WHITE PAPER

SDRF (Scientific Data Repository Framework)
Improving Data Sets' Machine Readabability and Interoperability with Published Research

A New Protocol for Sharing Scientific Data

Linguistic Technology Systems (LTS) is developing SDRF to encompass data models, publishing guidelines, and code libraries for deploying open-access research data sets associated with scientific publications. Nowadays there are many general-purpose and domain-specific portals hosting scientific data; there are also several available formats for describing and encoding scientific data, such as Research Objects, **schema.org/Dataset**, Digital Curation Center (DCC), SCIDATA, BIOCODER, and MIBBI (Minimum Information for Biological and Biomedical Investigations). The purpose of SDRF is to merge these different data-set formats into a unified, overarching standard which can be adapted to different publishing houses and pipelines.

Shaping Protocols to Conform to Current Specifications in Publishing

In order to conform to current specifications — such as FAIRsharing (Findable, Accessible, Interoperable, Reusable) or the Bill and Melinda Gates Foundation guidelines for authors (https://gatesopenresearch.org/for-authors/data-guidelines) — proper protocols must be implemented at several stages of the publishing process, in particular (1) publications should provide clear descriptions of accompanying data sets and where that data is hosted; (2) publication repositories should make data-set links and metadata clearly visible on web pages where documents are read or previewed; (3) data sets themselves need to include metadata and supporting files which help researchers properly access, visualize, and reuse the data; and (4) data sets need to be connected with software which has the correct features to load and display the relevant raw data files. SDRF will include technology applicable to each of these four facets of the publishing and data-sharing pipeline in order to conform to current standards.

It is important to emphasize that data sets are only truly valuable if they are machine-readable and seamlessly integrated into domain-specific software ecosystems. Scientists who examine and reuse published data sets are generally researchers doing technical work in a field closely related to that of the original authors; in many cases there are specialized software applications, computational methods, algorithms, and research protocols which are endemic to the relevant subject areas. When sharing research data, accordingly, publishers/authors should make it as easy as possible for scientists to examine the data within the digital ecosystem that they utilize for their own research. This often implies that document viewers — e.g., PDF viewers and/or HTML pages on publisher portals — should be ideally interconnected with scientific applications so that scientists, when reading books/articles, can seamlessly launch domain-specific software and visualize/examine associated data sets. Unfortunately, most scientific software does not incorporate code libraries to parse metadata (summarizing file types, download instructions, etc.) describing open-access data sets. This can be addressed by providing plugins or extensions that add data-set-accession capabilities to existing scientific software, so that publisher repositories and data-hosting repositories can be made truly interoperable with scientific applications.

To demonstrate how such plugins can work, as well as other facets of the data-publication process facilitated via SDRF, this paper will review two case-studies addressing research in the academic

literature, each involving papers that have been linked with multiple data sets (data which was either reused or newly created during the course of the research described).

First Case Study: "Parkinson's Disease Diagnosis: The Effect of Autoencoders on Extracting Features from Vocal Characteristics," by Ashena Gorgan Mohammadi, Pouya Mehralian, Amir Naseri, and Hedieh Sajedi (International Journal of Speech Technology, pending review)

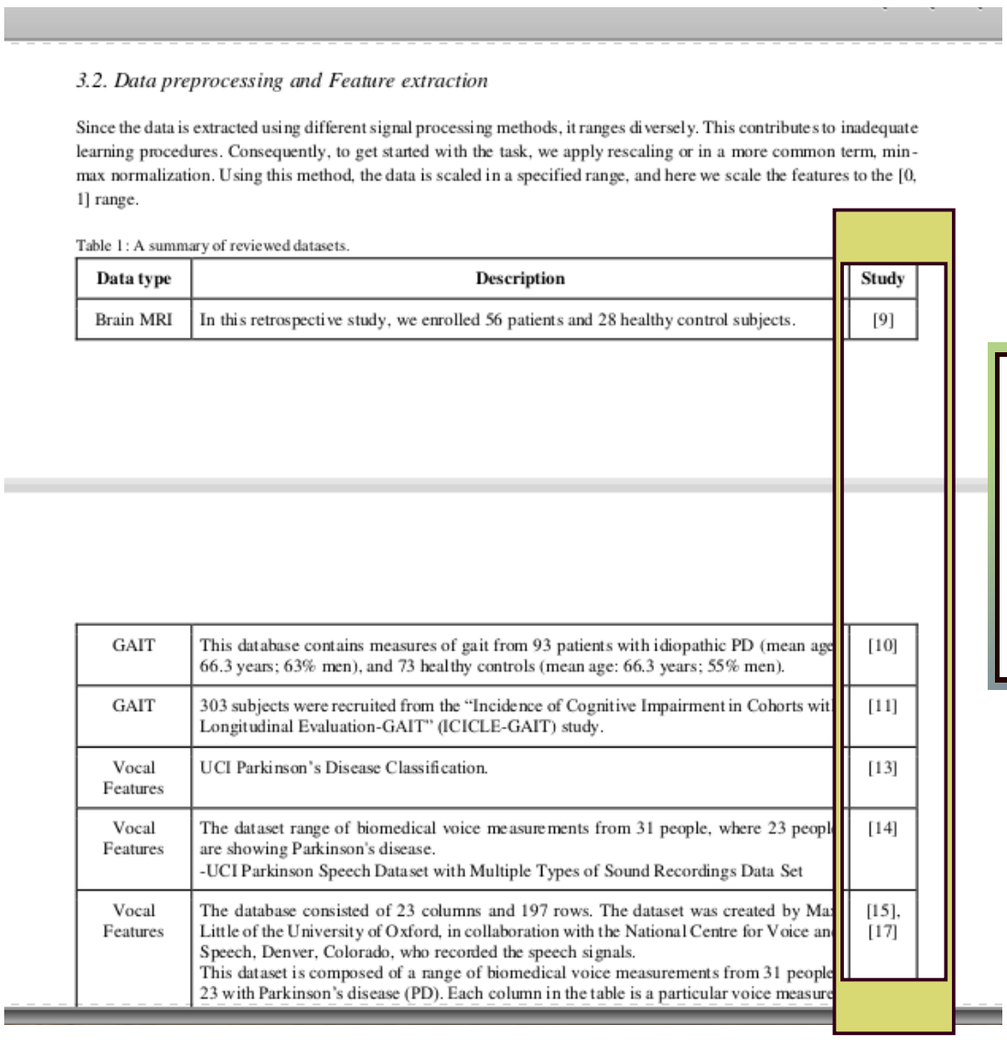
This case study demonstrates a scenario where an article reuses multiple pre-existing data sets. The article examines Parkinson's Disease symptomology from multiple perspectives, including gait (loss of motor function), speech impairment, and bioimaging (MRIs). The authors apply Machine Learning to data sets focused on these three different diagnostic areas, in an effort to advance research to refine Parkinson's predictors and diagnoses. The overall information can be summarized as follows:

1. **Gait Data**: this data was primarily drawn from a PhysioNet data set (https://physionet.org/content/gaitpdb/1.0.0/) obtained via sensors attached to subject's feet as they walked (the study includes both Parkinson's patients and healthy controls). This sensor data is provided as a collection of text files, each file corresponding to one patient (or control subject), with each line in a file representing a single time snapshot. The lines are divided into space-separated columns, each representing force exerted on a single sensor, plus two additional columns calculating total force on the left-foot and right-foot sensors respectively. This data set also includes demographic and clinical information for each patient in a spreadsheet format. The authors also use an additional source of gait data derived from a more recent (2019) study whose data is available only upon request (see https://www.nature.com/articles/s41598-019-53656-7#MOESM1).
2. **Speech Data**: this data was primarily drawn from a data set hosted by the University of California Irvine Machine Learning Repository (https://archive.ics.uci.edu/ml/datasets/Parkinsons). The central information is a CSV file, where each line represents a single voice recording from one of 31 subjects (consisting of 23 Parkinson's patients and 8 healthy controls). Each subject made multiple recordings. The individual lines in the CSV file present a quantitative model of subjects' speech via a collection of acoustic features/attributes. A similar "telemonitoring" data set from the same archive was used to study how the analysis of Parkinson's-related speech data may be applicable to samples obtained via devices such as smartphones.
3. **Radiological Data**: this data is not immediately available for reuse, but requires special (non-commercial) authorization from the Parkinson's Progressive Markers Initiative (https://www.ppmi-info.org/) or making a request of corresponding authors of referenced papers introducing the relevant data (https://www.frontiersin.org/articles/10.3389/fnins.2019.00874/full#h6).[1] Although this data is derived from bioimaging (so that the underlying raw data files are radiological) the authors of the IJST paper under review utilize the data in a more structured form, building off of image-feature extraction already performed when the data sets were first published by the prior authors. However, the republished unified data set itself (to appear in conjunction with the IJST submission) might include code to allow researchers to reproduce the original analytic workflow if desired: for instance, among other things, we can enable the implementations published via https://biomedia.doc.ic.ac.uk/software/malp-em/ to be embedded as a CaPTk module (see in particular https://cbica.github.io/CaPTk/tr_integration.html#tr_cppIntegration).
4. **Python Code Repository**: the authors also provide Python code, hosted on GitHub, which they used to analyze these various data sources.

Unifying the Data Sets into a Single Package

In this artice pending review, the authors summarize the data sets in a table within the main text (see Figure 1 here) and, in their bibliography, they cite these data sets either directly or by referencing

[1] When the unified data set is published, we will request permission to include a copy of the original data set to spare future readers from having to request this data on their own.



3.2. Data preprocessing and Feature extraction

Since the data is extracted using different signal processing methods, it ranges diversely. This contributes to inadequate learning procedures. Consequently, to get started with the task, we apply rescaling or in a more common term, min-max normalization. Using this method, the data is scaled in a specified range, and here we scale the features to the [0, 1] range.

Table 1: A summary of reviewed datasets.

Data type	Description	Study
Brain MRI	In this retrospective study, we enrolled 56 patients and 28 healthy control subjects.	[9]
GAIT	This database contains measures of gait from 93 patients with idiopathic PD (mean age 66.3 years; 63% men), and 73 healthy controls (mean age: 66.3 years; 55% men).	[10]
GAIT	303 subjects were recruited from the "Incidence of Cognitive Impairment in Cohorts wit Longitudinal Evaluation-GAIT" (ICICLE-GAIT) study.	[11]
Vocal Features	UCI Parkinson's Disease Classification.	[13]
Vocal Features	The dataset range of biomedical voice measurements from 31 people, where 23 peopl are showing Parkinson's disease. -UCI Parkinson Speech Dataset with Multiple Types of Sound Recordings Data Set	[14]
Vocal Features	The database consisted of 23 columns and 197 rows. The dataset was created by Ma Little of the University of Oxford, in collaboration with the National Centre for Voice an Speech, Denver, Colorado, who recorded the speech signals. This dataset is composed of a range of biomedical voice measurements from 31 people 23 with Parkinson's disease (PD). Each column in the table is a particular voice measure	[15], [17]

The authors provide citations for data sets used in their analyses ...

Figure 1: Table Listing Analyzed Data Sets in the Parkinson's Article

[8] Sakar, C. Okan, Gorkem Serbes, Aysegul Gunduz, Hunkar C. Tunc, Hatice Nizam, Betul Erdogdu Sakar, Melih Tütüncü, Tarkan Aydin, M. Erdem Isenkul, and Hulya Apaydin. "A Comparative Analysis of Speech Signal Processing Algorithms for Parkinson's Disease Classification and the Use of the Tunable Q-factor Wavelet Transform." *Applied Soft Computing* 74 (2019): 255-63. doi:10.1016/j.asoc.2018.10.022.

[9] Salvatore, C., A. Cerasa, I. Castiglioni, F. Gallivanone, A. Augimeri, M. Lopez, G. Arabia, M. Morelli, M.c. Gilardi, [illegible] Machine Learning on Brain MRI Data for Differential Diagnosis of Parkinson's Disease and Progressive Supranuclear Palsy." *Journal of Neuroscience Methods* 222 (2014): 230-37. doi:10.1016/j.jneumeth.2013.11.016.

[10] "Gait in Parkinson's Disease." Gait in Parkinson's Disease V1.0.0. February 25, 2008. https://physionet.org/content/gaitpdb/1.0.0/

[11] Rehman, Rana Zia Ur, Silvia Del Din, Yu Guan, Alison J. Yarnall, Jian Qing Shi, and Lynn Rochester. "Selecting Clinically Relevant Gait Characteristics for Classification of Early Parkinson's Disease: A Comprehensive Machine Learning Approach." Scientific Reports 9, no. 1 (2019). doi:10.1038/s41598-019-53656-7.

[12] Goetz, Christopher G. "The History of Parkinson's Disease: Early Clinical Descriptions and Neurological Therapies." *Cold Spring Harbor Perspectives in Medicine*, Cold Spring Harbor Laboratory Press, Sept. 2011.

[13] UCI Machine Learning Repository: Parkinson's Disease Classification Data Set. https://archive.ics.uci.edu/ml/datasets/Parkinson's Disease Classification.

[14] Sriram, Tarigoppula & Rao, M. & Narayana, G & Vital, T. & Dowluru, Kala[illegible] VGK. (2013). [illegible] Parkinson Disease Prediction Using Machine Learning Algorithms. IJETT. 3. 212-215.

[15] R. Das, "A Comparison of multiple classification methods for diagnosis of Parkinson disease." Expert Systems with Applications, vol. 37, no. 2, pp. 1568-1572, 2010.

[16] Erdogdu Sakar, Betul et al. "Analyzing the effectiveness of vocal features in early telediagnosis of Parkinson's disease." *PloS one* vol. 12,8 e0182428. 9 Aug. 2017, doi:10.1371/journal.pone.0182428

[17] M. Peker, B. Şen, D. Delen, Computer-aided diagnosis of Parkinson's disease using complex-valued neural networks and mRMR feature selection algorithm, J. Healthcare Eng. 6 (3) (2015) 281–302

[18] Ahlrichs, Claas, and Michael Lawo. "Parkinson's Disease Motor Symptoms in Machine Learning: A Review." Health Informatics - An International Journal 2, no. 4 (2013): 1-18. doi:10.5121/hiij.2013.2401.

[19] Khoury, Nicolas, Ferhat Attal, Yacine Amirat, Abdelghani Chibani and Samer Mohammed. "CDTW-based classification for Parkinson's Disease diagnosis." ESANN (2018).

[20] Brooks, David J. "Neuroimaging in Parkinson's Disease." NeuroRX 1, no. 2 (2004): 243-54. doi:10.1602/neurorx.1.2.243.

[21] Mohammad, Roohi, and Fatima Mubarak. "Neuroimaging in Parkinson Disease." Parkinson's Disease and Beyond - A Neurocognitive Approach, 2019. doi:10.5772/intechopen.82308.

[22] A. Kazeminejad, S. Golbabaei and H. Soltanian-Zadeh, "Graph theoretical metrics and machine learning for diagnosis of Parkinson's disease using rs-fMRI," 2017 Artificial Intelligence and Signal Processing Conference (AISP), Shiraz, 2017, pp. 134-139.

[23] Shiiba T, Arimura Y, Nagano M, Takahashi T, Takaki A. "Improvement of classification performance of Parkinson's disease using shape features for machine learning on dopamine transporter single photon emission computed tomography." PLoS ONE 15(1): e0228289, (2020). doi: 10.1371/journal.pone.0228289.

[24] Xu, Jiahang, Jiao, Huang, Yechong, Luo, Xu, Qian, Li, Ling, Liu, Zuo, Wu, Ping, and Xiahai. "A Fully Automatic Framework for Parkinson's Disease Diagnosis by Multi-Modality Images." Frontiers. August 05, 2019.

[25] Ting, Jiang, Lin, Wei, Wu, Ping, Zhou, Yongjin, Zuo, Wang, Jian, Yan, Zhuangzhi, Shi, Kuangyu,

Some data sets are directly available through the bibliography; others have to be located by reading the cited articles.

Figure 2: Bibliography (With Data Set Hyperrefs) in the Parkinson's Article

publications where the relevant data sets are described (see Figure 2 here). Doing so properly credits authorship to the researchers who curated the data sets, and it gives readers a means to locate the raw data. However, accessing and working with the raw data is inconvenient from a reader's point of view without most or all of the data sets being repackaged into a *single* archive that could be hosted and downloaded as one unit. Obtaining raw data from the resources identified in the bibliography requires several steps — for instance, the PhysioNet sensor data can be downloaded as a zipped folder, while the demographic data attached to it has to be downloaded separately. Furthermore, some of the information obtained from MRI and speech analysis (reported in papers cited as data sources for the IJST submission) is provided as supplemental materials within the secondary papers; this requires readers to browse one at a time through each of the relevant articles so as to find downloadable links (see Figure 4). In short, piecing all the source data together puts the onus on readers to manually inspect multiple web resources and to manually interconnect files once they are downloaded.

Another complicating factor is that certain information present in the data sets is implicit within how the data sets are organized, requiring extra effort to extract this information in a machine-readable manner. For instance, the PhysioNet sensor data uses a file-naming convention which encodes several pieces of information in the file names, such as whether the file presents a Parkinson's patient or a control subject (see Figure 3). Though by examining file names it would be possible to construct a table with additional information providing context for the file contents, such information is not directly included within the PhysioNet data set; it needs to be extracted by computer code.

Constructing Machine-Readable Supplemental Archives

This collection of data sets serves as an example of how technologies such as SDRF can fill the gap between publication/data repositories and scientific computing, making scientific data more "FAIR"

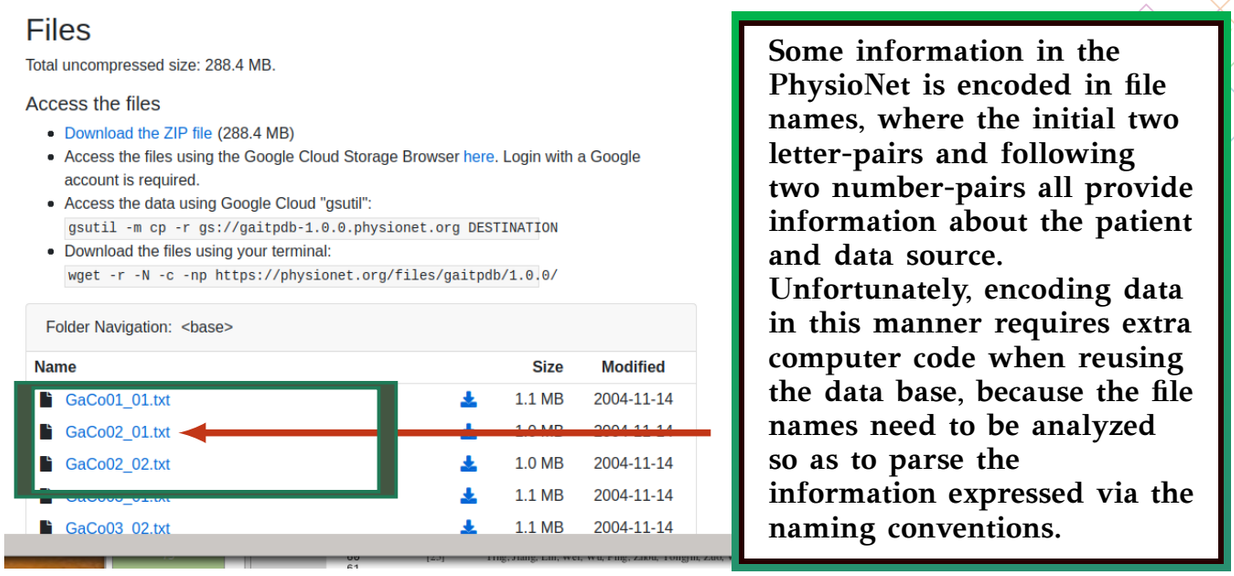


Figure 3: Extracting Information Encoded in File Names

(Findable, Accessible, Interoperable, Reusable). Upon publication of this submission in IJST (pending peer review) the disparate open-access data from the article's secondary sources would be provided as a single SDRF archive. This archive would provide *machine-readable* access to information spread across multiple sources, translated into a common file format. In general, SDRF encourages and implements features to help data sets conform to FAIR and related standards, such as (1) bundling multiple data sets into a single archive; (2) migrating data to general-purpose representations wherever possible — formats such as XML, HDF5, ARFF, or DICOM; (3) providing meta-data in multiple formats (DCC, **schema.org/Dataset**, MIBBI, BioCoder, etc.) to be compatible with different organizations' platforms; (4) identifying one or more "preferred applications" for examining/reusing the published data; (5) explicitly representing information encoded via file-names; (6) bundling raw data, meta-data, and (where possible) machine-readable article text into a single resource, which SDRF calls a "Supplemental Archive;" and (7) annnotating the data sets to support microcitations that granularly link the publication to its associated Supplemental Archive.

Once a Supplemental Archive has been downloaded, an important question for any SDRF archive is how researchers will productively access the data. Unlike the Flow Cytometry use-case discussed below, the Parkinson's archive spans several scientific disciplines; as such, there is no obvious application which could be preferred by default for examining the data files. As a fallback option, SDRF is designed to present data sets via Qt Creator, a C++ Integrated Development Environment associated with the Qt application-development framework. SDRF includes code libraries to represent research meta-data as C++ objects; these libraries can be opened as Qt projects. These may be supplemented with separate libraries extracting and managing information specific to individual data sets. In particular, the Supplemental Archive for the Parkinson's article under peer review would provide C++ classes encapsulating spreadsheet-like data (whether originally in **.xls**, CSV, or space-delimited formats) republished by the Journal in the unified data set.

An additional concern for SDRF archives is how to properly annotate publications and data sets side-by-side. In the Parkinson's article, individual C++ classes encapsulating tabular data serve as convenient microcitation targets: annotations within the relevant C++ code represent anchors through which the data set may be referenced (on a more precise scale than merely citing the Supplemental Archive as a whole). In some places, individual class attributes can also be linked to lines in the authors' Python source code. On the text side, certain paragraphs within the Parkinson's article can be linked to the corresponding C++ code annotations. This illustrates SDRF's recommended annotation/microcitation system, where segments in publication texts (identified for instance via LaTeX **phantomsection** commands or JATS **statement** tags) are linked to annotations or comments in code and/or raw data files in the Supplemental Archive.

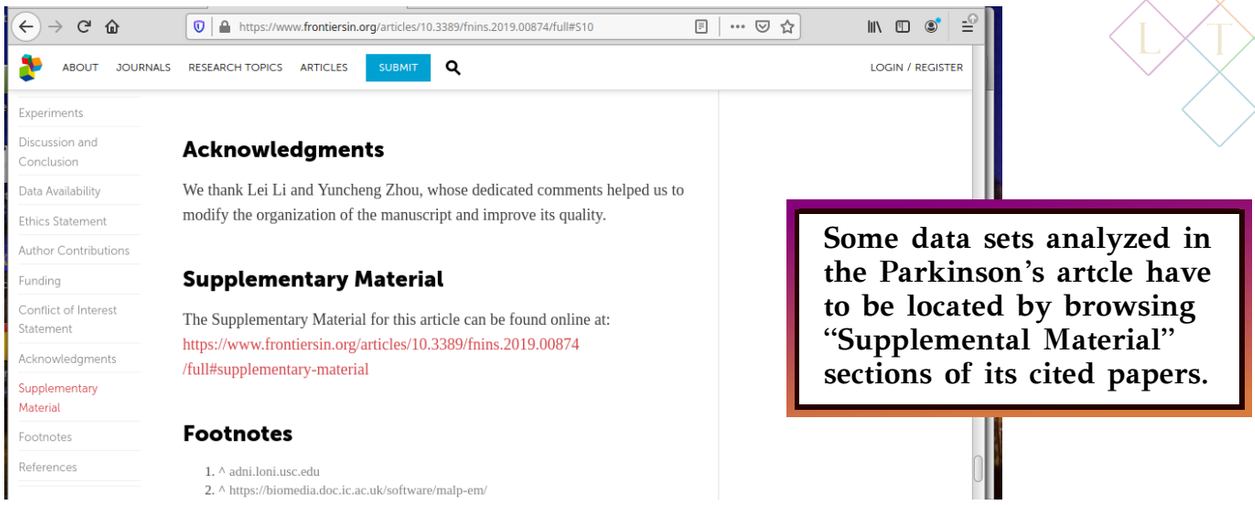


Figure 4: Indirectly Locating Data Sets from Cited Papers

Second Case Study: "Marked T cell activation, senescence, exhaustion and skewing towards TH17 in patients with COVID-19 pneumonia" from nature.com, 2020

This article presents a use-case with some noteworthy contrasts to the Parkinson's publication described in the previous section. This Covid-19 paper (https://www.nature.com/articles/s41467-020-17292-4) was published along with two data sets comprising Flow Cytometry Standard (**FCS**) files hosted via the Flow Repository (http://flowrepository.org/id/FR-FCM-Z2N4 and http://flowrepository.org/id/FR-FCM-Z2N5). Links to the data sets (via **flowrepository.org** pages) are explictly provided in the publication's "Data Availability" section. However, researchers still need to perform several steps to manually download the full set of relevant **FCS** and meta-data files.

One feature of this second use-case is that the technical information in the data sets belong to a single scientific area (Flow Cytometry) and are mostly encoded in a single format (**FCS**). As such, it is straightforward to identify the kind of software which researchers need to use to visualize the raw data — basically, any application that can parse **FCS** files. There are a variety of commercial as well as open-source Flow Cytometry (**FCM**) applications which can be used to access **FCS** data. Once readers have downloaded the Flow Repository archives, they may individually load the **.fcs** files to study data which, in the original article, is summarized via figure illustrations.

This workflow nonetheless requires researchers to perform several manual steps before being able to use the published data. The core problem is that existing Flow Cytometry software does not intrinsically have capabilities to read **PDF** files, locate **FCS** data sets, and interoperate with hosting platforms such as Flow Repository. Employing this use-case as an example with which to illustrate proper alignment between document viewers, publication/data repositories, and scientific software, We are developing a new Flow Cytometry application which *can* interoperate with **PDF** viewers and **SDRF** archives. This application is designed so that, when authors are reading a **PDF** file associated with an **FCS** data set, the **PDF** viewer can automatically launch and signal to the **FCM** application when a reader wishes to download and visualize **FCS** files. In short, the **FCM** application — having received data from a **PDF** viewer which implements an **SDRF** inter-application protocol — will automatically execute download and extraction steps that scientists otherwise would have to perform manually.

Note also that, although most of the relevant data for this Covid-19 article is in **FCS** form, there is, in addition, supplemental clinical information provided in other formats (...). For cases such as these, the LTS **FCM** software includes code libraries allowing researchers to parse non-**FCS** data in standard formats such as **XML**, **HDF5**, or **ARFF**.

This use-case illustrates a general principle: that research data is most convenient for scientists when it is deployed within an infrastructure where portals, document viewers, and scientific applications can seamlessly interoperate. Wherever possible, when researchers are reading published books or articles, they should be able to *automatically* launch the proper scientific application, download data

sets, and examine raw data files in the preferred software with only one or two clicks. These steps would be performed automatically as much as is feasible, instead of readers having to waste time with manually finding, downloading, merging/extracting, and then opening data files.

Conclusion

The two case-studies considered here are similar in that each involve articles which are linked to multiple data sets. For maximum convenience, it is optimal for researchers to be able to access this data without performing manual download and merging/extracting actions. There are also some differences between the two case-studies: in particular, the Parkinson's data spans multiple disciplines, whereas the Covid-19 data is more rigorously grounded in Flow Cytometry. As such, the operational requirements for the Covid-19 data, from a reader's point of view, are more clearly delineated: effective integration between the publication and its accompanying data sets is defined by launching Flow Cytometry software while a researcher is reading the publication, allowing the researcher to view the data via software similar to that used to generate/analyze the data while the reported research was being conducted. In the case of the Parkinson's data, in contrast, there is no single application which would seamlessly display the specrum of information considered in that article; as mentioned above, in such cases, SDRF would default to using QT Creator as a fallback for loading Supplemental Archives where no other software is available.

Regardless of whether one is using QT Creator or a domain-specific application, it is preferable that each SDRF archive be associated with one or more applications that researchers can use to view data (and extract information) from the archive. Moreover, these applications would ideally be linked to document viewers and also to publisher's portals, so that readers can automatically launch preferred applications and view Supplemental Archives while reading concomitant publications. In order to achieve this, scientific applications need to be augmented with plugins to parse SDRF data. To address this need, we are developing an inter-application messaging protocol so that disparate applications with SDRF plugins may interoperate. In particular, this protocol would entail PDF viewers being able to interoperate with scientific applications so that publications' data sets may be automatically downloaded and visualized via the preferred software.

Our prototype example for an application utilizing such plugins, as mentioned above, is software for Flow Cytometry. We are also working on a prototypic QT Creator plugin so that QT Creator (as the "default" SDRF software) can participate in SDRF networks using the same protocol. We will then expand the scope of this protocol via plugins for software in other domains, such as image-analysis, molecular visualization, radiology, 3D graphics, and so forth.

Future Projections: Operationalizing SDRF

This section will present a concrete outline of the steps necessary to package scientific data into an SDRF Supplemental Archive. Data Sets may be published with content and organization compatible with both SDRF and other formats, such as Research Object Bundles, so using SDRF does not preclude adopting other formats as well. For example, a data set which provides SDRF meta-data may also include the **meta-inf** files required by Research Objects. Indeed, it is recommended that authors aim for compatibility with multiple standards, not only SDRF. This section, however, will focus on the components of Supplemental Archives that are specific to SDRF.

This section will describe two different approaches to using SDRF. One approach employs SDRF to a limited extent, deferring to older technologies for such basic operations as encoding data or annotating publications. An alternative is to adopt SDRF more holistically, adopting experimental or under-development features in the SDRF libraries. When discussed in the following outline, these features will usually be characterized as "experimental" or "specific to SDRF code libraries."

Formulating Scientific Data Repository Models for Individual Publishing Environments

SDRF is intended to interoperate with mulitple publisher and data-hosting platforms, each of which have their own requirements and technology. Therefore, the precise contents and organization of a given SDRF Supplemental Archive will depend on specifications pertaining to the specific repository and hosting platform where the publication and data set, respectively, are to be deposited. To facilitate Supplemental Archive preparation for different environments, SDRF seeks to incorporate "Scientific Data Repository Models" (SDRMs) which are tailored to individual publishers. These SDRMs would encapsulate information targeted to the specific platforms where the relevant publication and data set are hosted. For example, research funded by the Bill and Melinda Gates Foundation must follow certain guidelines, which are documented whenever an article is submitted to the Gates Open Research portal. Consequently, the specific SDRF meta-data for such papers can be designed to notate how the associated research adheres to these organizational guidelines. In this type of situation, the layout and vocabulary for SDRF meta-data would be determined at least in part by the SDRM germane to the environment where the research is published.

The purpose of an SDRM is also to facilitate the implementation of technologies which interoperate with publishers' platforms — for instance, API clients which can request information from document/data portals (e.g., locating articles or data sets via keyword searches), or PDF viewers enhanced with capabilities to obtain data sets from hosting platforms (to automatically download data sets upon request from a user reading the corresponding publication). For these use-cases, individual SDRMs would include information and/or computer code about how to access publishers' APIs, how to download and extract a data set given a digital identifier, and so forth.

Because SDRF Supplemental Archives may differ somewhat according to the specific SDRM in effect, any description of these archives is provisional. To be precise, SDRF assumes a generic SDRM — dubbed "MOSAIC" — which can be extended or modified in consultation with individual publishers, as required. As such, the following outline will apply intrinsically to the MOSAIC SDRM; depending on the platform, the details for individual SDRF Supplemental Archives may be different than the generic model presented here.

Standard Components of SDRF Supplemental Archives

The components of an SDRF Supplemental Archive can be grouped into several facets, concerning meta-data, raw data, and machine-readable text, respectively.

1. **SDRF Meta-Data**

   Meta-Data Files SDRF uses a vocabulary for describing the contents of data sets which merges the data models of several existing formats, such as DCC and **schema.org/Dataset**. SDRF recommends encoding this data in TAGML ("Text-as-Graph Markup Language"), which is a very flexible, and computationally powerful representation format.[2] The SDRF-specific libraries include parsers for (an extended version of) TAGML.

   C++ Files SDRF also recommends that authors provide C++ code to initialize objects representing Supplemental Archive's meta-data. These objects may be directly constructed from the TAGML meta-data files, or authors may choose to customize the C++ logic as desired. For lab-based research, authors may employ BioCoder, which uses C++ code to notate research protocols and workflows (the SDRF libraries include a modified version of BioCoder). Moreover, for data sets whose "preferred application" for opening raw data files is implemented in C++ (as is the case for many, if not most, scientific-computing applications), the Supplemental Archive may include plugins or extensions to these applications. In general, using C++ objects specific to Supplemental Archive meta-data as a basis, authors (or programmers coding

[2] TAGML is "powerful" in the sense that, with suitable mappings between their disparate syntactic expressions, TAGML represents a superset of XML and other common data-representation languages, such as JSON.

on authors' behalf) may introduce additional C++ code demonstrating or enabling analysis, visualization, or application-integration for the Supplemental Archive's raw data.

SDRM-Specific Meta-Data Either via data files or computer code, information or capabilities specific to the platforms where papers and Supplemental Archives are hosted may be presented in conjunction with the applicable SDRMs. For instance, C++ code distributed with the archive may include procedures for accessing APIs of the hosting service where the archive is deposited.

Qt Project Files As explained earlier, SDRF defaults to using QT Creator as an application with which to examine data sets, if there is no obvious "preferred application" based on the research topic. Therefore, SDRF recommends that the C++ meta-data code be paired with QT project files and other assets needed to run the code in the QT Creator IDE (Integrated Development Environment), using **qmake** as the build system. In general, there should be a QT project specific to the SDRF meta-data; in some cases, there will also be separate QT projects for working with raw data files.

2. **Raw Data**: As mentioned above, SDRF recommends that raw data be encoded in general-purpose formats such as ARFF, HDF5, XML, or TAGML (as compared to formats such as **.xls**, which are associated with specific applications). An experimental "Hypergraph Exchange Format" (HGXF), being developed in conjunction with SDRF, may also be used.

   An exception to the guidelines for general-purpose formats is when data should be presented in optimized formats endemic to a certain scientific field, such as FCS files for Flow Cytometry, or DICOM files for bioimaging. In these cases, it is recommended to employ such formats for most of the raw data files but also to construct summaries of the data, which may facilitate properly loading and accessing the domain-specific files, represented in a more general-purpose format.

3. **Annotated Manuscripts** Where authors have permission to share full-text versions of their books/articles, SDRF recommends that a machine-readable representation of these publications, in formats such as TAGML or XML, be included in the Supplemental Archive (SDRF has an experimental "Hypergraph Text Encoding Protocol" which may be utilized as well). This machine-readable manuscript may then be annotated and cross-referenced with raw data and/or code files also included in the Supplemental Archive. An experimental SDRF LaTeX package provides an annotation framework which not only marks locations in document text, but also encodes PDF viewport coordinates for use by specialized PDF viewers which implement an SDRF protocol for integrating PDF viewers with scientific applications.

Checklist for Authors, Editors, and Publishers

To clarify the steps needed to deploy publications and data steps according to SDRF recommendations, this section will outline the steps that would be taken by individuals fulfilling different roles in the publishing pipeline.

Authors In most publishing environments, the process of depositing data sets for open-access hosting is entirely separate from that of submitting articles for peer-reviewed publication. Therefore, it is the responsibility of authors to ensure that these two resources — their manuscripts and Supplemental Archives — are properly interconnected. Some guidelines for ensuring that this process is carried out smoothly include: (1) cite the URL for the data set near the top of the document; (2) cite the URL for the publication in a **readme** or similarly prominent location in the data set; (3) if there are no access restrictions, include a PDF file showing the full article in a findable location in the data set; (4) use labels and **phantomsection** or similar tags/commands to mark locations in the manuscript which are conceptually linked to parts of the data set; and (5) request that future authors cite both the publication URL and the Supplemental Archive URL when

referencing the body of research.

Additional steps are also necessary when packaging files into the published archive. The exact details will depend on the SDRMs which apply to the environments where the publication and data set are hosted, and also on whether authors wish to comply with research standardizations other than those of SDRF. For example, archives which are published as a Research Object Bundle need to use a folder layout proscribed by that specification. To work flexibly with other standards, SDRF does not demand *a priori* that authors use specific conventions for file and folder names and hierarchies. In general, so as to pin the location of SDRF meta-data, SDRF recommends the following: C++ classes and **main.cpp** files specific to initializing SDRF meta-data should be marked accordingly with C++ comments; and files in formats such as TAGML, serializing SDRF meta-data, should be similarly identified. Once the meta-data source is locatable within the archive, all other information needed to fully parse the SDRF-specific data should be extracted from the initialized meta-data objects.

Editors Because (as just outlined) authors' submissions to data-hosting portals are usually operationally separate from their endeavors to publish books and articles, publication editors ordinarily cannot directly influence the authors' preparations for publishing their data sets. However, editors can make recommendations and double-check that authors' research (and other research which is cited within new articles) is properly referenced, ensuring that new (and relevant pre-existing) data sets are Findable. That is, editors can ensure that (1) URLs and digital identifiers for concomitant data sets are prominently notated in the publication text; (2) bibliographic references to other publications which have their own data sets include citations for and hyperref links to those data sets if possible; and (3) the authors provide a brief overview of their data set if doing so is necessary to help researchers access the raw data and to help readers understand how the raw data has informed the research or findings presented in the publication. Moreover, editors can review the data sets themselves and verify that they properly reference the accompanying publication.

It should be noted that the proper URLs and formats for referencing publications may change as manuscripts work their way through the publishing process. For example, a web link for accessing a peer-reviewed article may only be defined once the document is accepted for publication. Also, some institutions (e.g., Gates Open Research) publicly track a submission through different stages of peer review (articles may be accessible for reading even while they are still being reviewed or finalized). Consequently, when a data set links to its accociated publication, such information about the current state of the submission should be duly observed in the data set.

In short, the precise publication-related data within a Supplemental Archive may need to be modified one or more times as a document submission is processed, so editors should be prepared to notify authors when a change within the data set is necessary.

Publishers Assuming that authors and editors follow the steps just outlined, articles and Supplemental Archives may be published in any environment without directly altering publishers' workflows or technologies. However, their are steps which publishers may take to enhance the usefulness of SDRF resources, and to derive additional scientific and commercial value from hosting articles that utilize SDRF. These steps include:

- **Feature Data-Set References on Article Front Pages** The "front page" of an article is a URL which uniquely locates a given publication on a publisher's portal. Usually this front page includes an abstract, bibliographic citations, author names/affiliations, keywords, and other high-level info about the document; in some environments, the front-page also reproduces the full text of the article (usually in HTML format). Even with this information set forth, however, it can be difficult for readers to identify which publications are paired with open-access data sets and to locate those data sets when they are available. Publishers can rectify this situation by including links to data sets near the top of the front page.
- **Recognize Data-Set Microcitations in Manuscripts** Properly connecting publications and

data sets requires customized LaTeX commands or XML attributes, so as to mark and annotate relevant locations/passages within the document. To fully utilize SDRF, then — or, indeed, any rigorous data-sharing protocol — publishers need to expand their in-house media for internally representing manuscripts in the pipeline with a collection of additional tags, commands, and/or attributes. The details of these add-ons, depending on the technology used, may be codified within individual SDRMs.

- **Recognize Alternative Manuscript and PDF Formats** This applies to non-restricted publications where full text representations may be included within a Supplemental Archive. In this situation, the ideal PDF version of a document — as well as of the machine-readable encoding of the text — may be different from what is internally adopted by the publisher. For example, an article may appear as a chapter in a book, typeset according to the book's conventions; in the data set, however, that same article is shared as its own document, and might use different typesetting rules optimized for its specific content. Also, the machine-readable full-text representations presented in the data set should be optimized for Text and Data Mining (TDM), and may therefore differ from the internal encoding used by the publisher. For these scenarios, publishers should allow for alternate versions of both human-readable and machine-readable publication text being incorporated into the publishing process, where these alternatives end up becoming assets within Supplemental Archives. Such alternatives may not differ in terms of textual content (although a data-set-specific version of an article may include supplemental instructions for using the data set), but they can differ in the visual formatting of the text as well as how the text is encoded.
- **Register Cross-References Between Publications and Data Sets** Publishers' databases should be configured to clearly designate when a publication is linked to an associated data set. Moreover, some third-party biblioinformatics services, such as CrossRef, allow publishers to introduce additional attributes describing publications. Connections between publications and data sets may therefore be formally declared in these third-party contexts.
- **Incorporate Data-Sets Information into Publisher APIs** Most publishing portals use APIs to support application-level queries for finding or accessing publications. These APIs can be extended to systematically honor associations between publications and data sets, e.g.: (1) return the URL for a data set when given an article's Document Object Identifier; (2) return information about an article when queried by a software component designed to manage data sets; (3) search for publications whose data sets fit given criteria (file format, programming language, subject area, etc.); and (4) given a publication, return basic information about its associated data that may help a reader decide whether to download the data set (file format, total file size, preferred application, and so forth).

Benefits for Publishers

We believe adopting SDRF technology can derive several benefits for publishers (as well as instutions, such as universities or independent journals, which publish their own material). Bear in mind that the details are SDRF are adapted to different SDRMs; as such, the characterization of SDRF content presented here is provisional, and may take different forms in different publishing environments. Therefore, adopting SDRF does not require publishers to substantially (or at all) modify their existing software. In general, though, incorporating certain SDRF protocols and/or implementations — whether as software extensions or simply as recommended design patterns — can augment publishers' offerings in several ways:

1. **Increased Citations and Downloads** By explicitly linking publications and data sets, publishers improve the likelihood that researchers will find the publications. Data sets present an alternative route (alongside references in other sources, or keyword searches) for scholars to locate books and articles relevant to their research.

   It is important to stress that, despite standardizations such as DCC and Research Objects, the vast majority of published data sets are shared haphazardly, without proper organization or

meta-data. Therefore, those data-sets which *do* adopt rigorous curation standards are more likely to be found by software designed for contemporary data-publishing specifications. Well-curated resources also stand out from their peers on data hosting platforms. Some hosting environments explicitly introduce a "score" to measure how well each data set is organized and documented (e.g., the MIFLOWCYT, or "Minimumum Information about a Flow Cytometry Experiment", rating on **flowRepository.org**). Accordingly, data sets which score highly on existing or future metrics — assessing how systematically their data is organized — are more likely to attract researchers' attention, which in turn drives traffic to the publication sites where data sets' corresponding articles are indexed.

2. **Impact Factor** Articles which are paired with conscientiously curated, well-documented and reusable data sets are more likely to spur further research than publications whose raw data is not shared at all or is shared in ways which add extra effort for researchers seeking to re-use or re-examine the data. In general, scientists gravitate to using raw data which can be plugged most readily into the computing environments they use for their own research. For this reason, research work — expressed both in data sets and in publications discussing the research — is likely to be more influential when scientists deem the data convenient to use and expand upon.
3. **Enhancing the Publishing Ecosystem** As digital media becomes an increasingly important part of scientists' use and experience of published research, publishers are expected to provide ever more sophisticated technology for visualizing data and publications. Aside from just showing PDF or HTML views of articles, publishers are gradually introducing multi-media platforms which permit interactive data access, 3D data visualization, audio and video content, and similar features of enhanced "Reader Experience." Interfacing directly with scientific applications — e.g., via publisher-specific application plugins — is a natural corollary to this trend, adding yet more computational heft to publishers' platforms. But properly networking publications with specific software demands rigorously documented links between publications and data sets, such as provided via SDRF.

The Transparent Hypergraph Query Language

SDRF does not overtly proscribe any particular format for encoding raw data in SDRF archives; as such, SDRF cannot make definitive recommendations for how this data may be consumed. Nevertheless, SDRF draws on contemporary research into database engineering, particularly the body of literature concerning Hypergraph Database design and the use of hypergraphs as multi-faceted, general-purpose formats for representing information. Hypergraph Database engines are favored in part because hypergraph schema can incorporated many families of data models, so that Hypergraph Databases can store diverse, heterogeneous data, sourced from disparate origin-points. Nevertheless — despite some common features which make Hypergraph Database engines, collectively, good architectural choices for heterogeneous and decentralized data persistence — there are several distinct Hypergraph engines widely used in contemporary technology, each with slightly different data models and protocols. It is therefore a worthwhile project to define a multi-purpose Hypergraph data model which merges idiosyncratic details of these various platforms, offering a common framework for accessing Hypergraph data. Such a framework's value could be enhanced by applying it to information spaces which are not, necessarily, backed by a Hypergraph Database engine but which are amenable to an intermediate software layer that translates Hypergraph-oriented queries to a query language endemic to the actual database used. Although Hypergraph databases are gaining a greater foothold in some computing domains (biomedical, governmental, financial, etc.), a much larger proportion of database instances rely on more traditional data-storage technology. With proper software adapters, it is possible to construct "views" onto data spaces such that they can be operationally interacted with as if they employed hypergraph models internally.

These points apply not only to databases themselves but also to any systematic data compilation, including published data sets. Therefore, one use-case for SDRF is to package raw data in such a

way that it may be queried according to protocols established for Hypergraph Database engines (and then extended to other information architectures).

Such is the motivation behind the so-called "Transparent Hypergraph Query Language" (THQL) which LTS is designing as a paradigm for organizing data sets with the goal of formulating a hypergraph query model that may be extended to Hypergraph databases, as well as to information spaces that can be programmed to emulate such databases. THQL is characterized as "transparent" because this technology is grounded on queries expressed and evaluated directly in C++ code, so that query-processing logic may be openly observed in source code and through a debugger (this does not preclude a distinct THQL *language*, but such would be implemented by translating query expressions to C++ procedure calls; behind the scenes THQL is implemented as a C++ DSL).

In its design, THQL represents the different use-cases in which Hypergraph database technology is selected as the optimal strategy for data storage and information management or "knowledge engineering." In general, THQL recognizes four architectural paradigms which undergird the Hypergraph database model (that is, an idealized model abstracted from specific products):

Object Serialization One appeal of Hypergraph database engines is that they minimize the boilerplate code needed to serialize (and later retrieve) "objects," or typed data structures, in the sense of Object-Oriented Programming. These engines are not "object" databases where object-persistence happens automatically, but they are designed to manage objects as first-class integral values more fluently than other database architectures (relational, hierarchical, graph-oriented, etc.). This means that, when querying a hypergraph database, it should be easy to manipulate the query results as one single typed value which is an instance of an application-specific type implemented within the application which initiates the query, or as an iterator or "recursive factory" from which such typed values can be initialized. THQL accordingly incorporates logic to manage queries and query results according to such an object-factory design pattern.

Collections-as-Values Another beneficial feature of Hypergraph databases is that collections of similarly-typed values can be manipulated as single values, requiring less intermediate modeling than appears in SQL or RDF, for instance. THQL accordingly presents views on hypernodes which emulate collections data structures implemented in mainstream programming languages (stacks, queues, deques, etc.).

Graphs and Network Models Although "nodes" (or, more accurately, *hypernodes*) in a Hypergraph database have more internal structure than nodes in ordinary (non-"hyper") graphs, hypergraphs are also models of hypernodes connected in graph-like networks, so conventional graph queries and analyses (ignoring hypernodes' internal content and considering just connections between hypernodes) can be evaluated on hypergraphs no less than on ordinary graphs. Accordingly, THQL incorporates common features of conventional graph-query languages.

Hypernodes as Relational and Conceptual Structures This facet of THQL integrates analytic protocols derived from various existing Hypergraph technologies (as well as AI research that is operationalized and/or facilitated by these technologies), such as **Graken.ai**, HyperGraphDB, Conceptual Space Theory, Fuzzy Logic, Conceptual Role Semantics, and so forth. Each of these paradigms have distinct interpretations of the internal structure of hypernodes. To accommodate this variation, THQL allows hypernodes' internal structuration — the relationship between the overarching hypernode and its contents — to be notated according to several different systems.

LTS hopes to deploy a THQL prototype in conjunction with published SDRF data sets and also in the context of data-integration projects, including ones that will be examined in a forthcoming Elsevier volume concerning "Data-Integration and Conceptual-Space Models for Covid-19." Please contact LTS for more information.

For more information please contact:
Amy Neustein, Ph.D., Founder and CEO
Linguistic Technology Systems
amy.neustein@verizon.net • (917) 817-2184