BRIEF REPORT

REVISED *biogitflow*: development workflow protocols for bioinformatics pipelines with git and GitLab [version 3; peer review: 2 approved]

Choumouss Kamoun^{1-4*}, Julien Roméjon^{1-4*}, Henri de Soyres¹⁻⁴, Apolline Gallois¹⁻⁴, Elodie Girard¹⁻⁴, Philippe Hupé^{id}¹⁻⁵

¹Institut Curie, Paris, F-75005, France

²U900, Inserm, Paris, F-75005, France

³PSL Research University, Paris, France

⁴Mines Paris Tech, Fontainebleau, F-77305, France

⁵UMR144, CNRS, Paris, F-75005, France

* Equal contributors

V3 First published: 22 Jun 2020, 9:632
<https://doi.org/10.12688/f1000research.24714.1>
 Second version: 08 Dec 2020, 9:632
<https://doi.org/10.12688/f1000research.24714.2>
 Latest published: 19 Feb 2021, 9:632
<https://doi.org/10.12688/f1000research.24714.3>

Abstract

The use of a bioinformatics pipeline as a tool to support diagnostic and theranostic decisions in the healthcare process requires the definition of detailed development workflow guidelines. Therefore, we implemented protocols that describe step-by-step all the command lines and actions that the developers have to follow. Our protocols capitalized on the two powerful and widely used tools git and GitLab, and are based on gitflow, a well-established workflow in the software engineering community. They address two use cases: a *nominal* mode to develop a new feature in the bioinformatics pipeline and a *hotfix* mode to correct a bug that occurred in the production environment. The protocols are available as a comprehensive documentation at <https://biogitflow.readthedocs.io> and the main concepts, steps and principles are presented in this report.

Keywords

development workflow, bioinformatics pipeline, quality management, healthcare

This article is included in the **Bioinformatics** gateway.

Open Peer Review

Approval Status

	1	2
version 3		
(revision)		
19 Feb 2021	view	
version 2		
(revision)		
08 Dec 2020	view	view
version 1		
22 Jun 2020		
	view	

1. **Bernard Pope** , University of Melbourne, Parkville, Australia

2. **Julia Ponomarenko**, The Barcelona Institute of Science and Technology, Barcelona, Spain

Any reports and responses or comments on the article can be found at the end of the article.

Corresponding authors: Choumouss Kamoun (choumouss.kamoun@curie.fr), Philippe Hupé (philippe.hupe@curie.fr)

Author roles: **Kamoun C:** Conceptualization, Validation, Writing – Review & Editing; **Roméjon J:** Conceptualization, Validation, Writing – Review & Editing; **de Soyres H:** Conceptualization, Writing – Review & Editing; **Gallois A:** Validation, Writing – Review & Editing; **Girard E:** Validation, Writing – Review & Editing; **Hupé P:** Conceptualization, Supervision, Validation, Writing – Original Draft Preparation, Writing – Review & Editing

Competing interests: No competing interests were disclosed.

Grant information: This work was supported by the project French Bioinformatic Network for NGS Cancer Diagnosis, funded by the Institut National du Cancer (INCA).

The funders had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Copyright: © 2021 Kamoun C *et al.* This is an open access article distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

How to cite this article: Kamoun C, Roméjon J, de Soyres H *et al.* ***biogitflow: development workflow protocols for bioinformatics pipelines with git and GitLab [version 3; peer review: 2 approved]*** F1000Research 2021, 9:632 <https://doi.org/10.12688/f1000research.24714.3>

First published: 22 Jun 2020, 9:632 <https://doi.org/10.12688/f1000research.24714.1>

REVISED Amendments from Version 2

This version improves upon the main issues raised by the reviewers. The major change is the addition of an 'Exercises' section in the biogitflow documentation. The infinite lifetime of the release and hotfix branches has been motivated. We have emphasized in the conclusion that biogitflow can be straightforwardly applied on any kind of software development.

Any further responses from the reviewers can be found at the end of the article

Introduction

The importance of best practices for bioinformatics analysis and software have been highlighted by many authors who proposed very valuable guidelines for better reproducibility and traceability (Georgeson *et al.*, 2019; Hamburg & Mogyorodi, 2019; Noble, 2009; Sandve *et al.*, 2013). However, reproducing an analysis is often a challenge (Kim *et al.*, 2018). Therefore, guidelines have to be promoted by the computational labs and bioinformatics core facilities to federate the bioinformaticians across common practices for software development. This is especially essential when the bioinformatics pipeline is used in precision medicine to support the diagnostic and therapeutic activities. Indeed, many hospitals worldwide use High-Throughput Sequencing in routine clinical practice to guide the therapeutic decision. This new era of genomic medicine is even promoted in healthcare systems at a large scale within several national initiatives, such as France, USA, UK or Australia (Stark *et al.*, 2019).

This evolution has brought Bioinformatics at the forefront of the healthcare process with the bioinformatics pipeline being a fully integrated component of the clinical decision. Compliance with the healthcare laboratory accreditation standards and regulations is thus required for the development and exploitation of the bioinformatics pipeline. In this context, several authors (Hume *et al.*, 2019; INCa, 2018; Matthijs *et al.*, 2016; Roy *et al.*, 2018) recommended in their guidelines that an appropriate code repository tool should be used to enforce version control. It ensures to track the different releases of the bioinformatics pipeline, their validation and the developers involved in their implementation. This must be integrated in a quality management process with standardized protocols approved for a diagnostic use.

There is a wide ecosystem of tools (see Perez-Riverol *et al.*, 2016; Riesch *et al.*, 2020; Zolkifli *et al.*, 2018) that can support the implementation of such protocols. Among them, we capitalized on **git** for the version-control system as it became a very popular tool in the software engineering community and **GitLab** for the repository manager as it can be self-hosted. Both tools offer a large set of very powerful functionalities that can be used and combined in multiple ways thus increasing their usage complexity. It is thus mandatory to formalize through detailed guidelines how to use them on a daily basis for the development and deployment of the bioinformatics pipeline. Therefore, we implemented the set of protocols *biogitflow* (Kamoun *et al.*, 2020)

that describes step-by-step all the command lines and actions to be performed by the developers.

These protocols are mainly dedicated to bioinformaticians and software developers who are involved in the development, deployment and maintenance of bioinformatics pipelines that support routine production. For example, core facilities (such as sequencing platforms) generate, with a very high-throughput, many data that have to be processed and delivered on the fly to the end-users. Short time-to-delivery and continuity of service under any circumstances is a challenge that can be tackled by promoting these protocols to pave the way towards better industrialized processes for the software component in the context of production, in particular in healthcare.

A comprehensive documentation available at <https://biogitflow.readthedocs.io> provides all the technical details. We introduce here the main concepts, steps and principles.

Methods

Development workflow

The development workflow consists of four main steps. The first step is **software development**, which includes code writing and testing. The second step is **acceptance testing** by the end-users who validate that the expected functionalities have been correctly implemented. The third step is **check the installation process and new testing** to ensure that the bioinformatics pipeline can be installed in a similar environment than the one used in production. During this step, a new testing is performed such that bugs can be corrected before installing the bioinformatics pipeline in production. Finally, the fourth step is **production deployment**. During this last step, the new release of the bioinformatics pipeline with the new functionalities is installed in the production environment.

Multiple deployment environments

The four different steps have to be performed in separated environments in order to i) ensure that a stable version can be used in production, ii) allow the end-users to validate a new release without any impact on both the version used in production or the version under development and iii) allow the software developers to add new functionalities and modify the code without any impact on the end-users who are validating a new version and/or using the current version in production. Therefore, three deployment environments are used: a development (**dev**), a validation for pre-production (**valid**) and a production environment (**prod**). Besides these environments, each developer can deploy the bioinformatics pipeline in a **local** workspace to test the new functionalities that have been developed.

Version control and branching model

Several development workflows and branching models have been proposed including **GitHub flow**, **GitLab Flow** and **gitflow**. Our *biogitflow* protocols capitalized on the popular **gitflow** model proposed by Driessen (2010) ten years ago. The management of the different bioinformatics pipeline versions is based on four different **git** branches. Depending on the context and the step of

the development workflow the following branches on the remote repository are used:

devel contains the code of the current version under development.

release contains the code with both candidate and official releases. The release branch comes from the devel branch.

hotfix is a mirror of the release branch and is used to patch the code that is in production. If a critical bug occurs in production, this branch is used to fix the issue.

master is only used to archive the code from the release and hotfix branches. This branch is not used for development.

Among these four branches, the release, hotfix and master are protected branches such that only the developers with the Maintainer role in the GitLab repository can directly push code on these branches (the other users have to use the GitLab merge request functionality to push on protected branches).

In addition to these four branches, the developer can create local branches to i) implement a new feature (the branch is named with a prefix *feature* plus any meaningful suffix) and ii) fix a bug in production or resolve a problem during the third step (these branches are either named with the prefix *release* or *hotfix* depending on the use case plus some relevant contextual information).

User roles and permissions

Two levels of roles and permissions are considered. Firstly, in the GitLab remote repository a developer is either assigned to the role Developer (**D**) to push the developments only on the non-protected branches or Maintainer (**M**) to push the developments on any branches. Secondly, in the deployment environments a user is either granted with the permission **UD** to deploy in the dev environment only or **UVP** to deploy in the valid and prod environments (any user with the **UVP** permissions is also granted with the **UD** permissions).

Bioinformatics pipeline testing

Testing the bioinformatics pipeline occurs during all the steps of the development workflow. It involves all stakeholders including the developers, the users in charge of the deployments and the end-users. This should be done as often as possible to identify and resolve any unexpected behavior early in the development process. The International Software Testing Qualifications Board provides comprehensive guidelines for software testing (Hamburg & Mogyorodi, 2019). Among them, we strongly recommend to include the following testing. The *unit testing* confirms that a piece of code provides the expected output according to the input parameters. The *integration testing* checks that the interfaces of the different bioinformatics pipeline components are consistent with each other and that the result of their integration allows the expected functionalities

to be performed. The *system (or functional) testing* validates that the full bioinformatics pipeline works and fits well the end-user's needs. The *regression testing* checks that the correction of bugs or the development of new functionalities did not introduce defects in unchanged areas of the bioinformatics pipeline. In addition, we highlight the importance of the *operational testing* to check that the bioinformatics pipeline provides the expected results on a reference dataset (golden dataset) in the production environment. This testing is systematically launched prior to any new analysis by the bioinformatics pipeline in the production environment. This ensures that the results are reproducible as long as the exact same version of the bioinformatics pipeline is used.

Results

Two use cases have been addressed with dedicated protocols that detail the different actions step-by-step. The first one is the **nominal mode** (Figure 1) in which a new feature is implemented to improve the bioinformatics pipeline based on requirements and expectations from the end-users who operate it for their daily clinical practice. The second one is the **hotfix mode** (Figure 2) in which there is a bug in the bioinformatics pipeline in the production environment that hampers the delivery of the results for the patients. The *biogitflow* documentation provides all the technical details, on how to configure the remote repository in GitLab to develop a new bioinformatics pipeline, how to use *git* and *GitLab* depending on the roles and permissions during the time frame of the development workflow for both use cases.

Whatever the use case, the bioinformatics pipeline is deployed in production and operated by the end-users once it has successfully passed all the testing. Whenever new patient data has to be analyzed to deliver results for diagnostic or theranostic purposes, the bioinformatics pipeline can be launched by the end-users only if the *operational testing* reproduces the results from the golden dataset, otherwise it is blocked by internal control mechanism. In this case, the developers investigate the reason of the failure and have to fix it using the **hotfix mode**.

As a real example, we provided a repository that is available at <https://gitlab.com/biogitflow/biogitflow-template> (it is recommended that the user registers on GitLab.com to benefit from all its functionalities). This repository has been created according to the guidelines from the section *Create a new project in Gitlab* from the *biogitflow* documentation. It contains the different protected and non-protected branches according to the branching model, protected tags, templates for the issues and merge requests, and a set of labels for the issues. The user can fork the repository in a personal workspace and apply the **nominal mode** (Figure 1) and **hotfix mode** (Figure 2) following the proposed protocols.

Discussion

While *gitflow* was a main source of inspiration, *biogitflow* differs in several ways as:

- it uses a similar branching model with some adaptations:
 - *gitflow* relies on two branches with infinite lifetime,

1.1 **X** ☐ ☐ ☐ ☒ ☐ ☐ ☐ ☐

1.2 **X** ☐ ☐ ☒ ☒ ☐ ☐ ☐

1.3 **X** ☐ ☐ ☐ ☒ ☐ ☐ ☐

1.4 **X** ☐ **X** ☐ ☒ ☐ ☐ ☐

```

graph TD
    2.1[2.1] -- "If end-users validate and testing successful, go to Step 3.1, if not go to Step 1.1" --> 2.2[2.2]
    2.2 -- "If end-users validate and testing successful, go to Step 3.1, if not go to Step 1.1" --> 3.1[3.1]
    2.2 -- "If end-users validate and testing successful, go to Step 3.1, if not go to Step 1.1" --> 1.1[1.1]
  
```

3.1

1	2	3	4				
---	---	---	---	--	--	--	--

3.2

1	2	3	4	5	6		
---	---	---	---	---	---	--	--

3.3

1	2	3	4			7	8
---	---	---	---	--	--	---	---

If testing successful, go to Step 3.8, if not go to Step 3.4

3.4

1	2	3	4	5	6		
---	---	---	---	---	---	--	--

3.5

1	2	3	4	5	6		
---	---	---	---	---	---	--	--

3.6

1	2	3	4	5	6		
---	---	---	---	---	---	--	--

3.7

1	2	3	4	5	6		
---	---	---	---	---	---	--	--

go to Step 3.2

Diagram illustrating the steps of a two-phase commit protocol:

- 4.1: Coordinator sends 'X' to all nodes.
- 4.2: Coordinator sends 'X' to all nodes.
- 4.3: Coordinator sends 'M' to all nodes.
- 4.4: Coordinator sends 'P' to all nodes.
- 4.5: Coordinator sends 'M' to all nodes.

** only CHANGELOG is modified

Page 5 of 20

the master and devel(op) branches, and three branches with limited lifetime, the feature (a local branch on the developer's side), release and hotfix branches that could be eventually removed from the remote repository. In contrast, the release and hotfix branches have infinite lifetime in *biogitflow*. This way, the release branch is always ready to welcome the future version. The infinite lifetime of the hotfix branch is motivated to address the following situation: the release branch has been modified since a new version is under preparation but a bug occurs in the production environment. Therefore, the hotfix branch is the only branch that is similar to what has been deployed in production and it can be used straightforwardly to fix the bug.

- the release branch in *biogitflow* is used in a similar manner as the master branch in *gitflow*, in particular, the tag for a new version is added on the release branch (or hotfix branch in the hotfix mode).
- the master branch in *biogitflow* is used as an archive of all the developments.
- it provides more technical details about the git commands including the use of commits (with some naming conventions for the commit messages, the tagging).
- it describes the usage of git with GitLab and therefore describes the use of issues, labels and merge requests.
- it is more comprehensive as it actually goes beyond the usage of git and GitLab since it includes guidelines for deployment and testing.
- it defines roles and permissions for the branches and deployment environment.

Writing these protocols required some feedback from the developers in order to decide what was the best way to capitalize on git and GitLab taking into account our internal constraints and organization. The need of formalization required by the laboratory accreditation agency such that a bioinformatics pipeline can be used in healthcare was a great catalyst to implement these protocols. It was a major step to improve our daily practice of software engineering towards better quality management and was a source of motivation to change our work habits. More precisely, the use of the *biogitflow* protocols contributed to the harmonization of the development practices across developers. Every git pipeline repository is now structured the same way that simplifies the skill transfer from one person to another. This is very convenient, especially when you have to maintain different pipeline repositories with different developers involved. Moreover, due to multiple deployment environments along with the branching model, the developer can really perceive a constructive and positive pressure as long as the deployment in the production (the danger zone) is getting close: this is a source of stimulation for better quality.

In order to promote these protocols, it was necessary to train the users involved in the development workflow to demonstrate to the laboratory accreditation agency that the technical protocols are known, understood and mastered by the developers. Therefore, an internal accreditation process of our developers was implemented. The training consists of a series of exercises that cover all the different use cases, roles and permissions of the protocols. The exercises are first realized in pair with the tutor and the trainee, then the trainee performs the exercises alone twice, and finally the trainee performs the exercises alone but in the presence of the tutor who ask additional questions to ensure that the tricky parts of the protocols are understood. To be endorsed, the trainee has to perform the exercises fluently in full autonomy. All the actions of the training process are tracked in a dedicated GitLab remote repository. Endorsement is valid for one year that can be extended if the developer still masters the protocols. This is assessed during a dedicated yearly interview with the tutor. The exercises are available on <https://biogitflow.readthedocs.io>.

As depicted in Figure 1 and Figure 2, the development workflow protocols contains many steps that are complex. This complexity arises from the reality of the numerous tasks to be performed to ensure the development, deployment and maintenance of bioinformatics pipelines in the context of production as mentioned in the introduction. Writing the documentation was clearly mandatory in order to provide a formal description of our work methods and promote them internally. Obviously, such protocols contains many manual operations that are prone to error and misapplication. This is the reason why, not only training is important, but also practicing on a weekly (or at least on a monthly) basis is crucial to remain fluent. To circumvent these pitfalls, these protocols would benefit from improved automation to reduce the number of manual steps and avoid possible mistakes. This will be the scope of further improvements that could capitalize of additional valuable functionalities such as git hooks, GitLab CI/CD (continuous integration / continuous deployment) and Operations features.

Conclusions

We described two protocols that are used on a daily practice to develop bioinformatics pipelines compliant with the accreditation standards for healthcare. Bioinformaticians and software developers involved in the development, deployment and maintenance of bioinformatics pipelines in the context of production will benefit from these protocols. While some choices were made to match our internal constraints and organization, the protocols can be easily transposed in other institutes as the main concepts, steps and principles hold in most of the contexts. Indeed, the protocols were internally motivated by the development of bioinformatics pipelines to support the use of genomics sequencing in clinical diagnostic and can be straightforwardly applied not only to genomics but also to any kind of omics approaches (proteomics, metabolomics, etc.), medical informatics or more generally any software development whether it is bioinformatics or not. According to the principle of the Deming wheel of continuous improvement in quality management (Swamidass, 2000), the protocols we

described are intended to evolve in order to address future requirements, handle new risk management, integrate new tools or technical frameworks offering better efficiency of the overall process. We strongly encourage the promotion of such protocols in the context of production not only for the healthcare activities but also for research activities.

Data availability

Underlying data

All data underlying the results are available as part of the article and no additional source data are required.

Extended data

biogitflow documentation is available at: <https://biogitflow.readthedocs.io/>.

Archived source of the **biogitflow** documentation at time of publication: <https://doi.org/10.5281/zenodo.4537153>

License: **CeCILL Version 2.1**.

Author contributions

CK, HDS, JR and PH conceived the protocols. PH wrote the protocols and the manuscript that were reviewed by AG, CK, EG, HDS and JR. PH supervised the study.

Acknowledgments

We thank the **French Bioinformatic Network for NGS Cancer Diagnosis** and the Institut National du Cancer (INCA) for the fruitful discussions.

References

- Driessen V: **A successful git branching model**. 2010.
Reference Source
- Georgeson P, Syme A, Sloggett C, *et al.*: **Bionitio: demonstrating and facilitating best practices for bioinformatics command-line software**. *GigaScience*. 2019; **8**(9): giz109.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Hamburg M, Mogyorodi G: **Standard glossary of terms used in software testing**. Technical report, International Software Testing Qualifications Board. 2019.
Reference Source
- Hume S, Nelson TN, Speevak M, *et al.*: **CCMG practice guideline: laboratory guidelines for next-generation sequencing**. *J Med Genet*. 2019; **56**(12): 792–800.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- INCa: **Conception de logiciels pour le diagnostic clinique par séquençage haut-débit**. Technical report, collection Outils pour la pratique. 2018.
Reference Source
- Kamoun C, Roméjon J, De Soyres H, *et al.*: **bioinfo-pf-curie/biogitflow: version-1.1.0**. Zenodo. 2020.
Publisher Full Text
- Kim YM, Poline JB, Dumas G: **Experimenting with reproducibility: a case study of robustness in bioinformatics**. *GigaScience*. 2018; **7**(7): giy077.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Matthijs G, Souche E, Alders M, *et al.*: **Guidelines for diagnostic next-generation sequencing**. *Eur J Hum Genet*. 2016; **24**(10): 1515.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Noble WS: **A quick guide to organizing computational biology projects**. *PLoS Comput Biol*. 2009; **5**(7): e1000424.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Perez-Riverol Y, Gatto L, Wang R, *et al.*: **Ten simple rules for taking advantage of git and github**. *PLoS Comput Biol*. 2016; **12**(7): e1004947.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Riesch M, Nguyen TD, Jirauschek C: **bertha: Project skeleton for scientific software**. *PLoS One*. 2020; **15**(3): e0230557.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Roy S, Coldren C, Karunamurthy A, *et al.*: **Standards and guidelines for validating next-generation sequencing bioinformatics pipelines: A joint recommendation of the association for molecular pathology and the college of american pathologists**. *J Mol Diagn*. 2018; **20**(1): 4–27.
PubMed Abstract | **Publisher Full Text**
- Sandve GK, Nekrutenko A, Taylor J, *et al.*: **Ten simple rules for reproducible computational research**. *PLoS Comput Biol*. 2013; **9**(10): e1003285.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Stark Z, Dolman L, Manolio TA, *et al.*: **Integrating genomics into healthcare: A global responsibility**. *Am J Hum Genet*. 2019; **104**(1): 13–20.
PubMed Abstract | **Publisher Full Text** | **Free Full Text**
- Swamidass PM: **Deming cycle (PDCA)**. In the *Encyclopedia of Production and Manufacturing Management*. Springer. 2000; 155.
Publisher Full Text
- Zolkifli NN, Ngah A, Deraman A: **Version control system: A review**. *Procedia Comput Sci*. 2018; **135**: 408–415.
Publisher Full Text

Open Peer Review

Current Peer Review Status:

Version 3

Reviewer Report 01 March 2021

<https://doi.org/10.5256/f1000research.54697.r79930>

© 2021 Pope B. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Bernard Pope

Melbourne Bioinformatics, University of Melbourne, Parkville, Vic, Australia

I am happy that the authors have addressed all of my comments and suggestions from the previous version.

A few small typos remain and the article would benefit from a thorough proof read, but they are only minor issues.

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Bioinformatics. Genomics. Cancer. Software Development. Pipelines.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Version 2

Reviewer Report 21 January 2021

<https://doi.org/10.5256/f1000research.31091.r76180>

© 2021 Ponomarenko J. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Julia Ponomarenko

¹ Center for Genomic Regulation (CRG), The Barcelona Institute of Science and Technology,

Barcelona, Spain

² Center for Genomic Regulation (CRG), The Barcelona Institute of Science and Technology, Barcelona, Spain

This paper presents the protocol for the development and support of bioinformatics pipelines using git and GitLab. The intended users are bioinformaticians working in a clinical setting, while bioinformaticians working in research/academia can also benefit.

This reviewer questions if the scope of the intended audience can be expanded. For example, in a clinical setting, software developed in the field of medical informatics is often used along with the software developed in biology. Would the medical informatics benefit from the proposed approach as well? Also, the first paragraph mentions only genomics sequencing as a field of application. But could the same approach at the workflow development be applied to proteomics, metabolomics and other relevant for personalized medicine and diagnostics fields?

Minor comment:

- It shouldn't belie the value of the presented protocol, but the reader and potential user might benefit and become more enthusiastic about using the *biogitflow* if the presented resource provided and described an example of a specific pipeline with a small test dataset.
- The paper requires proofreading for typos; e.g., "As a real example, we provided a of the repository that is available ..."

Is the work clearly and accurately presented and does it cite the current literature?

Yes

Is the study design appropriate and is the work technically sound?

Yes

Are sufficient details of methods and analysis provided to allow replication by others?

Partly

If applicable, is the statistical analysis and its interpretation appropriate?

Not applicable

Are all the source data underlying the results available to ensure full reproducibility?

Partly

Are the conclusions drawn adequately supported by the results?

Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: bioinformatics

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard.

Author Response 12 Feb 2021

Philippe Hupé

First, we would like to thank the reviewer. We are very grateful for her time, contribution and very valuable comments that significantly helped to improve the article.

You will find below a detailed answer to the different issues that we have addressed in the revised manuscript.

Best regards,

Choumouss Kamoun, Julien Roméjon and Philippe Hupé

=== Detailed response to the reviewer2's comments ===

> This reviewer questions if the scope of the intended audience can be expanded. For example, in a clinical setting, software developed in the field of medical informatics is often used along with the software developed in biology. Would the medical informatics benefit from the proposed approach as well? Also, the first paragraph mentions only genomics sequencing as a field of application. But could the same approach at the workflow development be applied to proteomics, metabolomics and other relevant for personalized medicine and diagnostics fields?

This is really true that the proposed protocols are not restricted to genomics. We have added in the conclusion that they can be applied straightforwardly to any omics bioinformatics pipelines and even any software development.

> It shouldn't belie the value of the presented protocol, but the reader and potential user might benefit and become more enthusiastic about using the biogitflow if the presented resource provided and described an example of a specific pipeline with a small test dataset.

The biogitflow protocols come with a biogitflow-template that is publicly available on GitLab.com. In this new version, we have added in the biogitflow documentation an 'Exercises' section that first explains how the user can fork the GitLab repository. Then, the user can practice the different exercises. As biogitflow is mainly dedicated to the use git and GitLab and not the coding standard or the code itself, it is difficult to provide a specific pipeline with a test dataset. Therefore, we strongly encourage the user to start from the template and go through the different exercises just following the technical protocols available on the documentation.

> The paper requires proofreading for typos; e.g., "As a real example, we provided a of the repository that is available ..."

Typos have been fixed.

Competing Interests: No competing interests were disclosed.

Reviewer Report 19 January 2021

<https://doi.org/10.5256/f1000research.31091.r75901>

© 2021 Pope B. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Bernard Pope

¹ Melbourne Bioinformatics, University of Melbourne, Parkville, Vic, Australia

² Melbourne Bioinformatics, University of Melbourne, Parkville, Vic, Australia

I thank the authors for their careful consideration of the issues raised in the previous review. I believe the changes made to the paper in response to those issues have improved the manuscript. I believe the following, relatively minor, issues should be considered in the next revision of the paper:

- The abstract should mention that biogitflow is based on gitflow, a well-established workflow in the software engineering community.
- It is not clear what "for any registered user" means, please clarify.
- Typo: "remoterepository", should be "remote repository".
- Some of the key differences between gitflow and biogitflow are enumerated, but additional explanation as to why they are different would be useful. For example, "In contrast, the release and hotfix branches have infinite lifetime in biogitflow," would benefit from a short explanation as to why this is the case.
- "that makes easy" -> "that simplifies"?
- "make the developer feels" (rephrase).
- "As a corollary of these protocols was the necessity" (rephrase).
- Are the training materials used in the accreditation process publicly available? It was not clear if they are included in the biogitflow documentation. If they are available then perhaps a link could be provided. If they are not available then perhaps the authors could consider including them in the documentation.
- If possible, it would be useful to provide a citation for the "Deming wheel of continuous improvement in quality management".

Is the work clearly and accurately presented and does it cite the current literature?

Partly

Is the study design appropriate and is the work technically sound?

Partly

Are sufficient details of methods and analysis provided to allow replication by others?

Partly

If applicable, is the statistical analysis and its interpretation appropriate?

Partly

Are all the source data underlying the results available to ensure full reproducibility?

Partly

Are the conclusions drawn adequately supported by the results?

Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Bioinformatics. Genomics. Cancer. Software Development. Pipelines.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 12 Feb 2021

Philippe Hupé

First, we would like to thank the reviewer. We are very grateful for his time, contribution and very valuable comments that significantly helped to improve the article.

You will find below a detailed answer to the different issues that we have addressed in the revised manuscript.

Best regards,

Choumouss Kamoun, Julien Roméjon and Philippe Hupé

=== Detailed response to the reviewer1's comments ===

> The abstract should mention that biogitflow is based on gitflow, a well-established workflow in the software engineering community.

We have modified the abstract accordingly

> It is not clear what "for any registered user" means, please clarify.

We have rephrased and explained that it is recommended that the user registers on GitLab.com

> Typo: "remoterepository", should be "remote repository".

The typo was indeed present in the online pdf but not in the word file we used to submit the revised version. I guess the editor can fix the typo.

> Some of the key differences between gitflow and biogitflow are enumerated, but additional explanation as to why they are different would be useful. For example, "In contrast, the release and hotfix branches have infinite lifetime in biogitflow," would benefit from a short explanation as to why this is the case.

Indeed, more explanation was required. We have added few lines in the corresponding paragraph to motivate the infinite lifetime of the release and hotfix branch.

> "that makes easy" -> "that simplifies"?

We have modified the text.

> "make the developer feels" (rephrase).

We have rephrased the text.

> "As a corollary of these protocols was the necessity" (rephrase).

>

> Are the training materials used in the accreditation process publicly available? It was not clear if they are included in the biogitflow documentation. If they are available then perhaps a link could be provided. If they are not available then perhaps the authors could consider including them in the documentation.

The exercises were not initially included. We have mentioned in the article that they are now available in the version-1.1.0 of the biogitflow documentation.

> If possible, it would be useful to provide a citation for the "Deming wheel of continuous improvement in quality management".

The reference to Swamidass (2000) has been added.

Competing Interests: No competing interests were disclosed.

Version 1

Reviewer Report 22 October 2020

<https://doi.org/10.5256/f1000research.27261.r72478>

© 2020 Pope B. This is an open access peer review report distributed under the terms of the [Creative Commons Attribution License](#), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Bernard Pope ¹ Melbourne Bioinformatics, University of Melbourne, Parkville, Vic, Australia² Melbourne Bioinformatics, University of Melbourne, Parkville, Vic, Australia³ Melbourne Bioinformatics, University of Melbourne, Parkville, Vic, Australia

Motivated by laboratory accreditation requirements for the development of bioinformatics workflows related to high throughput sequencing in a clinical setting, the authors have developed a software development process formalisation based on gitflow, a popular model in mainstream software engineering practice.

The formalisation prescribes a branching, testing, merging and release process, with set roles and permissions for developers, and methods for managing the development of new features (nominal mode) and bug fixing (hotfix mode).

In addition to describing the biogitflow workflow, the paper also contains a short section on testing bioinformatics software pipelines, however, this is largely a recapitulation of standard testing terminology, and does not appear to be particularly bioinformatics-specific.

The paper has a brief discussion about the motivation for adopting the workflow and some unsubstantiated claims about improving their software engineering practices. It also has a brief discussion about the need to train developers in the workflow and formal accreditation.

Two large figures are included in the paper that attempt to illustrate the proposed workflow in nominal and hotfix modes.

While the topic of this paper is interesting and highly relevant to bioinformatics software developers working in areas of clinical application, where process accreditation is a key concern, there are a number of areas where I think the paper could be improved:

1. The current version of the paper is lacking in critical analysis of the proposed workflow. One does not have to look far to find criticism (positive and negative) of gitflow. However, this paper does not provide any significant discussion of the pros and cons of the approach. One of the main criticisms of gitflow is that it is overly complex (for some software development projects) and thus prone to error and misapplication. This complexity is illustrated in Figures 1 and 2 and the discussion in the paper about the need to train developers in its use. Given the apparent regimented nature of the process, it seems plausible that some of the complexity could be avoided by automation. Are there any useful tools that can help developers apply the model successfully and avoid the common criticisms of gitflow? If

developers are finding it difficult to follow, especially the "tricky" parts then that may be a sign of limitations in the process or limitations in the tools supporting its application.

2. It is not clear how significantly biogitflow differs from gitflow. In what sense does the proposed workflow differ from established software engineering practices?
3. What alternative competing development models exist?
4. How can the authors be sure that the workflow is in fact making a positive difference to their software development?
5. It would be useful to clarify the intended audience of the paper. How can they apply and benefit from this work?
6. The discussion of the provided biogitflow templates sounds useful, but the discussion in the paper is unclear. Perhaps this could be expanded upon?
7. The paper ends with "We strongly encourage the promotion of such protocols not only for the healthcare activities but also for research activities." However, due to the aforementioned lack of critical analysis, it is not clear from the paper why this is true.

Is the work clearly and accurately presented and does it cite the current literature?

Yes

Is the study design appropriate and is the work technically sound?

Yes

Are sufficient details of methods and analysis provided to allow replication by others?

Yes

If applicable, is the statistical analysis and its interpretation appropriate?

Not applicable

Are all the source data underlying the results available to ensure full reproducibility?

Yes

Are the conclusions drawn adequately supported by the results?

Partly

Competing Interests: No competing interests were disclosed.

Reviewer Expertise: Bioinformatics. Genomics. Cancer. Software Development. Pipelines.

I confirm that I have read this submission and believe that I have an appropriate level of expertise to confirm that it is of an acceptable scientific standard, however I have significant reservations, as outlined above.

Author Response 27 Nov 2020

Philippe Hupé

First, we would like to thank the reviewer. We are very grateful for his time, contribution and very valuable comments that significantly helped to improve the article.

You will find below a detailed answer to the different issues that we have addressed in the revised manuscript.

Best regards,

Choumouss Kamoun, Julien Roméjon and Philippe Hupé

Detailed response to the reviewer1's comments

> Motivated by laboratory accreditation requirements for the development of bioinformatics workflows related to high throughput sequencing in a clinical setting, the authors have developed a software development process formalisation based on gitflow, a popular model in mainstream software engineering practice.

> The formalisation prescribes a branching, testing, merging and release process, with set roles and permissions for developers, and methods for managing the development of new features (nominal mode) and bug fixing (hotfix mode).

> In addition to describing the biogitflow workflow, the paper also contains a short section on testing bioinformatics software pipelines, however, this is largely a recapitulation of standard testing terminology, and does not appear to be particularly bioinformatics-specific.

> The paper has a brief discussion about the motivation for adopting the workflow and some unsubstantiated claims about improving their software engineering practices. It also has a brief discussion about the need to train developers in the workflow and formal accreditation.

> Two large figures are included in the paper that attempt to illustrate the proposed workflow in nominal and hotfix modes.

> While the topic of this paper is interesting and highly relevant to bioinformatics software developers working in areas of clinical application, where process accreditation is a key concern, there are a number of areas where I think the paper could be improved:

> 1. The current version of the paper is lacking in critical analysis of the proposed workflow. One does not have to look far to find criticism (positive and negative) of gitflow. However, this paper does not provide any significant discussion of the pros and cons of the approach. One of the main criticisms of gitflow is that it is overly complex (for some software development projects) and thus prone to error and misapplication. This complexity is illustrated in Figures 1 and 2 and the discussion in the paper about the need to train developers in its use. Given the apparent regimented nature of the process, it seems plausible that some of the complexity could be

avoided by automation. Are there any useful tools that can help developers apply the model successfully and avoid the common criticisms of gitflow? If developers are finding it difficult to follow, especially the "tricky" parts then that may be a sign of limitations in the process or limitations in the tools supporting its application.

We fully agree that the protocols are complex and it is not straightforward to master them. Of note, we pointed out in our reply to your comment number 5 that the protocols are mainly relevant in the context of production to deliver to process data on the fly which is obviously a challenge especially when the bioinformatics pipelines have to deal with a very high-throughput of data. The complexity of the workflow just reflects the reality of the development and deployment cycle in the context of production as we intend to promote industrialized processes for the software implementation in order to offer better service, better reactivity, better traceability to end-users. Formalism was thus required and this is the reason why we wrote the biogitflow documentation.

You are also fully right when you say that it is "prone to error and misapplication" and that some "complexity could be avoided by automation". This is clearly an improvement according to the Deming wheel as the next step would be to capitalize on additional functionalities of git/gitlab using git hooks or gitlab-CI/CD (continuous integration, continuous deployment) and gitlab Operations features for example.

Anyway, as you have mentioned, this critical analysis and limitations were absolutely not highlighted. We, therefore, added a new paragraph at the end of the discussion to explain limitations and foreseen improvements with automation.

> 2. It is not clear how significantly biogitflow differs from gitflow. In what sense does the proposed workflow differ from established software engineering practices?

This comparison was clearly missing. We added a new paragraph at the beginning of the discussion that explains the main differences.

> 3. What alternative competing development models exist?

Other alternatives exist such as GitHub flow and GitLab flow. We added them in the section "Version control and branching model".

> 4. How can the authors be sure that the workflow is in fact making a positive difference to their software development?

This is a good question that is difficult to answer. This would require something like an experimental design comparing two or more workflows based on the assessment of objective performance metrics, and then compare the metrics to figure out if they are statistically different. We did not perform such an approach.

However, we can say that the benefit of the biogitflow protocols was two-fold: 1/ "harmonization of the development practices" and 2/ the "constructive and positive

pressure" for "better quality". These points have been added at the end of the second paragraph of the discussion.

> 5. It would be useful to clarify the intended audience of the paper. How can they apply and benefit from this work?

Indeed, the target audience was not clearly mentioned. Therefore, we added in the introduction that the protocols are appropriate for software developers in the "context of production" to "support core facilities" that generate "high-throughput of data". The protocols aim at promoting better "industrialized processes". The "context of production" has been reminded in the conclusion as well.

Obviously, any bioinformatician involved in a custom development to analyze data in a research project would not straightforwardly benefit from our work: for this use case, it might just be sufficient to track the developments with git using a single branch. However, we encourage anyone to read the proposed protocols for general knowledge as it can also give ideas for more elaborated needs.

> 6. The discussion of the provided biogitflow templates sounds useful, but the discussion in the paper is unclear. Perhaps this could be expanded upon?

We realized that both the link to the template and its content were indeed missing. We, therefore, added the url to the template and a more detailed description that also refers to the dedicated section in the biogitflow documentation website.

> 7. The paper ends with "We strongly encourage the promotion of such protocols not only for the healthcare activities but also for research activities." However, due to the aforementioned lack of critical analysis, it is not clear from the paper why this is true.

As mentioned in point number 5, we have detailed in the introduction who was the targeted audience and highlighted that these protocols are useful in the "context of production". We have modified the quoted sentence above highlighting again in the conclusion the usefulness of the protocols in the "context of production".

Competing Interests: No competing interests were disclosed.

The benefits of publishing with F1000Research:

- Your article is published within days, with no editorial bias
- You can publish traditional articles, null/negative results, case reports, data notes and more
- The peer review process is transparent and collaborative
- Your article is indexed in PubMed after passing peer review
- Dedicated customer support at every stage

For pre-submission enquiries, contact research@f1000.com

F1000Research