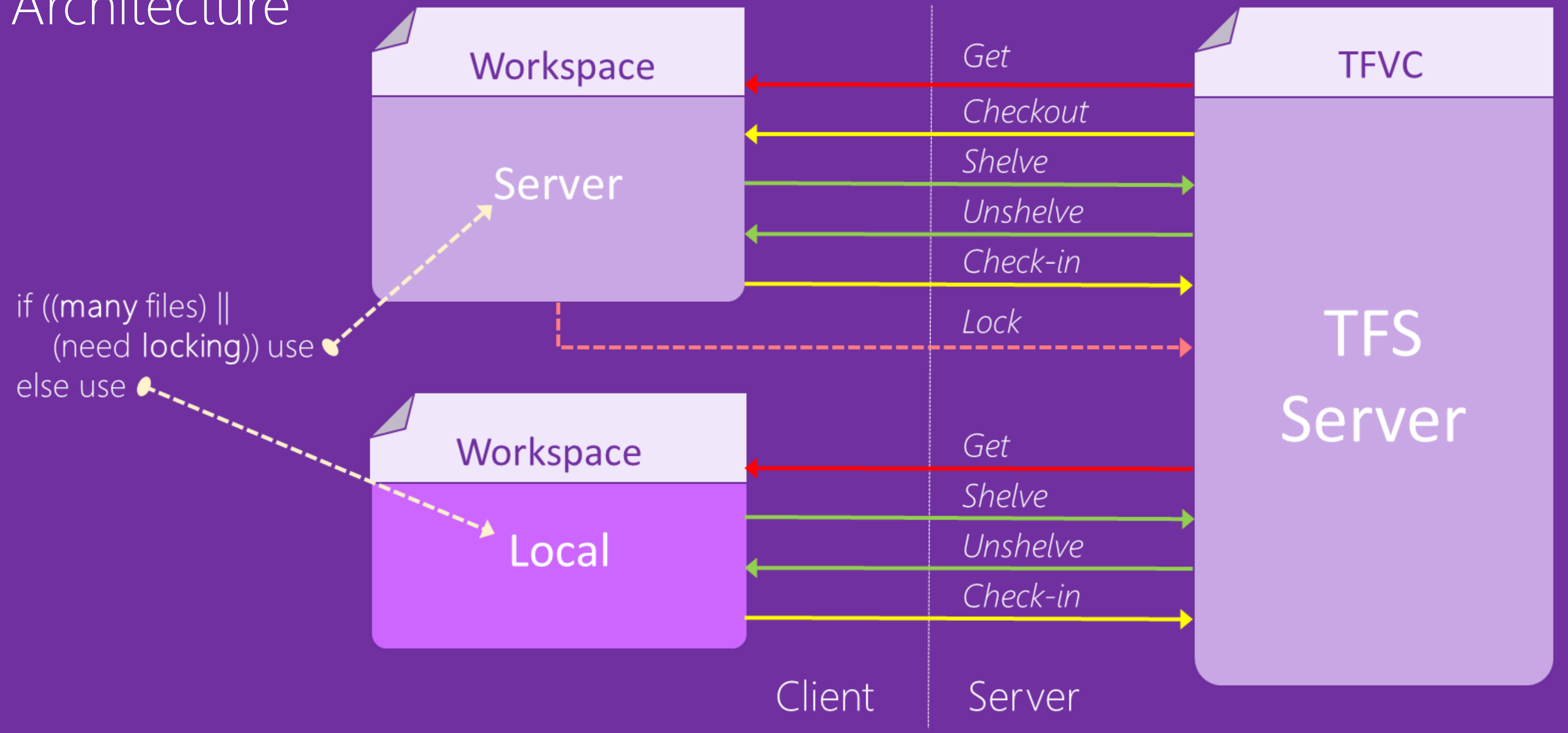
Version Control Cheat Sheet for TFVC and Git

Visual Studio
ALM Rangers

TFVC

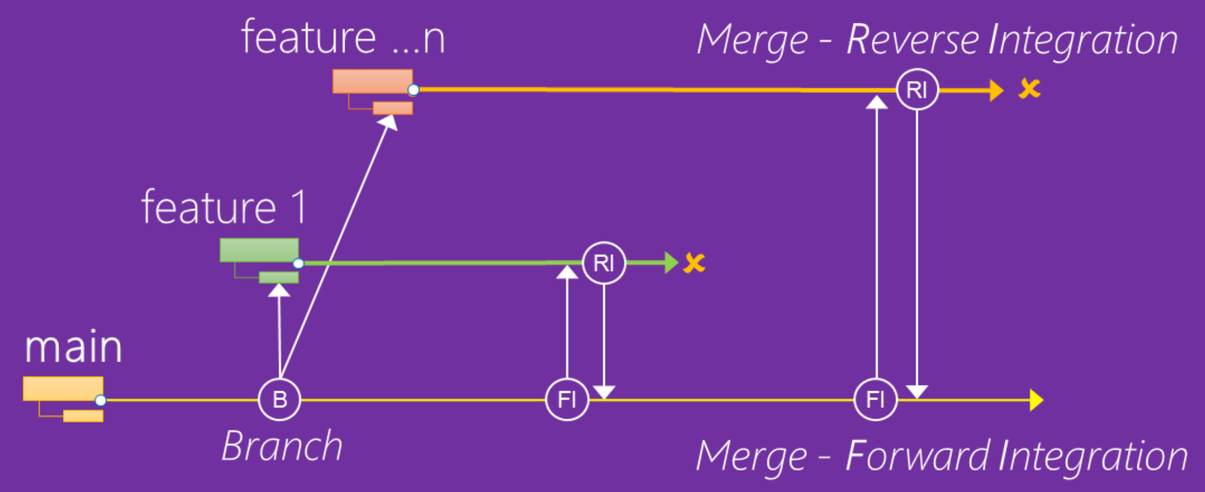
Architecture



Concepts

- **Branch** is an isolated copy of item metadata and version control history.
- **Changeset** is a logical container for changes of a single check-in.
- **Check-in** commits pending changes in workspace to server as a changeset.
- **Label** mutable grouping of specific version of a set of source files.
- **Shelveset** is a set of pending changes are temporarily saved on the server.
- **Workspace** is a local copy of your team's codebase. Create multiple workspaces and switch among them to work on different branches or copies of the codebase.

Feature Isolation Branching Example



Commands

TF *command*
Team Foundation Version Control (TVFC) command line tool with a variety of commands.

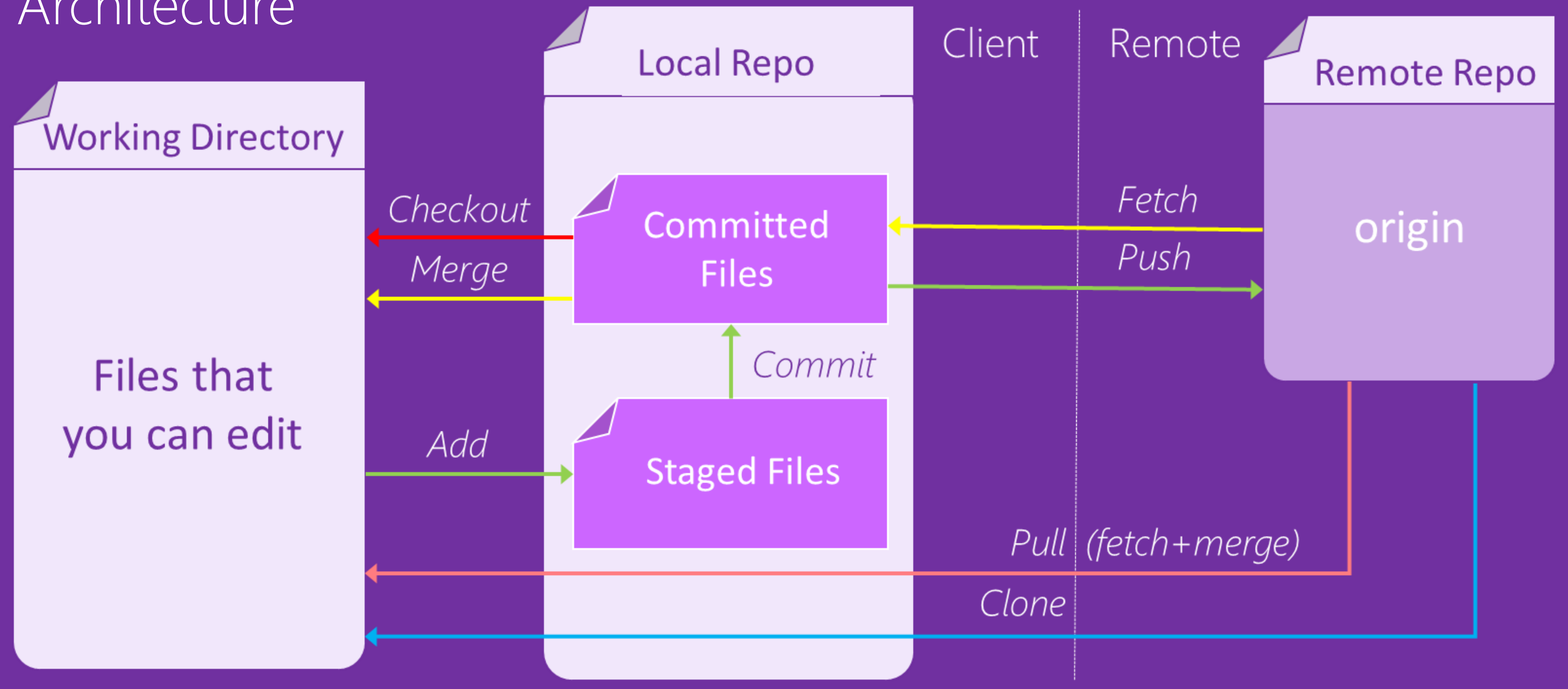
- everyday
  - add adds new files and folders from a local file system location
  - get retrieves a copy of items from TFVC to the workspace
  - checkin commits pending changes in current workspace to TFVC
  - checkout makes local file writable and changes status to "edit"
  - delete removes files and folders from TFVC and disk
  - history displays the revision history for files and folders
- branching
  - branch creates a copy of items, preserving a relationship to the original items
  - merge applies changes from one branch into another
- shelving
  - shelve stores a set of pending changes in TFVC without a commit
  - shelvesets used to view details of a shelveset or view shelvesets belonging to a specific use
  - unshelve restores shelved changes from TFVC to current workspace
- uncommon
  - changeset displays information about a changeset
  - destroy permanently deletes, version-controlled files from TFV. Admin only!
  - lock locks or unlocks to prevent against checkin/checkout of items
  - rename changes the name or the path of items
  - rollback reverts the changes of one or more changesets
  - undelete restores items that were previously deleted
  - workspace creates, deletes, displays, or modifies properties and mappings associated with a workspace.
- help

  Type **TF /?** on the command line for a complete list of commands and arguments.

TFSDeleteProject
Command line tool which deletes a team project from a TFS team project collection. It is a non-recoverable operation!

Git

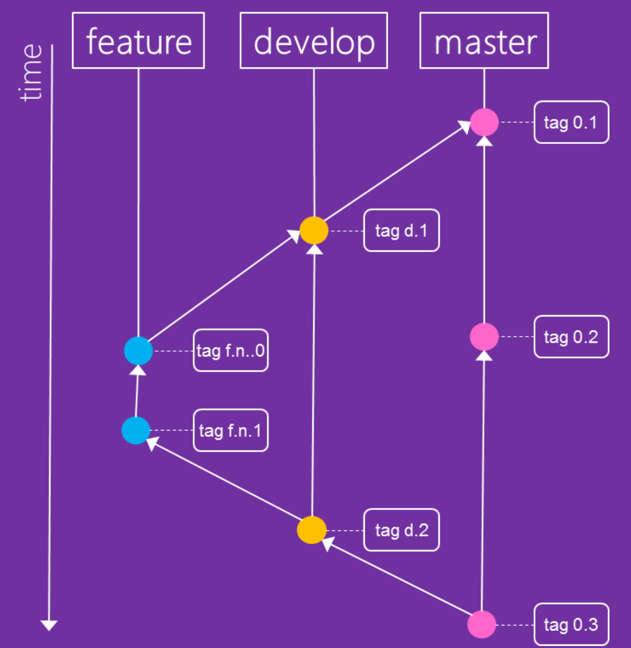
Architecture



Concepts

- **Branch** is a named pointer to a commit history in the repository. Used to isolate changes from each other.
- **Commit** (noun) is equivalent (mostly) to Changeset, but can also be an action.
- **HEAD** is a pointer to the branch your commits will be associated with.
- **Stash** is a set of pending changes are temporarily saved in the repository.
- **Tag** is a named pointer to a commit in the repository. Useful for marking a point-in-time in your repository.

Feature Isolation Branching Example



Links

msysgit.github.io	... Windows client downloads
git-scm.com	... documentation
github.com	... code host
bit.ly/gitscc	... Git source control provider VS2008-2012
bit.ly/gitext	... Git extensions

Commands

git *command*
git command line tool with a variety of commands.

- everyday
  - add adds the current content of existing paths
  - clone create a working copy from existing repository
  - commit (verb) all the staged local changes
  - fetch the latest changes from origin, not merging
  - pull the latest changes from origin and merge into working copies
  - push commits changes to origin
  - rm removes files from the working tree and from the index
  - status shows uncommitted changes in the working directory
  - tag mark a version or milestone
- branching
  - branch [name] creates branch called *name* based on HEAD
  - branch [–d name] deletes branch called *name*
  - checkout [id] switch to the *id* branch
  - diff shows changes to tracked files
  - merge two branches
- uncommon
  - blame show who changed what and when
  - bisect find regressions
  - grep search working directory
  - log show history of changes
  - show [id:file] show a specific *file* from a specific *id*
- help

  Type **git -help** on the command line for a complete list of commands and arguments.

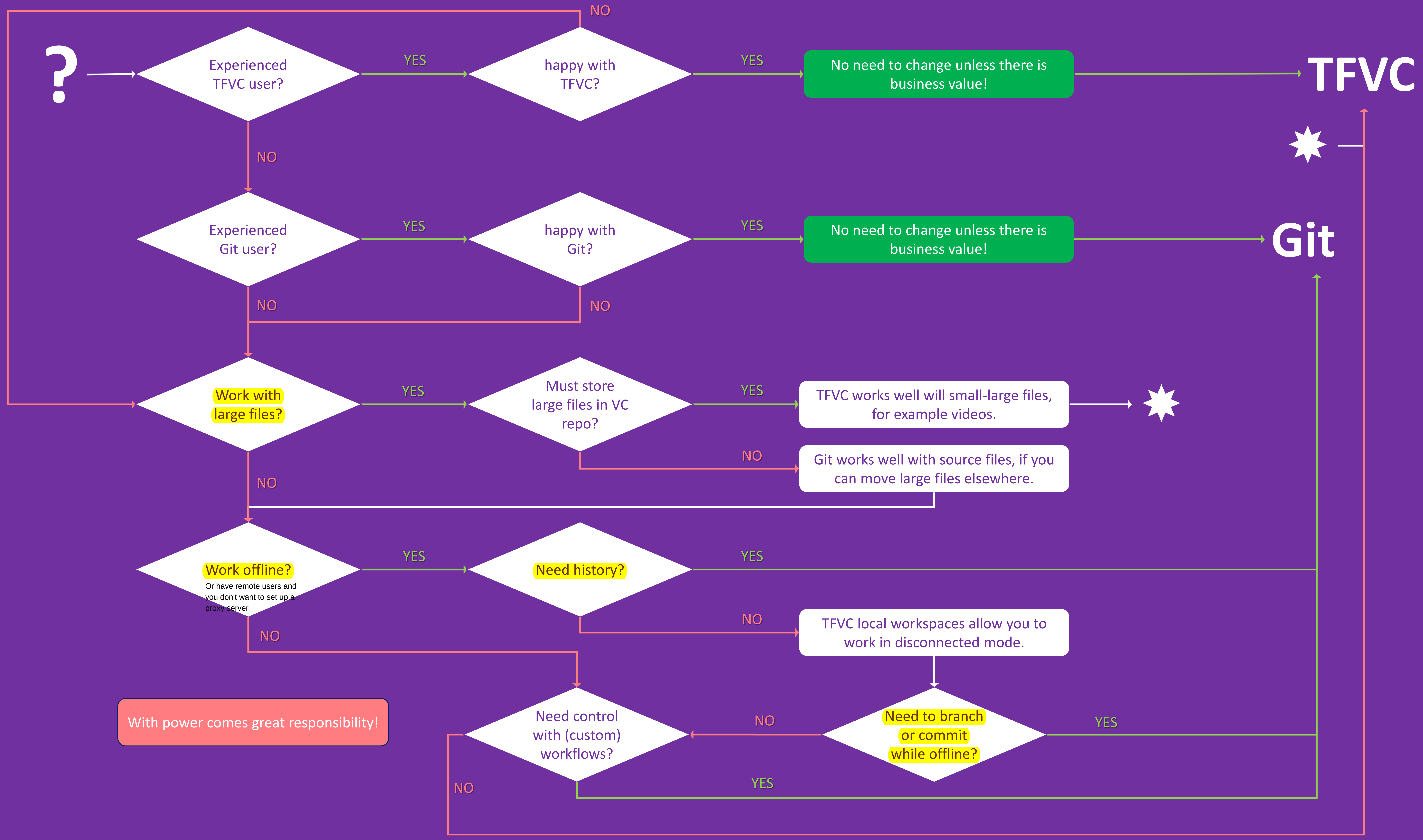
Microsoft

Visual Studio ALM Rangers Solutions – http://aka.ms/vsarsolutions

2014-03-20 v3

Version Control consideration aid for TFVC vs. Git

Visual Studio
ALM Rangers



?

Experienced TFVC user?

YES

happy with TFVC?

YES

No need to change unless there is business value!

TFVC

NO

NO

Experienced Git user?

YES

happy with Git?

YES

No need to change unless there is business value!

Git

NO

NO

Work with large files?

YES

Must store large files in VC repo?

YES

TFVC works well will small-large files, for example videos.

NO

Git works well with source files, if you can move large files elsewhere.

NO

Work offline?

Or have remote users and you don't want to set up a proxy server

YES

Need history?

YES

NO

TFVC local workspaces allow you to work in disconnected mode.

NO

With power comes great responsibility!

Need control with (custom) workflows?

NO

Need to branch or commit while offline?

YES

YES

NO

Microsoft

Visual Studio ALM Rangers Solutions – http://aka.ms/vsarsolutions

2014-03-20 v3