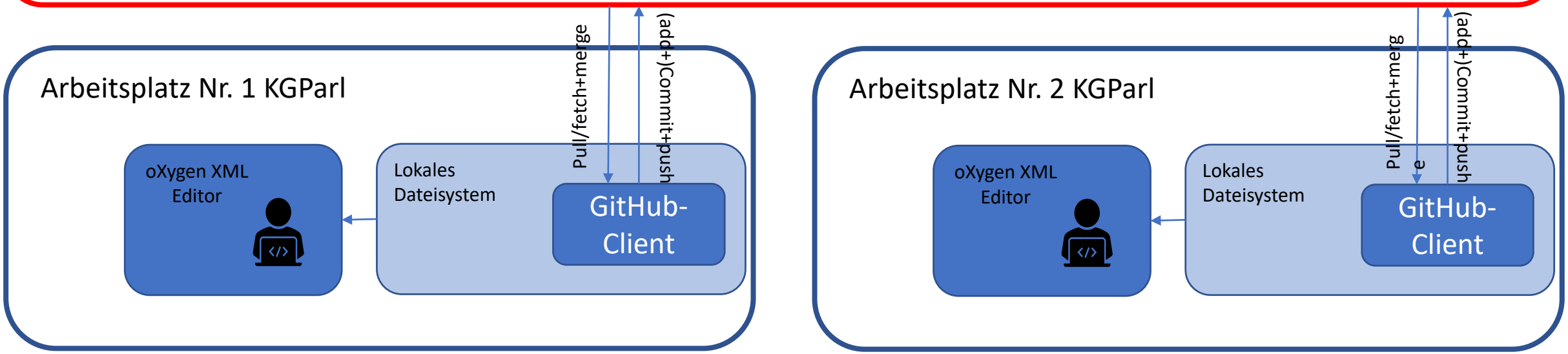
Webinar „KGParl und die Arbeit mit oXygen XML Editor, GitHub und eXist-db“ (17.05.2021)

- Projekt-Infrastruktur
- Git-“Terminologie“
- Einrichten des Arbeitsplatzes
  - 1.1 oXygen Editor
  - 1.2 Github-Client
  - 1.3 Aussehen Client, oXygen und File System
- Arbeitsworkflow
  - Neu anlegen/Bearbeiten eines Protokolls im oXygen Editor
  - Commit-Vorgang
  - Fetch/merge/discard

Projekt-Infrastruktur

GitHub-Projekt
(https://github.com/Fraktionsprotokolle-de)

- Protokolle-Repositorium (https://github.com/Fraktionsprotokolle-de/Protokolle.git)
- Altprotokolle (https://github.com/Fraktionsprotokolle-de/altprotokolle.git)
- Framework-Repositorium (https://github.com/Fraktionsprotokolle-de/framework.git)



git ”Terminologie” - 1

Ein entferntes Repositorium auf den lokalen Rechner kopieren:

- **clone**: erstellt eine Arbeitskopie des auf Github-Server gespeicherten Repositorums; die lokale Kopie besteht wie im Bild gezeigt aus drei Bereichen: dem **working directory**, dem **Index** und dem **HEAD.** Lokal an den Dateien vorgenommen Änderungen warden zunächst im working directory ausgeführt.

Eigene Änderungen hinzufügen:

- **add**: schlägt Änderungen für das entfernt liegende Repositorium vor; bei Nutzung des GitHub-Clients **entfällt dieser Schritt:** Sobald eine Datei geändert und lokal gespeichert wurde, erkennt der Client dies und führt automatisch ein “add” aus.
- **commit**: die Änderungen werden bestätigt, sie befinden sich nun im HEAD
- **push**: sendet die im HEAD befindlichen Änderungen an das entfernt liegende Repositorium, der Arbeitsprozess ist abgeschlossen.

Die drei Bereiche des Lokalen Repository

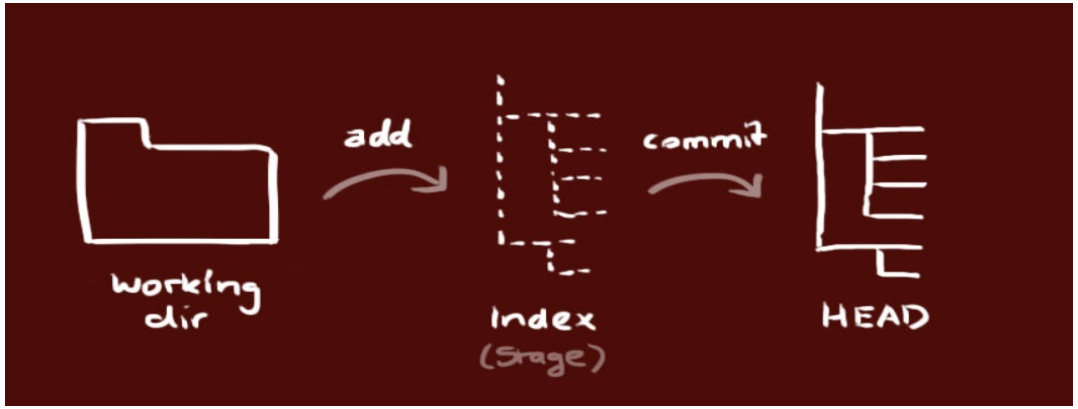


Bild: https://rogerdudler.github.io/git-guide/index.de.html

git ”Terminologie” - 2

Lokale Arbeitskopie aktualisieren:

- **pull**: aktualisiert die lokale Arbeitskopie mit den neusten Änderungen des entfernten Repositoriums (fetch + merge)
- **fetch**: Holt sich zunächst den aktuellen Stand des entfernten Repositoriums ohne aber gleichzeitig einen merge mit der lokalen Arbeitskopie durchzuführen. So kann zunächst abgeglichen werden, welche Dateien geändert wurden.
- **merge**: nachdem ein fetch entfernte Änderungen festgestellt hat, können diese mit der lokalen Arbeitskopie zusammengeführt werden.
- **revert**: macht die Änderungen im entfernten Repositorium rückgängig

git ”Terminologie” - 3

Konflikte:

- **“merge conflict”**: Kann bei einem pull oder fetch+merge auftreten, wenn entfernte Änderungen dieselben Dateien betrifft, die in der lokalen Arbeitskopie geändert wurden (diese wurden demnach auf einer “alten” Version der Daten vorgenommen, beide Datei-Versionen sind “auseinander gelaufen”).

- **“discard”**: Um den “merge conflict” aufzulösen, müssen die lokalen Änderungen zurückgenommen werden. “discard” **verwirft die lokalen Änderungen**, der lokale Stand der Datei ist wieder derselbe, auf dem die entfernten Änderungen vorgenommen wurden, sodass diese in das lokale Arbeitsverzeichnis integriert werden können. (*Um die eigenen Änderungen nicht zu verlieren, sollte die Datei vor dem “discard” separate gespeichert werden, um anschließend einen Dateivergleich durchführen und die Änderungen schneller in die aktualisierte Datei einfügen zu können.)* Um Merge-Konflikte zu vermeiden, sollte durch regelmäßiges “pullen” sichergestellt werden, dass das lokale Arbeitsverzeichnis auf dem aktuellen Stand des entfernten Repositoriums ist.