The CIVET project intends to consistently provide developers with the tools they need to conduct experiment analyses, and researchers at the McConnell Brain Imaging Centre (BIC). Simultaneously, it should allow developers and programmers with the flexibility of modifying the processes that operate on their data.

CIVET is designed as an interface linking ever-growing and improving imaging software with the researcher’s demands for reproducibility and consistency of results. It will enable researchers to perform vertex-based corticometric tasks (VBM), voxel-based morphometric tasks (VBM), symmetry and symmetry analysis.

The CIVET project will attempt to provide extensive documentation that would boost the value of users’ existing degrees of computational experience.

**Methods**

Development of CIVET has been pursued in parallel along eight main directions:

1. **Centralization of maintenance and debugging of quarantined software.**
2. **Modularization of CIVET code-base to minimize inter-component dependency so that each component (set of processing activities, see Fig. 1) can be independently developed, and then combined into a customized analysis pipeline.**
3. **Implementation of Automated Quality-Control (AQC) to register, classification, and surface-fitting using reference distributions to identify outlier voxels and vertices. (see Fig. 1)**
4. **Development of an independent debugging tool that assures integrity of CIVET.**
5. **Debugging and standardization of CIVET modules; Further developed PMP developed methods for constructing data-based sets with AQC outputs.**
6. **Kelvin Mok: Developed independent debugging tool; Oleg Ivanov: Developed the script building the GUI wrapper;Debugging and standardization of CIVET modules; Further developed PMP developed methods for constructing data-based sets with AQC outputs; Integrated CIVET’s architecture is now modular in design, with on-going parallel task-specific development.**
7. **Development of a JAVA-based graphical-user interface (GUI) wrapper for CIVET (Fig. 2) that activates a Perl-script which in turn sources the desired quarantine versions for hundreds of software packages available at the BIC in a controlled manner on most platforms.**
8. **Detailed ongoing documentation by the community of developers working on any component of the project.**

**References & Credits**

- **CIVET was developed collaboratively by:**
  - Yasser Ad-Dabagh: Developed modular versions of CIVET; Project coordination; Assessment of output quality; Interface design for GUI wrapper; Documentation...
  - Dale Einanson: Major role in developing the early quarantine-building process; Perl-script that automates the process of installing the latest stable versions for platforms.
  - J-Sebastian Muehlboeck: Developed a JAVA-based graphical-user interface (GUI) wrapper for CIVET (Fig. 2) that activates a Perl-script which in turn sources the desired quarantine versions for hundreds of software packages available at the BIC in a controlled manner on most platforms.
  - Robert D. Vincent: Developed pre-modular versions of CIVET; Project coordination; Assessment of output quality; Interface design for GUI wrapper; Documentation...
  - Kelvin Mok: Developed independent debugging tool; Oleg Ivanov: Developed the script building the GUI wrapper;Debugging and standardization of CIVET modules; Further developed PMP developed methods for constructing data-based sets with AQC outputs; Integrated CIVET’s architecture is now modular in design, with on-going parallel task-specific development.
- **AQC was developed collaboratively by:**
  - Claude Lepage: Developed the 3rd edition of CLASP; Further developed and improved the quarantine-building architecture; Contributed tools such as Diffuse and mosio2; Debugging and standardization of CIVET modules.
  - Kelvin Mok: Developed independent debugging tool; Oleg Ivanov: Developed the script building the GUI wrapper;Debugging and standardization of CIVET modules; Further developed PMP developed methods for constructing data-based sets with AQC outputs; Integrated CIVET’s architecture is now modular in design, with on-going parallel task-specific development.
  - Claude Lepage: Developed the 3rd edition of CLASP; Further developed and improved the quarantine-building architecture; Contributed tools such as Diffuse and mosio2; Debugging and standardization of CIVET modules.
  - Kelvin Mok: Developed independent debugging tool; Oleg Ivanov: Developed the script building the GUI wrapper;Debugging and standardization of CIVET modules; Further developed PMP developed methods for constructing data-based sets with AQC outputs; Integrated CIVET’s architecture is now modular in design, with on-going parallel task-specific development.

**Results and Conclusions**

CIVET’s architecture is now modular in design, with on-going parallel task-specific development. The process of development is collaborative among developers and depends on regular feedback from end-users. Based on the Perl-based Poor Man’s PipeLine (PMP) architecture, it is composed of a pipeline-invoking shell and core modules that control the parallel processing of data (using MINC, MINI-AutoSeg, INSEG, ANIMAL, CLASPP, SURFREG3, Diffuse), among others, enabling both voxel- and vertex-based analysis of data in a safe, dependency- and integrity-managed environment. This core architecture can be invoked to run any set of processing tasks with a single command-line, or through the use of a very user-friendly GUI.

CIVET is a comprehensive, easy to use and implement pipelining environment for fully automated image-processing of large data-sets that meets the needs of both developers and end-users and enables researchers to focus on scientific questions both at the biological as well as the computational ends.