Gitについて

コマンド動作の簡易的説明

Gitって何?

- バージョン管理ツール
  - ファイルの変更履歴を記録するシステム
  - サーバー上に本体のファイル群が存在する
    - GitHub: ネットワーク上(クラウド上)
    - GitBucket: 社内サーバー
      - ※ 上記は一例
  - 集中管理方式
  - 分散管理方式

Gitの運用モデル(Webmaster's Git-Flowについて)

- masterブランチ
  - リリース可能な完全品質を保証するブランチ
  - stagingブランチからのマージのみで更新される
- stagingブランチ
  - リリース前の最終確認を行うブランチ
  - masterブランチと同じ内容になる
- releaseブランチ(主に操作するブランチ)
  - リリース内容毎に管理される(例: リリース日やバージョン名)
  - 1つ前のreleaseブランチから派生(ない場合はstagingブランチから派生)

上記ブランチは権限上位者(管理者)が操作するのでエンジニアは関係しないことが多い

Gitの運用モデル(ブランチについて)

- featureブランチ
  - 追加機能及び修正作業を行うためのブランチ
  - releaseブランチから派生させる
  - 作業完了してレビュー(プルリク)が通ったら、 releaseブランチにマージする
    - マージされたら該当のfeatureブランチは削除する

用語解説

- リポジトリ(repository)
  - ファイル, プログラム, 設定情報などの「保管場所」
- リモート(remote)
  - サーバ上に存在する ex) リモートリポジトリ
- ローカル(local)
  - PC上に存在する ex) ローカルリポジトリ
- リモートリポジトリ
  - Gitではこれを中心として作業を行う
  - 各メンバーの作業の終着点
- ローカルリポジトリ
  - リモートリポジトリを元に自分の作業を行う
  - リモートレポジトリに反映させるためには別途手順が必要(後述)

用語解説

- クローン(clone)
  - ブランチ(運用モデルのrelease又はfeature)を指定してリモート上からローカル上にコピーする
- リモート追跡ブランチ(ex: origin/master)
  - クローンしてきたブランチの情報をローカル上に保持し、リモート上と比較するために存在する(編集は不可)
- マージ(merge)
  - 変更情報をブランチに更新する(ローカル→リモート/リモート→ローカル)
  - 新たな起点となる
  - ファストフォワード(fast-forward)
    - 該当ブランチをメインブランチ(今回はreleaseブランチ)にする → 集団作業でやるべきではない
    - ログがどのブランチからのマージか判別できなくなるため
- フェッチ(fetch)
  - リモートブランチの最新状態を、リモート追跡ブランチに反映
  - マージ処理はないので、差分の確認などで使用する

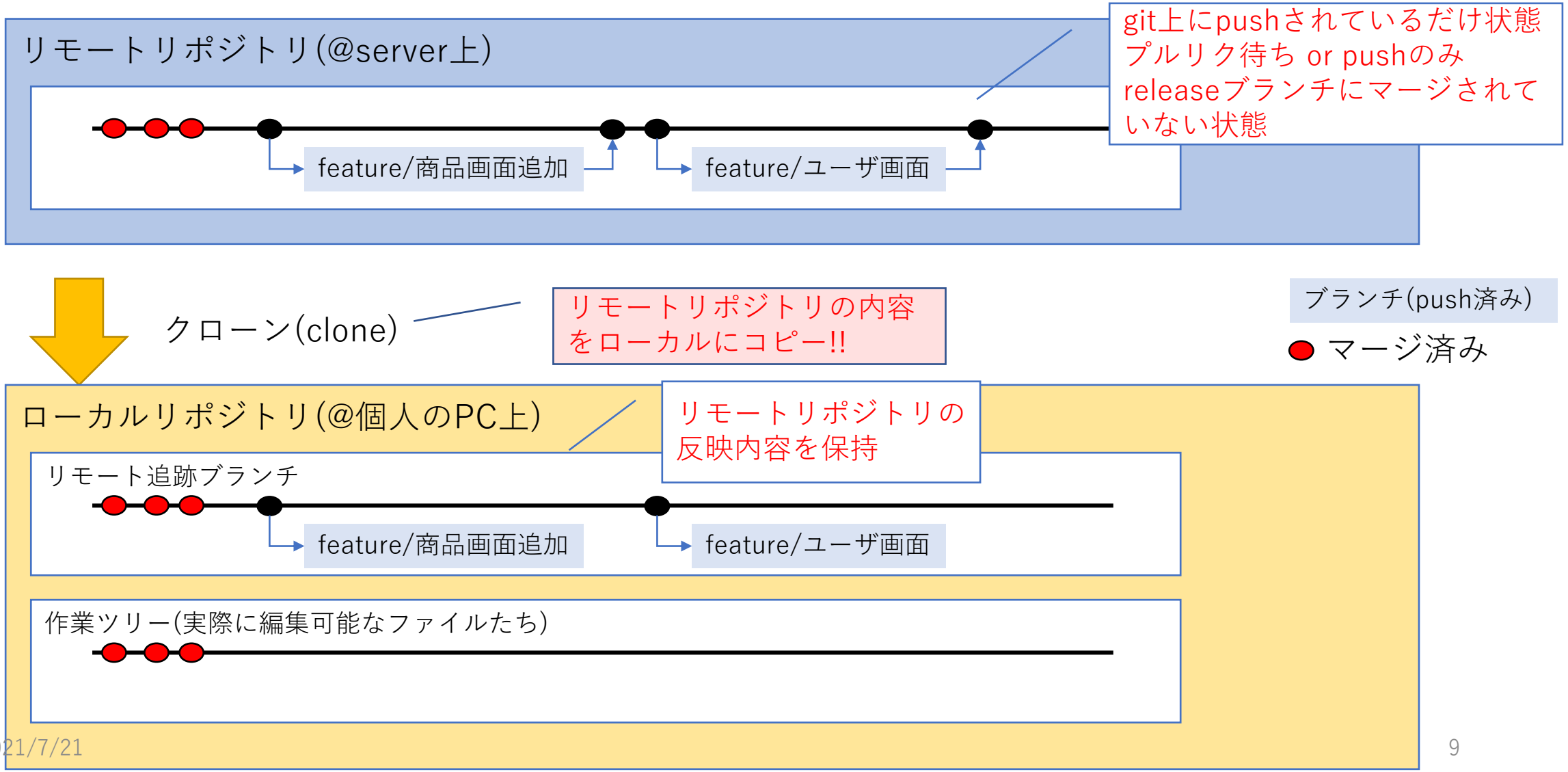
用語解説

- プル(pull)
  - fetch + merge
  - リモートブランチの内容の差分を確認せずにmergeする
- リベース(rebase)
  - ブランチの起点を最新に付け替える→きれいな直線的な履歴グラフになる(どこでバグが発生したのか記録を正確に残したい場合)
  - しかし、コミット日時を変更することになるので非直線的な履歴グラフを推奨するエンジニアもいます
- アド(add)
  - 変更したファイルをコミット対象(indexに追加)にする
- コミット(commit)
  - アドした内容を対象にコミットメッセージを記述(どんな変更したか)して、ローカルブランチに変更を反映する
- プッシュ(push)
  - ローカルブランチの内容をリモートブランチに反映させる
  - featureブランチ単位の場合、プルリクが発生するため即時マージはしない
- プルリクエスト(pull request)
  - レビューの機会を与える(コミュニケーションの場を与える): バグの発生を抑える
  - 結果問題なしの場合: releaseブランチにマージする
  - 問題ありの場合: 再度、担当者に修正を依頼する

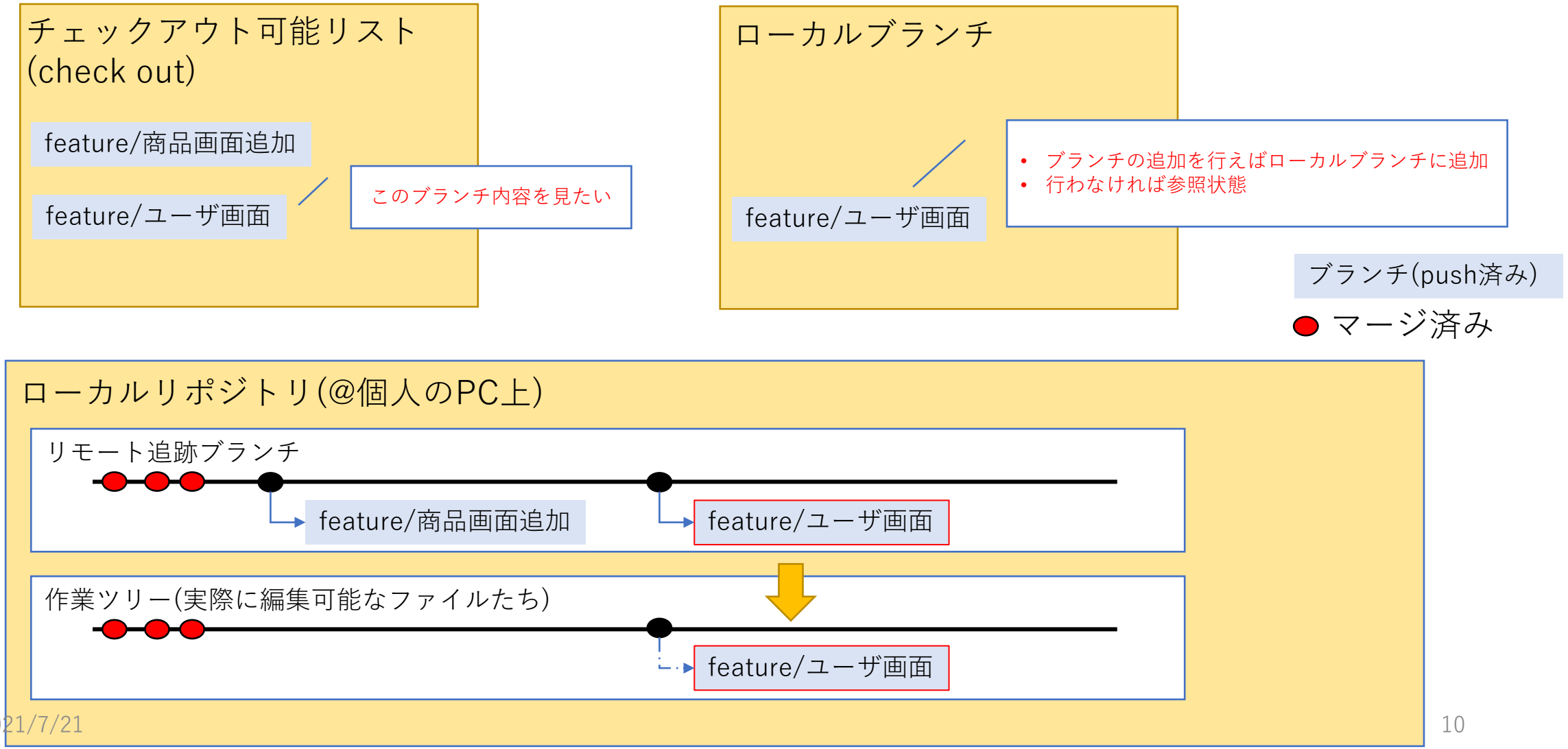
環境

- TortoiseGitを使用
  - コマンドではなくGUIでの操作を前提
    - ただ図解のイメージはコマンドの場合も同じ
- 使用Git
  - GitBucket or GitHubどちらでも
  - 今回は、イメージしやすいGitBucketを想定

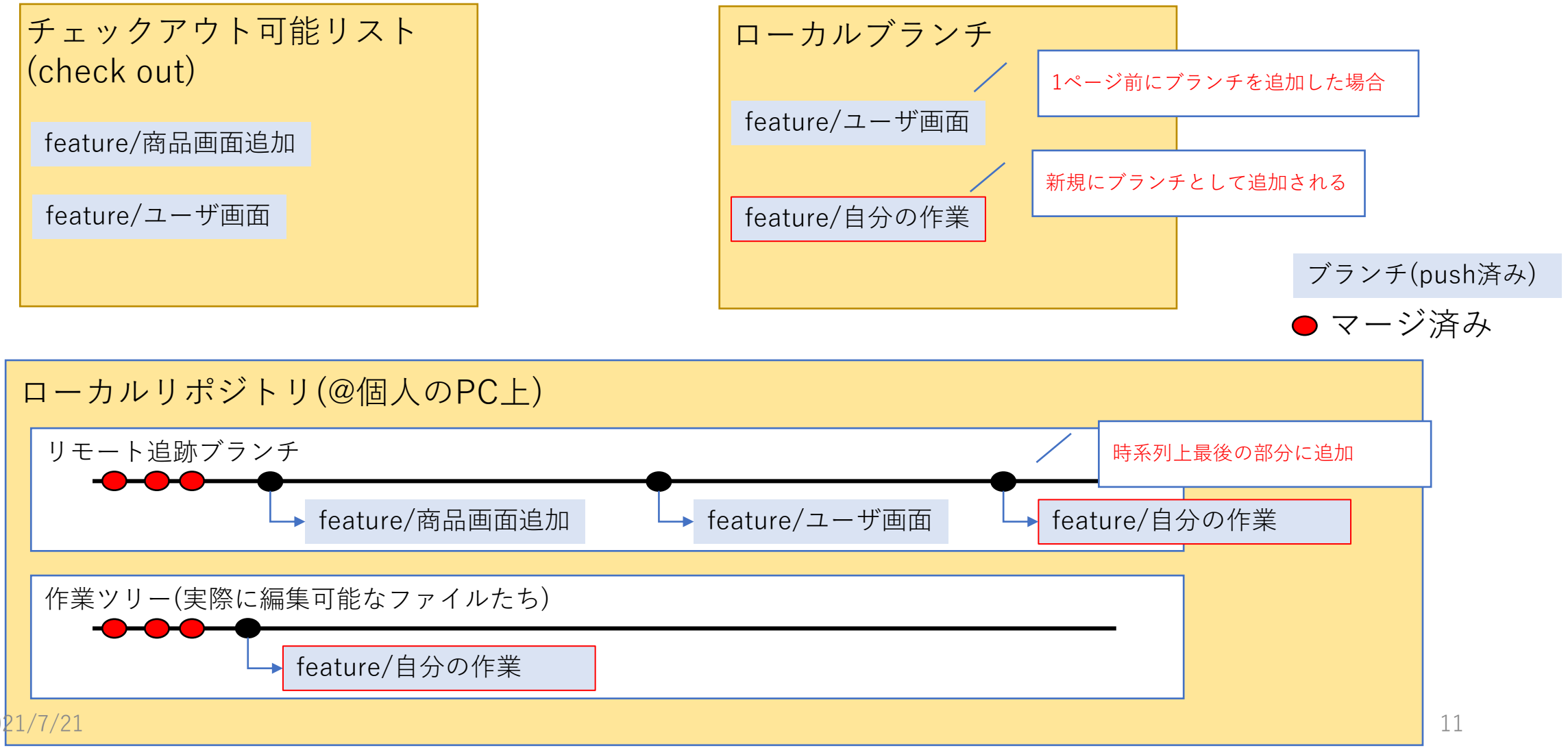
図解(例: ECサイト)


リモートリポジトリ(@server上)
feature/商品画面追加
feature/ユーザ画面
git上にpushされているだけ状態
プルリク待ち or pushのみ
releaseブランチにマージされて
いない状態
クローン(clone)
リモートリポジトリの内容
をローカルにコピー!!
ブランチ(push済み)
マージ済み
ローカルリポジトリ(@個人のPC上)
リモートリポジトリの
反映内容を保持
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
作業ツリー(実際に編集可能なファイルたち)

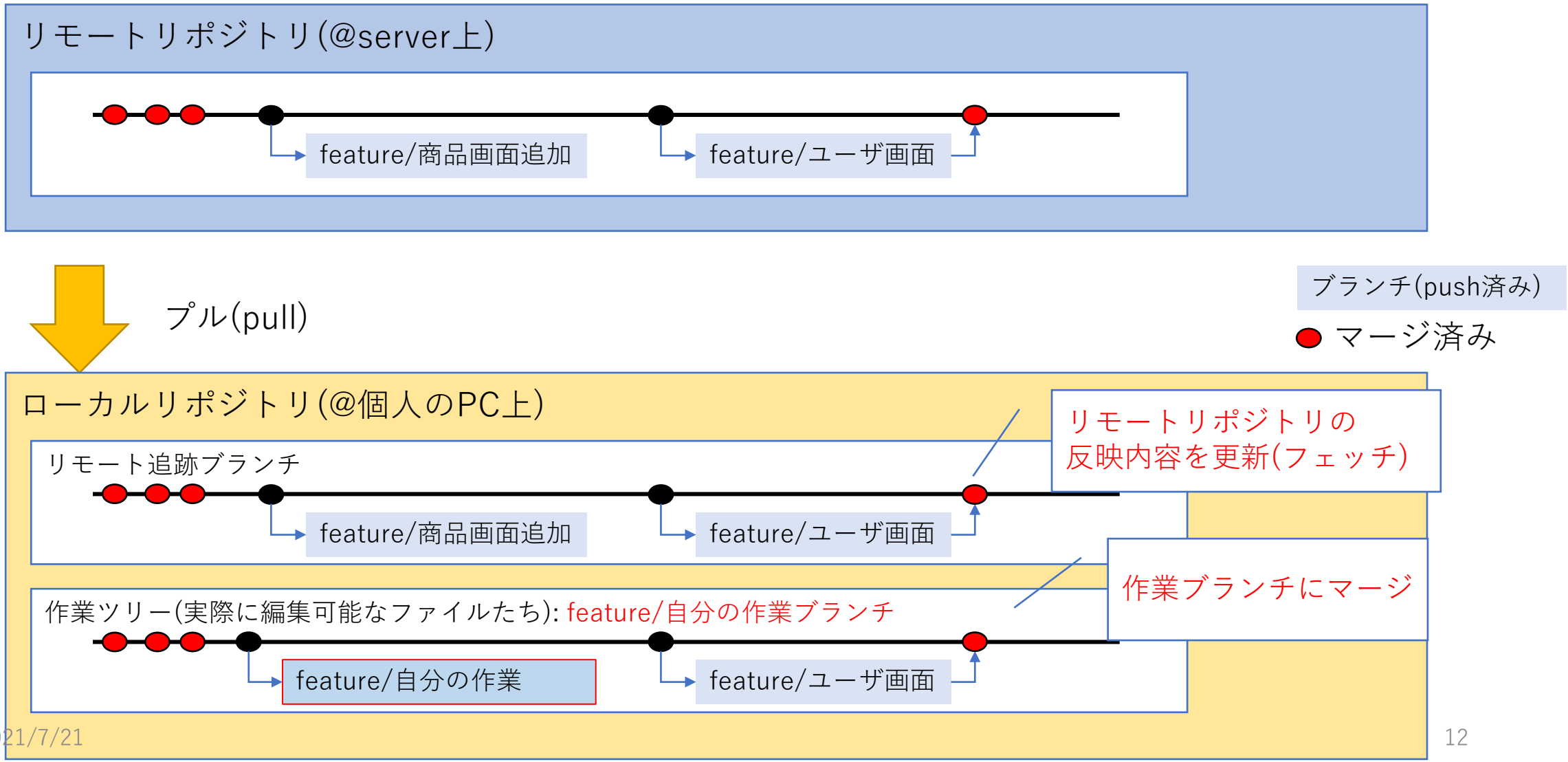
図解(チェックアウト)


チェックアウト可能リスト
(check out)
feature/商品画面追加
feature/ユーザ画面
このブランチ内容を見たい
ローカルブランチ
feature/ユーザ画面
ブランチの追加を行えばローカルブランチに追加
行わなければ参照状態
ブランチ(push済み)
マージ済み
ローカルリポジトリ(@個人のPC上)
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
作業ツリー(実際に編集可能なファイルたち)
feature/ユーザ画面

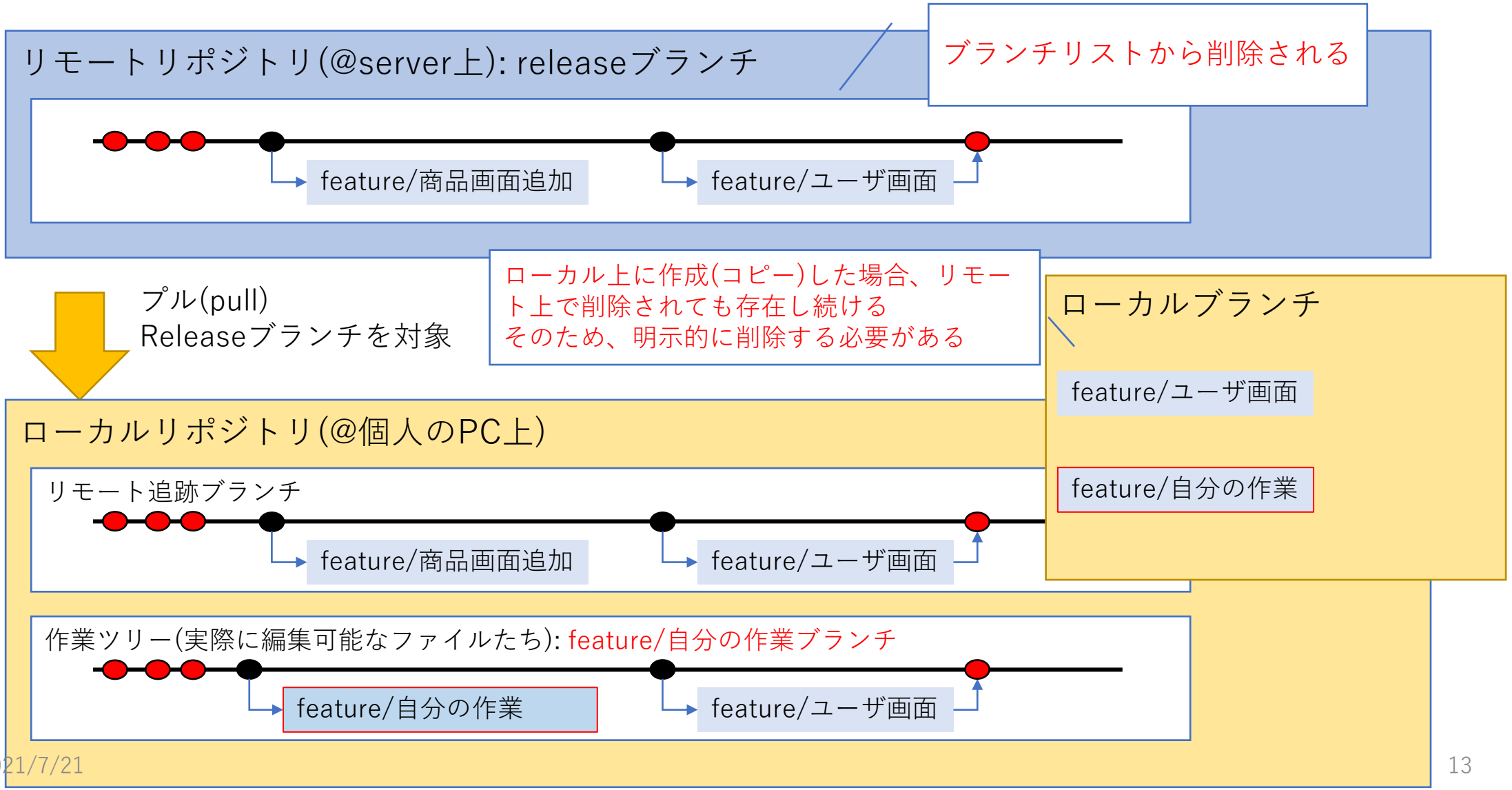
図解(ブランチ作成)


チェックアウト可能リスト
(check out)
feature/商品画面追加
feature/ユーザ画面
ローカルブランチ
feature/ユーザ画面
1ページ前にブランチを追加した場合
feature/自分の作業
新規にブランチとして追加される
ブランチ(push済み)
マージ済み
ローカルリポジトリ(@個人のPC上)
リモート追跡ブランチ
時系列上最後の部分に追加
feature/商品画面追加
feature/ユーザ画面
feature/自分の作業
作業ツリー(実際に編集可能なファイルたち)
feature/自分の作業

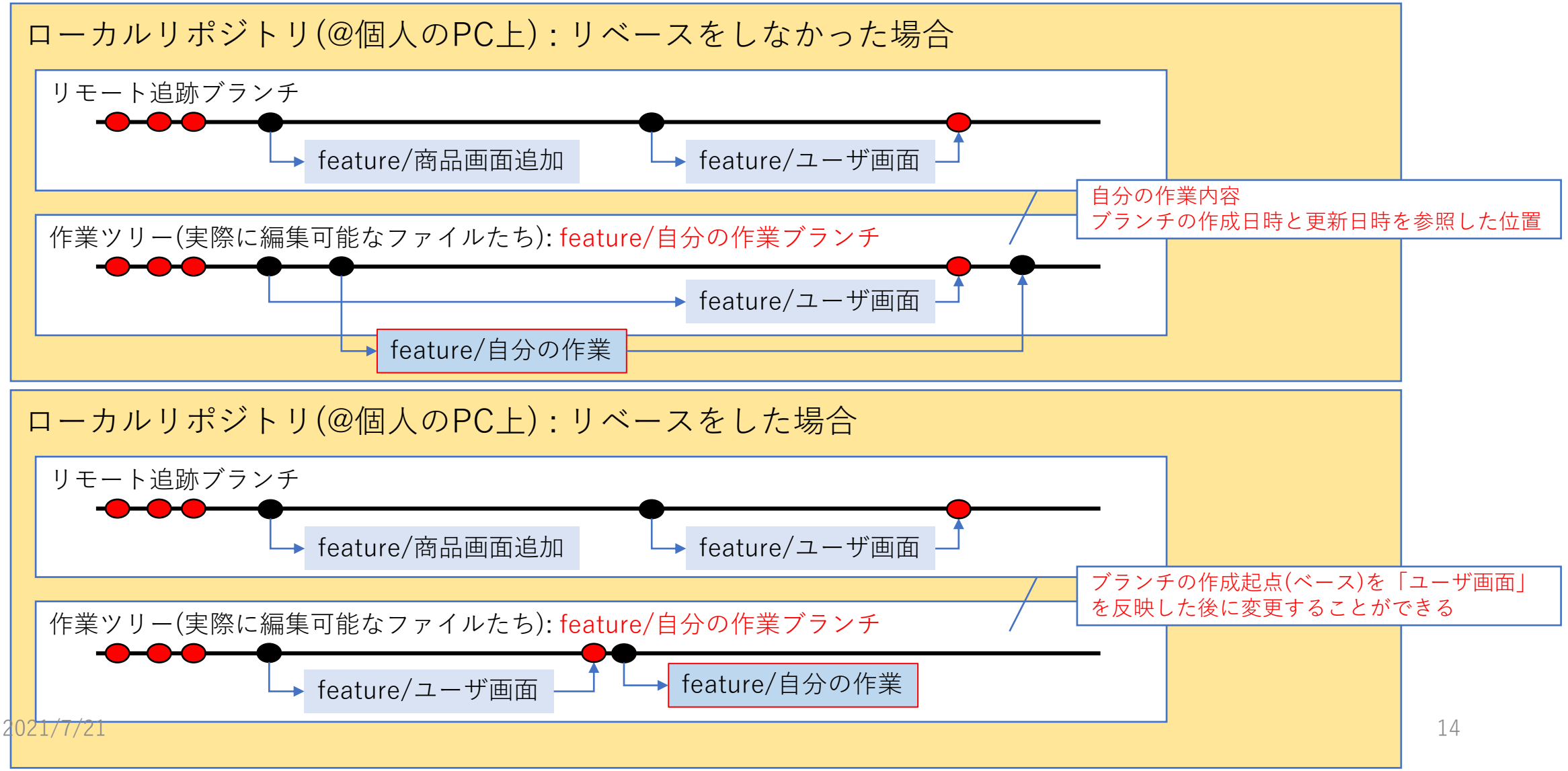
図解(例: プル)


リモートリポジトリ(@server上)
feature/商品画面追加
feature/ユーザ画面
プル(pull)
ブランチ(push済み)
マージ済み
ローカルリポジトリ(@個人のPC上)
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
リモートリポジトリの
反映内容を更新(フェッチ)
作業ツリー(実際に編集可能なファイルたち): feature/自分の作業ブランチ
feature/自分の作業
feature/ユーザ画面
作業ブランチにマージ

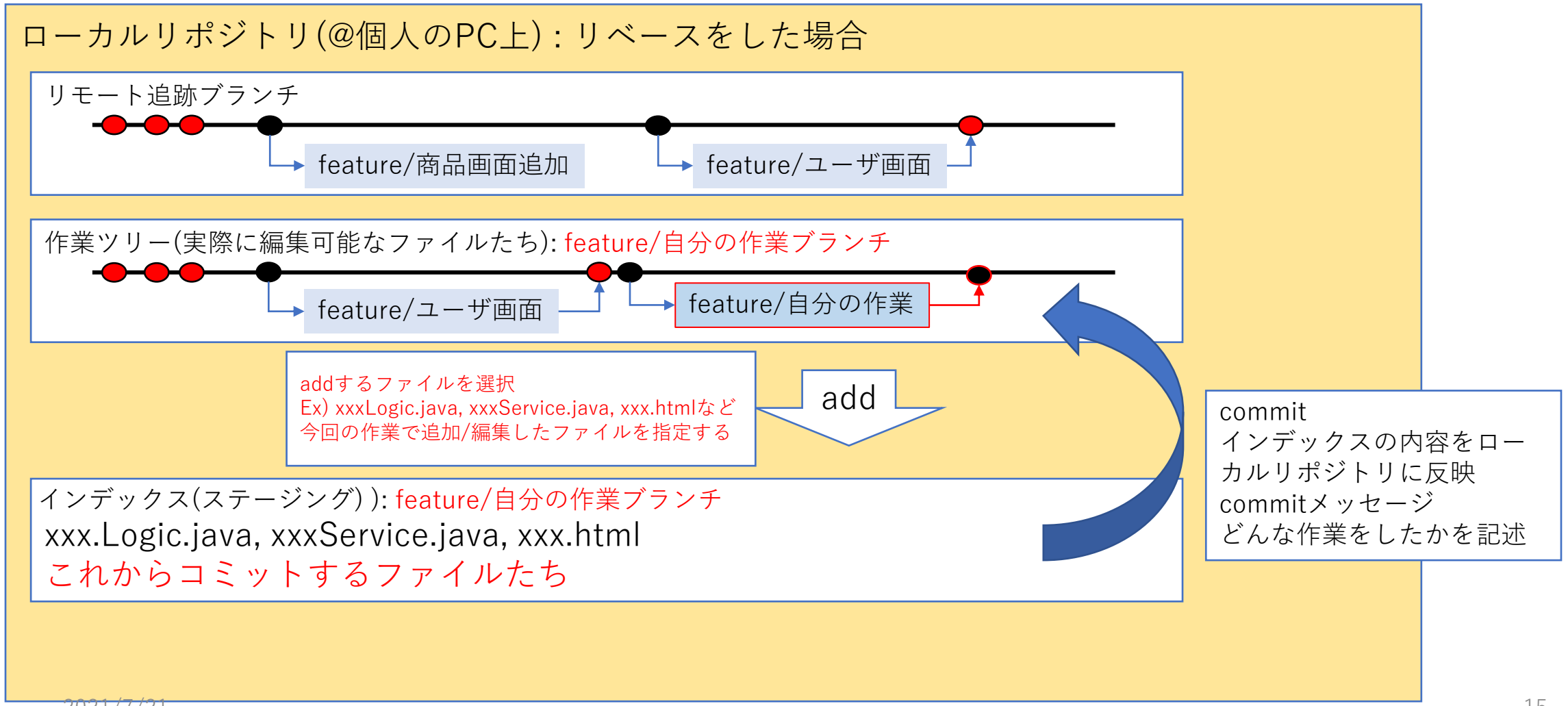
図解(例: プル - ブランチの動き)



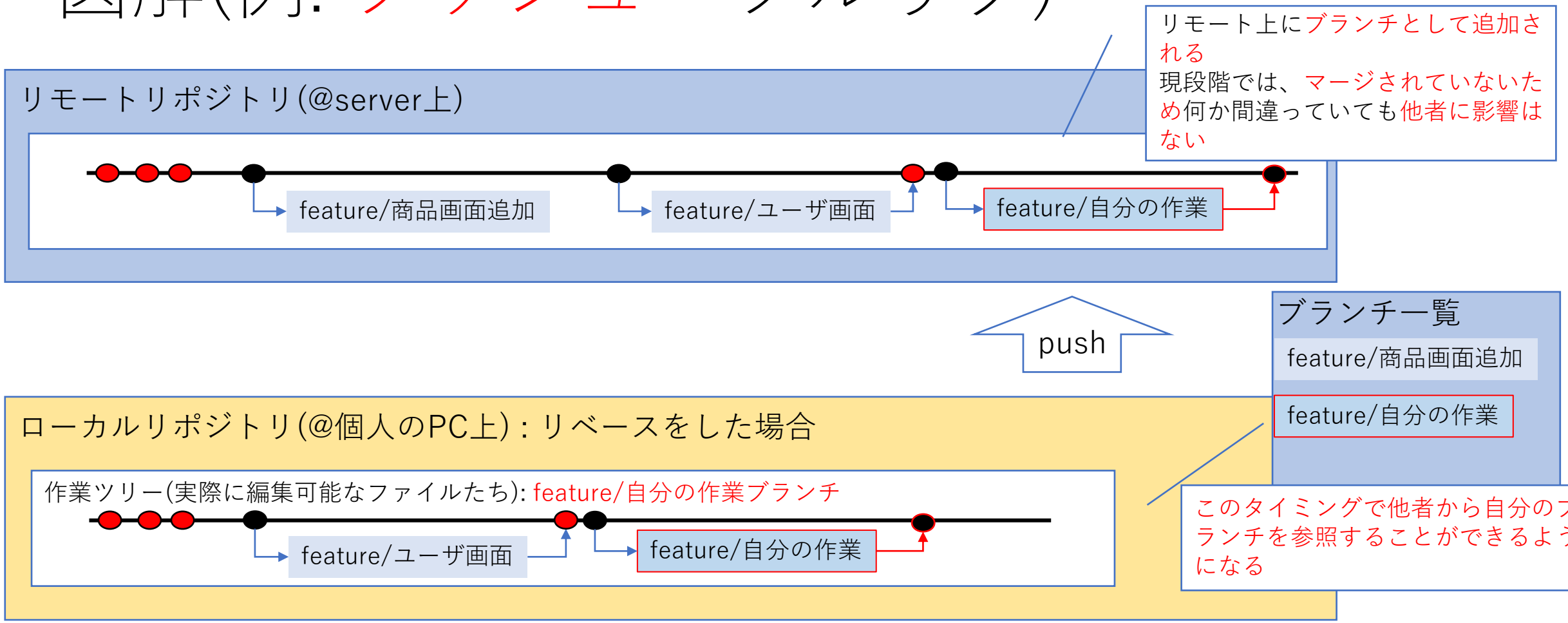
図解(例: リベース)


ローカルリポジトリ(@個人のPC上) : リベースをしなかった場合
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
作業ツリー(実際に編集可能なファイルたち): feature/自分の作業ブランチ
feature/ユーザ画面
feature/自分の作業
自分の作業内容
ブランチの作成日時と更新日時を参照した位置
ローカルリポジトリ(@個人のPC上) : リベースをした場合
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
作業ツリー(実際に編集可能なファイルたち): feature/自分の作業ブランチ
feature/ユーザ画面
feature/自分の作業
ブランチの作成起点(ベース)を「ユーザ画面」
を反映した後に変更することができる

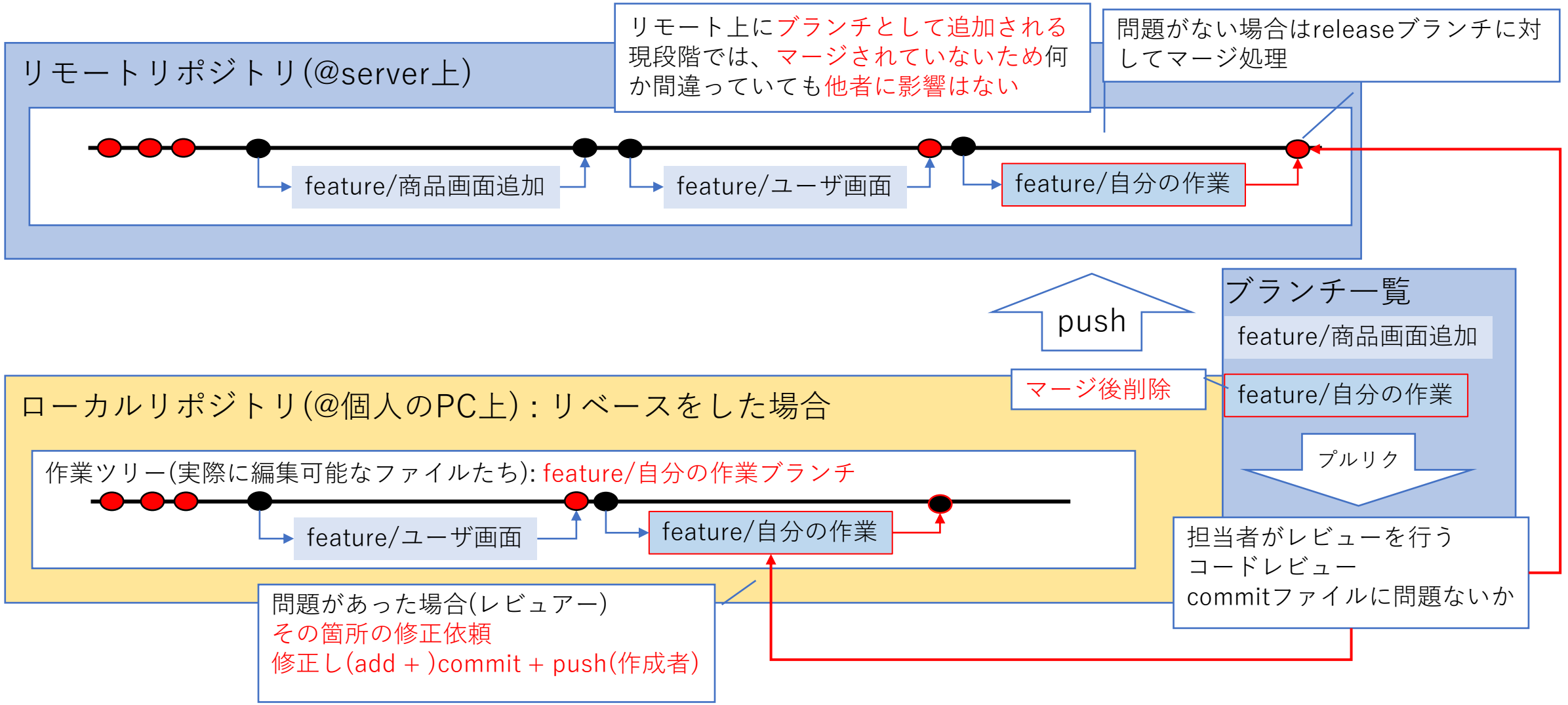
図解(例: アド→コミット)


ローカルリポジトリ(@個人のPC上) : リベースをした場合
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
作業ツリー(実際に編集可能なファイルたち): feature/自分の作業ブランチ
feature/ユーザ画面
feature/自分の作業
addするファイルを選択
Ex) xxxLogic.java, xxxService.java, xxx.htmlなど
今回の作業で追加/編集したファイルを指定する
add
インデックス(ステージング)): feature/自分の作業ブランチ
xxx.Logic.java, xxxService.java, xxx.html
これからコミットするファイルたち
commit
インデックスの内容をローカルリポジトリに反映
commitメッセージ
どんな作業をしたかを記述

図解(例: プッシュ – プルリク)



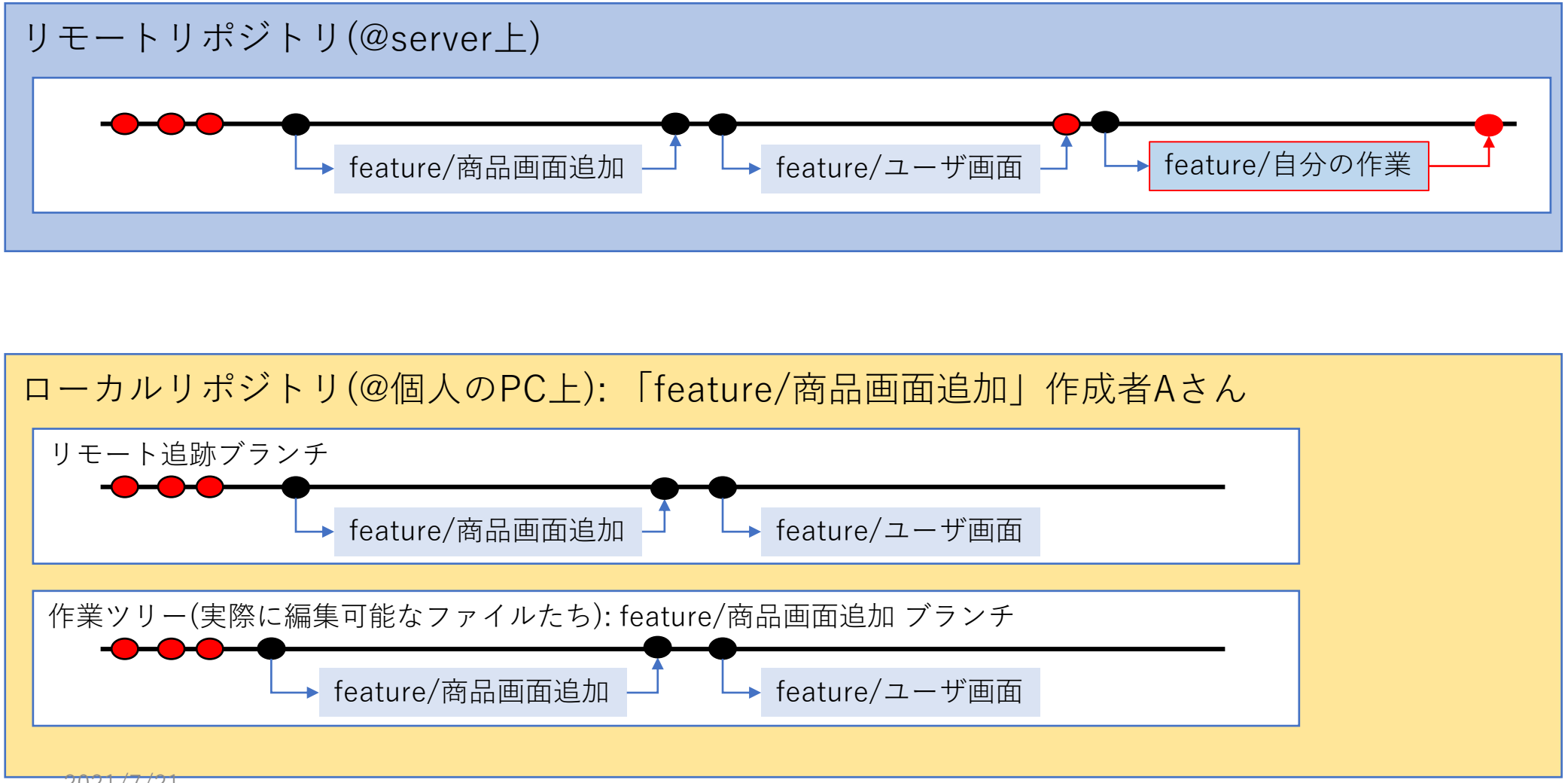
図解(例: プッシュ ー プルリク)



問題

- 「feature/商品画面追加」をきれいにreleaseブランチにマージする方法は?
- 前提条件
  - 条件としては、プルリクはOKをもらっている
  - リモートレポジトリのreleaseブランチの状態は前頁を参照
  - 競合がないため、Web上からマージ可能
    - これを行う場合のメリット、デメリットも教えて

問題


リモートリポジトリ(@server上)
feature/商品画面追加
feature/ユーザ画面
feature/自分の作業
ローカルリポジトリ(@個人のPC上): 「feature/商品画面追加」作成者Aさん
リモート追跡ブランチ
feature/商品画面追加
feature/ユーザ画面
作業ツリー(実際に編集可能なファイルたち): feature/商品画面追加 ブランチ
feature/商品画面追加
feature/ユーザ画面