3. Verzovací systémy

1. Princip

- komplexní správa zdrojového kódu, souboru, adresáře
- hlavní jednotkou je *repoztář*, adresář, do něhož se provádějí *commity*
  - commit by měl splňovat: atomicita, popisnost, jednoznačnost
- push, pull, merge
- branch = větev, slouží k odložení práce

2. Využití

- verzování
- historie vývoje
- týmová spolupráce
  - řesení konfliktů: jednako pomocí zamykání (upravovatel při pushování lockne repo a ostatní commiteři mohou pouze číst), jednak pomocí kombinací incrementů (více úprav najednou, nutné sledovat incrementy, problémy u netextových souborů)
- větvení, případná diverzifika týmu

3. Druhy

Centralizované VCS - vše je na centrálním serveru, nutnost připojení k internetu, soubor se zamyká - nemůže dojít k chybě, když selže centrální server, tak jsme nenávratně přišli o data, CVS nebo Subversion

Distribuované VCS - každý uživatel má kopii repozitáře, kopie je po úpravě nasdílena ostatním, pokud uživatelovi selže počítač, ostatní mu poskytnou kopii repo, takže pro práci není třeba internet, ale při klonování je vysoká vytíženost, zástupci: Mercurial či GIT, mohou vzniknout kolize → vzniknutí tzv. hybridních VCS - jeden main repo, do něj všichni ukládají a zároveň u sebe mají kopii

4. Git

- byl vytvořen Linusem Torvaldsem, byl použit na vývoj Linuxího kernelu
- ovlivnět projekty bitkeeper, monotone
- charakteristika:
  - podpora pro nelineární vývoj, branche
  - distribuivita
  - uložiště přes HTTP, FTP, rsync
  - efektivní práce na velkých projektech
- kryptografická autentizace commitů
- toolkit, řada programů pro správu repo
- používání:
  1. `git init`
  2. `git add .`
  3. `git commit -m "message"`
  4. `git push -u origin main` (origin je kam a main je branch)