東京エリア Debian 勉強会
Debian パッケージング道場

第 130 回 2015 年 9 月度

岩松 信洋

2015 年 9 月 12 日

設営準備にご協力ください。

会場設営よろしくおねがいします。

Agenda

- 最近あった Debian 関連のイベント報告
  - 第 129 回 東京エリア Debian 勉強会
  - OSC 2015 Niigata
- パッケージングチュートリアルを使った解説
- パッケージング道場
- 今後のイベント
- 今日の宴会場所

イベント報告

第 129 回東京エリア Debian 勉強会

- 場所はスクウェア・エニックスさんのセミナルームをお借りしての開催でした。
- セミナは野島さんによる「APT1.1 超☆牛さんパワー炸裂!」と題した APT 1.1 の解説でした。
- 残りの時間で hack time を行い、成果発表をしました。

OSC 2015 Niigata

- 9/8 に 新潟で OSC が開催され、野島さんと吉田さんがさんかされました。
  /http://www.ospn.jp/osc2015-niigata/
- セミナと展示を行い、セミナーは野島さんによる「Debian update」と題した Debian 8.0 の紹介が行われました。

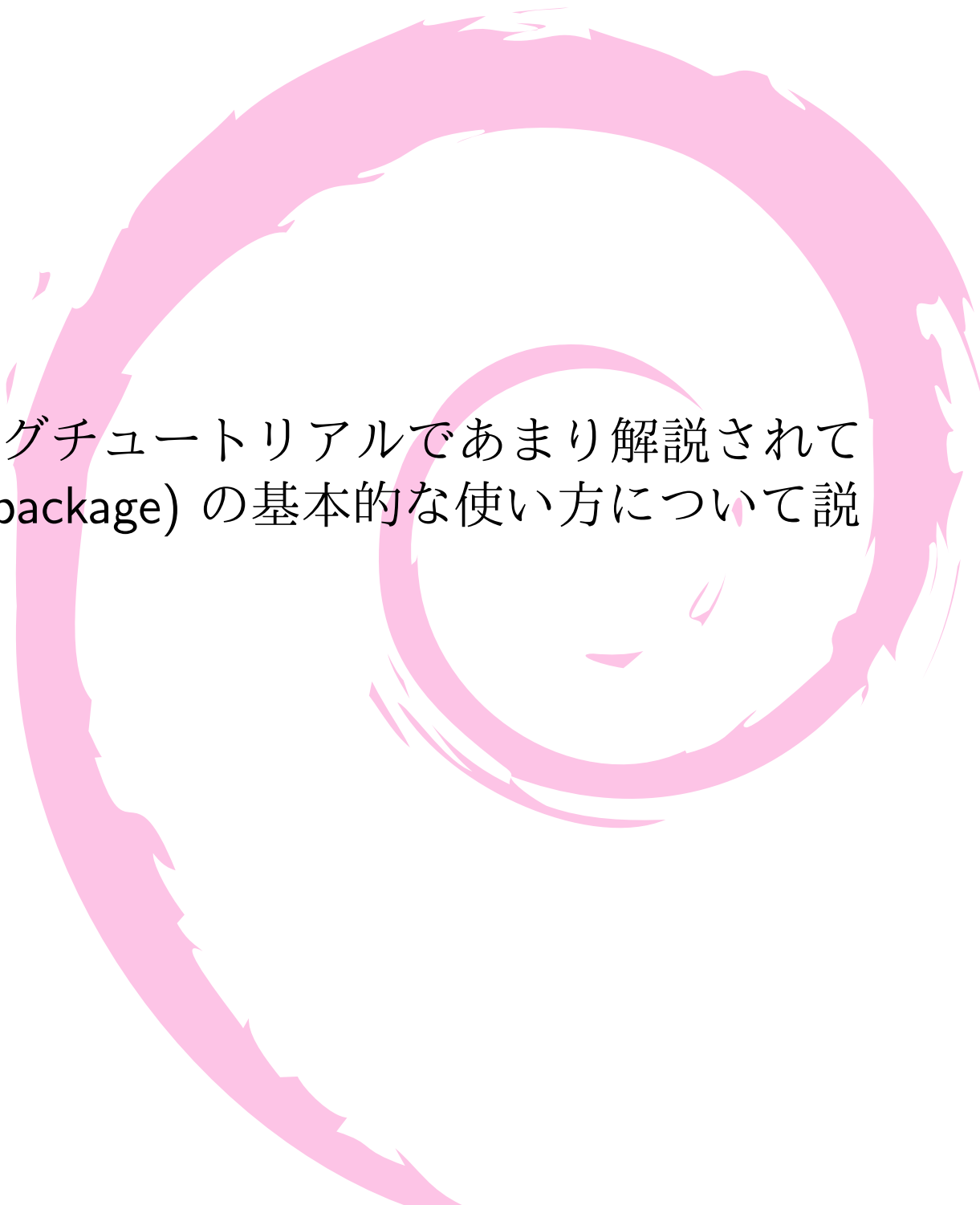
Debian パッケージング道場

本日の流れ

- 午前の部
  - パッケージングチュートリアルを使った Debian パッケージの解説
  - git buildpackage の使い方を解説
- お昼ごはん
- 午後の部 (13:00 -)
  - パッケージ作業
  - 成果報告

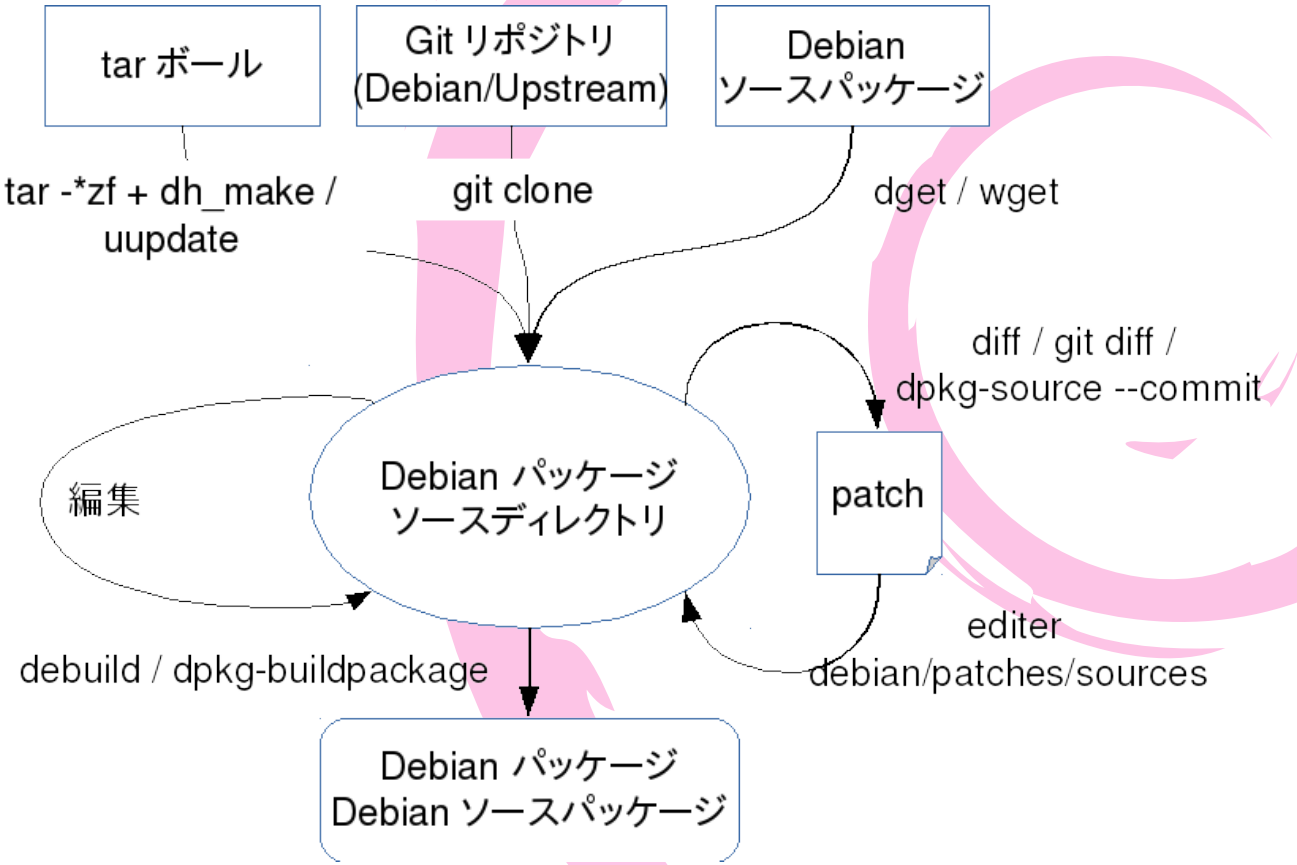
パッケージングチュートリアル

git build-package

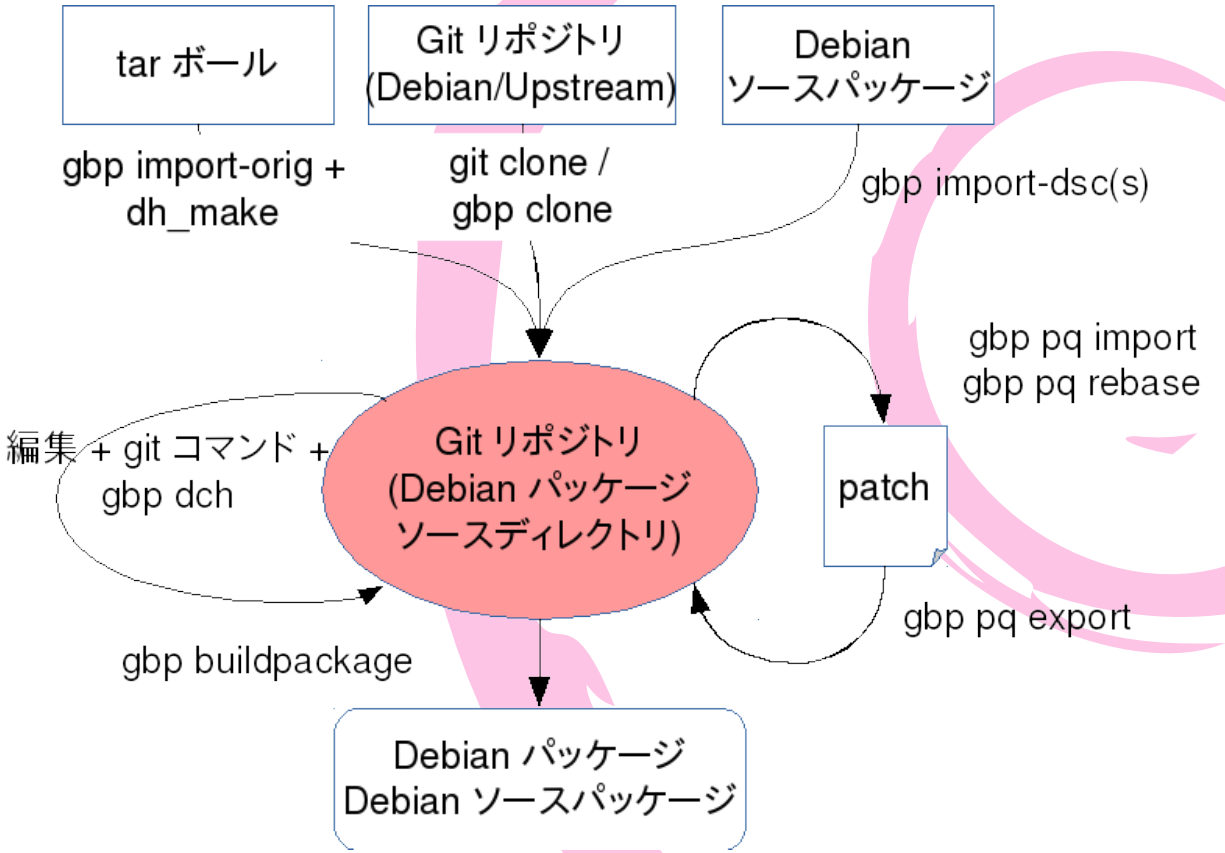


Debian パッケージングチュートリアルであまり解説されていない gbp (git buildpackage) の基本的な使い方について説明する。

VCS で管理しない場合



VCS で管理する場合


tar ボール
Git リポジトリ
(Debian/Upstream)
Debian
ソースパッケージ
gbp import-orig +
dh_make
git clone /
gbp clone
gbp import-dsc(s)
gbp pq import
gbp pq rebase
編集 + git コマンド +
gbp dch
Git リポジトリ
(Debian パッケージ
ソースディレクトリ)
patch
gbp pq export
gbp buildpackage
Debian パッケージ
Debian ソースパッケージ

Upstream から tar ボールがリリースされている場合

Upstream から tar ボールがリリースされている場合、tar ボールを Git リポジトリにコミットした後、パッケージ化を行う。

- tar ボールをダウンロードする

```
$ wget package-name-0.0.1.tar.gz
```

- Git リポジトリを作成する

```
$ git init package-name
$ cd package-name
```

必要であれば git config で Git の設定を変更する。

- tar ボールを指定して、ソースコードを Git にコミットする

```
$ gbp import-orig --pristine-tar \
        ../package-name-0.0.1.tar.gz
$ git tag
$ upstream/0.0.1
$ git branch
* master
  upstream
```

- Upstream のコードをは upstream ブランチで管理され、同時にタグが設定される。
- pristine-tar オプションはオリジナルの tar ボールの差分を保存するための仕組みを有効にする。Upstream のコードは upstream ブランチで管理され、Debian パッケージを作成するときに、そこから orig.tar.gz ファイルを作成する。この時に作成されるファイルが同じものにならない場合がある。Debian では取り込んだ tat.gz ファイルのハッシュ値と Debian ソースパッケージとしてアップロードされる tar.gz ファイルが同じである必要があるため、本オプションを用いて orig.tar.gz との差分をバイナリパッチとして保存し、orig.tar.gz を構築する度に再適用することで、ハッシュ値が同じ orig.tar.gz を再構築できるようにしている。

- dh_make で debian ディレクトリの雛形を作成する

```
$ dh_make -p package-name_0.0.1
```

- debian ディレクトリ内をいろいろ変更する

```
$ いろいろ修正
$ git add debian
$ git commit
```

- パッケージ化作業ができたら パッケージを構築する

```
$ gbp buildpackage --git-pristine-tar
```

- piuparts などでインストール、アンインストールのテスト
- 最後にクリーンな環境でビルドテスト

```
$ git-pbuilder
or
$ gbp buildpackage --git-pbuiler --git-pristine-tar
```

- タグを設定して リモートリポジトリにプッシュする

```
$ gbp buildpackage --git-tag-only
$ git push
```

upstream に更新があった場合

- upstream の tar ボールを取得する

```
$ wget package-name-0.0.2.tar.gz
```

- tar ボールを指定してリポジトリにソースコードをコミットする

```
$ gbp import-orig --pristine-tar \
                  ../package-name-0.0.2.tar.gz
```

--uscan オプションでダウンロード → コミットが一度にできる。また、ソースコードは自動的にマージされます。マージしたくない場合は--no-merge を指定して実行する。

- debian/changelog を修正

```
$ dch -i
or
$ gbp dch
```

- パッケージをビルド

```
$ gbp buildpackage --git-pristine-tar \
                   --git-pristine-tar-commit
```

Upstream が Git で管理されている場合

- Upstream では tar ボールでリリースされず、Git のタグのみでリリースされる場合もある。
- Github で開発されているプロジェクトが良い例。
- このような場合は リポジトリをクローンした後、Debian 独自のブランチルールを用いてソースパッケージの管理を行う。

- リポジトリをクローンし、作成されたディレクトリに移動する

```
$ git clone git://example.org/git/package-name.git
$ cd package-name
```

- ベースにしたいバージョンのコードをチェックアウトする

```
$ git reset --hard 0.0.1
```

- タグを設定する

```
$ git tag upstream/0.0.1
```

upstream ネームスペースは gbp デフォルト参照先。Upstream のタグを使いたい場合や独自のネームスペースを使いたい場合は gbp の upstream-tag が利用できます。

```
[git-buildpackage]
upstream-tag = v%(version)s
```

- dh_make で debian ディレクトリの雛形を作成する

```
$ dh_make -p package-name_0.0.1
```

- debian ディレクトリ内をいろいろ変更する

```
$ いろいろ修正
$ git add debian
$ git commit
```

- パッケージ化作業ができたら パッケージを構築する

```
$ gbp buildpackage --git-pristine-tar \
                   --git-pristine-tar-commit
```

tar ボールを使う場合と異なるのは --git-pristine-tar-commit オプションを指定すること。このオプションを指定することによってタグから orig.tar.gz を生成する。

- piuparts などでインストール、アンインストールのテスト
- 最後にクリーンな環境でビルドテスト

```
$ git-pbuilder
or
$ gbp buildpackage --git-pbuilder
```

- タグを設定して リモートリポジトリにプッシュする

```
$ gbp buildpackage --git-tag-only
$ git push
```

upstream に更新があった場合

- upstream のリポジトリ情報を取得する

```
$ git remote update
```

- 変更をマージ

```
$ git tag upstream/0.0.2 0.0.2
$ git merge upstream/0.0.2
```

- debian/changelog を修正

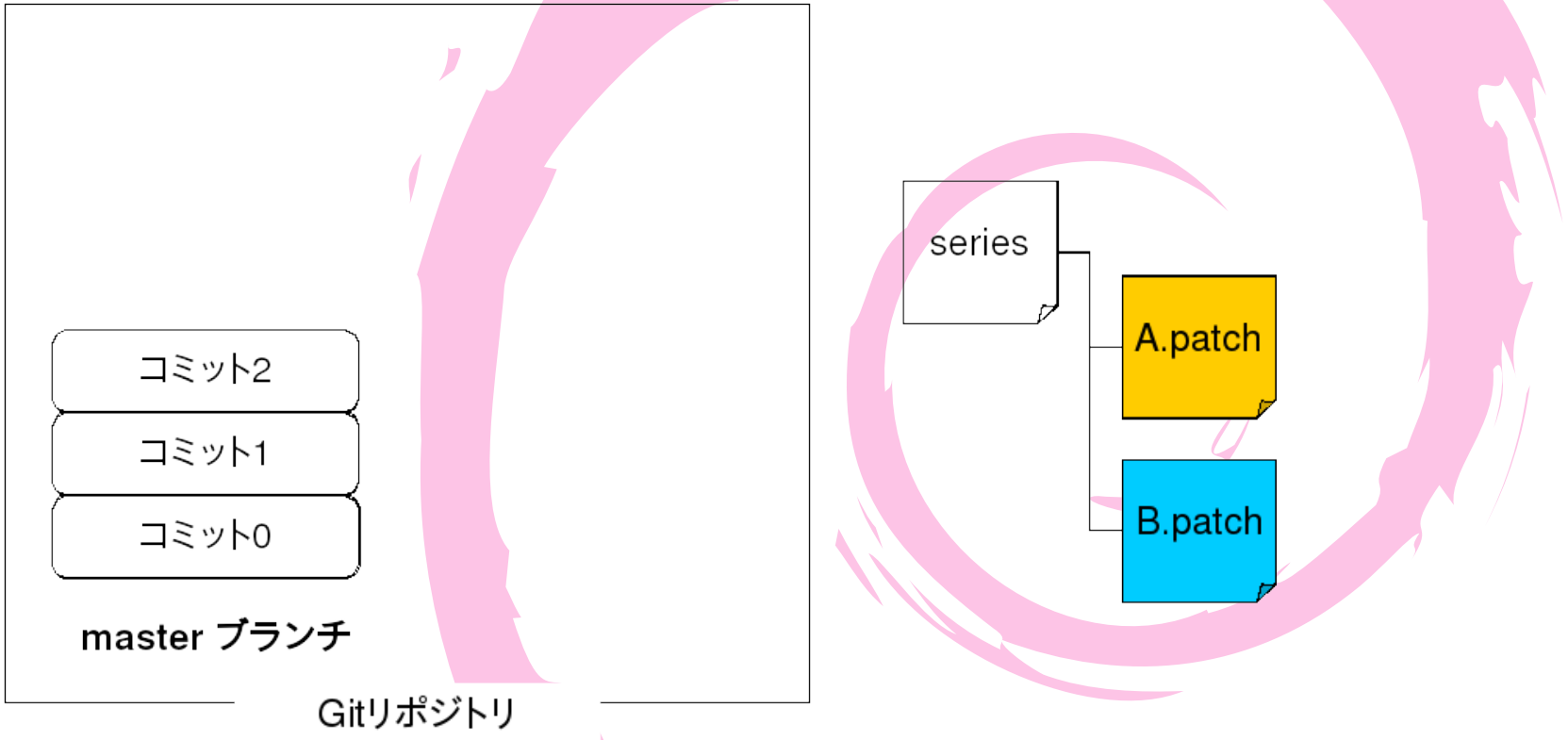
```
$ dch -i
or
$ gbp dch
```

- パッケージをビルド

```
$ gbp buildpackage --git-pristine-tar\
                   --git-pristine-tar-commit
```

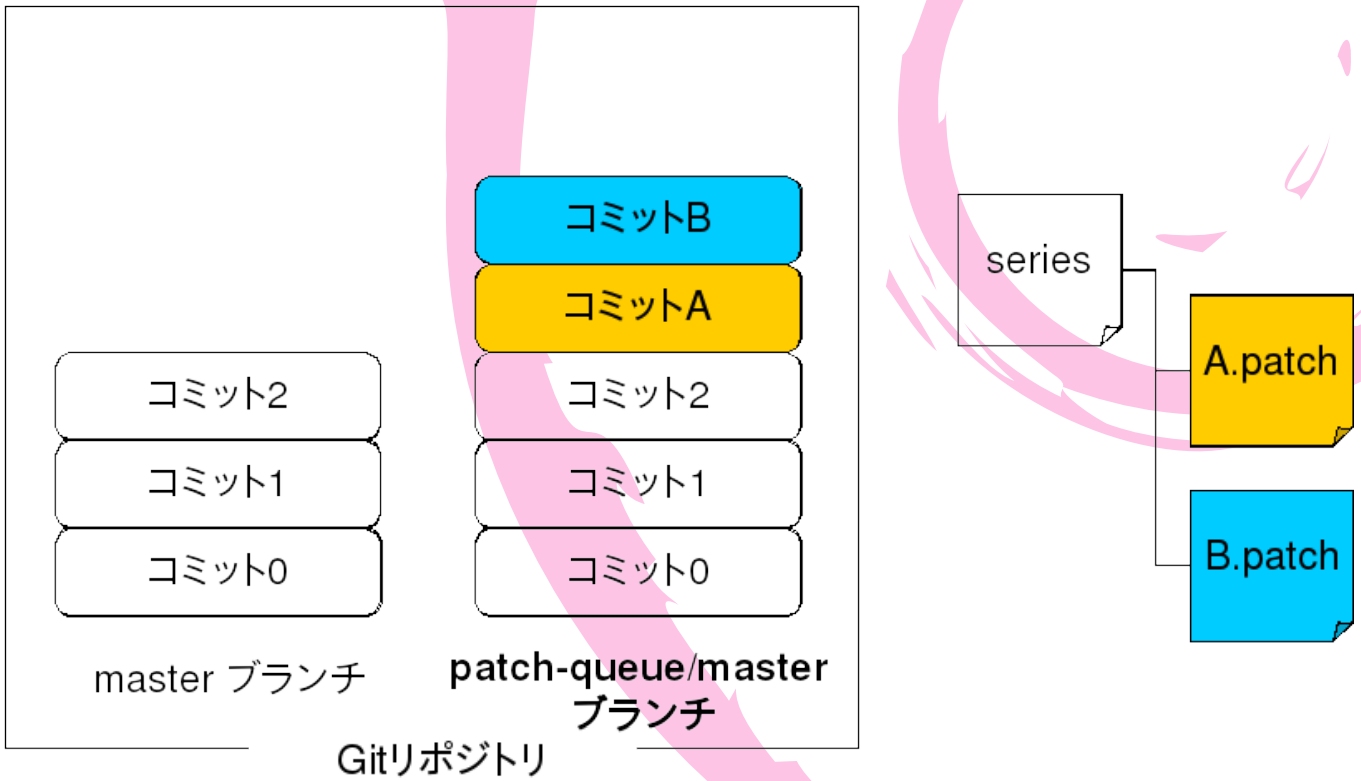
アップストリームのソースコード変更

1. gbp pq import
2. Upstream ソースコードの修正
3. git commit
4. gbp pq export
5. git commit


コミット2
コミット1
コミット0
master ブランチ
Gitリポジトリ
series
A.patch
B.patch

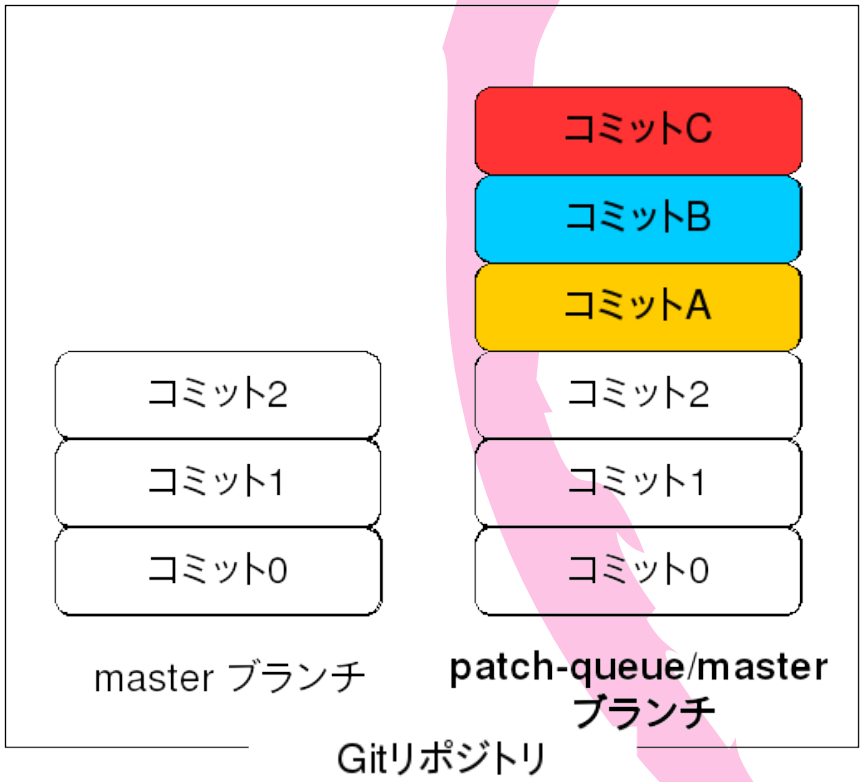
gbp pq import

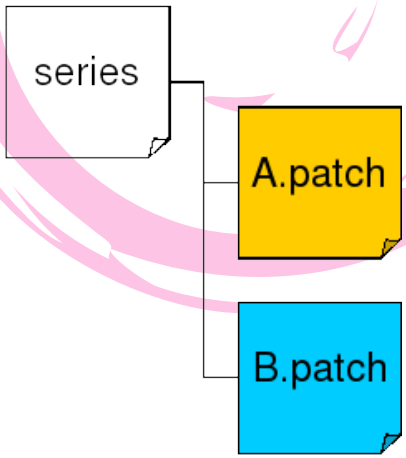
1. HEAD を patch-queue/master ブランチとしてチェックアウト
2. debian/patches/series にあるパッチをコミット



修正 & git commit

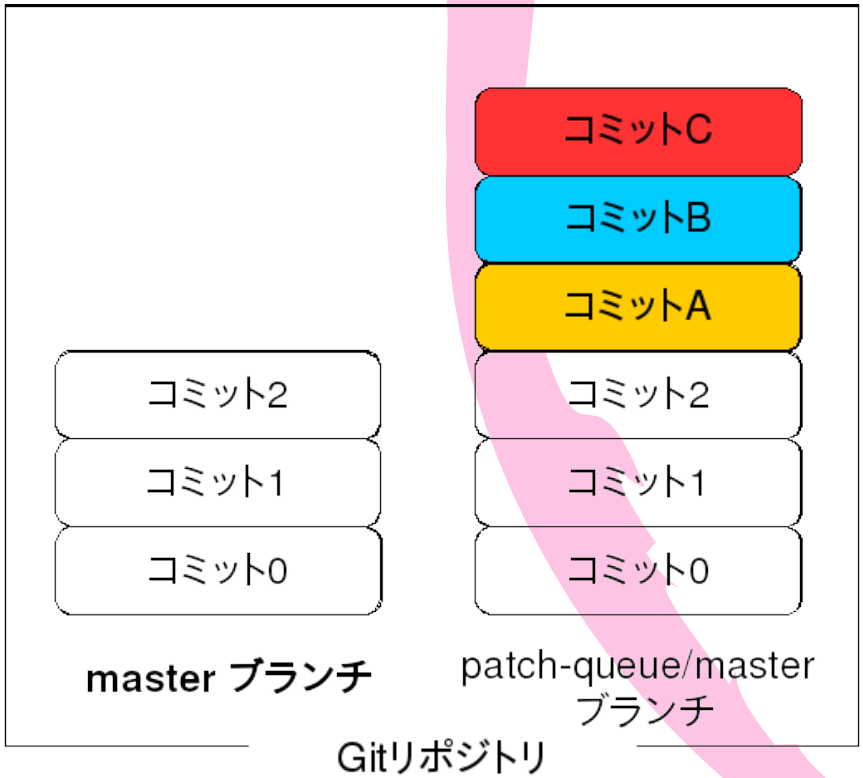
1. Upstream ソースコードの修正
2. git commit

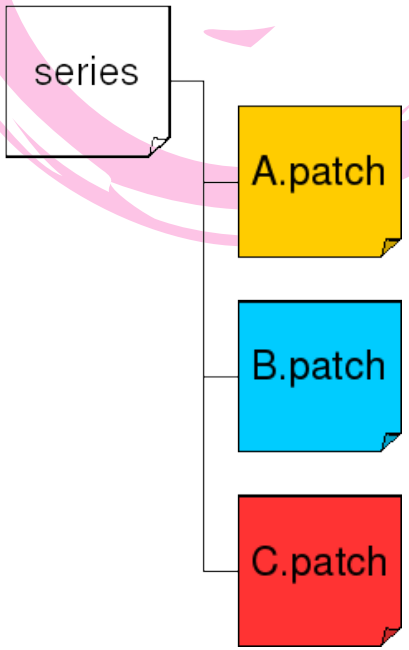




gbp pq export

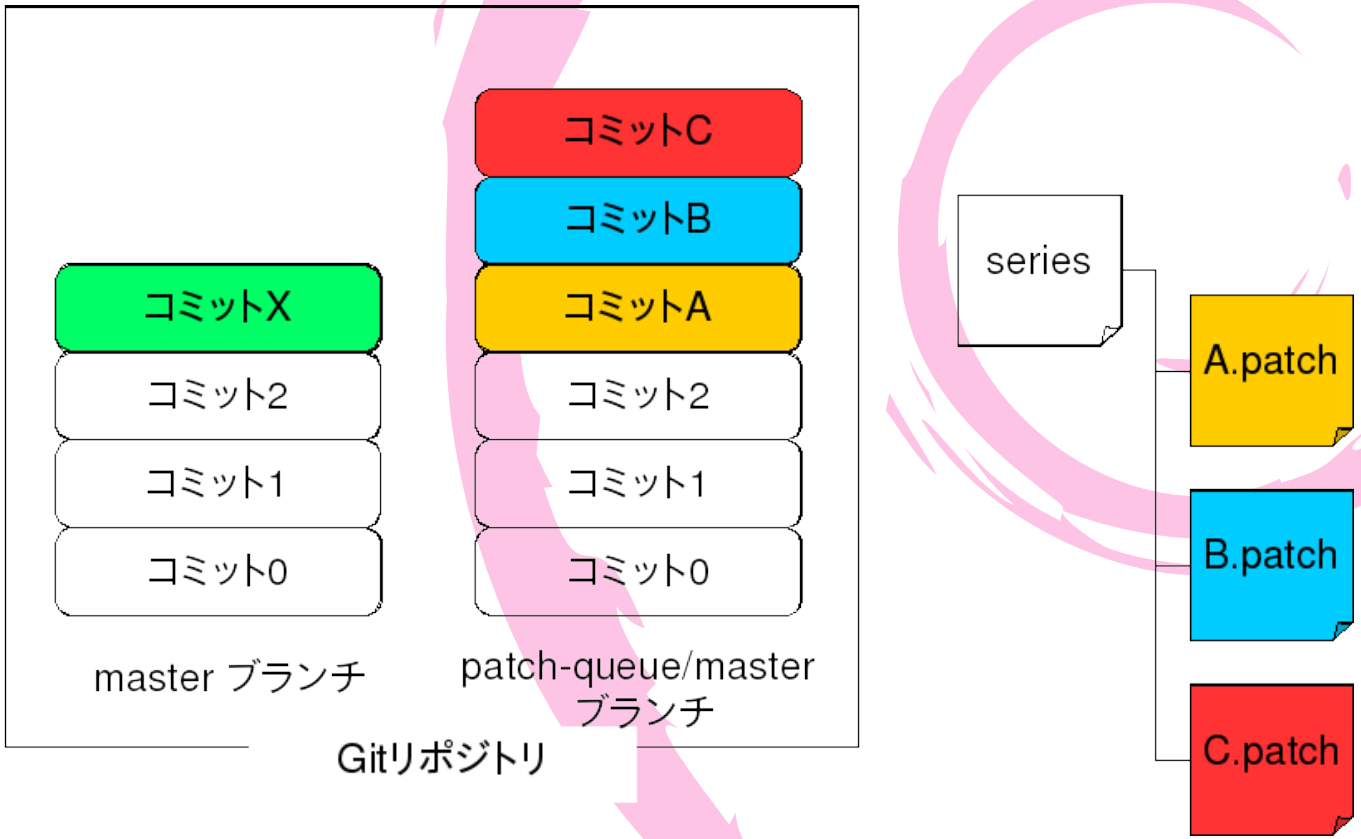
1. patch-queue/master と master ブランチの差分をパッチとして debian/patches に出力
2. debian/patches/series を更新
3. master ブランチをチェックアウト





git commit

1. パッチ更新をリポジトリにコミット



まとめ

- gbp (git buildpackage) はデファクトスタンダート
- gbp import-* でリポジトリ取り込み
  `--pristine-tar` を忘れずに
- gbp dch で debian/changelog を更新
- gbp buildpackage でパッケージビルド
- gbp pq でパッチ操作

パッケージ作業

成果報告

今後のイベント

今後のイベント

- 関西エリア Debian 勉強会
- 10/24(土)、10/25(日) OSC 2015 Tokyo Fall でセミナ＆ブース

  http://www.ospn.jp/osc2015-fall/

今日の宴会場所

今日の宴会場所

未定