

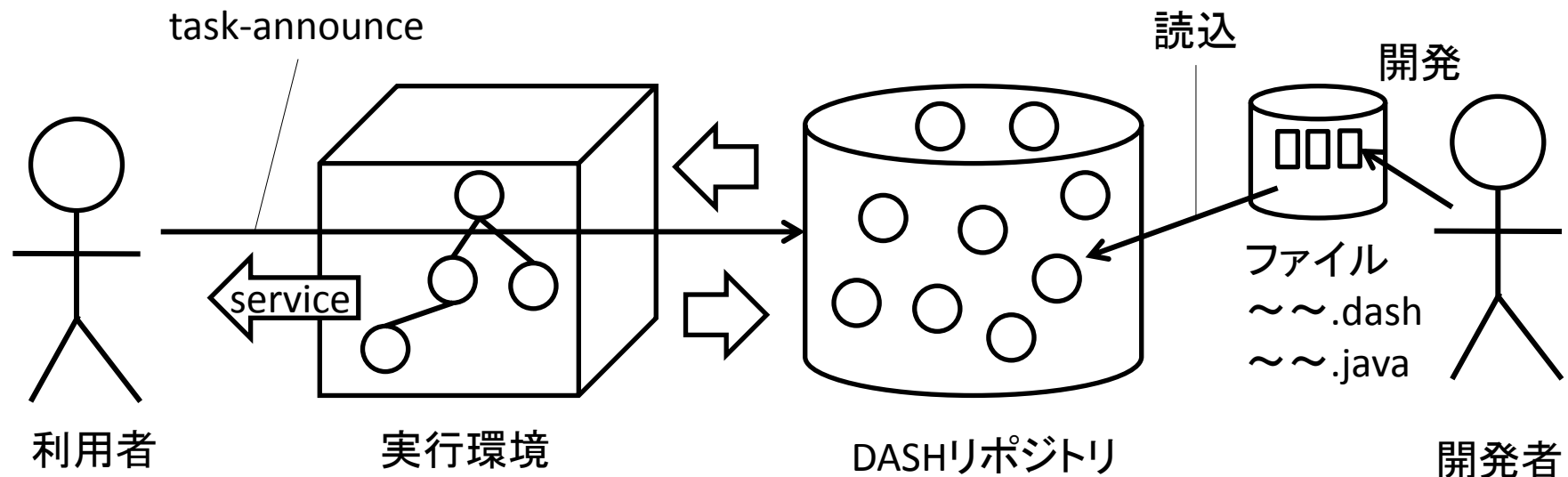
DASHリポジトリとgitリポジトリ 接続のための設計

千葉工業大学情報工学科

藤田茂、柄沢桃子(B4)

DASHエージェント開発の利点

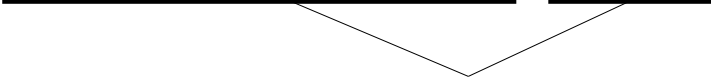
1. ソフトウェア部品の自己組織化による自律的サービス提供
2. 知識型のソフトウェア部品管理
3. 動作履歴の利用



「版管理」 ソフトウェア開発の課題

良く聞く例：ファイル名-日付-版番号.xls

課題：仕様書-201607220814-1023.xls



手作業管理なので間違える

変更理由が、ファイル中に「コメント」で記述
時として、それも無い

複数人が同時に変更すると「競合」が発生
時として、気づかずに上書きされる

「版管理システム」の課題

SCCS, CVS, Subversion, Mercurial, Visual SourceSafe, Team Foundation Server, git, etc.

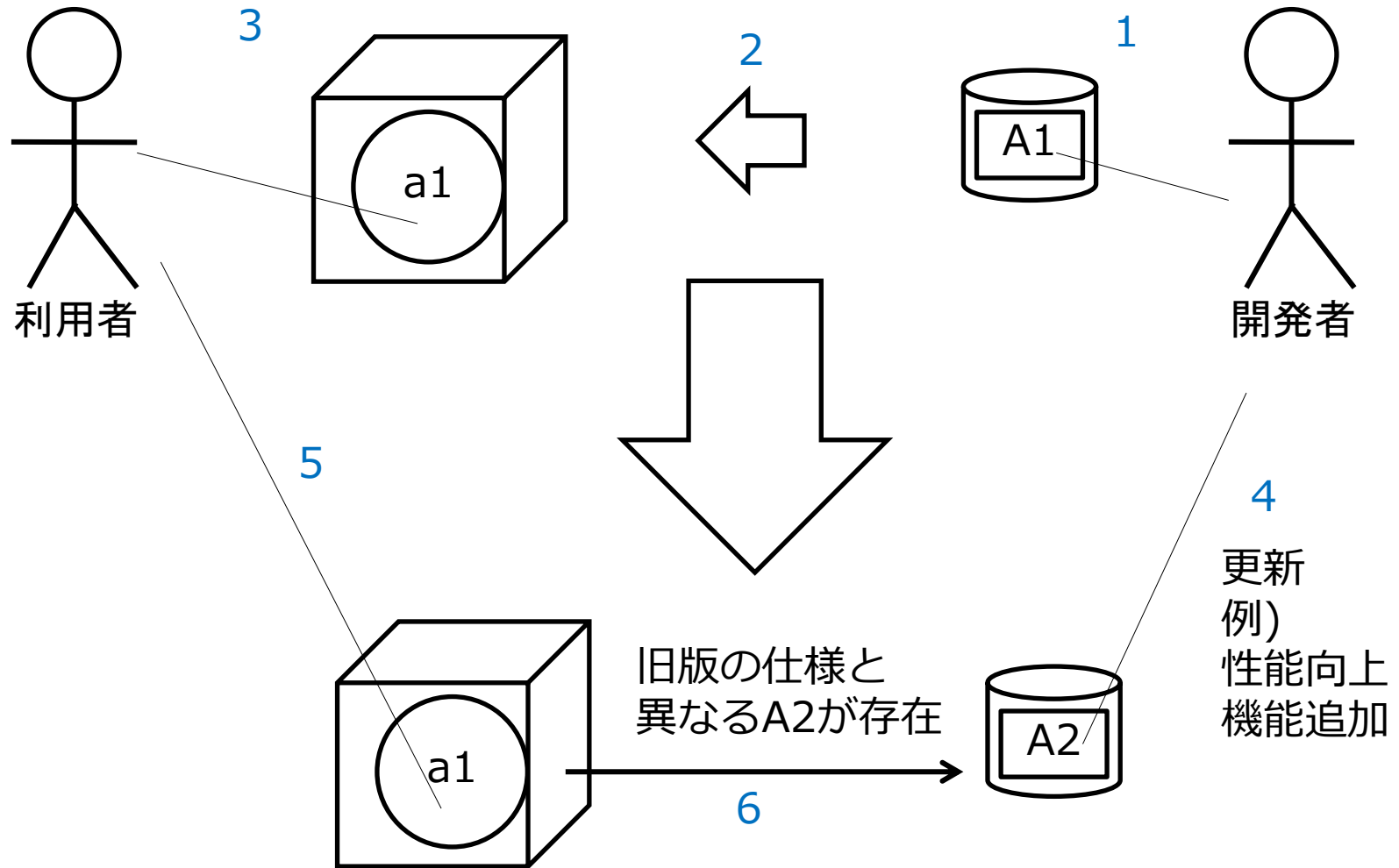
「プロジェクト」により「ソフトウェア開発支援」
ゆるやかな連携は考慮されていない？

ソフトウェア開発の知識蓄積は属人性高い

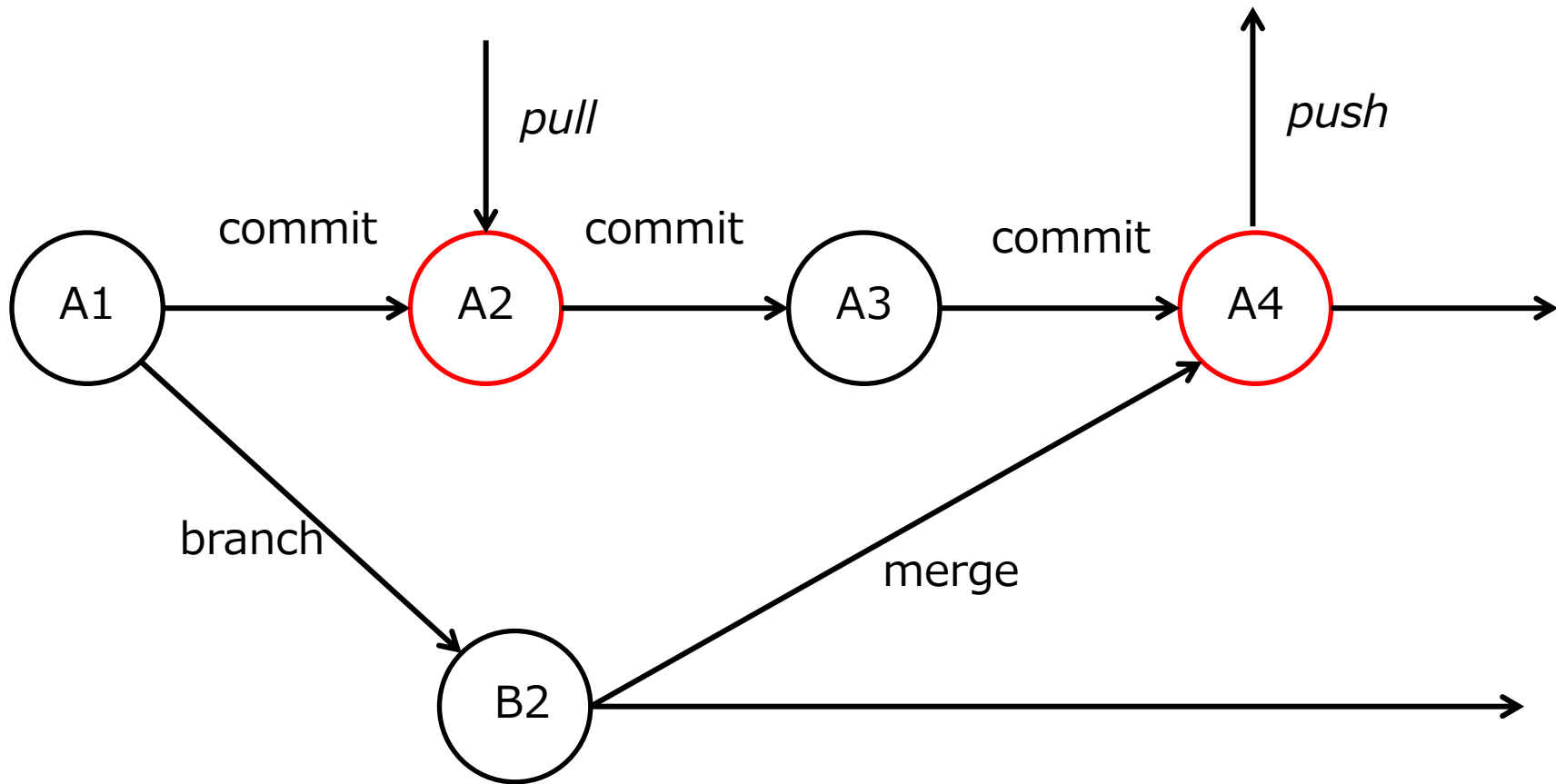
競合解消は人手、変更理由は人手で残る

操作が煩雑で「使いにくい」と公言する人も

エージェント開発のユースケース



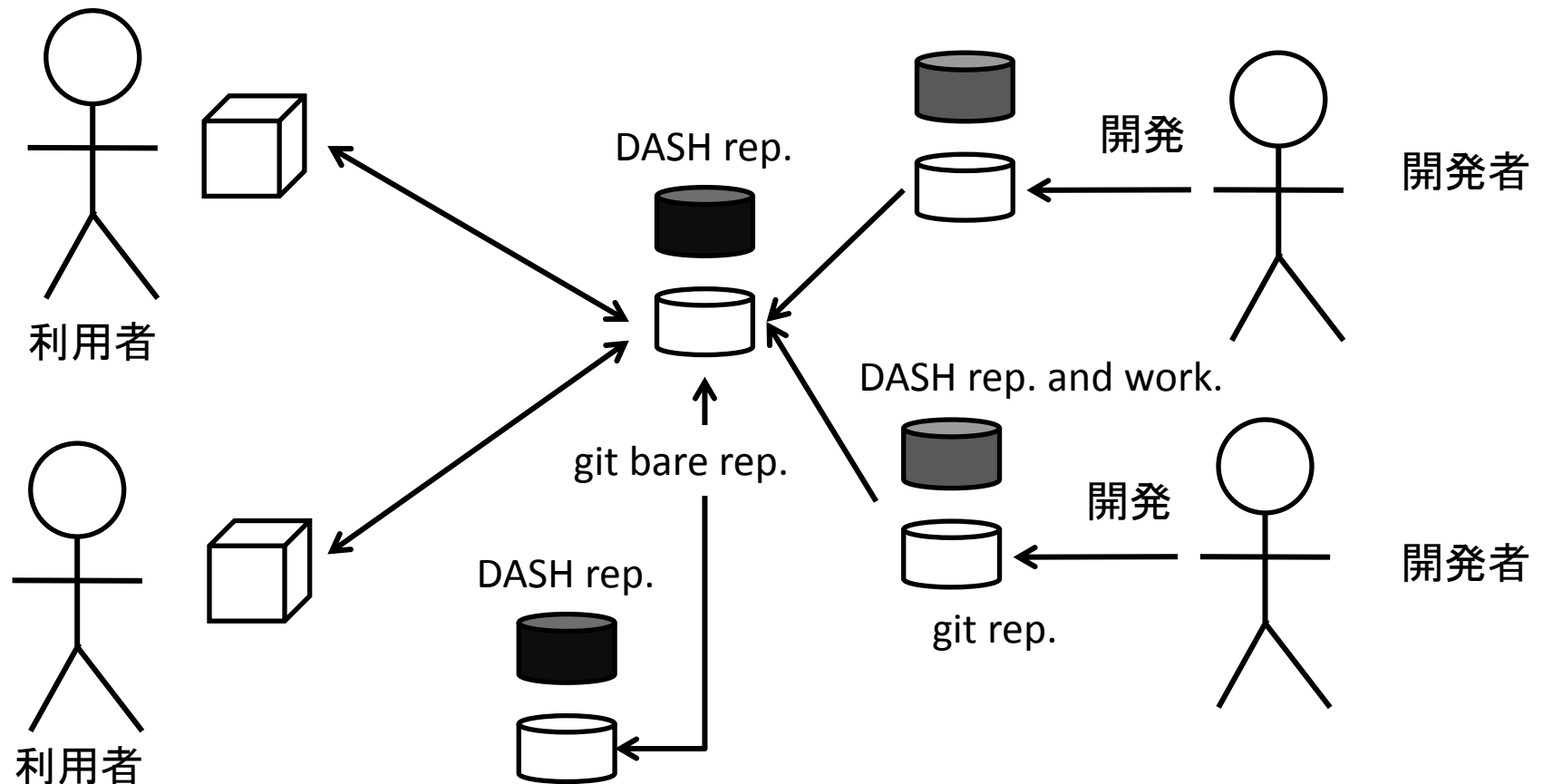
エージェント開発の版管理



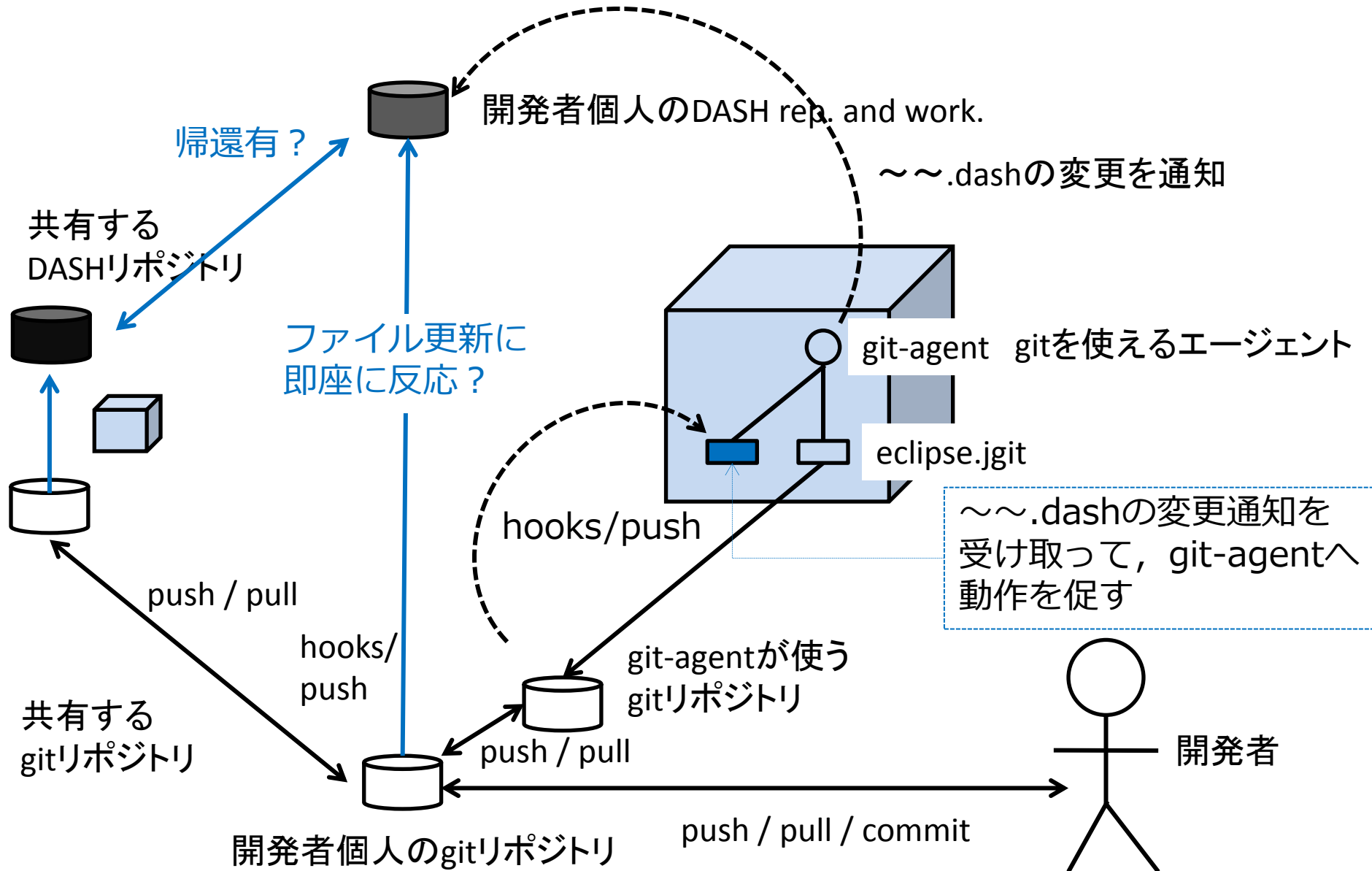
開発者がcommit/branchした時には
エージェントとして「何か」が異なる

エージェント部品の開発

更新を行う、が、古いエージェントが必要な場合も



DASH and git repositories



git-agent 目標

1. hooks/pushを受け取って, DASHリポジトリへ通知する機能
 1. DASHリポジトリが新しいファイルを読み込む
2. commitメッセージと差分からエージェントの仕様変更点を抽出
 1. 知識獲得や開発者支援へ
3. 帰還型エージェントを取り込む
 1. 実行時の知識獲得へ

コメント

- エージェント自体に， gitの差分ファイルを持たせる
- 「上手くいった時の記憶」を持っていると良い
- リポジトリが再起動無しで読込めると良いですね（やはり， DASHには無いらしい）