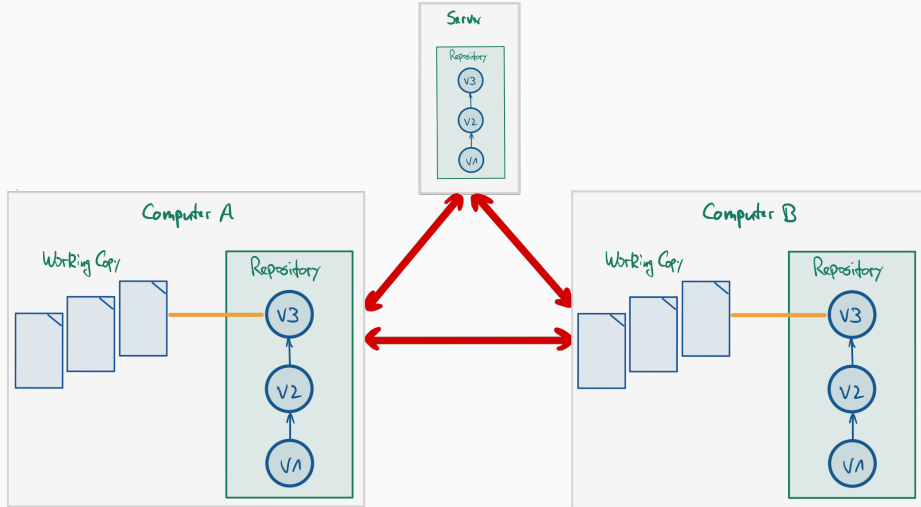


Branching-Strategien mit Git

Carsten Gips (FH Bielefeld)

Unless otherwise noted, this work is licensed under CC BY-SA 4.0.

Nutzung von Git in Projekten: Verteiltes Git (und Workflows)



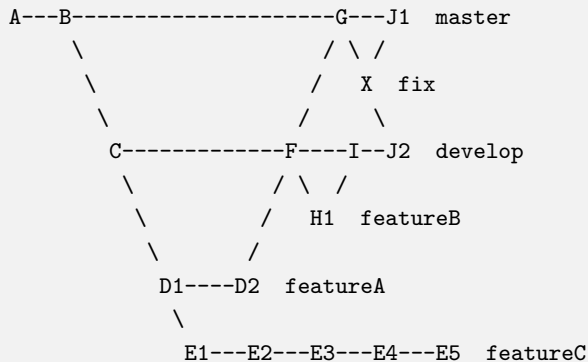
Umgang mit Branches: Themen-Banches

```
      I---J---K  wuppieV1
      /
    D---F  wuppie
    /
A---B---C---E  master
    \
      G---H  test
```

Umgang mit Branches: Langlaufende Branches

```
A---B---D master
  \
   C---E---I develop
     \
      F---G---H topic
```

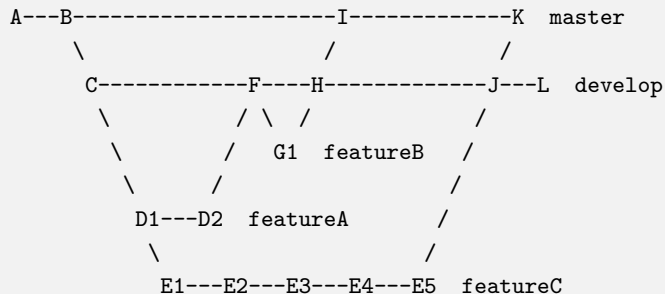
Komplexe Branching-Strategie: Git-Flow



Git-Flow: Hauptzweige *master* und *develop*

```
A---B-----E-----J  master
  \      /              /
   C---D---F---G---H---I---K  develop
```

Git-Flow: Weitere Branches als Themen-Branches



Git-Flow: Umgang mit Fehlerbehebung

```
A---B---D-----F1  master
      \    \      /
      \    E1---E2  fix
      \    \      \
      C1-----F2  develop
```


Vereinfachte Braching-Strategie: GitHub Flow

```
A---B---C---D-----E  master
      \   \   /
        \  ta1  topicA /
          \      /
            tb1---tb2---tb3  topicB
```

- Einsatz von Themenbranches für die Entwicklung
- Unterschiedliche Modelle:
 - Git-Flow: umfangreiches Konzept, gut für Entwicklung mit festen Releases
 - GitHub Flow: deutlich schlankeres Konzept, passend für kontinuierliche Entwicklung ohne echte Releases



Unless otherwise noted, this work is licensed under CC BY-SA 4.0.