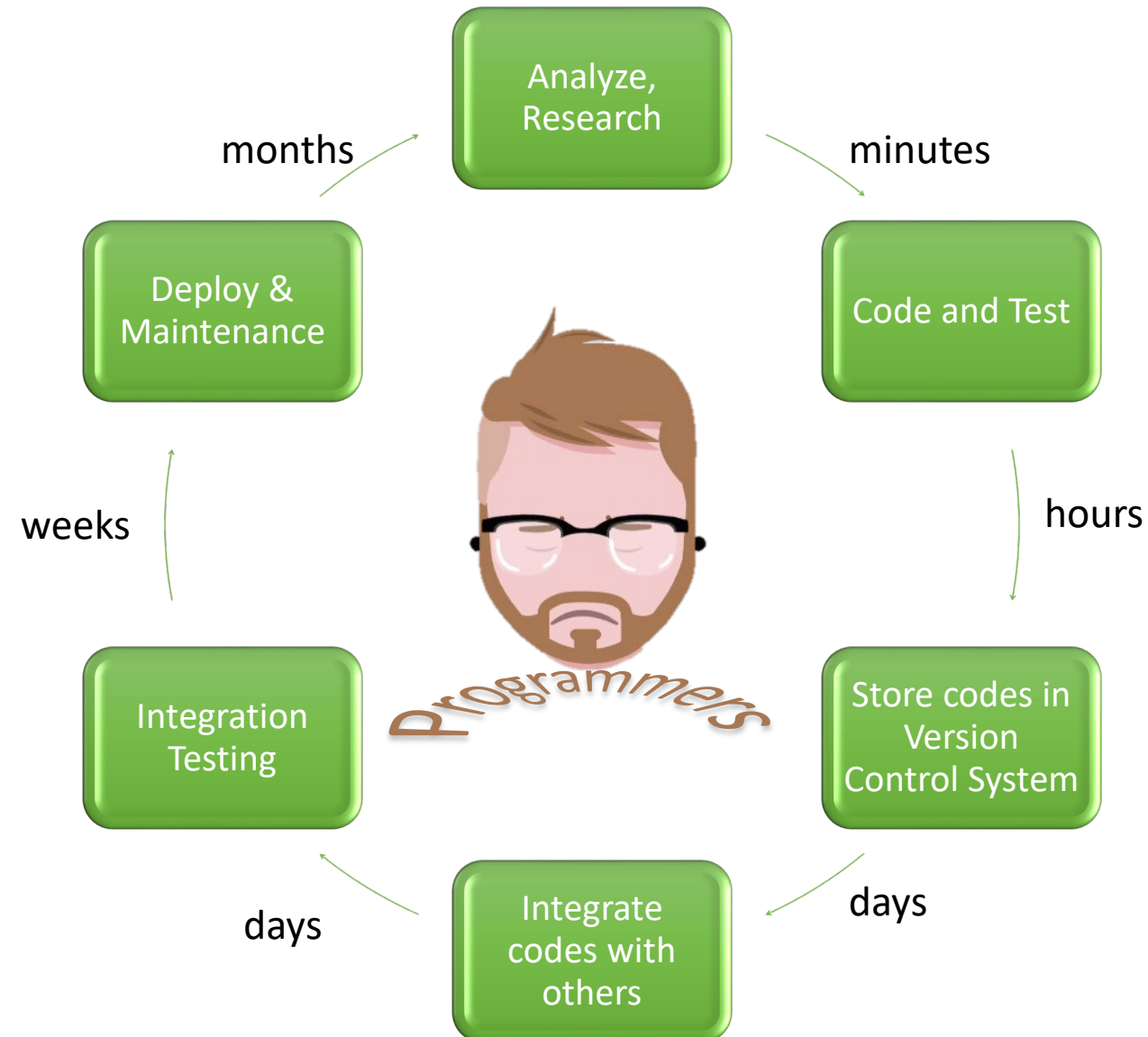
No.	Item name	Unit	Qty	Subtotal	Action
1	Shoes	25 \$	<input type="text" value="1"/>	25 \$	×
2	Books	0.5 \$	<input type="text" value="8"/>	4 \$	×
Total:				29 \$	

The Bear doll is removed that caused Total changed to 29 \$, and Badge changed to 9.

III. Development process

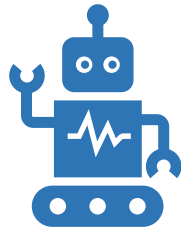
- The process of analysis, research, code, unit testing, and deployment of one specific task.
 - **Analysis:** understand the problem, [cause of problem](#), symptom, and match with digital solution. Example: List items as table view. The problem is [how to display it correctly](#) such as strings are left aligned, numbers are right aligned, currencies are suffixed with \$. The item data should be as ordered list or associative array? Adapts to the situation.
 - **Research:** After understanding the problem, if any technical problem, search online for solution. Example: “[how to right aligned the text in Java](#)”.
 - **Code:** write the instructions, build it, and run it to check if its [output as expected](#). Example: after found solution on how to display text as right alignment, try the code in local PC, build and run it to see if it is really worked.
 - **Unit testing:** After tested each piece of codes, we put them together and test the result and if it is not working try to [inspect the problem](#) and re-test it again.
 - **Deployment:** After successfully tested, [commit and push code](#) to Version Control System (VCS) then wait for integration test results. If the integration test failed, try to fix it yourself and/or with others, re-commit, re-push the code, and wait again for the result.

Development process



How to make it shorter time spent between phases?

- Apply automations where applicable

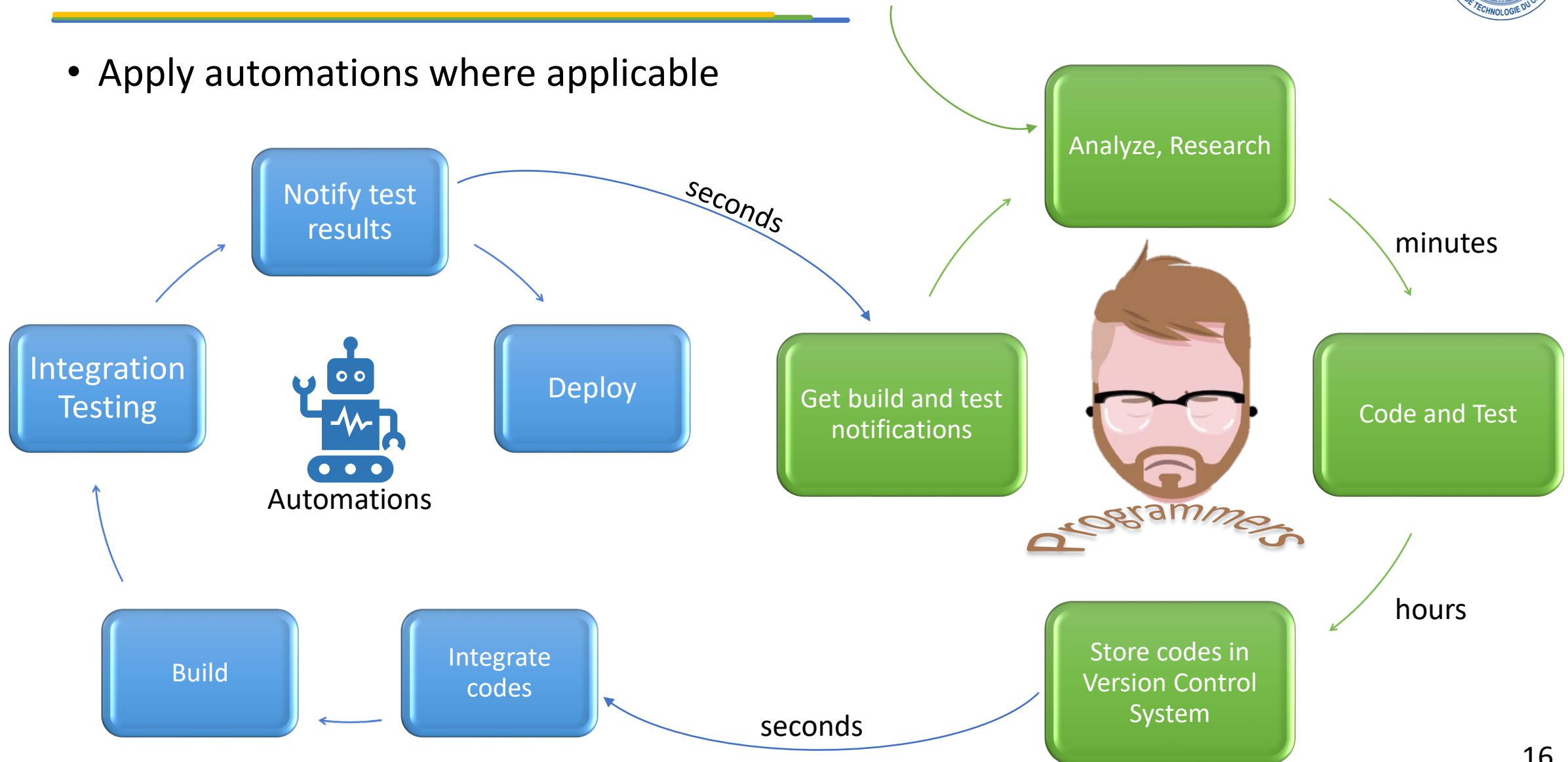


Automations



How to make it shorter time spent between phases?

- Apply automations where applicable



IV. Version Control System

- Daily tasks of developers
 - Create text
 - Save text
 - Edit text
 - Save it again
- What is VCS?
 - About managing multiple versions of
 - Documents
 - Programs
 - Websites etc.
 - Tracks History of collection of files
 - Version control software keeps track of every modification to the code in a special kind of database

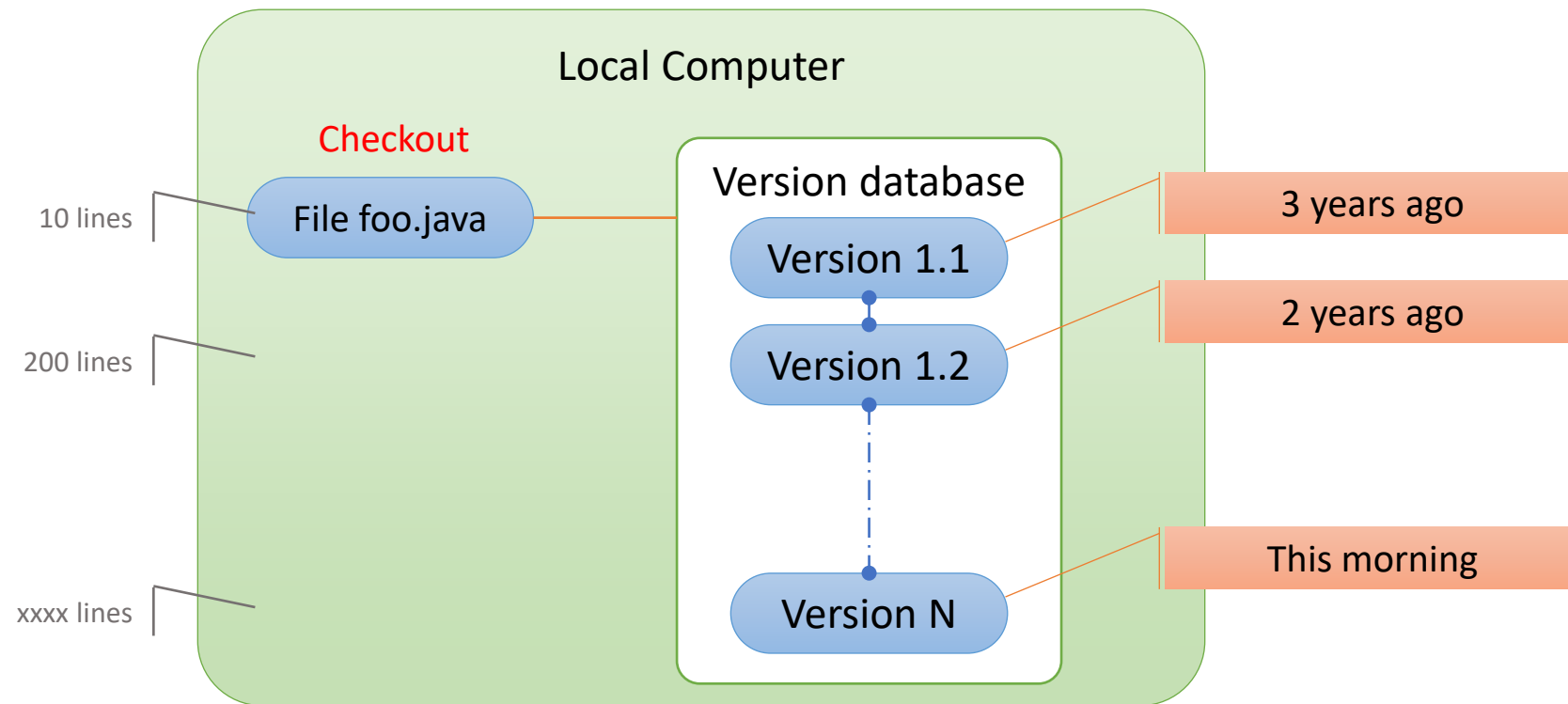
Why VCS?

- For Individual Help:
 - Gives you a “time machine” for going back to earlier versions
 - Gives you great support for different versions (standalone, web app, etc.) of the same basic project
- For Working with Team:
 - Greatly simplifies concurrent work, merging changes
- Management of changes to files.
 - Keep track of what changes occurred.
 - Allows People to work Together.

Localized and Centralized VCS

- A localized version control system keeps local copies of the files.
- In centralized source control, there is a server and a client. The server is the master repository which contains all of the versions of the code.

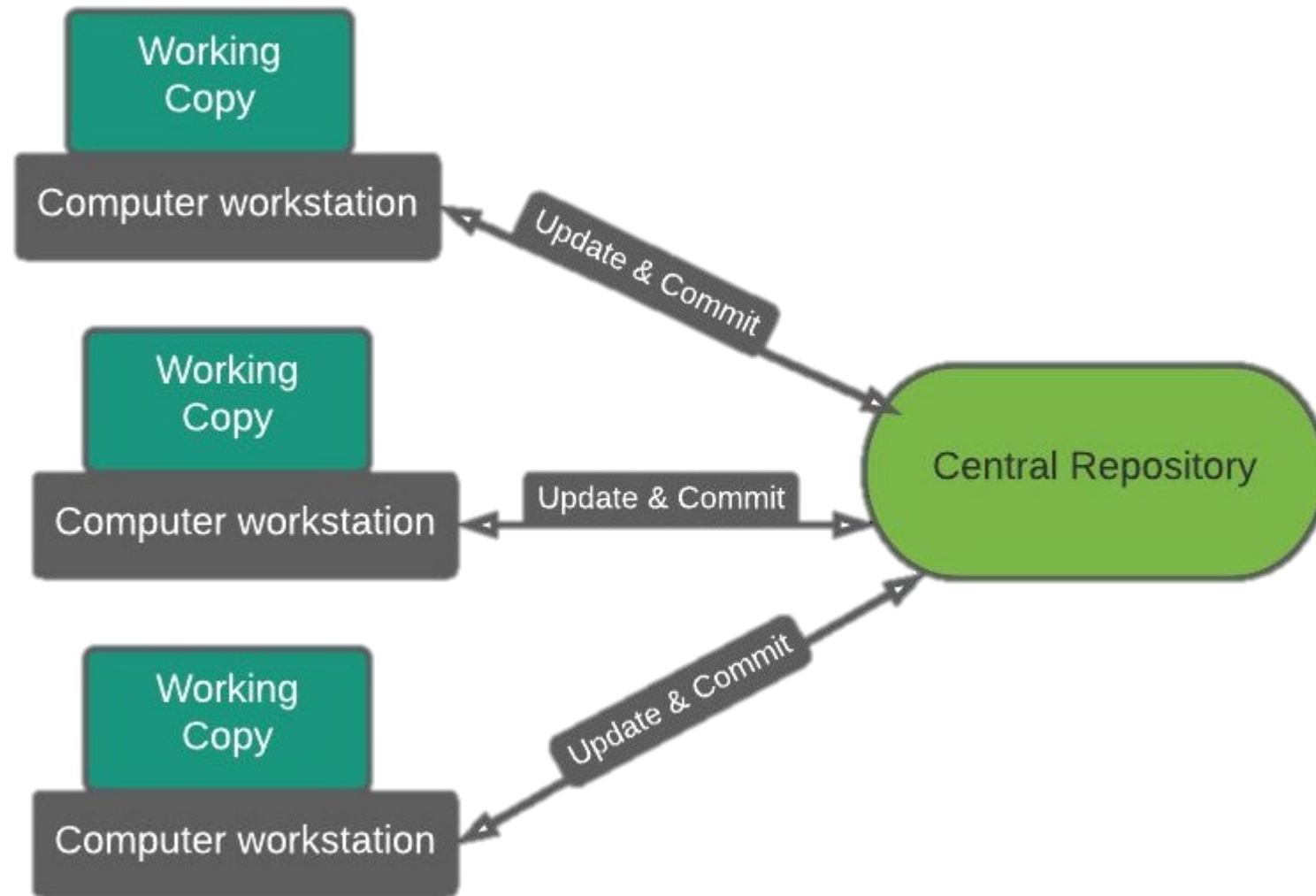
Localized VCS



Centralized VCS

- In Subversion, CVS, Perforce, etc. A central server repository (**repo**) holds the "**official copy**" of the code.
 - the server maintains **the sole version history** of the repo
- You make "**checkouts**" of it to your local copy
 - you make local modifications
 - your changes are not versioned
- When you're done, you "**check in**" back to the server
 - your checkin increments the repo's version

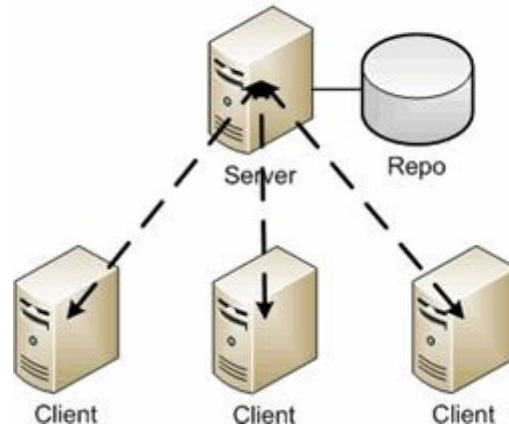
Centralized VCS



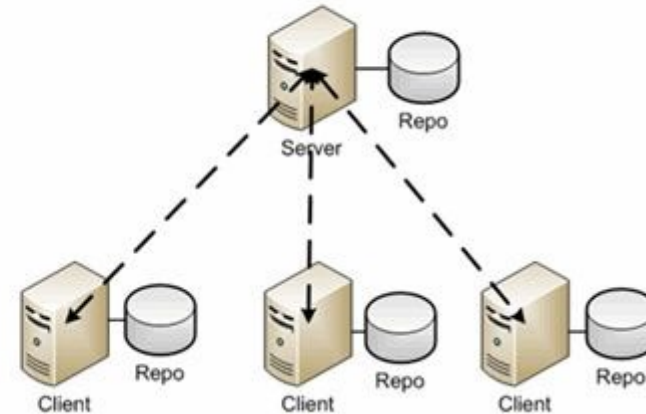
Distributed version control systems

- In a distributed version control system each user has a complete local copy of a repository on his individual computer.

Traditional



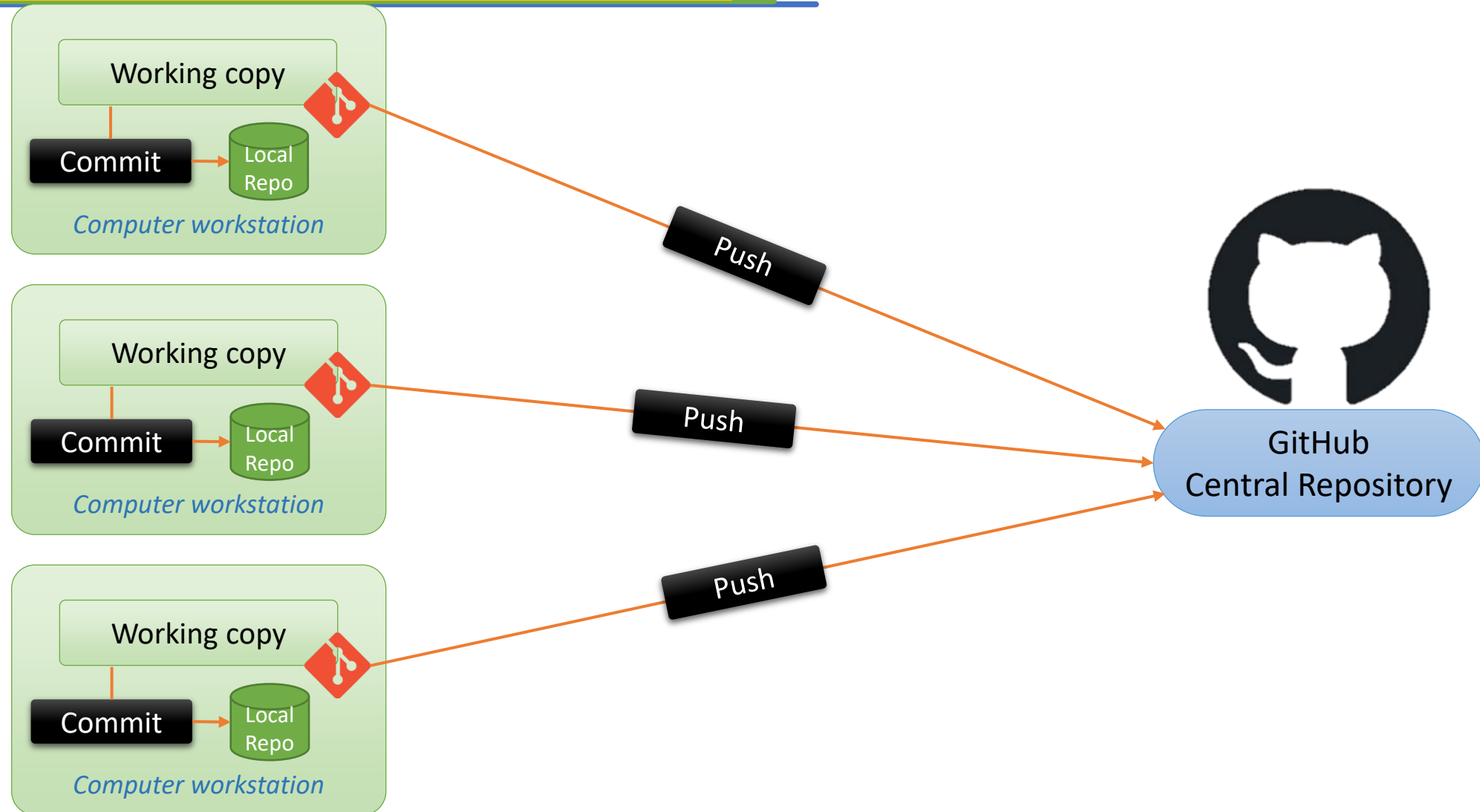
Distributed



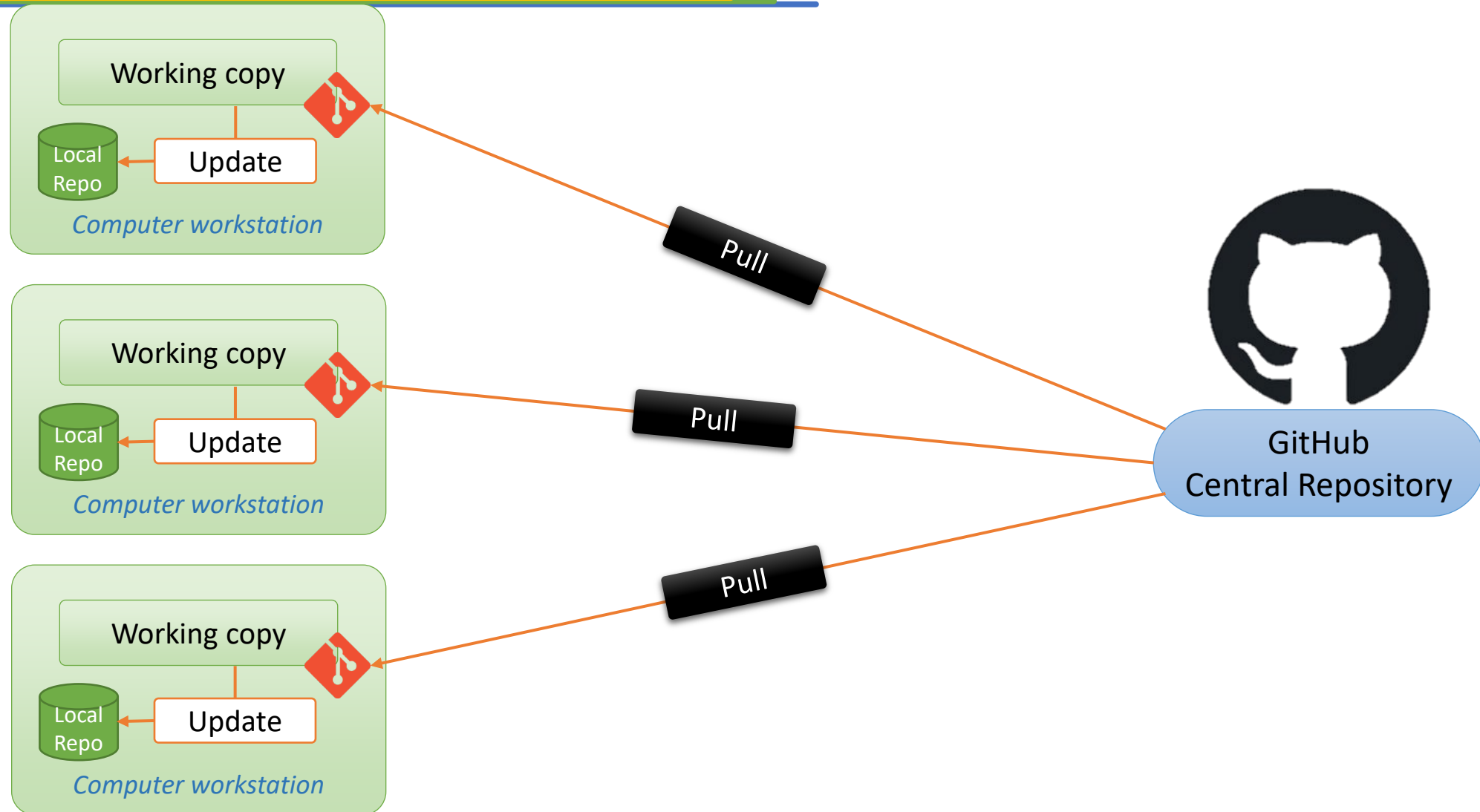
Distributed version control systems

- In git, mercurial, etc., you don't "checkout" from a central repo
 - you "clone" it and "pull" changes from it
- Your local repo is a complete copy of everything on the remote server
 - yours is "just as good" as theirs
- Many operations are local:
 - check in/out from local repo
 - commit changes to local repo
 - local repo keeps version history
- When you're ready, you can "push" changes back to server

Distributed version control systems

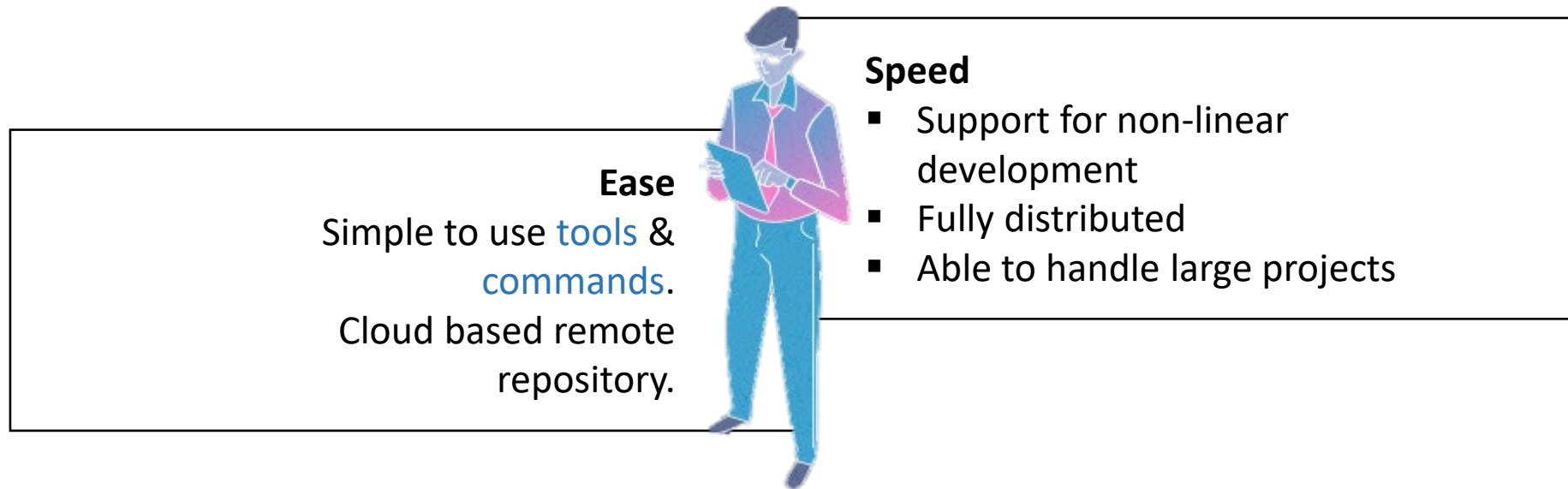


Distributed version control systems



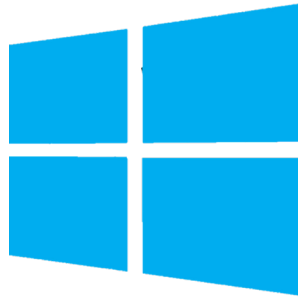
V. What is Git?

- Git is a distributed version control system
- Git is a Tree History storage system
- Git is content tracking management system



Git Quick Start

- Created by Linus Torvalds, creator of Linux, in 2005
 - Came out of Linux development community
 - Designed to do version control on Linux kernel
- Installing Git



Install Git Bash



Install via HomeBrew



Install via Package manager
(yum, apt, snap etc)

Local Repository Setup

1. Set the name and email for Git to use when you commit:
 - ✓ `git config --global user.name "SOK Bopha"`
 - ✓ `git config --global user.email bopha.sok@email.com`
2. Create a directory
3. Initialize directory with
 - ✓ `git init`
4. Create Readme.md file
 - ✓ `git add` (Staging)
 - ✓ `git commit` (Local commit)

Remote Repository

- Create Remote repository on
 - ✓ GitHub, Gitlab, bitbucket etc.
- Clone Repo to local
 - ✓ git clone URL
- Local to Remote integration
 - ✓ cd to local repo
 - ✓ git remote add origin ssh://git@github.com/[username]/[repository-name].git
 - ✓ git push
 - ✓ git pull (to fetch latest changes)