



# INTRODUCTION AU DEVELOPPEMENT COLLABORATIF AVEC GIT ET GITHUB

Extraits de la formation académique BTS SIO SLAM 2013

Académie de Lyon – B. Pesquet



# LES ENJEUX DU DÉVELOPPEMENT COLLABORATIF

# || Caractéristiques d'un projet logiciel

- Durée de vie importante (plusieurs années)
- Phases de maintenance/évolutions souvent plus longues que la phase de développement initial
- Plusieurs versions successives
- Multiplicité des intervenants :
  - Travail en parallèle
  - Equipes de maintenance souvent distinctes des équipes de développement

# La gestion du code source

- Code source = cœur du projet logiciel
- Besoins associés :
  - Assurer la pérennité du code source
  - Permettre le travail collaboratif :
    - Récupérer le code commun
    - Publier ses modifications
    - Gérer les conflits (modifications concurrentes)
  - Identifier les différentes versions du projet



- Nécessité d'une gestion du code source
- Utilisation quasiment obligatoire d'un outil de SCM (*Source Code Management*)

# ■ Anatomie d'un outil de SCM

- Dépôt qui héberge le code source d'un ou plusieurs projets logiciels
- Chaque développeur accède au code via le dépôt
- Il peut ensuite effectuer plusieurs actions :
  - Récupération du code commun (*update*)
  - Publication de ses modifications personnelles (*commit*)
- Traçage des modifications successives (historique)
- Gestion des versions (étiquetage ou *tagging*)
- ...

# Le marché des outils de SCM

- Plusieurs dizaines d'outils, deux grandes catégories
- Gestion centralisée (plus ancienne) :
  - Un seul dépôt de code fait référence
  - Connexion au dépôt en mode client/serveur
  - Exemples : CVS, SVN (Subversion), TFS
- Gestion décentralisée (récente) :
  - Autant de dépôts de code que de développeurs
  - Synchronisation entre tous les dépôts (mode pair-à-pair)
  - Plus adaptée aux équipes distribuées géographiquement
  - Exemples : Git, Mercurial



# GIT & GITHUB



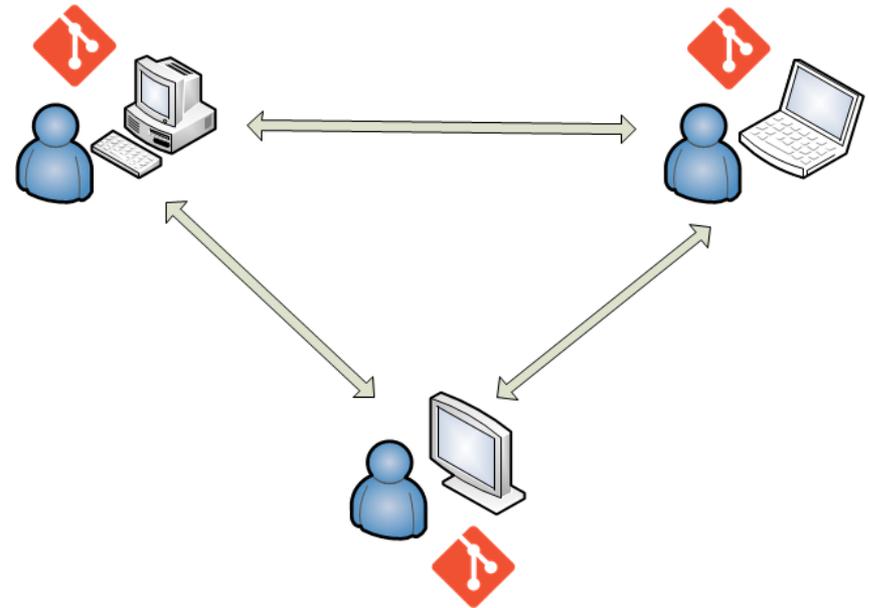
# Présentation de Git

- Outil de SCM créé en 2005 par Linus Torvalds pour gérer les contributions au noyau Linux
- Logiciel libre sous licence GNU GPL
- Multi-langages, multiplateformes
- Optimisé pour la performance



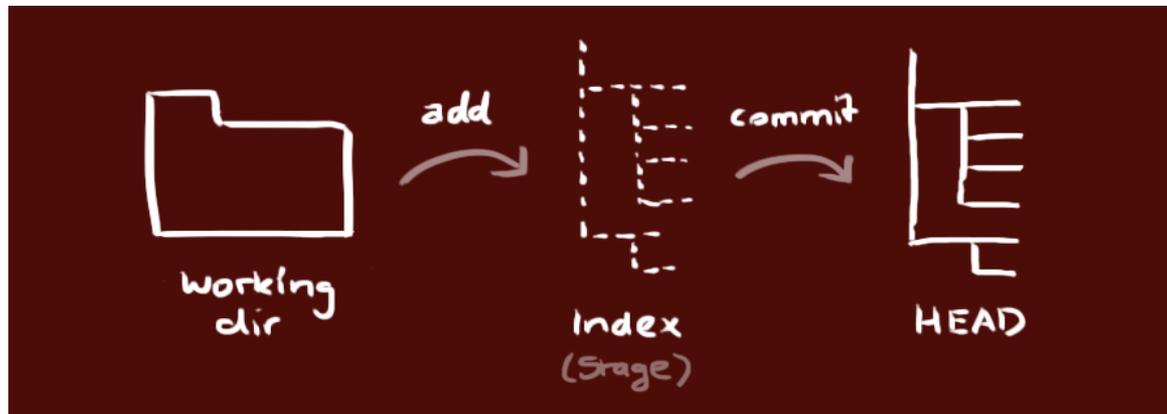
# ■ Fonctionnement de Git

- Gestion décentralisée (modèle pair-à-pair) :
  - Chaque développeur a son dépôt Git local (« *repository* »)
  - Tout dépôt Git contient l'intégralité des données (code source, historique, versions, etc)



# Structure d'un dépôt Git

- Dépôt Git = répertoire *.git*
- Constitué de trois arbres
  - Espace de travail (« *working directory* ») : fichiers actuels
  - Index : espace de transit
  - HEAD : derniers fichiers archivés



# Principales commandes Git

- ***init***: crée un nouveau dépôt local
- ***clone*** : duplique un dépôt distant
- ***add*** : ajoute des fichiers à l'Index
- ***commit*** : archive des fichiers (de Index vers HEAD)
- ***tag*** : pose une étiquette
- ***status*** : affiche les modifications par rapport au dépôt
- Utilisation en ligne de commande ou via un IDE

# Présentation de GitHub

- Plate-forme Web de publication de code
- <https://github.com/>



# ■ Fonctionnalités de GitHub

- Fourniture de dépôts Git en ligne
- Mais aussi :
  - Gestion des tâches à réaliser (« *issues* »)
  - Travail collaboratif
  - Contrôle d'accès aux dépôts
  - Réseau social de développeurs



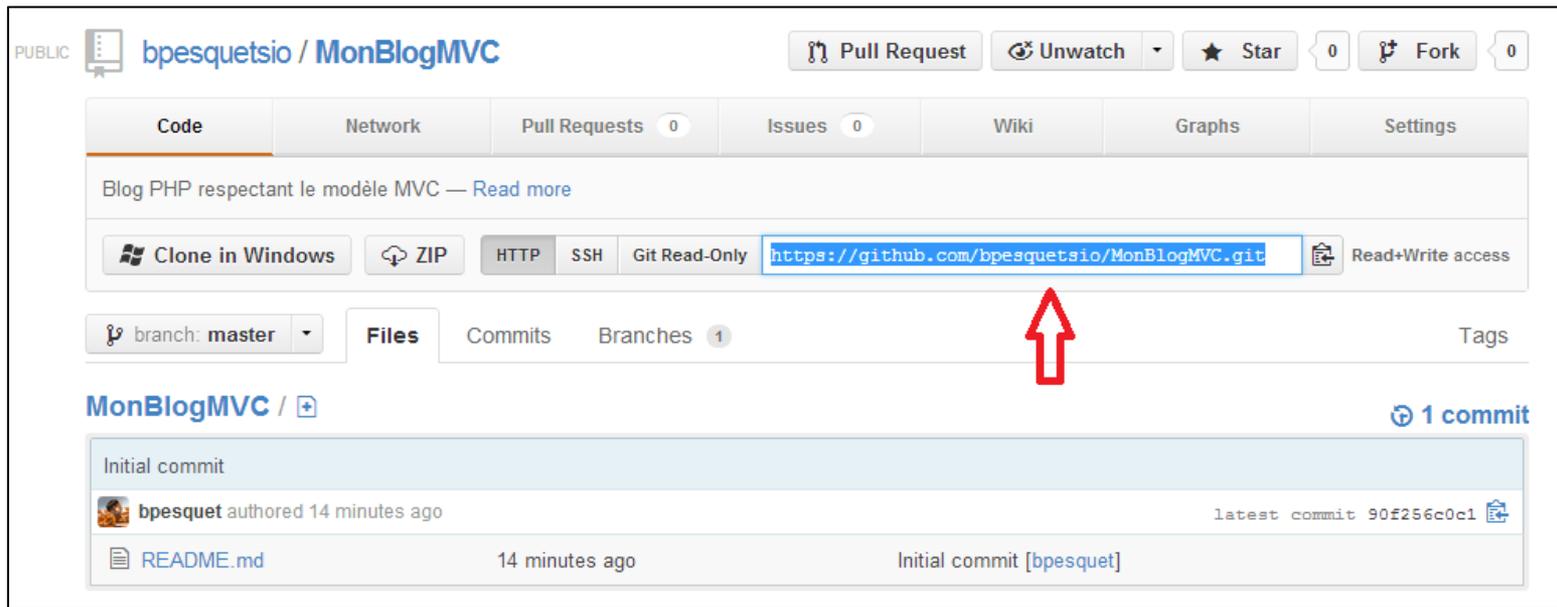
Véritable **forge logicielle**

# Business model de GitHub

- Gratuit :
  - Inscription sur la plate-forme (obligatoire)
  - Création d'organisations pour gérer le travail en équipe
  - Création de dépôts Git *publics* (accès libre)
- Payant :
  - Création de dépôts *privés* (accès restreint)
- Offre éducation : 10 dépôts privés gratuits
- Alternative : BitBucket (dépôts privés gratuits)

# Création d'un dépôt GitHub

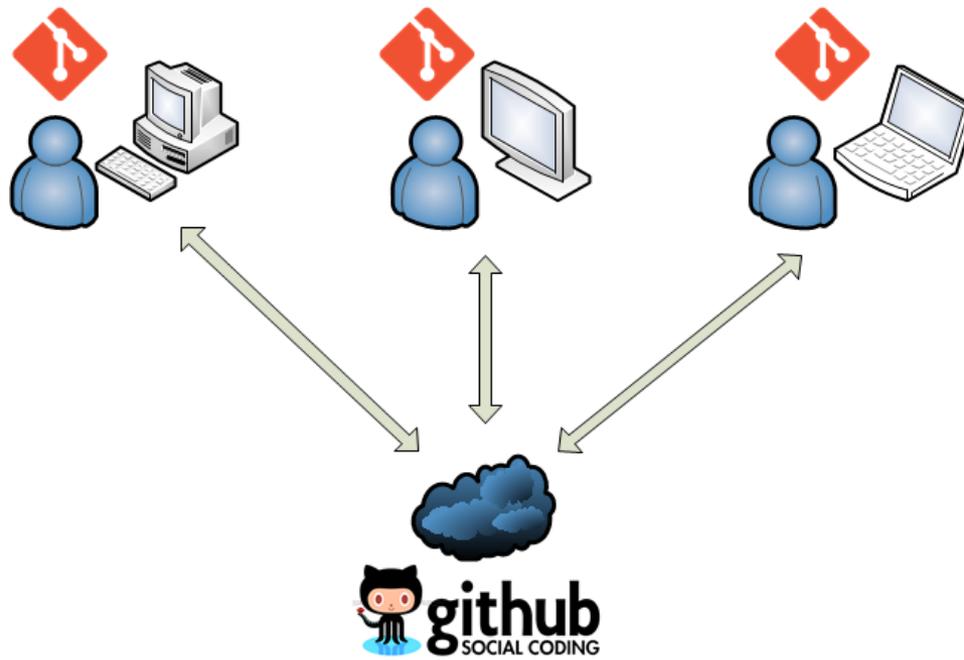
- Dépôt GitHub = dépôt Git accessible en ligne (URL de type <https://...git>)
- Créé pour un utilisateur GitHub ou une organisation



The screenshot shows the GitHub interface for a repository named 'MonBlogMVC' by user 'bpesquetsio'. The repository is public and has 0 stars and 0 forks. The main content area shows a description: 'Blog PHP respectant le modèle MVC — Read more'. Below this, there are options to clone the repository: 'Clone in Windows', 'ZIP', 'HTTP', 'SSH', and 'Git Read-Only'. The 'Git Read-Only' option is selected, and the URL 'https://github.com/bpesquetsio/MonBlogMVC.git' is displayed in a blue box. A red arrow points to this URL. To the right of the URL is a 'Read+Write access' button. Below the cloning options, there are tabs for 'Files', 'Commits', and 'Branches' (with 1 branch). The 'Files' tab is active, showing a list of files: 'README.md' (14 minutes ago) and 'Initial commit [bpesquet]' (14 minutes ago). The 'Initial commit' is the latest commit, with a hash of 90f256c0c1. The repository has 1 commit in total.

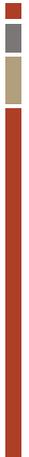
# Travail en équipe avec GitHub

- Chaque développeur travaille sur son dépôt Git
- Le dépôt GitHub permet la synchronisation



# Processus de travail en équipe

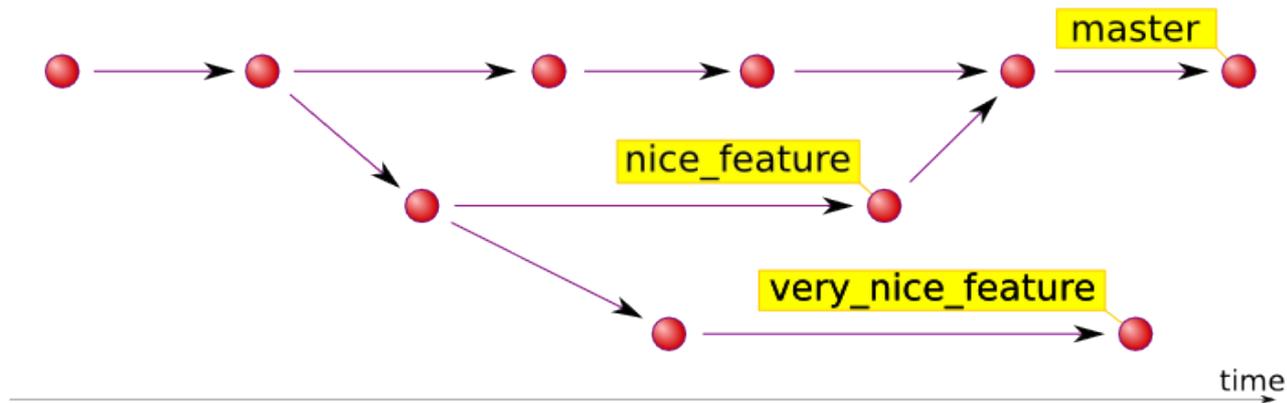
1. Dépôt GitHub initialisé par un membre de l'équipe ou le chef de projet
2. Clonage du dépôt GitHub (***git clone***) par les autres membres de l'équipe => dépôt Git local
3. Travail individuel sur chaque dépôt Git local (***git add, commit, tag***, etc)
4. Publication sur GitHub : ***git push***
5. Récupération des modifications apparues sur GitHub : ***git pull***



# ALLER PLUS LOIN AVEC GIT & GITHUB

# ||| Gestion des branches

- Branche principale par défaut : **master**
- Possibilité de créer d'autres branches pour travailler en parallèle de la branche principale, puis éventuellement fusionner
- Branches créés avec Git (***git branch***)



# GitHub : tâches à réaliser (« issues »)

- Unités d'information (sujets de discussion)
- Assignables à un ou plusieurs membres de l'équipe
- Cycle de vie simple (ouvert/fermé)
- Permettent (entre autres) le suivi des incidents

Browse Issues   Milestones   Search: Issues & Milestones...   **New Issue**



 rflagrange is assigned    No milestone 

Write   **Preview**   Comments are parsed with GitHub Flavored Markdown

Les champs du formulaire d'inscription sont mal alignés.

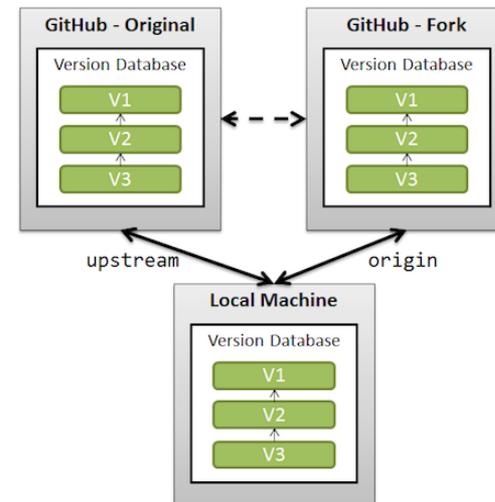
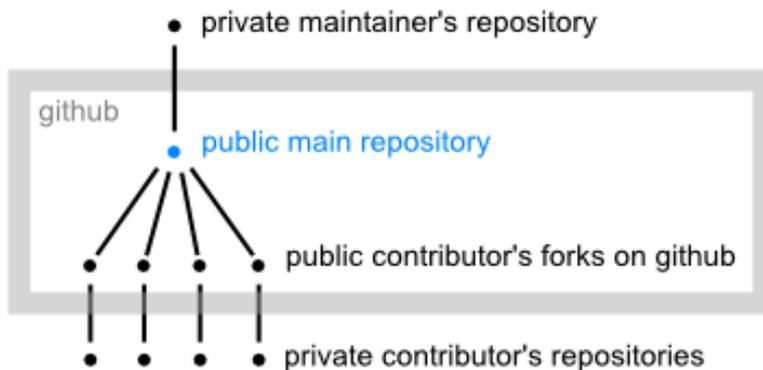
**Submit new issue**

**Add Labels**

-  bug
-  duplicate
-  **enhancement**
-  invalid
-  question
-  wontfix

# GitHub : modèle « fork & pull »

- Modèle de travail collaboratif pour équipes nombreuses (exemple : projets Open Source)
- Dépôt commun « forké » par chaque développeur
- Le responsable du dépôt commun intègre les évolutions des contributeurs





Merci de votre attention