Flexible Data Access Framework for Neuroinformatic Databases

D. MacFarlane[1], X. Lecours-Boucher[1], R. Abou-Haider[1], D. Lo[1], M. Legault[2], L. MacIntrye[1], Samir Das[1], A.C. Evans[1]

[1]McGill Center for Integrative Neuroscience, Montreal Neurological Institute, McGill University, Montréal, QC, Canada

[2]Montreal Neurological Institute, McGill University, Montréal, QC, Canada

Introduction

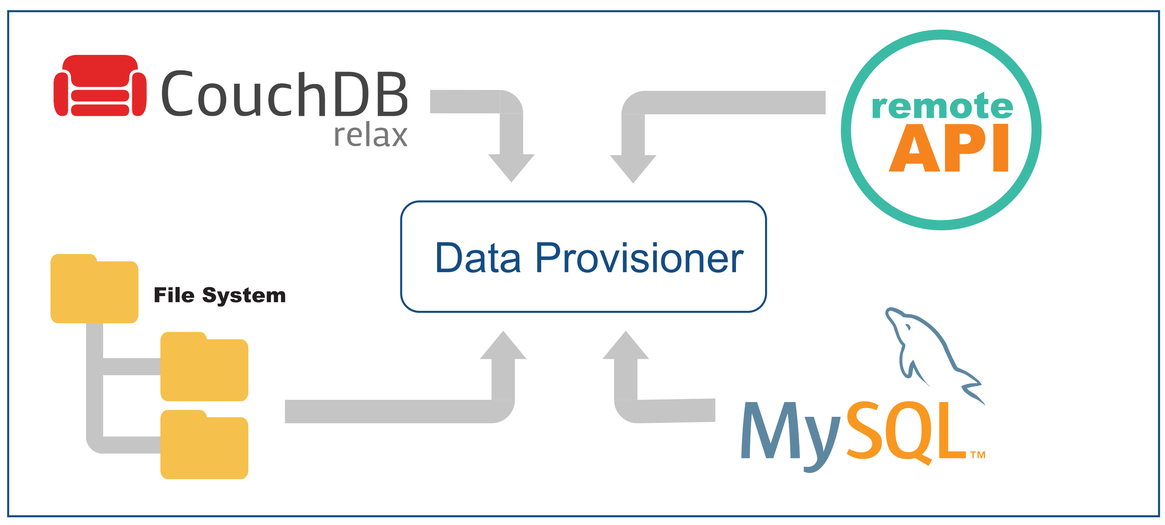
Large databasing frameworks, such as LORIS[1] have a number of features[2,3,4] across many modules to support neuroimaging studies. Maintaining modules becomes increasingly difficult with the addition each of new feature or module. LORIS created a Data Access Framework that simplifies gathering and filtering data in a consistent manner to address this difficulty.

Prior to the Data Access Framework, LORIS modules accessed data on an ad-hoc basis by writing bespoke SQL queries generated from object oriented base classes. Features such as site-based permissions required updating every query across all LORIS modules to filter user-restricted data. Rather than object-oriented principles, the Data Access Framework provides a high level interface inspired by functional programming and considers data presentation to be a pure function of the user and the data itself. Actions are defined in immutable terms[5], while lazy evaluation[6] ensures efficiency of data retrieval. This has unified the data access in LORIS and resulted in the ability to easily add more granularity to permissions.

Methods

The Data Access Framework is modelled in terms of “data provisioners”, responsible for retrieving data from a data source and returning a data model.

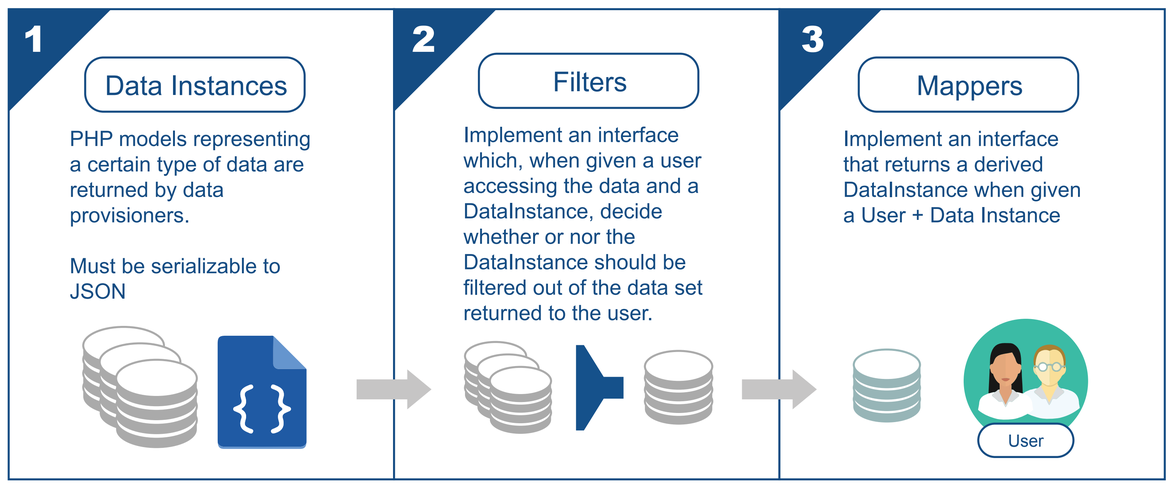
The Data Access Framework takes a high level view of “data” and conceptually separates the filtering from the retrieval of data. This allows the same framework to be used for different data sources accessed by LORIS while using the same framework for filtering and modifying the data representation presented to the user.



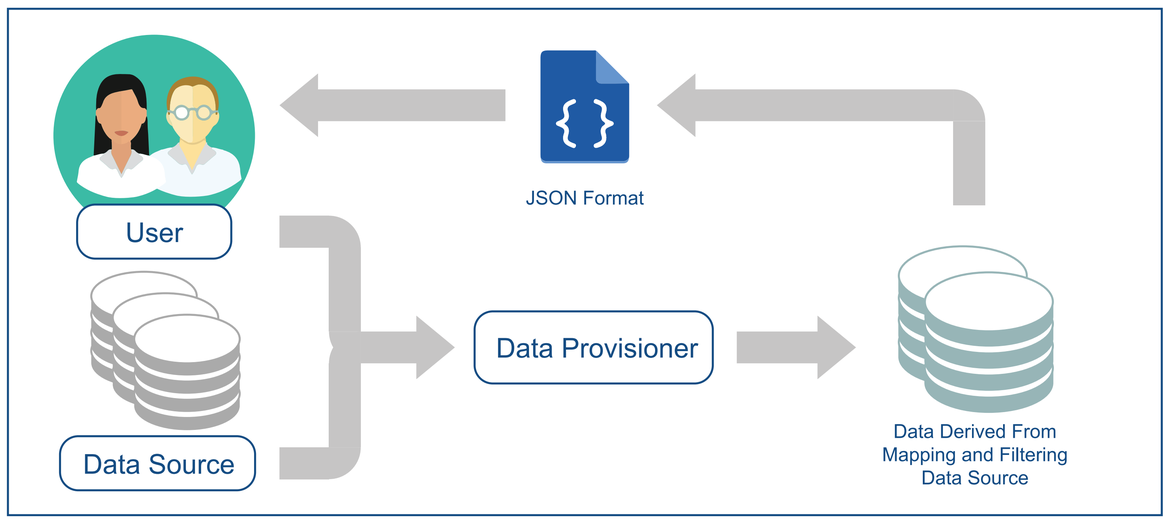
Data can be retrieved from different sources using a common interface

Architecture

The framework is implemented in terms of three core concepts:



Data provisioners are responsible for retrieving the data, and applying relevant maps and filters. Standard filters included in LORIS can be applied to implement consistent filtering across data models. For instance, the UserSiteMatch filter checks whether the model for the DataInstance that it is given has a getCenterID method. If so, it will filter out DataInstances for users which are not part of that centre. The filter can be applied regardless of the source(s) that the data provisioner used to build the data model (SQL, NoSQL, or anywhere else) without knowledge of the underlying source, so long as the PHP data model for the DataInstance has a getCenterID() method.



Data provisioners map and filter data based on the data and the user accessing it.

Results

An implementation of this framework in LORIS has been done including the UserSiteMatch filter described in the architecture section and an anonymization mapper in the DICOM Archive module.

Benchmarking showed performance as comparable to the previous direct SQL implementation with less variability across users, while code re-use is improved by writing the filtering logic in higher level classes that deal with data models, not directly with the database.

Conclusion

We were able to improve the reliability and robustness of the data access in LORIS by implementing a novel framework inspired by functional, rather than object oriented programming concepts.

In future work, the framework will be used to consistently implement more granular permissions such as project, cohort, or imaging modality specific permissions consistently throughout LORIS.

References

1. Das et al, 2011. LORIS: a web-based data management system for multi-center studies. Frontiers in Neuroinformatics (5:37)
2. MacFarlane et al. Enhanced Data Querying For Neuroinformatics Databases. OHBM 2014
3. Mohadez et al. Web-based Imaging Uploader For LORIS. OHBM 2015
4. Rogers et al. LORIS Neuroinformatics Platform for Imaging Genetics. OHBM 2015
5. P. Helland. Immutability changes everything. CACM Jan. 2016 (DOI 10.1145/2844112)
6. B. W. Lampson. Lazy and speculative execution in computer systems. ICFP 2018 (DOI 10.1145/1411203.1411205)