PIsCO: A Performance indicators framework for collection of bioinformatics resource metrics

Haydee Artaza Corresp., [1] , **John M. Hancock** [1] , **Rafael C. Jimenez** [2] , **Manuel Corpas** [3]

[1] Earlham Institute, Norwich, Norfolk, United Kingdom
[2] ELIXIR-Hub, Wellcome Genome Campus, Cambridge, United Kingdom
[3] Repositive, Cambridge, United Kingdom

Corresponding Author: Haydee Artaza
Email address: haydeeartaza@gmail.com

We present PIsCO, a server-side JavaScript framework for the collection, registration and sharing of metrics that can be used to evaluate the impact of bioinformatics-related resources such as software, repositories, training or databases. The metrics framework can be used to capture standard definitions of metrics, facilitate the collection of data, monitor resources and share data to be reused by other teams, laboratories or academic institutions. In addition, PIsCO is able to collect those metrics and present them in a visual way to allow their easy interpretation.

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017

PIsCO: A Performance Indicators Framework for COllection of Bioinformatics Resource Metrics

Haydee Artaza[1,*], John M. Hancock[1], Rafael C Jimenez[2] and Manuel Corpas[3]

[1]Earlham Institute, Norwich NR4 7UZ, U.K.

[2]ELIXIR Hub, Wellcome Genome Campus, Hinxton, Cambridge, CB10 1SD, UK.

[3]Repositive, Cambridge, CB4 2HY, U.K.

* Corresponding author

Abstract

We present PIsCO, a server-side JavaScript framework for the collection, registration and sharing of metrics that can be used to evaluate the impact of bioinformatics-related resources such as software, repositories, training or databases. The metrics framework can be used to capture standard definitions of metrics, facilitate the collection of data, monitor resources and share data to be reused by other teams, laboratories or academic institutions. In addition, PIsCO is able to collect those metrics and present them in a visual way to allow their easy interpretation.

Subjects

Software Engineering, Computer Architecture, Bioinformatics

Keywords

metrics, sharing, automatic collection, registries, key performance indicators.

Introduction

Biological communities work across a range of domains and use a variety of research resources (Wilsdon, 2015). The selection of a particular resource can be aided by performance indicators to allow investigators to make informed decisions about alternatives. Furthermore, scientists may also need these indicators to justify the funding of a particular resource.

Using metrics, scientists can assess the quality of academic resources and their broader impact (Ball and Duke, 2015). Moreover, impact metrics may be used to encourage best practice and 'FAIR' (Findable, Accessible, Interoperable, Reusable) principles (Wilkinson et al., 2016) in biological resources (although well-founded metrics for the FAIR principles remain to be established). The adoption of standardised metrics and methods of collection is needed to facilitate evaluation and comparison of resources (Artaza et al., 2016) by scientists, funders and academic institutions as performance indicators to assess resource impact and to support decision-making. For example Durinx et al (2016) describe a suite of indicators to evaluate potential core data resources as part of the ELIXIR project, following on from earlier efforts of

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017

this kind such as BioDBCore (Gaudet et al, 2011). The Human Variome Project has described a suite of metrics to determine the quality of gene sequence variation databases (Vihinen et al, 2016). It is important to avoid ad hoc development of metrics software which may be easily lost and not reproducible.

Here we describe PIsCO, a Node.js JavaScript framework for collection, registration, dissemination and reuse of biological resource metrics. PIsCO can be used to: a) provide standard definitions of metrics; b) facilitate software to collect metrics; and c) by automatically executing each metric's functionality, facilitate the monitoring and analysis of the stored metrics.

PIsCO Framework Design and Functionality

The PIsCO framework is implemented primarily with Node.js, which facilitates the reuse of libraries in the client and the server sides (Tilkov and Vinoski, 2010), and the NoSQL MongoDB document-oriented database (Chodorow, 2013).

This framework consists of three elements (Fig. 1), which work together to carry out the complete registration and monitoring of metrics processes. The first element, Component, defines the component schema and its functionality; the second element, Components Registry, allows the registration of component metadata, making components available for use; the third element, Data and Monitoring Repository, installs and executes components, and collects data from the component's execution. Metrics results generated from each component's execution are stored in a MongoDB database to allow them to be used and interpreted.

Component

A Component is the basic unit defined in the PIsCO Framework (Fig. 2). It consists of two descriptors:

- Standard definition, following a common schema described in a XML metadata file (Fig. 3) that defines a set of parameters used in deploying a component (name, dependencies, frequency, resource, output, repository, etc) (see Specification 3.1.1 in Supplementary Material).
- Implementation/functionality, which follows a basic structure: code (written in JavaScript for NodeJS applications), documentation, guidelines, examples, and other element of interest for this component which can be added on an ad hoc basis. This directory structure should be stored in some source code management system: a software tool used by teams of programmers to manage source code (e.g. GitHub, GitLab, SourceForge, etc). When the component is installed this structure will be fetched, transferred and installed into this component.

Components Registry

This element allows the registration of component metadata (XML metadata file with a set of parameters, see Fig. 3), making components discoverable and available for use. The component's metadata are used to install components into the common repository.

Data and Monitoring Repository

This element installs and executes components using the component metadata registered by the Components Registry element. Moreover, data collected from each component's execution (the "metrics") are stored in a MongoDB database to allow them to be used and interpreted.

The metrics database organizes the collected data, grouping them in a three-dimensional format: resource-metric-frequencies. Each resource is associated with one or more metrics and the resource-metric pair is monitored at a specified frequency (see Specification 3.1.3 in Supplementary Material). This data can be exported as a csv file or it can be accessed using a GUI (Graphical User Interface) where different metrics graphs are accessible.

Operation

The PIsCO framework has been designed to make installation as simple as possible. The software requirements are:

1. Operating system: Linux or Mac OS
2. Nodejs (last version tested: v6.0.0)
3. Npm: a package manager for the JavaScript programming language (last version tested: v3.8.6)
4. MongoDB (last version tested: shell v3.0.4)

The GUI has been developed to be run in Google Chrome (version 56.0.2924.87/64-bits) and Mozilla Firefox (version 51.0.1/64-bits). Both of them were tested. The documentation, user manual and specification, is available in GitHub repository.

Example of an Application / Use Case

Using a bioinformatics resource as part of a scientific project could depend on having performance indicators that allow investigators to make informed decisions on different alternatives (Ball and Duke, 2015). In this very simple use case, we consider a metric to assess the frequency with which selected bioinformatics tools or packages, with an emphasis on alignment software, are looked up in Wikipedia. It should be noted that this scenario aims to provide a simple example to show the applicability of the PIsCO framework; we would expect real-life examples to be more complex.

This scenario uses the Pageviews metric. This metric gets the articles' pageviews trends on specific articles or projects in Wikipedia using the Wikipedia Analytics/Pageview API. The Pageviews metric was implemented as follows:

- A selection of bioinformatics resources was extracted from the list of sequence alignment software provided by Wikipedia (List of sequence alignment software, Wikipedia, 2016).
- Metadata for the Pageviews metric were described in a XML file (available in GitHub, see Software Availability).
- The metric was registered through the Registry GUI (see User Manual in Supplementary Material). Its metadata were collected and stored in the framework Registry.
- Once registered, the Pageviews metric was ready to be installed, bringing the code from the source code management system and setting up external dependencies, executed, and monitored automatically.
- Data collected from these metrics were stored in the framework Repository (metrics database) and were available to use.

Metric data, the number of visits in the previous 24 hours, were collected daily through July 2016. Pages for ten selected bioinformatics tools or packages were assessed: BFAST, Bioconductor, BioPerl, BLAST, Clustal, FASTA, HMMER, SAMtools, T-Coffee and UGENE. Each has an entry in a Wikipedia article. The trends of Wikipedia views for these ten resources, in the 31 days of July, are shown in Figure 4. The total number of views for each resource are provided in Table 1. Using these results, in addition to the graphical results, it may be seen that, of those considered, BLAST was the most accessed bioinformatics tool or package in Wikipedia over this period (15396 views), and BFAST the least accessed (108 views). On average, BLAST had more than ten times as many accesses as other tools (see Table 1).

According to Neumann et al. (2013) BLAST is widely used because of its high speed and efficient algorithm. Moreover, biologists employ BLAST as a first choice for sequence database searching because of the widely available public interfaces, in particular NCBI BLAST (http://blast.ncbi.nlm.nih.gov). This is likely to explain why BLAST has many more views in Wikipedia than the other tools. This simple example only uses a single metric. Gathering more metrics using the PIsCO framework would enable a more rounded view of the relative popularity of BLAST and other tools and these can be used by the user to do new interpretations.

Conclusion

We describe the PIsCO framework for collection, dissemination and reuse of biological resource metrics. Data collected from metrics can be used by scientists, funders and academic institutions as performance indicators to assess the impact of a variety of biological resources and as performance indicators useful to complement decision-making.

Unlike other similar projects like ImpactStory, ReaderMeter, or Altmetrics, the PIsCO framework is totally open source software (available via a web based interface and a command line interface), easy to install and use. Moreover, metrics executed in our framework can be used for different purposes (not just citation-based metrics), according to user's needs. In this way, users can define metrics to be applied for databases, training material, software, repositories, etc., including outside the bioinformatics domain. All these metrics can be shared, reused and disseminated because these will be located in a common repository. Defining, developing and interpreting the metrics themselves are the domain of experts and developers.

Software Availability

Latest source code for the pipeline is publicly available on GitHub:
https://github.com/BioPisCO/pisco-metrics-framework.
https://github.com/BioPisCO/metrics-module-pageviews.

Licence: MIT

Author contributions

All of the authors participated in designing the study, carrying out the research, and preparing the manuscript. All authors were involved in the revision of the draft manuscript and have agreed to the final content.

Competing interests

No competing interests were disclosed.

Grant Information

H.A. was supported by the ELIXIR Implementation Study “Metrics discovery and implementation in life sciences”.

Supplementary Material

Supplementary material for this article can be found online at …..

References

Artaza H, Chue Hong N, Corpas M, Corpuz A, Hooft R, Jimenez RC, Leskosek B, Olivier BG, Stourac J, Svobodova Varekova R, Van Parys T, and Vaughan D. 2016. Top 10 metrics for life science software good practices. *F1000Res* 5. 10.12688/f1000research.9206.1

Ball A, and Duke M. 2015. How to track the impact of research data with metrics. *DCC How-to Guides*.

Chodorow K. 2013. *MongoDB: The Definitive Guide: Powerful and Scalable Data Storage*: " O'Reilly Media, Inc.".

Durinx C, McEntyre J, Appel R, Apweiler R, Barlow M, Blomberg N, Cook C, Gasteiger E, Kim JH, Lopez R, Redaschi N, Stockinger H, Teixeira D, and Valencia A. 2016. Identifying ELIXIR Core Data Resources. *F1000Res* 5. 10.12688/f1000research.9656.2

Gaudet P, Bairoch A, Field D, Sansone SA, Taylor C, Attwood TK, Bateman A, Blake JA, Bult CJ, Cherry JM, Chisholm RL, Cochrane G, Cook CE, Eppig JT, Galperin MY, Gentleman R, Goble CA, Gojobori T, Hancock JM, Howe DG, Imanishi T, Kelso J, Landsman D, Lewis SE, Karsch Mizrachi I, Orchard S, Ouellette BF, Ranganathan S, Richardson L, Rocca-Serra P, Schofield PN, Smedley D, Southan C, Tan TW, Tatusova T, Whetzel PL, White O, Yamasaki C, and Bio DWG. 2011. Towards BioDBcore: a community-defined information specification for biological databases. *Database (Oxford)* 2011:baq027. 10.1093/database/baq027

Neumann RS, Kumar S, and Shalchian-Tabrizi K. 2014. BLAST output visualization in the new sequencing era. *Brief Bioinform* 15:484-503. 10.1093/bib/bbt009

Parmenter D. 2007. Key performance indicators, Hoboken. John Wiley & Sons, Inc.

Tilkov S, and Vinoski S. 2010. Node. js: Using JavaScript to build high-performance network programs. *IEEE Internet Computing* 14:80-83.

Vihinen M, Hancock JM, Maglott DR, Landrum MJ, Schaafsma GC, and Taschner P. 2016. Human Variome Project Quality Assessment Criteria for Variation Databases. *Hum Mutat* 37:549-558. 10.1002/humu.22976.

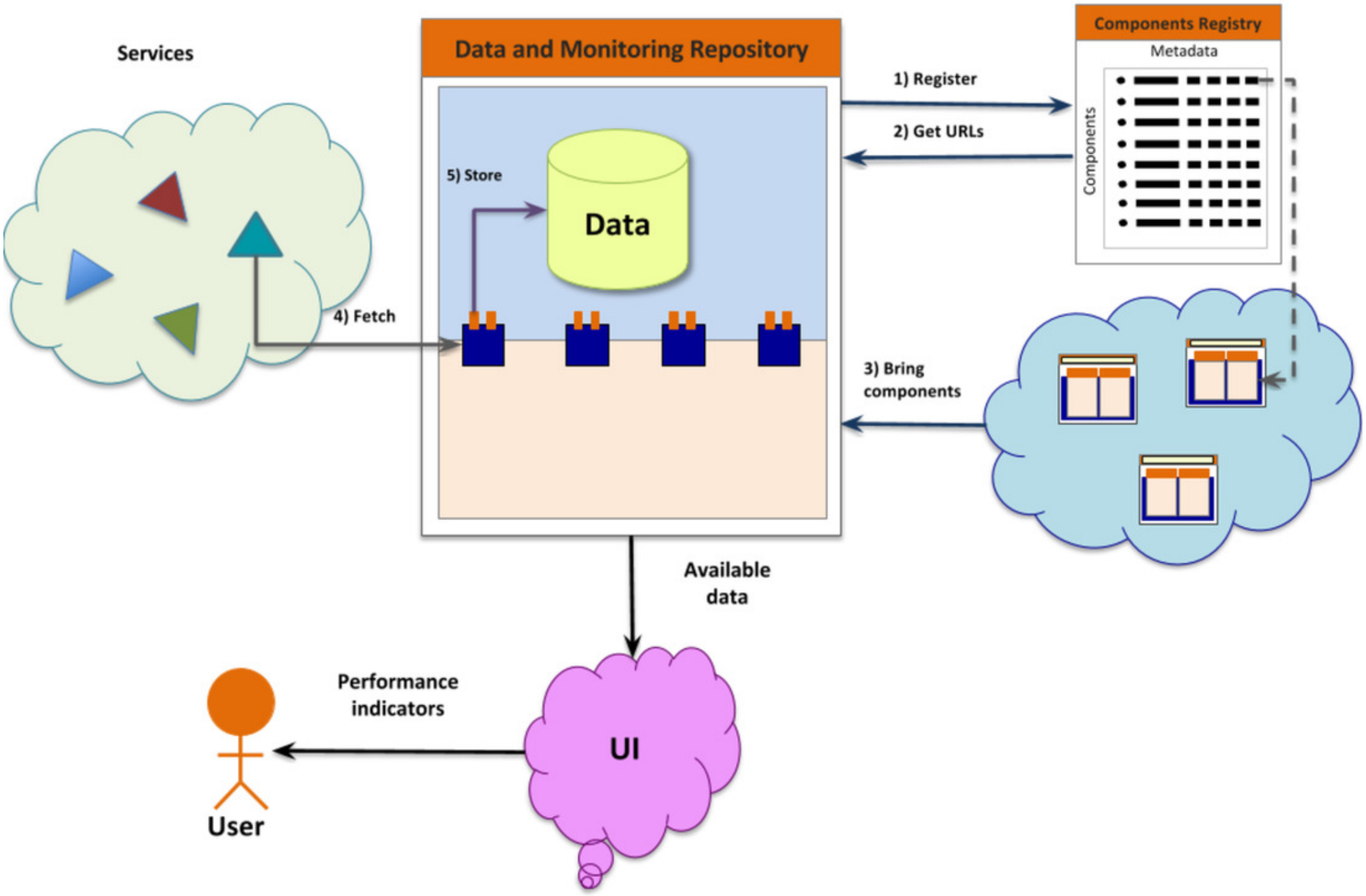
Wilkinson MD, Dumontier M, Aalbersberg IJ, Appleton G, Axton M, Baak A, Blomberg N, Boiten JW, da Silva Santos LB, Bourne PE, Bouwman J, Brookes AJ, Clark T, Crosas M, Dillo I, Dumon O, Edmunds S, Evelo CT, Finkers R, Gonzalez-Beltran A, Gray AJ, Groth P, Goble C, Grethe JS, Heringa J, t Hoen PA, Hooft R, Kuhn T, Kok R, Kok J, Lusher SJ, Martone ME, Mons A, Packer AL, Persson B, Rocca-Serra P, Roos M, van Schaik R, Sansone SA, Schultes E, Sengstag T, Slater T, Strawn G, Swertz MA, Thompson M, van der Lei J, van Mulligen E, Velterop J, Waagmeester A, Wittenburg P, Wolstencroft K, Zhao J, and Mons B. 2016. The FAIR Guiding Principles for scientific data management and stewardship. *Sci Data* 3:160018. 10.1038/sdata.2016.18

Wilsdon J, Allen L, Belfiore E, Campbell P, Curry S, Hill S, Jones R, Kain R, Kerridge S, and Thelwall M. 2015. The Metric Tide: Report of the Independent Review of the Role of Metrics in Research Assessment and Management. Bristol: Higher Education Funding Council for England.

Figure 1

PIsCO framework.

PIsCO consists of three elements for carrying out the complete registration and monitoring processes. The first element, **Component**, defines the metric schema and functionality; the second element, **Components Registry**, registers the component metadata into a registry to make it available for use; the third element, **Data and Monitoring Repository**, installs and executes metric components and collects data from the component execution. Data generated from each component can be visualised for their further analysis and interpretation.

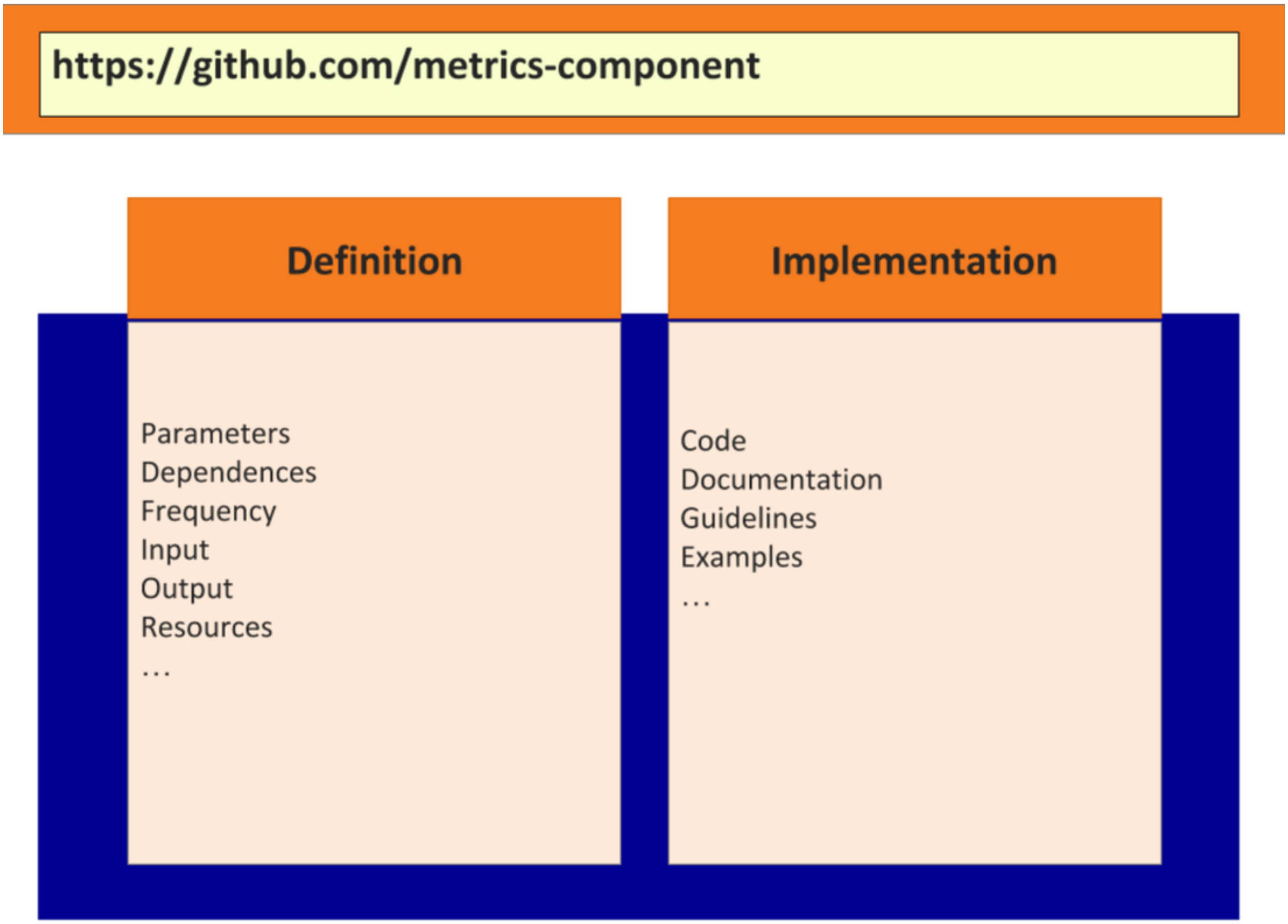


PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017

Figure 2

Basic component structure.

This figure shows the two differentiated parts in a component: standard definition (metadata) and functionality (Code, documentation, guidelines, and examples).

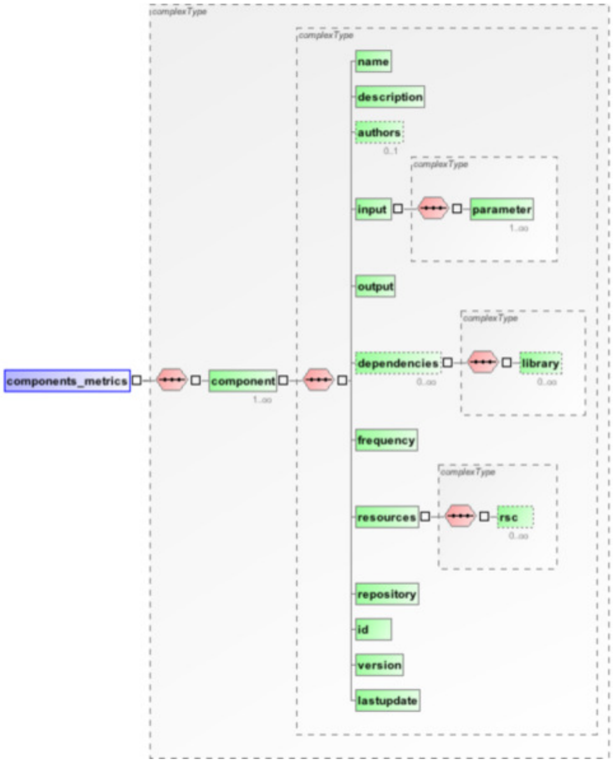


PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017

Figure 3

Graphical component metadata schema.

This graphic shows the component schema hierarchy. This metadata defines a set of parameters used in deploying a component: name, dependencies, frequency, resource, output, repository, etc.

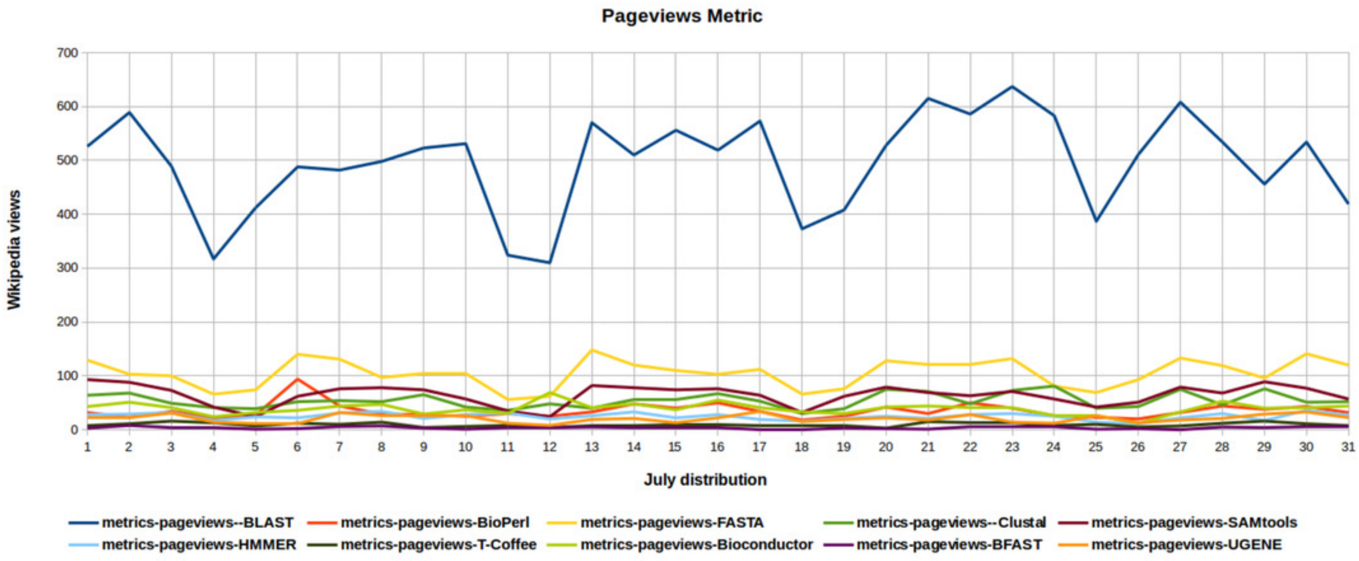


PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017

Figure 4

Daily Pageviews data from Wikipedia.

Visualizations associated with each bioinformatics resource, monitored daily on July. The blue line shows the BLAST resource trend, this resource is the Wikipedia article more visited.



PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017

Table 1(on next page)

Metric Pageviews results.

Total number of views of the Wikipedia article for each resource during July 2016.

Metric	Wikipedia division	Resource	Total
Pageviews	Database search	BLAST	15396
		FASTA	3256
		SAMtools	1996
		HMMER	768
	Pairwise alignment	Bioconductor	1211
		BioPerl	1081
	Multiple sequence alignment	Clustal	1682
		UGENE	643
		T-Coffee	292
	Short-read sequence alignment	BFAST	108

PeerJ Preprints | https://doi.org/10.7287/peerj.preprints.3112v1 | CC BY 4.0 Open Access | rec: 27 Jul 2017, publ: 27 Jul 2017