L1. Agile Cloud Automation

S1. Introduction

Dr A Boronat

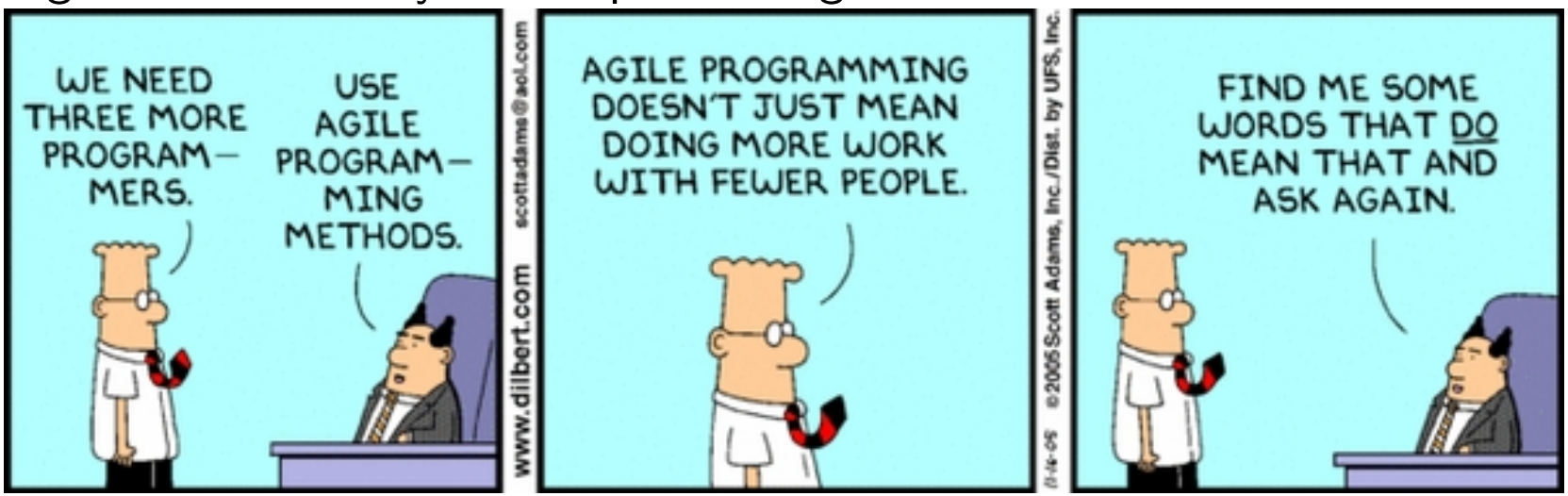
Reasons Why Software Projects Fail

- as complexity appears, abstraction is required
- communication between domain experts and developers
- fast pace in technology changes: new languages

modelling techniques are useful to capture abstractions
agile methodologies are needed
domain-specific languages are ubiquitious

Agile Methodologies

- According to the 9th Annual State of Agile™ survey, 94% of all organizations surveyed now practice agile

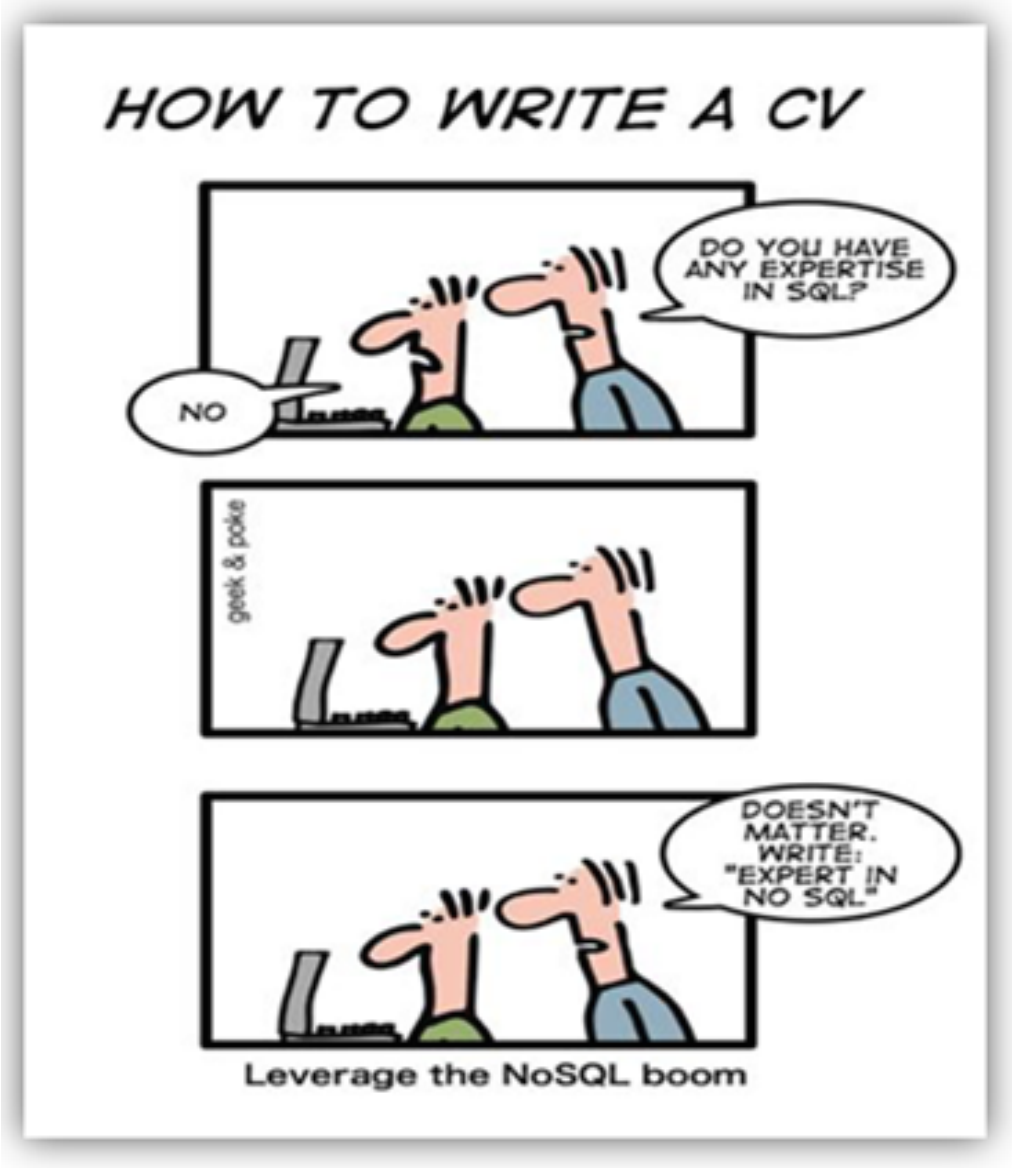


...recommended core practices:

  - automated builds
  - automated testing

Current sw development practices

- Domain-specific languages are pervasive in software development for the Cloud
  - Frameworks, such as Spring, offer high-level languages to configure cloud-based applications
  - DSLs built atop Javascript are born on a weekly basis
  - Facebook is using traditional compiler techniques to analyse dynamic language programs
  - Relational databases are not the sole star in a cloud-based era


HOW TO WRITE A CV
DO YOU HAVE ANY EXPERTISE IN SQL?
NO
geek & poke
DOESN'T MATTER. WRITE: "EXPERT IN NO SQL"
Leverage the NoSQL boom

Current sw development practices

- NoSQL databases have become a real alternative to relational RDBMSs
  - currently MongoDB ranked 4th in popularity (out of 258), just behind Oracle, MySQL and MS SQL Server.
  - Rapid prototyping
  - Facilities to deal with big data: scalability

Scope of this Module

- Focus on the use and design of DSLs in order to
  - deal with polyglot persistence (NoSQL and SQL)
  - agile software development using DSLs and model-driven development
  - develop integration solutions based on cloud-based technology
- This module is for you if you are interested in...
  - learning new DSLs and new programming languages
  - learning conceptual differences between SQL and NoSQL, and technical implications:
    MongoDB and Spring Framework
  - using tools for agile modelling (including the specification of DSLs):
    MDA, EMF, xText
  - improving your (sw) productivity as a software developer:
    build automation with Gradle
  - improving the (sw) productivity of your organization as a technical manager

Module Units

Part I: Twitter

- Introduction to continuous delivery with Gradle
- Polyglot Persistence - the advent of NoSQL

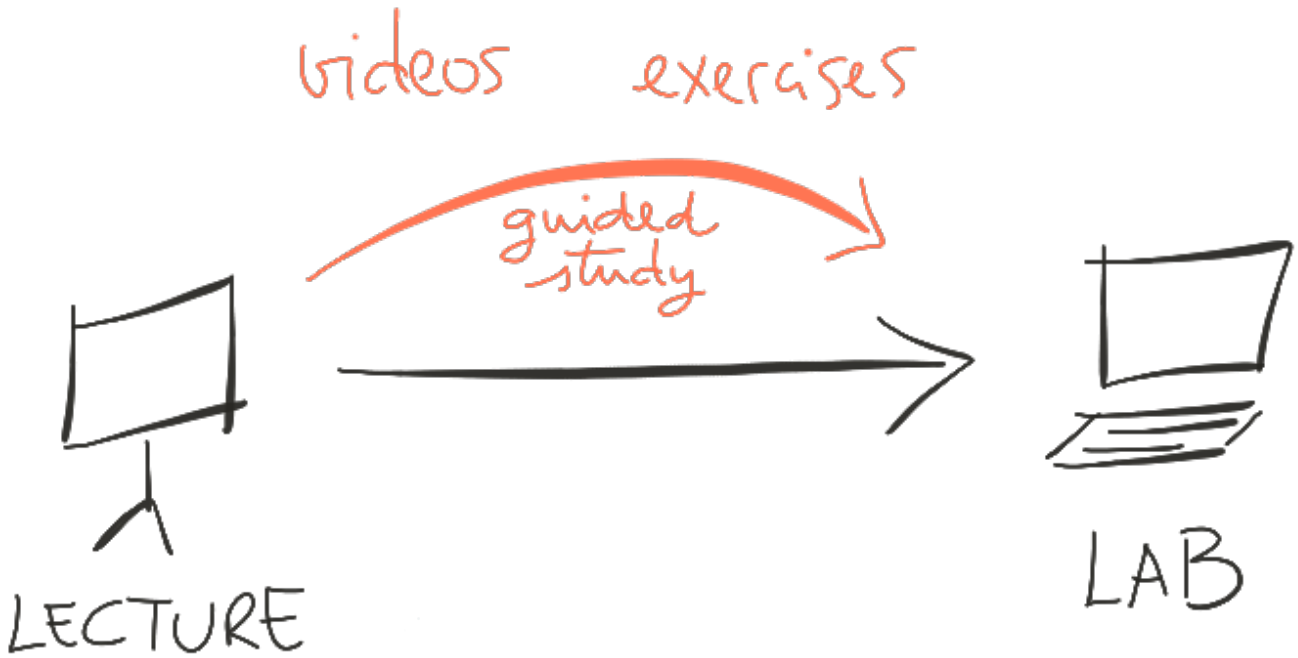
Part II: Facebook's GraphQL

- Domain-Driven Design and MDE
- Concrete syntax (text to model)
- Semantics (model to text)
- Model transformations (model to model)

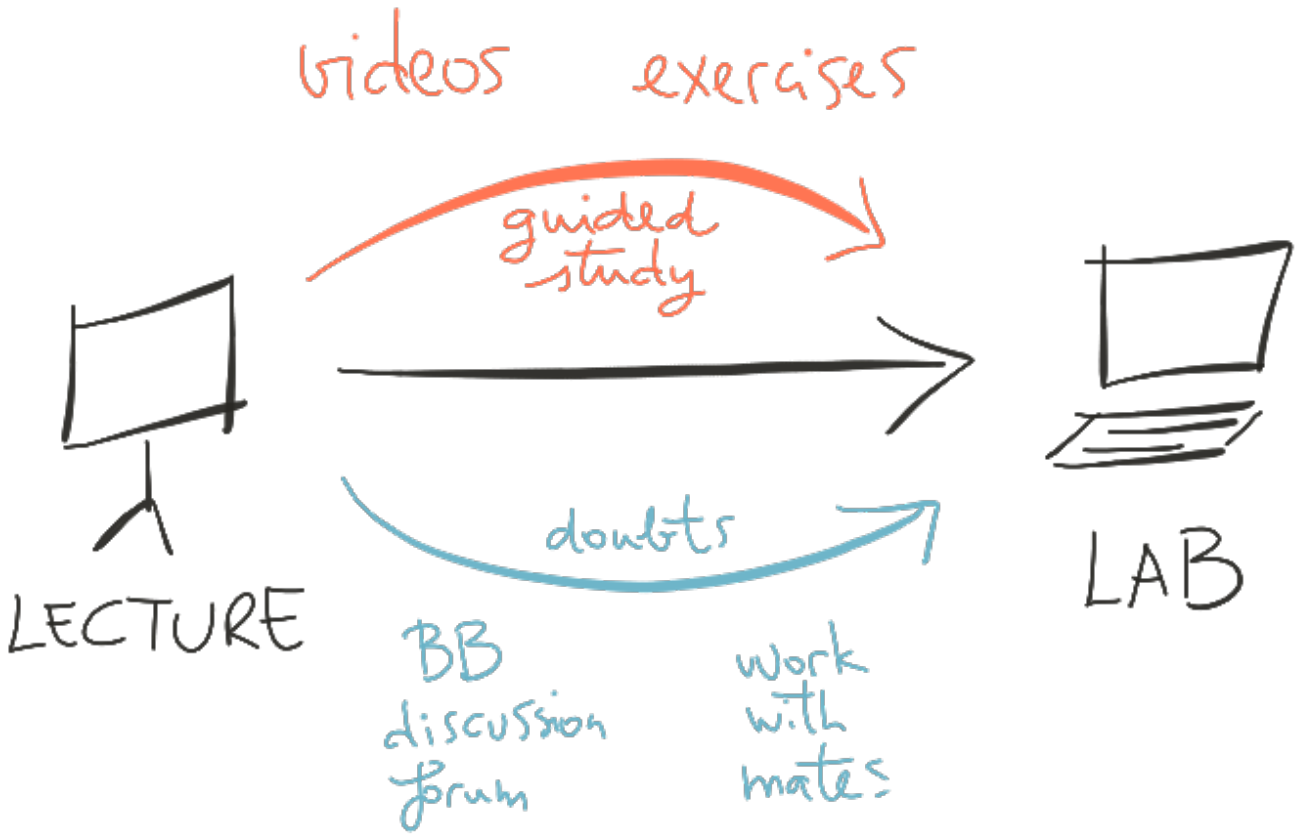
ORGANISATION

BLENDED LEARNING

BLENDED LEARNING



BLENDED LEARNING



SCHEDULE

Blackboard

Assessment and feedback

Part I

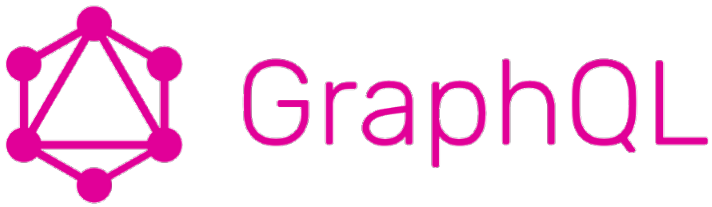
Miniproject 1: Thu 27 Oct - 30%

- 60% programming
- 40% essay: scalability concerns behind Twitter sw architecture

Part II

Miniproject 2: Thu 14 Dec - 30%

- 60% programming
- 40% essay: Facebook's GraphQL DSL



Final Test

Thu 23 Nov - 40%

Communication

- Blackboard discussion forum
  - check whether your question is answered already
  - if no one knows the answer, open a thread in the discussion forum with:
    - a meaningful title
    - some context: explanation of the problem (for technical questions, you may want to use a GitHub gist, a git repository with code, etc) that is required to understand the question
    - a question
- Office hours
- Laboratory sessions/tutor meetings
- GitHub

COMMUNICATION

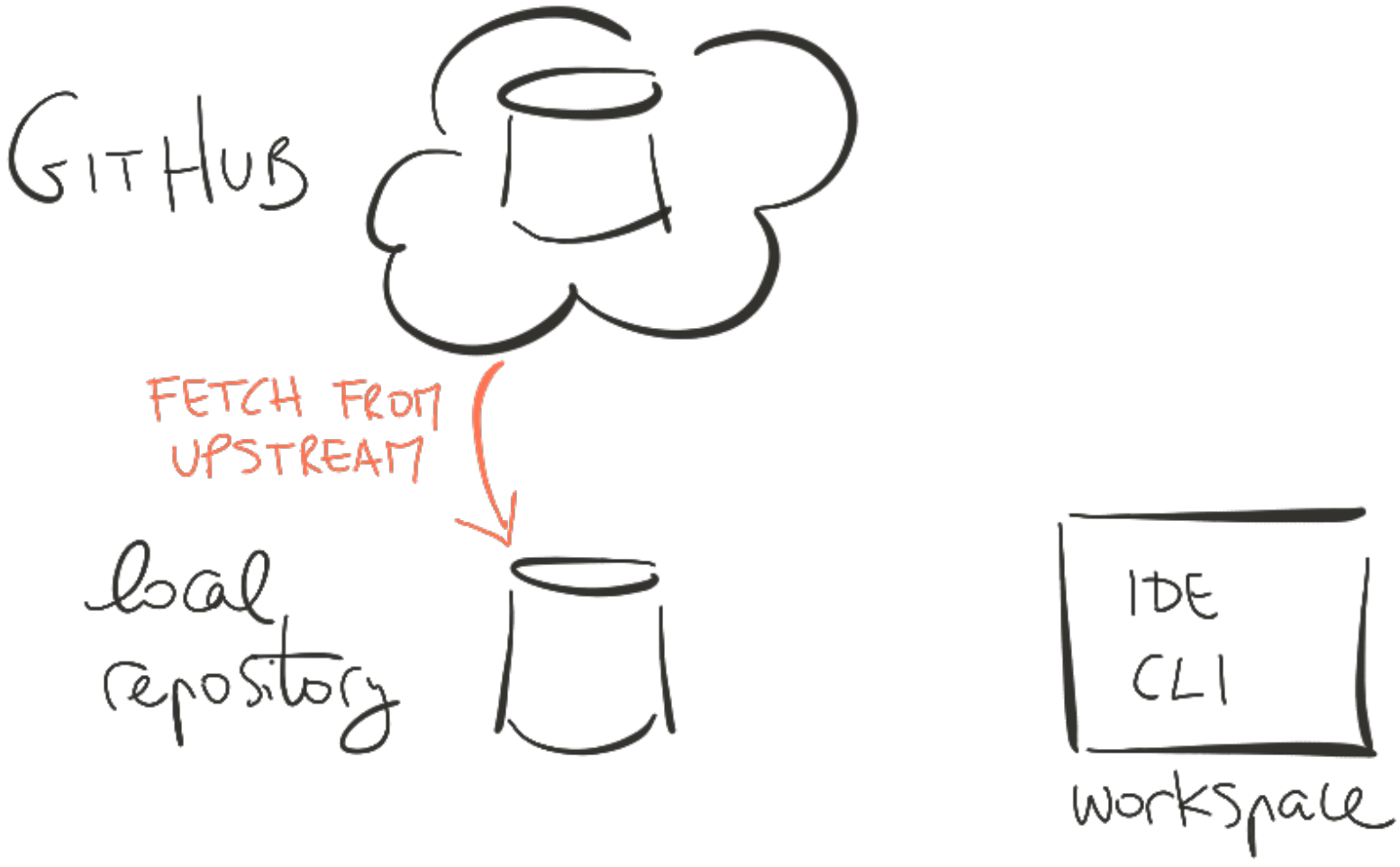
GITHUB

local repository

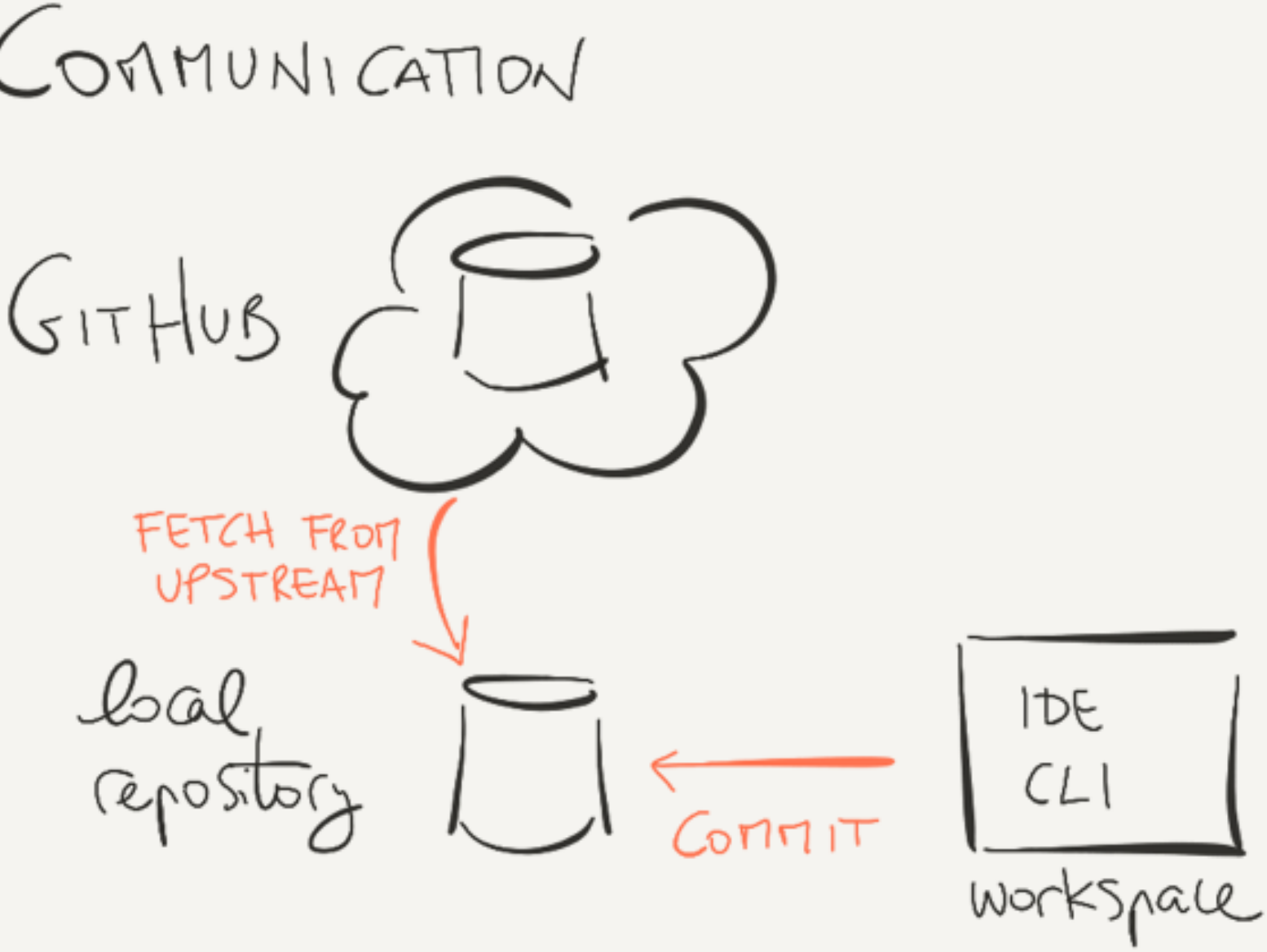
IDE
CLI

workspace

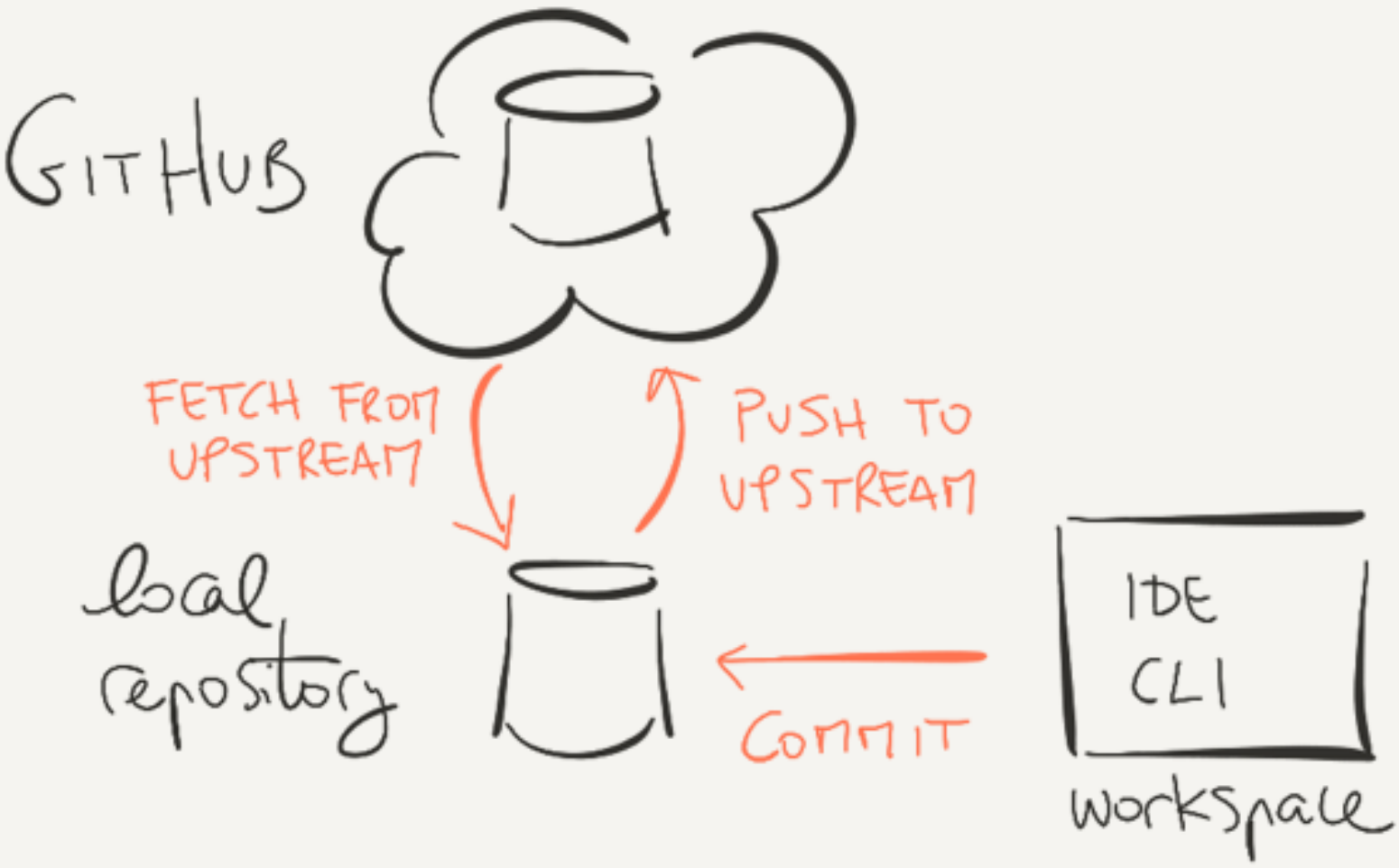
Communication



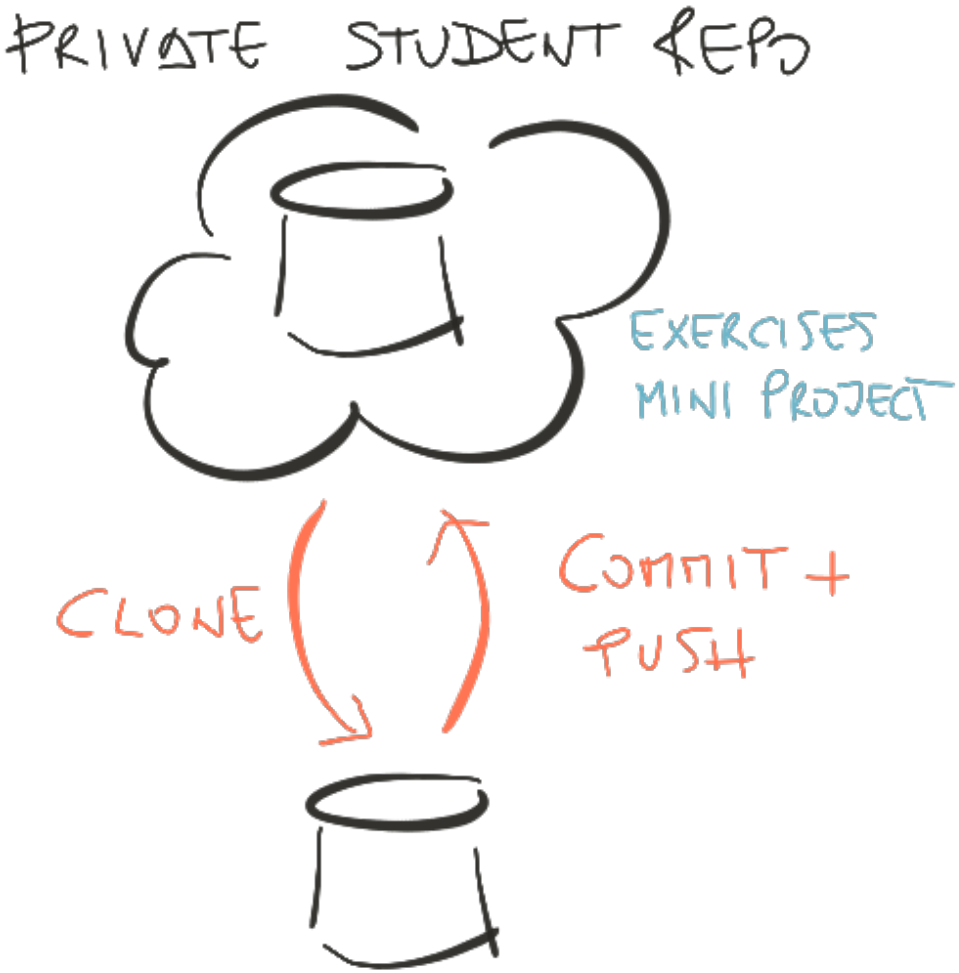
Communication



Communication



Communication



Communication