Journal of
open research software

SOFTWARE METAPAPER

Transplant2Mongo: Python Scripts that Insert Organ Procurement and Transplantation Network (OPTN) Data in MongoDB

Christine Harvey[1,2] and R. S. Weigel[2]

[1] The MITRE Corporation, US

[2] George Mason University, US

Corresponding author: Christine Harvey (ceharvs@gmail.com)

`transplant2mongo` allows users to transform Standard Transplant Analysis and Research (STAR) ASCII data files from the Organ Procurement and Transplantation Network (OPTN) into a MongoDB database [1, 2]. The STAR data are a complex collection of tab-separated files with inter-related records that are not amenable to complex queries. A researcher planning to use OPTN STAR data can use `transplant2mongo` to convert the data into a MongoDB database and then use open-source tool software for analysis. The source code for `transplant2mongo` is available on GitHub at https://github.com/ceharvs/transplant2mongo and includes sample data files for initial testing and queries.

Keywords: Python; OPTN; UNOS; organ transplant; organ donation; health care; analysis; MongoDB

Funding Statement: The software referenced herein is copyright of The MITRE Corporation and the result of MITRE's Early Career Research program and work done in the Computational Science and Informatics program at George Mason University. Approved for Public Release; Distribution Unlimited. Case Number 18-0298. The author's affiliation with The MITRE Corporation is provided for identification purposes only and is not intended to convey or imply MITRE's concurrence with, or support for, the positions, opinions, or viewpoints expressed by the author.

(1) Overview

Introduction

The OPTN keeps a record of all organ donations, transplants, and waiting list registrations since 1987 in the United States [1]. This complex data set can be used to review, query, and analyze the US organ donation system over time and across various facets. The total database size is ~9 GB and contains records on hundreds of thousands of donors and over a million patients [2]. OTPN provides STAR data as tab-separated variable (TSV) files, SPSS [3] files, or SAS [4] files. SPSS and SAS are proprietary software and the TSV files are not suitable for non-trivial queries. The `transplant2mongo` Python scripts insert the TSV data into a MongoDB database. `transplant2mongo` was developed to allow researchers to use open-source software to explore, manage, and analyze the data without the need for proprietary software. There is no similar open-source software available for transferring STAR data into a database that does not require proprietary software for use.

Due to the sensitive nature of STAR data, the development of `transplant2mongo` was completed in a secure lab environment at The MITRE Corporation and the code repository does not include any STAR files. Special consideration was made to process the data and remove erroneous symbols and characters that are not compatible with MongoDB. The resulting MongoDB database has been used to perform an analysis of the OPTN dataset and the results of the analysis will be presented in a future publication.

Users are able to test the code using sample synthetic TSV data. In order for users to run the code on actual UNOS STAR data, they must update the Makefile to specify the file types they have and the data location using instructions detailed fully in the README file. Once they have updated the Makefile to their specifications, the `make all` command will populate the database with all referenced STAR files.

Implementation and architecture

This software was developed on a machine running CentOS with Python 3.6 and MongoDB 3.6 and was tested using all available OPTN STAR TSV files obtained from the OPTN in June 2014 and then in March 2015. The general file configuration of OPTN STAR files and directories is shown in **Figure 1**. (Users of `transplant2mongo` may

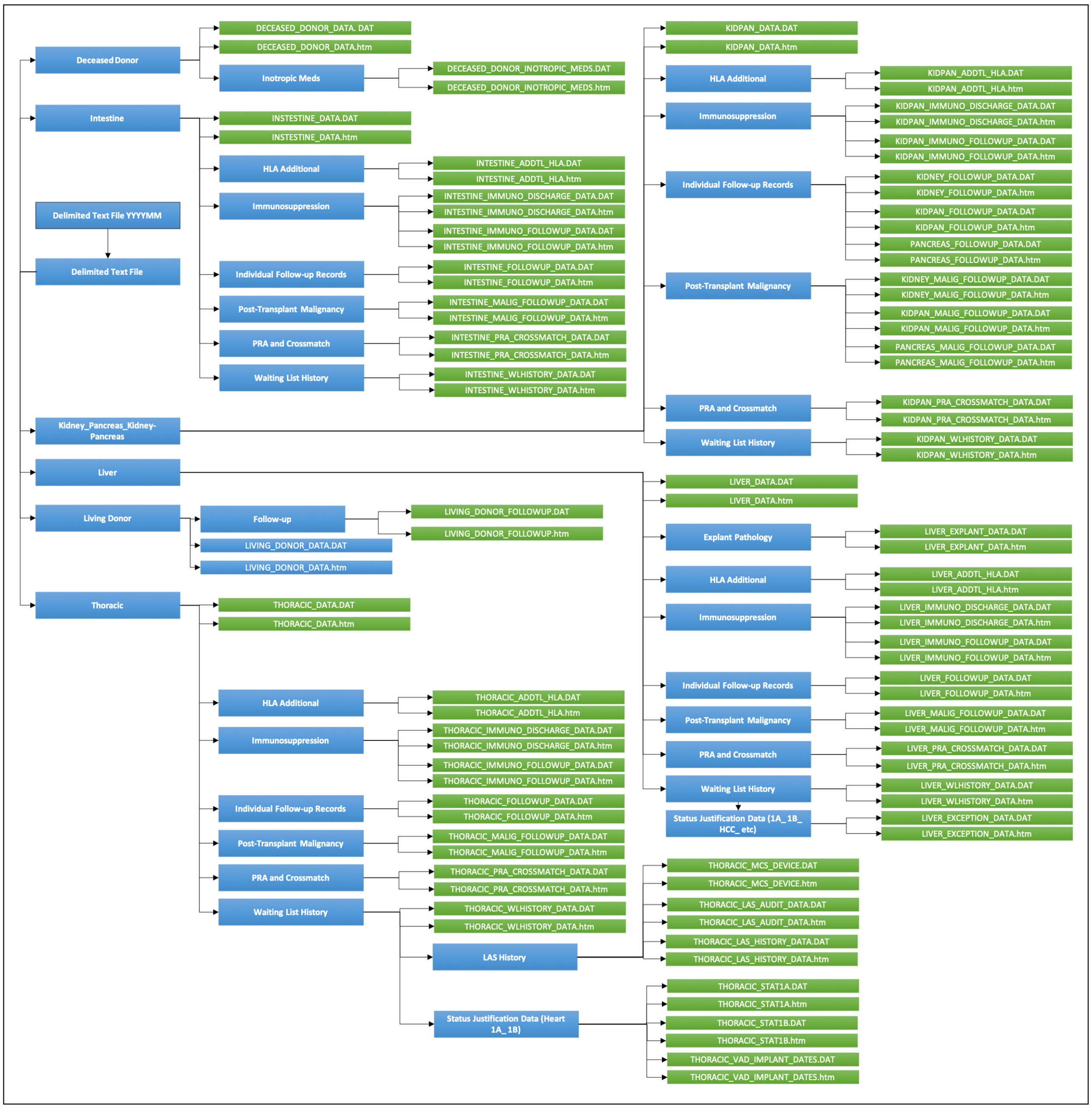
Deceased Donor
DECEASED_DONOR_DATA. DAT
DECEASED_DONOR_DATA.htm
Inotropic Meds
DECEASED_DONOR_INOTROPIC_MEDS.DAT
DECEASED_DONOR_INOTROPIC_MEDS.htm
Intestine
INSTESTINE_DATA.DAT
INSTESTINE_DATA.htm
HLA Additional
INTESTINE_ADDTL_HLA.DAT
INTESTINE_ADDTL_HLA.htm
Immunosuppression
INTESTINE_IMMUNO_DISCHARGE_DATA.DAT
INTESTINE_IMMUNO_DISCHARGE_DATA.htm
INTESTINE_IMMUNO_FOLLOWUP_DATA.DAT
INTESTINE_IMMUNO_FOLLOWUP_DATA.htm
Delimited Text File YYYYMM
Delimited Text File
Individual Follow-up Records
INTESTINE_FOLLOWUP_DATA.DAT
INTESTINE_FOLLOWUP_DATA.htm
Post-Transplant Malignancy
INTESTINE_MALIG_FOLLOWUP_DATA.DAT
INTESTINE_MALIG_FOLLOWUP_DATA.htm
PRA and Crossmatch
INTESTINE_PRA_CROSSMATCH_DATA.DAT
INTESTINE_PRA_CROSSMATCH_DATA.htm
Waiting List History
INTESTINE_WLHISTORY_DATA.DAT
INTESTINE_WLHISTORY_DATA.htm
Kidney_Pancreas_Kidney-Pancreas
Liver
Living Donor
Follow-up
LIVING_DONOR_FOLLOWUP.DAT
LIVING_DONOR_FOLLOWUP.htm
LIVING_DONOR_DATA.DAT
LIVING_DONOR_DATA.htm
Thoracic
THORACIC_DATA.DAT
THORACIC_DATA.htm
HLA Additional
THORACIC_ADDTL_HLA.DAT
THORACIC_ADDTL_HLA.htm
Immunosuppression
THORACIC_IMMUNO_DISCHARGE_DATA.DAT
THORACIC_IMMUNO_DISCHARGE_DATA.htm
THORACIC_IMMUNO_FOLLOWUP_DATA.DAT
THORACIC_IMMUNO_FOLLOWUP_DATA.htm
Individual Follow-up Records
THORACIC_FOLLOWUP_DATA.DAT
THORACIC_FOLLOWUP_DATA.htm
Post-Transplant Malignancy
THORACIC_MALIG_FOLLOWUP_DATA.DAT
THORACIC_MALIG_FOLLOWUP_DATA.htm
PRA and Crossmatch
THORACIC_PRA_CROSSMATCH_DATA.DAT
THORACIC_PRA_CROSSMATCH_DATA.htm
Waiting List History
THORACIC_WLHISTORY_DATA.DAT
THORACIC_WLHISTORY_DATA.htm
LAS History
Status Justification Data (Heart 1A_1B)
KIDPAN_DATA.DAT
KIDPAN_DATA.htm
HLA Additional
KIDPAN_ADDTL_HLA.DAT
KIDPAN_ADDTL_HLA.htm
Immunosuppression
KIDPAN_IMMUNO_DISCHARGE_DATA.DAT
KIDPAN_IMMUNO_DISCHARGE_DATA.htm
KIDPAN_IMMUNO_FOLLOWUP_DATA.DAT
KIDPAN_IMMUNO_FOLLOWUP_DATA.htm
Individual Follow-up Records
KIDNEY_FOLLOWUP_DATA.DAT
KIDNEY_FOLLOWUP_DATA.htm
KIDPAN_FOLLOWUP_DATA.DAT
KIDPAN_FOLLOWUP_DATA.htm
PANCREAS_FOLLOWUP_DATA.DAT
PANCREAS_FOLLOWUP_DATA.htm
Post-Transplant Malignancy
KIDNEY_MALIG_FOLLOWUP_DATA.DAT
KIDNEY_MALIG_FOLLOWUP_DATA.htm
KIDPAN_MALIG_FOLLOWUP_DATA.DAT
KIDPAN_MALIG_FOLLOWUP_DATA.htm
PANCREAS_MALIG_FOLLOWUP_DATA.DAT
PANCREAS_MALIG_FOLLOWUP_DATA.htm
PRA and Crossmatch
KIDPAN_PRA_CROSSMATCH_DATA.DAT
KIDPAN_PRA_CROSSMATCH_DATA.htm
Waiting List History
KIDPAN_WLHISTORY_DATA.DAT
KIDPAN_WLHISTORY_DATA.htm
LIVER_DATA.DAT
LIVER_DATA.htm
Explant Pathology
LIVER_EXPLANT_DATA.DAT
LIVER_EXPLANT_DATA.htm
HLA Additional
LIVER_ADDTL_HLA.DAT
LIVER_ADDTL_HLA.htm
Immunosuppression
LIVER_IMMUNO_DISCHARGE_DATA.DAT
LIVER_IMMUNO_DISCHARGE_DATA.htm
LIVER_IMMUNO_FOLLOWUP_DATA.DAT
LIVER_IMMUNO_FOLLOWUP_DATA.htm
Individual Follow-up Records
LIVER_FOLLOWUP_DATA.DAT
LIVER_FOLLOWUP_DATA.htm
Post-Transplant Malignancy
LIVER_MALIG_FOLLOWUP_DATA.DAT
LIVER_MALIG_FOLLOWUP_DATA.htm
PRA and Crossmatch
LIVER_PRA_CROSSMATCH_DATA.DAT
LIVER_PRA_CROSSMATCH_DATA.htm
Waiting List History
LIVER_WLHISTORY_DATA.DAT
LIVER_WLHISTORY_DATA.htm
Status Justification Data (1A_1B_HCC_etc)
LIVER_EXCEPTION_DATA.DAT
LIVER_EXCEPTION_DATA.htm
THORACIC_MCS_DEVICE.DAT
THORACIC_MCS_DEVICE.htm
THORACIC_LAS_AUDIT_DATA.DAT
THORACIC_LAS_AUDIT_DATA.htm
THORACIC_LAS_HISTORY_DATA.DAT
THORACIC_LAS_HISTORY_DATA.htm
THORACIC_STAT1A.DAT
THORACIC_STAT1A.htm
THORACIC_STAT1B.DAT
THORACIC_STAT1B.htm
THORACIC_VAD_IMPLANT_DATES.DAT
THORACIC_VAD_IMPLANT_DATES.htm

Figure 1: Structure of the OPTN STAR files. The top-level directory is `Delimited Text Files` and the subdirectories are shown to the right.

not have the complete set of UNOS STAR files, and may only have certain subsets of this information. In this case, the Makefile can be edited so that only a subset of the STAR files are processed.)

As shown in **Figure 1**, the TSV files are all stored in subdirectories of the `Delimited Text Files` directory. This directory contains a sub-directory for each of the data types: `Deceased Donor`, `Intestine`, `Kidney_Pancreas_Kidney-Pancreas`, `Liver`, `Living Donor`, and `Thoracic`. The `Kidney_Pancreas_Kidney-Pancreas` sub-directory contains information on all patients and transplant recipients for kidney and pancreas, and combined kidney/pancreas transplants. The `Thoracic` sub-directory includes data on both heart and lung transplant patients. The `Deceased Donor` and `Living Donor` sub-directories contain details on all organ donors, while the other subdirectories provide data on all patients registered to the waiting list and those who received a transplant. Each of these major groups is represented as collections in the generated MongoDB database with the patient or donor information as documents and the sub-folders as sub-documents.

MongoDB was chosen as the database due its NoSQL format, allowing multiple patient fields to be combined into a single document with multiple sub-documents. Many of the STAR file fields have changed over the years, and many of the patient fields are missing information, making this data suitable for NoSQL storage. Alternatively, a SQL database system could be used. However, with such a structured database implementation, the number of columns would be very large and there would be many NULL fields in the database.

A Makefile is used to execute the scripts that build the database from the contents of the `Delimited Text Files` folder. The Makefile has targets for cleaning, processing, and inserting the data into a MongoDB database. Users need to specify their database server and port in the Makefile before execution. The default server and port is `localhost:27017` and the default database name is `organ_data`. Users also need to define the components to process, which corresponds to the STAR files that the researcher has obtained from the OPTN (corresponding to the subdirectories of `Deceased Donor` shown in **Figure 1**). These values are selected from the following list: `deceased, living, intestine, kidpan, liver,` and `thoracic`. Data are first copied from the original location and file structures are flattened to establish a simple structure for parsing.

Once the flattened data files have been generated, another Python script is run to generate JSON and add the documents into the MongoDB database. For the base files, corresponding to the main donor and organ data files, the `add_patients.py` script is run to generate major documents in the appropriate tables. For files that contain sub-document information such as follow-up visits or medications, the `supplemental_data.py` script adds this information as a sub-document to the main patient or donor document. This script uses a unique identifier, such as `DONOR_ID`, `TRR_ID_CODE`, or `WL_ID_CODE`, to match the supplemental data file entry with a unique record for the donor or organ type. Once a match is found, the supplemental data is added to the document in the database. The TSV files, having the extension `.DAT`, only contain data and do not include column names; columns are determined by the `.htm` files corresponding to each `.DAT` files.

Data are cleaned before insertion into the database. The cleaning process is handled in the Python scripts `add_patients.py` and `supplemental_data.py`, which use the `clean_string` function. These scripts remove extraneous values such as extra commas, quotations, and newlines. The scripts do not assume a particular data type for any column. The database contains many different columns and the typing is automatically determined by Python and MongoDB. All data are by default imported as a string. The Python scripts check if the data is a date or an integer, and if not, the data are kept as strings.

Following the execution of each of the import scripts, follow-up scripts are run to add age bins to all age fields of the data to allow fast and simple aggregation queries.

Quality control

The scripts have been tested on two distinct UNOS STAR file data sets from 2014 and 2015. Each script prints out the number of lines in the file being processed and then the number of patient documents successfully imported into the database. With the two data sets, the line count results match the imported entry counts in all scenarios with the exception of a particular data file where a newline was hidden with quotation marks and the reported file length is one more than the imported record count. The data were manually confirmed in this situation and additional manual spot checks were performed.

The GitHub repository includes sample data that can be used for testing the installation. These data contain no real patient information and were generated by the developers of the software. The Makefile is set up by default to run on this sample data set with the `localhost` MongoDB database by following the directions provided in the `README`. A `test` target in the Makefile can be used to verify that the sample data was imported correctly and a query can be performed.

A MongoDB viewer, such as Robo 3T, is recommended for browsing and spot checking the data after the sample data or STAR data has been inserted into the database. A Python script is included, `query.py`, that performs database queries and prints the output to a comma-separated values (CSV) file. A Jupyter Notebook, `query-examples.ipynb`, is also included that can be used as a starting point for analysis and demonstrates querying the database, performing statistical analysis, and graphics.

(2) Availability

Operating system

All operating systems should be compatible.

Programming languages

Python 2.7 or 3.6+ (Tested with Python 2.7, 3.6, and 3.7)

Additional system requirements

None.

Dependencies

MongoDB 2.6+ (Tested with 2.6 and 3.4)

Python packages: `pymongo, pandas, tqdm, and seaborn`

Users need to obtain UNOS STAR files directly from the OPTN. This data contains private health data and can only be obtained directly from the OPTN.

List of contributors

Christine Harvey was the lead developer on this project and R.S. Weigel provided contributions.

Software location

Archive

Name: Transplant2Mongo Version 1.0
Persistent identifier: 10.5281/zenodo.2528840
Licence: GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007
Publisher: Christine Harvey
Version published: 1.0
Date published: 29/12/2018

Code repository

Name: GitHub
Persistent identifier: https://github.com/ceharvs/transplant2mongo
Licence: GNU GENERAL PUBLIC LICENSE Version 3, 29 June 2007
Date published: 15/02/17

Language

English

(3) Reuse potential

This tool is available for re-use by anyone working with UNOS STAR files seeking to do research on the organ transplant system. This tool generates a database that can be used with open-source software.

The software has a high potential for extension. This database creation scripts are written in Python 3 and they could be modified to allow the insertion of the data into alternative databases such as SQL. A GUI could be developed that allows visualization and query of the contents of the database that is customized for the type of analysis done by organ transplant researchers.

There is no official support for this software, but those with questions or interested in collaboration can contact Christine Harvey at ceharvey@mitre.org or via GitHub.

Acknowledgements

The data discussed in this work have been supplied by the United Network for Organ Sharing as the contractor for the Organ Procurement and Transplantation Network. The interpretation and reporting of these data are the responsibility of the authors and in no way should be seen as an official policy of or interpretation by the OPTN or the U.S. Government.

The authors would also like the thank the Early Career Research Program and the Innovation Area Lead, Rob Case for supporting this research.

Competing Interests

The authors have no competing interests to declare.

References

1. **Organ Procurement and Transplantation Network** 2015 OPTN Website. Available at: http://optn.transplant.hrsa.gov/ [Accessed May 8, 2015].
2. **Organ Procurement and Transplantation Network** 2017 Standard Transplant Analysis and Research (STAR) Dataset Files. Available at: https://optn.transplant.hrsa.gov/data/request-data/ [Accessed September 27, 2017].
3. **SAS** 2018 About SAS. Available at: https://www.sas.com/en_us/company-information.html [Accessed December 5, 2018].
4. **SPSS** 2018 IBM SPSS software. Available at: https://www.ibm.com/analytics/spss-statistics-software [Accessed December 5, 2018].

How to cite this article: Harvey, C and Weigel, R S 2019 Transplant2Mongo: Python Scripts that Insert Organ Procurement and Transplantation Network (OPTN) Data in MongoDB. *Journal of Open Research Software*, 7: 5. DOI: https://doi.org/10.5334/jors.229

Submitted: 04 May 2018 **Accepted:** 13 February 2019 **Published:** 14 March 2019