Powertrain Documentation

Release 0.0

VUIIS

May 21, 2014

Powertrain is not a new idea. Many groups around the world are attempting to create scalable systems for large-scale medical image processing. In this respect, Powertrain is simply yet another attempt at building such a system.

Powertrain relies on three separate systems:

- REDCap, the web-based data-capture system from Vanderbilt.
- XNAT, eXtensible Neuroimaging Archive Tool from WUSTL.
- ACCRE, Vanderbilt's compute cluster.

Contents:

CHAPTER 1

The Problem & Goals of Powertrain

Powertrain is written for researchers using VUIIS MR imaging infrastructure at Vanderbilt. While ideas implemented within Powertrain may have broad applicability, we are designing Powertrain to be most useful to Vanderbilt researchers who capture MR data at VUIIS. As such, this is not a generalized system for running large-scale imaging analyses; we will make design & implementation decisions specific to infrastructure available at Vanderbilt. If similar infrastructure is available elsewhere, other research engineers may find Powertrain useful.

1.1 The Problem

Data management (DM) of large neuroimaging projects is wide-ranging & difficult. Some things to consider:

- Storage & archival of raw data.
- Storage & processing of analyzed data.
- Linkability between MR data, subject demographics and out-of-/in-magnet behavioral tasks.
- Data provenance. Can we reliably reproduce the processing done to a particular dataset?

Unfortunately poor DM planning & implementation can ruin projects and good DM is difficult to design and more difficult to achieve. However, good DM enables quite a few features important to a projects success including:

- Automatically-generated subject- and group-level analyses
- Integrated data security & backup
- Sharing of best practices with respect to neuroimaging analyses
- Reduced costs of equipment & personnel through shared infrastructure.

The majority of laboratories running neuroimaging experiments consider & run each project separately. Different people, computing resources & standard operating procedures all decrease replicability & reproducibility while increasing costs. More advanced labs may develop software and resources to manage their studies together. Rarely do they consider how other labs might operate.

The principle idea behind Powertrain is that at a high level, all neuroimaging studies are the same and can be modeled appropriately. Given such a model, proper data management should be a service executed at the institution level. Only at this level can resources be optimally used & shared among all users.

1.2 Standing on the shoulders of giants

Briefly, Powertrain is built upon the following existing (and externally managed) systems:

1.2.1 XNAT

XNAT is a widely used informatics platform for medical imaging research. The platform provides a central archive for multiple imaging systems (in this case, MR scanners) to push data using the DICOM standard for medical imaging. To access data, researchers can use both a web-based user interface or REST-based Application Programming Interface (API). Powertrain will predominantly use the XNAT API to download raw images and upload processed data.

The VUIIS installation of XNAT can be found here. For Powertrain, the VUIIS XNAT serves as the archive for all images, both raw and processed. Raw data from the MR scanners go directly to XNAT and all processed data is uploaded back to XNAT. Many layers of data security & backup are running against this system.

1.2.2 REDCap

REDCap is a web-based system for capturing & validating both research and clinical data. Users can easily create production-quality databases in an afternoon and quickly begin capturing data for research or clinical purposes. REDCap also provides an API to export & import both data and files saved through the web interface or generated elsewhere. Powertrain will make heavy use of this API to securely download subject demographics & image session metadata and upload non-imaging data produced during image analyses.

By putting data extracted during image processing alongside subject demographics & behavioral data, researchers can quickly build group-level analyses. For instance, selecting subjects for a functional MRI (fMRI) analysis involves filtering by demographics (age, race, etc), out-of-magnet behavioral data (IQ, task performance, etc), in-magnet behavioral data (fMRI task performance) and image quality (SNR, motion, pre-processing artifact detection, etc). All of these measures can be stored in REDCap and be used to programmatically & reproducibly build second-level analyses.

On the other hand, metadata about the images collected on a session basis including the order of images captured can likewise be captured in REDCap. Powertrain will rely heavily on this information to appropriately organize raw images into "paradigms" or groups of related images.

Because Powertrain and not a human operator is organizing the images, many levels of image analyses can be automatically generated & processed.

1.2.3 ACCRE

The Vanderbilt Advanced Computing Center for Research & Education (ACCRE) provides advanced computing infrastructure for large-scale image processing. Including a computing cluster with over 7000 processor cores, networked storage between compute nodes and an interface for submitting processing jobs to the compute cluster, ACCRE forms the backbone of the computing infrastructure Powertrain requires for large-scale neuroimage processing.

In general, image processing requires the following steps:

1. Organization of raw data.
2. Script generation around publicly-available or in-house analysis programs.
3. Submission of these scripts to the compute cluster.
4. Asynchronous handling of finished and crashed jobs.
5. Manual Quality Assurance (QA) for correctness & usability.

Powertrain can handle steps 1-4 but because different research groups may perform QA with different parameters of interest, Powertrain will instead make it as easy as possible for researchers to view rendered images of their data and make QA judgements. Powertrain will send this QA classification to REDCap and/or XNAT, based on the researchers needs.

1.3 Powertrain Web Interface

Users will interact with Powertrain primarily through a web-based interface. Through the UI, they will be able to accomplish the following tasks:

- Create an account for themselves and request permission to view and/or administer projects.
- View available projects and sessions within each project.
- Within each session, view completed processing tasks. Captured logs of the processing as well as any rendered images will be presented.
- If the user is an administrator to the project, they will be able to make QA judgements through the interface.
- If the user is an administrator, they can reset processing tasks so that Powertrain will re-submit the required processing to the compute cluster.
- If the user can only view sessions, they can flag certain processing results for review by an administrator. The administrators of the project will get an email notifying this action.

1.4 Powertrain Programmatic Interface

Initially, the programmatic interface to Powertrain will be limited to just receiving Data Entry Triggers from REDCap. These triggers are generated by the REDCap server when data is saved within projects. These triggers give reseachers the power to drive imaging data management within Powertrain simply by using their own REDCap database(s).

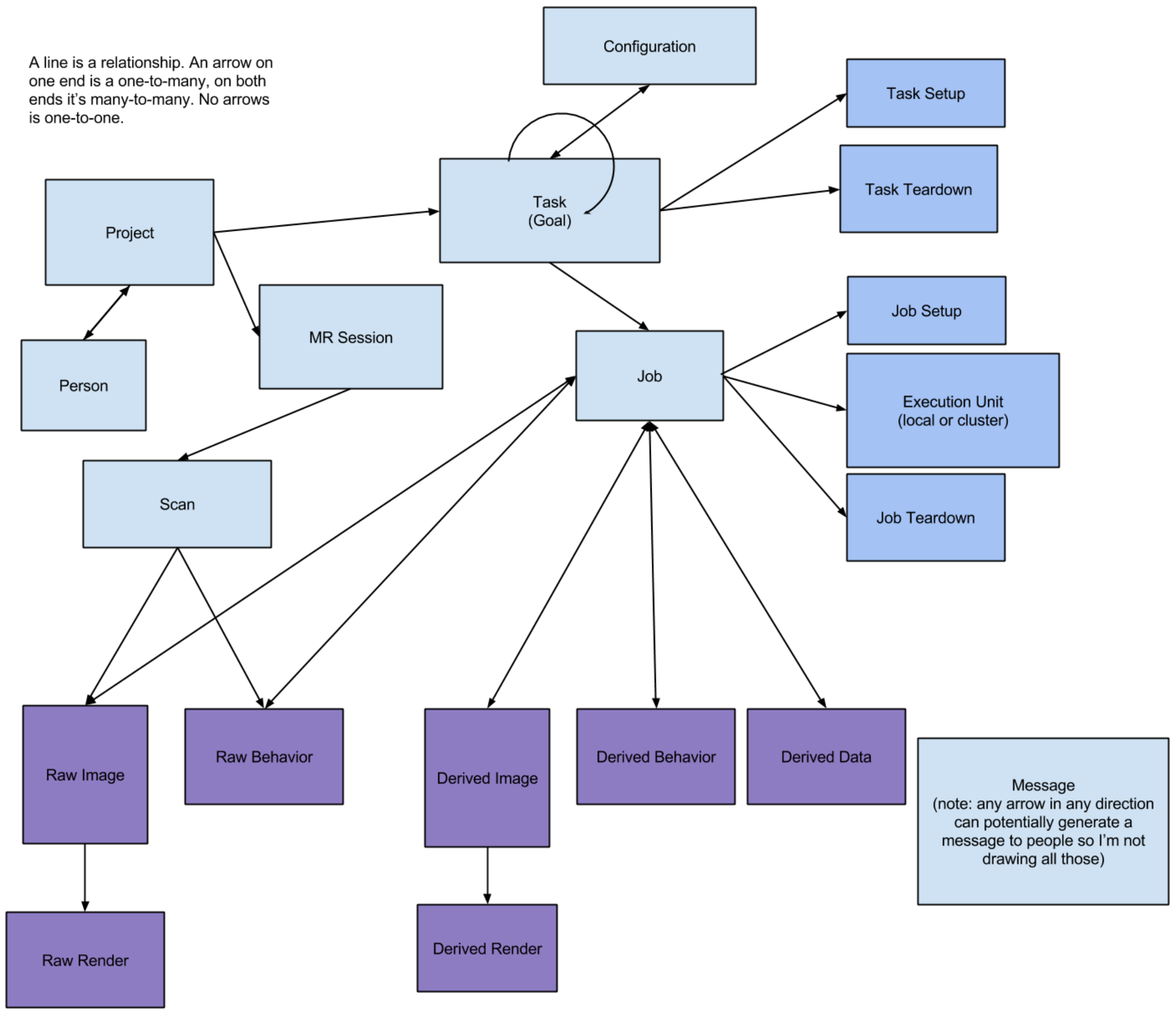
In the future we may explore implementing a RESTful API on top of Powertrain but this is not concrete.

CHAPTER 2

Technical Specification

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in RFC 2119

2.1 Models


A line is a relationship. An arrow on one end is a one-to-many, on both ends it's many-to-many. No arrows is one-to-one.
Configuration
Task Setup
Task Teardown
Task (Goal)
Project
MR Session
Person
Job
Job Setup
Execution Unit (local or cluster)
Job Teardown
Scan
Raw Image
Raw Behavior
Derived Image
Derived Behavior
Derived Data
Message (note: any arrow in any direction can potentially generate a message to people so I'm not drawing all those)
Raw Render
Derived Render

CHAPTER 3

Application Programming Interface

3.1 Models

High Level

class `powertrain.models.`**`Project`**(***kwargs*)
TODO: Documentation for Project

class `powertrain.models.`**`User`**(***kwargs*)
TODO: Documentation for User

`confirm`(*token*)
Confirm `User.email` is valid and the actual human has access to this account.

Parameters **token** (*str*) – Confirmation token. Generated with `generate_confirmation_token`

`generate_confirmation_token`(*expiration=86400*)
Generate a token used to confirm email addresses

Parameters **expiration** (*int*) – Time (seconds) after which token doesn't work

`generate_reset_token`(*expiration=86400*)
Generate a token for resetting a password

Parameters **expiration** (*int*) – Time (seconds) after which token doesn't work

`reset_password`(*token*, *new_password*)
Reset a password when given a proper token.

Handling matching passwords is done elsewhere.

Parameters

- **token** (*str*) – Generated with `User.generate_reset_token`
- **new_password** (*str*) – New password

`verify_password`(*password*)
Returns True when incoming password matches hashed version

Within Projects

class `powertrain.models.`**`MRSession`**(***kwargs*)
TODO: Documentation for MRSession

class `powertrain.models.`**`Configuration`**(***kwargs*)
TODO: Documentation for Configuration

class `powertrain.models.`**`Task`**`(`***kwargs*`)`
TODO: Documentation for Task

class `powertrain.models.`**`TaskSetup`**`(`***kwargs*`)`
TODO: Documentation for TaskSetup

class `powertrain.models.`**`TaskTeardown`**`(`***kwargs*`)`
TODO: Documentation for TaskTeardown

Within MRSessions

class `powertrain.models.`**`Scan`**`(`***kwargs*`)`
TODO: Documentation for Scan

class `powertrain.models.`**`RawBehavior`**`(`***kwargs*`)`
TODO: Documentation for RawBehavior

class `powertrain.models.`**`DerivedImage`**`(`***kwargs*`)`
TODO: Documentation for DerivedImage

class `powertrain.models.`**`RawImage`**`(`***kwargs*`)`
TODO: Documentation for RawImage

class `powertrain.models.`**`RawRender`**`(`***kwargs*`)`
TODO: Documentation for RawRender

class `powertrain.models.`**`DerivedRender`**`(`***kwargs*`)`
TODO: Documentation for DerivedRender

class `powertrain.models.`**`Job`**`(`***kwargs*`)`
TODO: Documentation for Job

class `powertrain.models.`**`ExecutionUnit`**`(`***kwargs*`)`
TODO: Documentation for ExecutionUnit

class `powertrain.models.`**`JobSetup`**`(`***kwargs*`)`
TODO: Documentation for JobSetup

class `powertrain.models.`**`JobTeardown`**`(`***kwargs*`)`
TODO: Documentation for JobTeardown