THE DIGITAL CLASSICIST 2013

BULLETIN OF THE INSTITUTE OF CLASSICAL STUDIES SUPPLEMENT 122

DIRECTOR & GENERAL EDITOR: JOHN NORTH

DIRECTOR OF PUBLICATIONS: RICHARD SIMPSON

THE DIGITAL CLASSICIST 2013

EDITED BY STUART DUNN AND SIMON MAHONY

INSTITUTE OF CLASSICAL STUDIES
SCHOOL OF ADVANCED STUDY
UNIVERSITY OF LONDON

2013

The cover image is of a torso of Pothos (Roman 1st century BC – 1st century AD) in the Museu Calouste Gulbenkian, Lisbon, Portugal.

ISBN 978-1-905670-49-9

Designed and computer typeset at the Institute of Classical Studies.

Printed by Short Run Press Limited, Bittern Road, Exeter EX2 7LW.

This volume is dedicated to the memory of two people whose untimely death marks a great loss, both personally and to our communities.

Elaine Matthews (died 26 June 2011): one of our esteemed contributors, ambassador and advocate of the Digital Humanities and the place there for Classics, thank you for all your many contributions to scholarship, to this volume, and your generous words on the cover of the earlier *Digital Classicist* (Ashgate 2010) volume.

Gerhard Brey (1954-2012): a valued friend, colleague, and collaborator with whom we shared intellectual ideas as well as coffee and biscuits. Gerhard was always willing to seek out new areas of 'interest' and so could be willingly called upon to review chapters in this and the earlier Ashgate volume.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS

The editors would like to thank the Institute for Classical Studies, and specially the Deputy Director and Administrator Olga Krzyszkowska, for their continued support and generosity in hosting and supporting the Digital Classicist seminars. Thanks are also due to all the members of our community who have presented papers at our seminars and conference panels as well as all those who have come along to listen and support these events.

We are grateful to the following scholars for comments, criticism, and advice on individual chapters in this volume; it is through their input that we are able to ensure the high quality of the final work: Chris Blackwell; Gabriel Bodard; John Bodel; Kalina Bontcheva; Gerhard Brey; Hugh Cayless; Graeme Earl; Michael Fulford; Sebastian Heath; Tim Hill; Kathryn Piquette; Dot Porter; Julian Richards; Matteo Romanello; Charlotte Roueché; Melissa Terras; Notis Toufexis; Charlotte Tupman; Michelle Wienhold.

ABSTRACTS

Andrew Bevan *Travel and interaction in the Greek and Roman World. A review of some computational modelling approaches* pp. 3-24

Inferring dynamic past behaviours from the static archaeological record is always a challenge, but computational and quantitative techniques can be helpful. In particular, they can provide useful insight on patterns of movement and interaction, by better characterising existing archaeological evidence, suggesting simple models of mobile decision-making or proposing expected patterns against which the observed record can be compared. This paper reviews the range of modelling options now available for understanding the movement and interaction behind the archaeological and historical record. There are increasing opportunities not only to pick and choose between different modelling approaches, but also to integrate them in a more theoretically and practically satisfactory way.

Vince Gaffney, Phil Murgatroyd, Bart Craenen, and Georgios Theodoropoulos *'Only individuals': moving the Byzantine army to Manzikert* pp. 25-43

Traditionally, history has frequently emphasized the role of the 'Great Man or Woman', who may achieve greatness, or notoriety, through the consequences of their decisions. More problematic is the historical treatment of the mass of the population. Agent-based modelling is a computer simulation technique that can not only help identify key interactions that contribute to large scale patterns but also add detail to our understanding of the effects of all contributors to a system, not just those at the top. The Medieval Warfare on the Grid project has been using agent-based models to examine the march of the Byzantine army across Anatolia to Manzikert in AD 1071. This article describes the movement model used to simulate the army and the historical sources on which it was based. It also explains why novel route planning algorithms were required in order to surmount problems with standard solutions.

Elton Barker, Leif Isaksen, Nick Rabinowitz, Stefan Bouzarovski, and Chris Pelling *On using digital resources for the study of an ancient text: the case of Herodotus's 'Histories'* pp. 45-62

Involving the collaboration of researchers from Classics, Geography, and Archaeological Computing, and supported by funding from the AHRC, *Hestia* aims to enrich contemporary discussions of space by developing an innovative methodology for the study of an ancient narrative, Herodotus's *Histories*. Using the latest digital technology in combination with close textual study, we investigate the geographical concepts through which Herodotus describes the conflict between Greeks and Persians. Our findings nuance the customary

topographical vision of an east versus west polarity by drawing attention to the topological network culture that criss-crosses the two, and develop the means of bringing that world to a mass audience via the internet.

-In this chapter we discuss three main digital aspects to the project: the data capture of place-names in Herodotus; their visualization and dissemination using the web-mapping technologies of GIS, Google Earth, and Timemap; and the interrogation of the relationships that Herodotus draws between different geographical concepts using the digital resources at our disposal. Our concern will be to set out in some detail the digital basis to our methodology and the technologies that we have been exploiting, as well as the problems that we have encountered, in the hope of contributing not only to a more complex picture of space in Herodotus but also to a basis for future digital projects across the Humanities that spatially visualize large text-based corpora. With this in mind we end with a brief discussion of some of the ways in which this study is being developed, with assistance from research grants from the Google Digital Humanities Awards Program and JISC.

Marco Büchler, Annette Geßner, Monica Berti, and Thomas Eckart

Measuring the Influence of a Work by Text Re-Use pp. 63-79

Over the centuries an incredible amount of ancient Greek texts have been written. Some of these texts still exist today whereas other works are lost or are available only as fragments. Without considering intentional destruction, one major question remains: why did some texts remain and others get lost? The aim of this chapter is to investigate this topic by trying to determine the influence of certain ancient Greek works through detecting text re-use of these works. Text re-use measures if and how an author quotes other authors and in this chapter we differentiate between *re-use coverage* and *re-use temperature*.

Tobias Blanke, Mark Hedges, and Shrija Rajbhandari

Towards a virtual data centre for Classics pp. 81-90

A wide variety of digital resources have been created by researchers in the Classics. These tend to focus on specific topics that reflect the interests of their creators; nevertheless they are of utility for a much broader range of research, and would be more so if they could be linked up in a way that allowed them to be explored as a single data landscape. However, while the resources may be reusable, the variety of data representations and formats used militates against such an integrated view. We describe two case studies that address this issue of interoperability by creating virtual resources that are independent of the underlying data structures and storage systems, thus allowing heterogeneous resources to be treated in a common fashion while respecting the integrity of the existing data representations.

Ryan Baumann *The 'Son of Suda On-line'* pp. 91-106

The Son of Suda On-Line (SoSOL) represents the first steps towards a collaborative, editorially-controlled, online editor for the Duke Databank of Documentary Papyri (DDbDP). Funded by the Andrew W. Mellon Foundation's Integrating Digital Papyrology Phase 2

(IDP2), SoSOL provides a strongly version-controlled front-end for editing and reviewing papyrological texts marked up in EpiDoc XML.

Elaine Matthews and Sebastian Rahtz

The Lexicon of Greek Personal Names and classical web services pp. 107-24

This chapter documents the data resources of the long-term classical research project, *The Lexicon of Greek Personal Names* (LGPN), published in six volumes since 1987. It explains and demonstrates the web interfaces and services which now make available online the bulk of the LGPN, providing both powerful searching tools for scholars and an interface to allow other systems to link to LGPN data. Making the data available online provides direct, unmediated access to the material and supports exploitation of the data for further research both individual and collaborative.

We describe the work that went into creating the Lexicon, detail the granularity of the data structures, and explain the history of the project's record management. We then move onto the work undertaken in recent years to provide an archival XML-based format for the Lexicon's long-term preservation, and show how this has allowed us to build new web services, including exposure of Resource Description Framework (RDF) metadata, using the ontology of the CIDOC Conceptual Reference Model (CRM) ontology for semantic web applications.[1]

Simon Mahony: HumSlides *on Flickr: using an online community platform to host and enhance an image collection* 125-46

Moving a teaching and research image collection from an analogue to a digital medium for delivery brings with it many advantages but at the same time it also presents many new problems and ones probably not previously considered. This chapter discusses the move of a departmental slide collection, firstly to a proprietary in-house format, and then subsequently to the online community platform Flickr. It draws on the experience and model of the Library of Congress in partnership with Flickr and *The Commons*, as well as initiatives at Oxford and at New York University, and in doing so critically analyses and evaluates the possibilities for the future development of this collection. It asks why this collection is not currently being used to its potential and examines how the development of a user community would help to enrich the collection and ensure long term sustainability and future growth.

[1] CIDOC CRM is an ISO standard (21127:2006) that 'provides definitions and a formal structure for describing the implicit and explicit concepts and relationships used in cultural heritage documentation' <http://www.cidoc-crm.org/>.

One of the great potentials of the internet is its capacity to aggregate and unify information from diverse sources. Information in the Classics, and data generated by classicists, is inherently fragmented, and organized according to different standards. This paper describes a project at King's College London which sought to provide a set of aggregating services to humanities scholars. www.arts-humanities.net provides a platform, a library, and a taxonomy to organize and present data: we describe its facilities for supporting a multi-source dataset tracing the paths of Romano-British inscriptions, both in space and conceptually. Itinerant geographies of metrical versus text inscriptions are discussed, including how these can be published in a variety of non-conventional platforms, such as Twitter. We argue that, in the future, these platforms will come to play a critical role in the wider scholarly discourse of the Classics.

ABBREVIATIONS

ABM	Agent-based modelling
ADS	Archaeology Data Service
AHRC	Arts and Humanities Research Council
API	Application Programming Interface
APIS	Advanced Papyrological Information System
AWIB	Ancient World Image Bank
CC	Creative Commons
CI	Collective Intelligence
CIDOC	International Council of Museums
CRM-CIDOC	CIDOC Conceptual Reference Model
CSV	Comma-separated data fields
DANS	Data Archiving and Networked Services
DARIAH	Digital Research Infrastructure for the Arts and Humanities
DDbDP	Duke Databank of Documentary Papyri
DPI	Dots per inch
DVCS	Distributed Version Control Systems
EDM	Europeana Data Model
GAP	Google Ancient Places
GIS	Geographical Information Systems
HEA	Higher Education Academy
HEFCE	Higher Education Funding Council for England
HESTIA	Herodotus Encoded Space-Text-Image Archive
HGV	*Heidelberg Gesamtverzeichnis der griechischen Papyrusurkunden Ägyptens*
IAph	*Inscriptions of Aphrodisias*
IDP	Integrating Digital Papyrology
ISAW	Institute for the Study of the Ancient World
JDI	Image Digitization Initiative
JISC	Joint Information Systems Committee
JSON	JavaScript Object Notation
KML	Keyhole Markup Language
LaQuAT	Linking and Querying Ancient Texts
LCCW	Longest Common Consecutive Words
LGPN	*The Lexicon of Greek Personal Names*
LoC	The Library of Congress
MDID	Madison Digital Image Database
OAI-ORE	Open Archives Initiative Object Reuse and Exchange
OER	Open Education Resources
OGSA-DAI	Open Grid Service Architecture–Data Access and Integration
OGSA-DQP	Distributed Query Processing
PDF	Portable Document Format

PN	Papyrological Navigator
PRM	Probabilistic Road Map
RDF	Resource Description Framework
RIB	*Roman Inscriptions of Britain*
SGML	Standard Generalized Markup Language
SOL	Suda On-Line
SoSOL	Son of Suda On-Line
SQL	Structured Query Language
SVG	Scalable Vector Graphic
TEI	Text Encoding Initiative
TLG	*Thesaurus Linguae Graecae*
URI	Uniform Resource Identifiers
V&A	Victoria and Albert Museum
VRE	Virtual Research Environment
WFS	Web Feature Service
WMS	Web Map Service
WYSIWYG	What-You-See-Is-What-You-Get
XML	Extensible Markup Language

THE *SON OF SUDA ON-LINE*

RYAN BAUMANN

Introduction

Integrating Digital Papyrology (*IDP*) is a multi-institutional project aimed at establishing and improving relationships between three digital papyrological resources: the *Duke Databank of Documentary Papyri* (*DDbDP*), the *Heidelberger Gesamtverzeichnis der griechischen Papyrusurkunden Ägyptens* (*HGV*), and the *Advanced Papyrological Information System* (*APIS*). Started in 1983, the *DDbDP* collects a number of digital transcriptions of ancient documentary papyri from print editions. *HGV* and *APIS*, meanwhile, collect metadata (place of origin, date, keywords, bibliography, *etc*.) and images of much of the same material. A unification of these data sources would allow linking the digital transcriptions of texts with the images, dates, and other metadata, and in 2007 the Andrew W. Mellon Foundation funded a project, *Integrating Digital Papyrology*, to begin this process. Over the years the *DDbDP* has undergone a number of transitions, and this project also supported its transition from an idiosyncratic SGML encoding to standards-based EpiDoc XML markup.[1] In addition, the grant provided funding to improve and finish the first generation of a tool for searching and browsing the unified collection of materials, called the Papyrological Navigator (or PN). At the conclusion of the *IDP* grant, Mellon funded a second phase of the project (called *IDP*2) with the following goals:[2]

1. Improve operability of the PN search interface on the merged and mapped data from the *DDbDP*, *HGV*, and *APIS*.
2. Facilitate third-party use of the data and tools.
3. Create a version-controlled, transparent and fully audited, multi-author, web-based, real-time, tagless, editing environment, which – in tandem with a new editorial infrastructure – will allow the entire community of papyrologists to take control of the process of populating these communal assets with data.

The environment described in the last item, inspired by the *Suda On-Line* (*SOL*),[3] was named the *Son of Suda On-Line* (*SoSOL*). Though it takes its name and inspiration from *SOL*, *SoSOL* was written from the ground up to incorporate new technologies, address

[1] SGML (Standard Generalized Markup Language), first formalized as a standard in 1986, is the generalized document markup language of which XML (Extensible Markup Language) is a descendant. EpiDoc <http://epidoc.sourceforge.net> is a set of community guidelines for marking up digital editions of ancient texts.

[2] J. D. Sosin *et al.*, 'Integrating Digital Papyrology 2' Mellon Foundation (New York 2008): <http://www.duke.edu/~jds15/IDP2-FinalProposalRedacted.pdf>

[3] The *Suda On-Line*: <http://www.stoa.org/sol/> is a project aimed at collaborative translation of the massive tenth century Byzantine encyclopedia known as the *Suda*.

project-specific problems, and move toward more open data and tooling.[4] This chapter aims at not only a description of the resulting software, but also of the challenges encountered and solutions chosen in its formulation to encourage broader adoption or discussion of both.

The Son of Suda On-Line

Though collaborative online editing environments, such as Wikipedia, have the advantage of allowing anyone to contribute, many question the scholarly integrity of resources which can be edited by anyone, unvetted. The *Suda On-Line*, which actually predated the existence of Wikipedia by two years, addressed this problem by marking submitted translations with their level of editorial vetting.[5] This combination of openness to contribution with strong editorial control was the guiding principle in the design of the *Son of Suda On-Line*.

However, even more than *SOL*, the papyrological projects encompassed in *Integrating Digital Papyrology* value the scholarly integrity of data published under their aegis. Thus, *SoSOL* attempts to digitally replicate the scholarly mechanisms of the peer review these projects would normally enforce. This results in somewhat of an inversion of where and how editorial control is exerted in comparison with *SOL*. While *SOL* users are authorized by editors during registration and are assigned work or must request a specific entry,[6] in *SoSOL* users are not screened and at present work on whatever they feel needs emendation or inclusion in the corpus. However, this distinction in the assignment of work may just arise naturally from the differing natures of the texts involved; whereas the *Suda* – while large – is a bounded unit of work, the papyri do not yet show signs of halting their expanding numbers in transcriptions and publications.

Standing in starker contrast is how submissions that have not received editorial oversight are handled: in *SOL*, they are immediately publicly searchable and accessible but marked as 'draft'; in *SoSOL*, submissions undergo review and voting by an editorial board before publication as 'canonical' and being made available for searching in the Papyrological Navigator. This may seem restrictive, or even contradictory to claims of openness. It is indeed the former, but only inasmuch as the editorial boards are controlling the quality of what they are willing to put their names to in the tradition of peer review. The latter requires some discussion.

Data and openness

We no longer see *IDP* as representing at any given moment a synthesis of fixed data sources, directed by a central management; rather, we see it as a constantly changing set of fully open data sources, governed by the scholarly community and maintained by all active scholars who care to participate. One might go so far as to say that we see this

[4] T. Elliott, *Background and funding: integrating digital papyrology* (2008): <http://idp.atlantides.org/trac/idp/wiki/BackgroundAndFunding>; J. D. Sosin, 'Digital papyrology', in *26th Congress of the International Association of Papyrologists* (2010): <http://www.stoa.org/archives/1263>.

[5] R. Finkel, W. Hutton, P. Rourke, R. Scaife, and E. Vandiver, 'The *Suda On Line*', *Syllecta Classica* 11 (2000) 178-90: <http://www.stoa.org/sol/about.shtml>.

[6] A. Mahoney, 'Tachypaedia Byzantina: the *Suda On Line* as collaborative encyclopedia', *Digital Humanities Quarterly* 3.1 (2009).

nexus of papyrological resources as ceasing to be 'projects' and turning instead into a community.[7]

What do we mean here by 'fully open'?

On one level, it is the terms under which data is published. Though asserting any sort of copyright on the 2,000-year-old texts themselves is perhaps nonsensical (at least in the USA),[8] the complete set of *IDP* XML files are published with a Creative Commons Attribution 3.0 Licence,[9] explicitly permitting the typical varieties of scholarly reuse and citation anticipated for the data, in line with other recent calls for open access in the humanities.[10] (Atypical and unanticipated forms of reuse would be even more exciting.)

On another level, it is the manner in which data is published. For collaborative online projects, this is usually a challenge. If the data is constantly changing, how do you publish it in any traditional sense? Perhaps even more challenging is this: how do you publish the changes themselves, both retroactively and proactively?

By retroactively, we mean that the revision history of the data up to the present may itself be important; by proactively, we mean that if a user has already obtained the complete revision history at some point in time, it is better to allow them to simply download the changes since that point. Many online collaborative environments, such as MediaWiki and the original *SOL*, store all changes in a database system. This usually makes distribution of the complete revision history, particularly proactive distribution, extremely difficult. As an example, the English-language Wikipedia was unable to distribute its complete dataset for several years, and was only most recently able to dump its revision history in January 2010, with no successful exports since.[11] Even if they were able to do so, the only mechanism for updating such a data dump is to download the entire several-hundred-gigabyte file each time.

Next-generation version control

We felt that the best way to approach this problem was to use a Revision Control System as the backend for the data itself, instead of a traditional database. Though there are many

[7] R. Bagnall, 'Integrating digital papyrology' in *Online humanities scholarship: the shape of things to come*, ed. F. Moody and B. Allen, Mellon Foundation (New York 2010): http://hdl.handle.net/2451/29592;

[8] See *e.g.* Bridgeman Art Library *v.* Corel Corp.: <http://www.law.cornell.edu/copyright/cases/36_FSupp2d_191.htm>, which rules that 'slavish copies' of public domain works are not copyrightable. For scholarly transcriptions and images of ancient texts, much of the goal is to produce as faithfully slavish a copy as possible.

[9] Creative Commons Attribution 3.0 License: <http://creativecommons.org/licenses/by/3.0/>.

[10] G. Crane, 'Give us editors! Re-inventing the edition and re-thinking the humanities', in *Online humanities scholarship: the shape of things to come*, ed. F. Moody and B. Allen, Mellon Foundation (New York 2010): <http://cnx.org/content/m34316/latest/>.

[11] Wikipedia: 'Database download – latest complete dump of English Wikipedia': <http://en.wikipedia.org/w/index.php?title=Wikipedia:Database_download&oldid=393163797\#Latest_complete_dump_of_English_Wikipedia>.

well-established centralized systems such as CVS and Subversion,[12] in recent years there has been an explosion in the popularity of Distributed Version Control Systems (DVCSs). Typically this means there is no 'central' server except by social convention; all copies of the repository are, in a sense, equal and can share changes with one another. This allows for a variety of workflow styles, and has a number of other important impacts.

One of the most popular distributed version control systems is Git, initially developed by Linus Torvalds for managing the Linux kernel software project. Due to its broad acceptance, design choices, and proven performance on a number of large projects, it is the backend we selected for data in *SoSOL*. The *SoSOL* codebase itself was also managed with Git from the beginning, and is available online.[13]

As a result of using a DVCS for the data backend, it is possible to use Git not only to retrieve the complete revision history of the *IDP* data as managed by *SoSOL*[14] (retroactive publication), but also to easily update your copy of the repository as changes are published (proactive publication). Due to the distributed nature of Git, the concepts of branching development and merging changes have been integrated into its design, making it easy to keep your copy of the data up-to-date even if you have made your own modifications. (After all, if anyone can pull changes from any other copy of the repository, merging needs to be fast and easy.) The long-running version of this behaviour of splitting off your own modifications is known in the open source world as 'forking'. Git reduces the overhead of both forking a project, as well as of contributing your forked changes back.

That a DVCS makes these things trivial also represents a significant decision in the design of *SoSOL*: for the 'canonical' data repository it interacts with, it does not need to care about any external mechanisms or workflows used to introduce changes. *SoSOL* only needs to keep track of changes within its domain; it is merely a front-end and social infrastructure for easing and managing contributions. When a user edits data in *SoSOL*, it is forked from the main repository to allow them to do their work without interruption. They then submit their changes for editorial review, and when they pass muster they are merged back into the canonical repository. However, the repository may be updated by any external process in the interim, typically without drastically impacting the work that must be done to perform the merge.

Thus, the fact that *IDP* now uses Git for its public data repository, in combination with the licence the data is distributed under, represents the complete realization of 'a constantly changing set of fully open data sources governed by the scholarly community and maintained by all active scholars who care to participate'.[15] For us, 'participate' in fact has two senses: participating within our system (that is, participating in our editorial

[12] CVS, or the 'Concurrent Versions System,' was started in 1986 as a version control system built atop the even-earlier 'Revision Control System', which only operated on a single file. 'Subversion' was started as a later project for version control similar to CVS, but with various fixes and improvements.

[13] The *Son of Suda On-Line*: <https://github.com/papyri/sosol>.

[14] *IDP* Data: <http://github.com/papyri/idp.data>. See also *IDP* Data available on GitHub: <http://digitalpapyrology.blogspot.com/2011/01/idp-data-available-on-github.html>.

[15] Bagnall, 'Integrating digital papyrology' (n. 7 above).

review process), or participating in any enterprise you choose with the complete dataset which we make freely available.

Implementation

The *Son of Suda On-Line* environment itself is written in Ruby using the popular Rails web framework.[16] Instead of the mainline Ruby interpreter written in C (usually referred to as Matz's Ruby Interpreter, or MRI, after the language's creator), we use a Java implementation called JRuby. Though this was initially done to enable deployment of *SoSOL* in any Java Servlet Container such as Tomcat, *SoSOL* has come to use a number of Java libraries (particularly for interacting with XML data), facilitated by JRuby's tight Java integration.

Git internals

Some discussion of how Git works and internally represents version history is perhaps necessary to illustrate how its design enables and informs other design decisions in *SoSOL*. In Git, the version history of a project is encoded as a directed graph of three kinds of internal objects, all of which are identified by the unique SHA-1 hash of the object's content.[17] These objects form the 'nodes' in Git's graph, while their contents contain the directional arrows linking them together.

The simplest instance of version history (in Git, and conceptually) is a single piece of content with one version. Bare content is the 'blob' object in Git, and has no additional metadata associated with it – these can be thought of as leaf nodes in the graph (that is, nodes which do not point to other nodes).

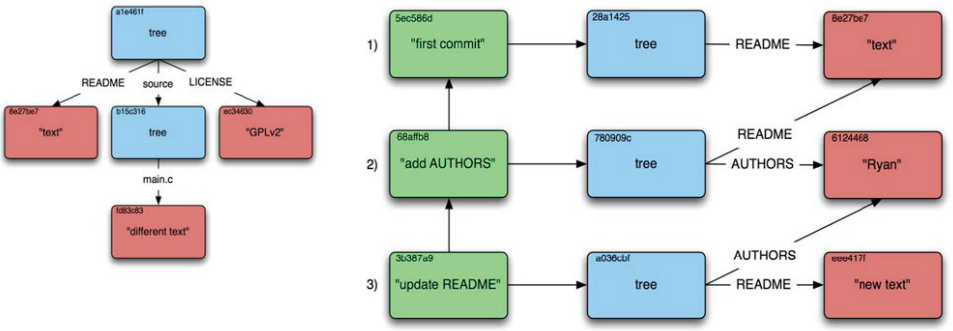
However, having just one file is not very useful for most projects. Git organizes and collects multiple blob objects into a file structure using tree objects. These tree objects are simple, plaintext files, which list identifying hashes with their filenames. Each tree object represents a single directory; for subdirectories, trees can point to other tree objects in addition to blobs, as in figure 1a.

Revisions in Git are stored as commit objects. These point to a single tree, and contain metadata about the commit (such as author, time, commit message) as well as pointers to one or more parent commit objects. Thus, a simple merge would have two parent commits, while a branch or fork would be two different commits pointing to the same commit. The fact that all of this history is stored as a connected graph is what allows the Git system itself to examine things such as where a fork occurred when attempting to merge concurrent changes, and intelligently make the right decision based on the chosen merge strategy.

Let us walk through a simple, linear commit history to illustrate things, winding up with the graph shown in figure 1b.

[16] 'Ruby' is a dynamic, object-oriented, interpreted programming language. 'Rails' is a web framework written for and in Ruby which uses the Model-View-Controller design pattern to organize web applications.

[17] Think of the SHA-1 hash as a 160-bit number that uniquely identifies any given input string. Even a small change in the string results in a very different hash, and collisions are designed to be rare.



(a) A tree graph in Git (b) A series of commits in Git

Figure 1: Visualizations of Git's internal graph structure. Red squares are blobs, blue are trees, green are commits. Text in the top-left corner is the object's truncated hash.

1. We start our project with just a README file containing the string 'text'. Since we want to record this momentous occasion, we commit the state of our repository with the commit message 'first commit'. This commit points to the hash of the tree, which contains the hash of our README.
2. Since this project is sure to be our *magnum opus*, we quickly decide we want to immortalize contributors by crediting them in an AUTHORS file. We write this out and hastily make a new commit to add it. Since the README is unchanged, the same object from before is reused. However, because we have added a new file, the tree has changed, so a new tree object is made for the commit to point to.
3. Later, the mood strikes us to update the README, so we do so, and make a new commit. Again, a new blob object is constructed for the new content, although this time the AUTHORS blob is able to be reused because we did not modify it. Because changing the README changes the blob object's hash, the tree object must store the new hash and receive a new hash itself.

Though there are many other facets and features, this is the core of Git.

The consequence of all objects being identified by a unique hash of their contents means these objects can easily be shared between copies of the repository – knowing both that there will not be conflicts between objects with different content having the same hash, and that the same object will have the same hash no matter where it is. Since all links between objects are part of the hashed content, any change in the graph would result in a cascade of changing hashes (as in step three of our example). Because Git verifies and uses these hashes for its own operation, the integrity of the repository is incredibly robust; if you have a copy of the repository, and someone tries to rewrite history by removing or altering an object which is already referenced, you will notice when you try to pull their changes because all descendant objects will have different hashes from yours. If an object file is corrupted, it can easily be restored from any copy of the repository; likewise, any copy of the repository is a complete copy of the repository.

Data sources, publications, and workflow

As outlined in the introduction, *IDP* represents the synthesis of a variety of papyrological projects managed by different institutions. This has informed *SoSOL*'s design in how it interacts with and models these disparate data sources.

Because these papyrological resources evolved separately, the concept of a 'publication' and what defines an object or text may in some cases be slightly different. As an example, two distinct hands may have written two different texts on a single piece of papyrus; *HGV* keeps two metadata records, while the *DDbDP* keeps a single transcription (but still indicating the distinct hands inside it). These relationships can become quite complicated with things such as reprints of texts, or ancient, military receipt records containing hundreds of texts. In addition, the data itself is different enough that different methods may be preferable for editing or interacting with data from a given resource. Although *IDP* has standardized on EpiDoc XML encoding, it still collects a variety of different kinds of information about papyri, including transcriptions, translations, and complex metadata.

SoSOL deals with this internally by representing each 'publication' as a collection of one or more resources, which we call 'identifiers'. Because there are a variety of types of resources, we use an identifier base class which implements common methods for all types of resources (such as getting or setting the identifier's content in the Git repository), while using subclasses of this to implement behaviour specific to a given type of resource. Each identifier is actually called such because it is named with a string which it is assumed corresponds to exactly one resource – for example, *HGV*'s or *DDbDP*'s name for an object. Because these resources are stored as separate XML files with their own particular directory structure, each type of identifier has its own method for turning its name into a file path.

SoSOL uses the 'publication' as the unit of work for the editorial workflow – each publication corresponds to a development branch in the Git backend. When the identifiers belonging to a publication have been modified, it can be submitted to the editorial boards for review. Because we have had to deal with disparate types of resources from the beginning, the review process is able to have different editorial boards for each type of identifier. Currently this is implemented as a sequential workflow; if a user submits a publication with changes to *HGV* metadata, *DDbDP* transcription, and *HGV* translation, it will be reviewed in that order, requiring approval from each board before going to the next. Each editorial board has their own membership and voting rules. If a submission is rejected, the user is able to see the reasons given during voting and revise and resubmit their work. Users can also see a list of other users working on the same 'publication', as well as their contact information. If they desire to coordinate amongst themselves, they can share links to their publications, which are viewable (but not editable) by any logged-in user who knows the link.

Because of Git's design, *SoSOL* is actually able to maintain a separate Git repository for each user and editorial board in its backend (see figure 2). Despite the fact that *IDP*'s canonical data repository is around 1GB in size, each copy can be on the order of kilobytes because it can simply reference objects already stored in the repository it has been forked from. This means that when a user begins editing a publication, the branch for that publication is made on their copy of the repository, and only new objects which they create in the course of making updates must be stored in it. When they submit the publication, this branch and its related objects are copied to the board's repository, and, when they approve the publication, they then merge this branch back into the canonical data

repository. Eventually, this design could be integrated with a Git server (in the style of GitHub),[18] allowing each user to have direct access to their own Git repository to make changes easily, using any process they choose, before submission.

Figure 2: A user's dashboard in SoSOL, with publications being worked on.

Another advantage of Git's design is that accurate, transparent attribution is easily maintained in the history of each piece of data. Interventions made by the editorial board after submission are preserved as being authored by them, rather than by the submitter. Because Git allows a distinction between 'author' (who wrote a commit) and 'committer' (who put a given commit in the repository), we can record which editor made the merge to the canonical repository without losing information about who actually authored the underlying changes. We also record the members of the editorial board at the time a submission is accepted, by adding that they have signed off to the commit message. All of this is done as part of a process we call 'finalization' – after a board approves something, it is assigned to a random member of the board to undergo any final revisions and manual oversight of the merge into the canonical repository. As part of this, we flatten multiple commits made before submission into a single commit (as each time the user saves it introduces a commit, which was deemed more revision granularity than necessary for our

[18] Secure source code hosting and collaborative development – GitHub: <http://github.com/>.

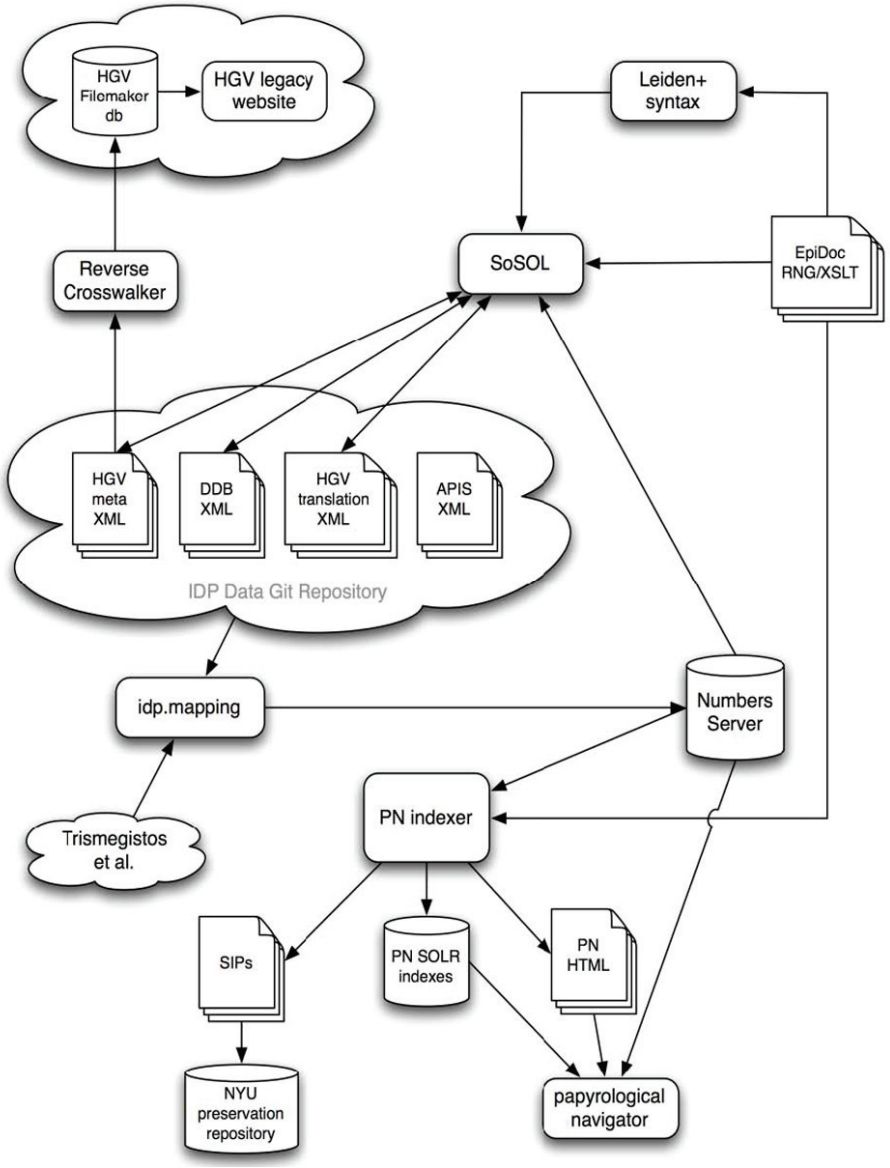


Figure 3: Software components and data flow at the conclusion of IDP2

use), which is rewritten to have the content of any individual commit messages as well as the submission reason and editorial sign-off messages.

The advantage of all this is that the core design of *SoSOL* deals mainly with this identifier/publication model, and providing functionality for using this abstraction to have an editorial workflow on top of a Git repository. The goal was to make these core components reusable, while providing your own implementations for how your own identifiers are edited and aggregated into publications. One can imagine the simplest implementation as being an identifier whose name is the file path, which just presents the

plaintext contents of the file for editing, and which has no relationships with other identifiers, so that each publication is a single identifier.

Under *IDP*2, *SoSOL* manages the complex relationships between identifiers by interfacing with a piece of software developed for the project, which we call the Numbers Server. This is implemented as an RDF triplestore,[19] which is built up by processing the entire canonical *IDP* dataset and looking for associations between resources. *SoSOL* can then simply query any single identifier in the Numbers Server to find all other identifiers related to it, in order to aggregate them into a logical publication. The relationships and data flow between *SoSOL*, the Papyrological Navigator, and the Numbers Server are illustrated in Figure 3.

Alternative syntax for XML editing

One of the proposed items for the *SoSOL* environment was to provide a 'tagless' editing environment for the EpiDoc XML data used by *IDP*. For metadata, where there is some fixed number of possible elements, this can be achieved by simply presenting the user with a form specific to the needed data types which translates to and from XML. This is what we have done for *HGV* metadata, as shown in Figure 4.

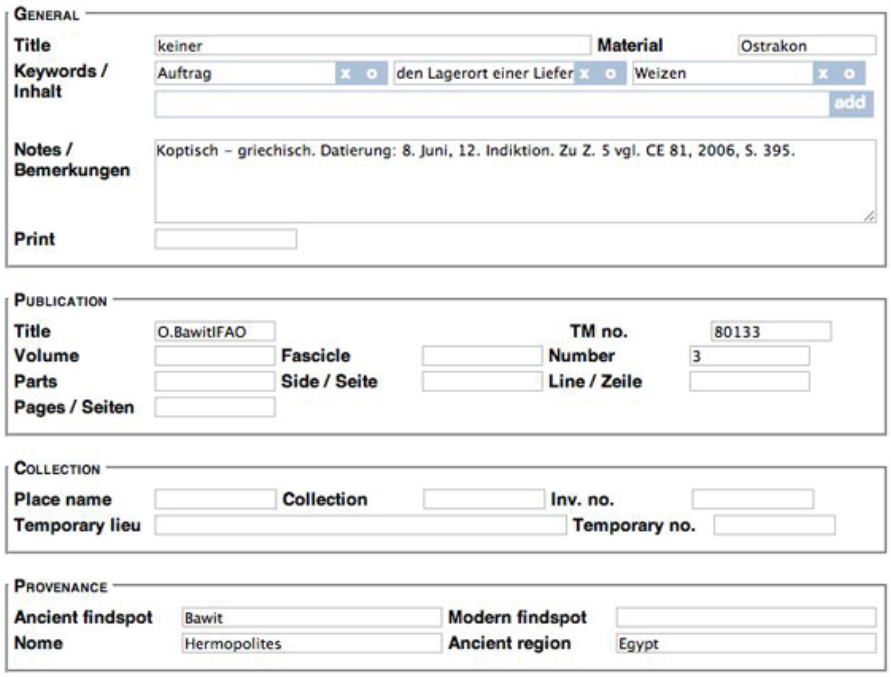


Figure 4: HGV metadata entry form in SoSOL

[19] An RDF (Resource Description Framework) triplestore provides a way of storing and querying the relationships between resources as 'triples' in subject-predicate-object form, such as 'dogs are animals' or, in this case, '*DDbDP*'s P.Oxy. 1 53 relates to *HGV* number 20715'.

However, for freeform text transcriptions like those recorded by the *DDbDP*, the concept of a 'tagless' environment is more challenging. Because many papyrological materials are damaged and difficult to read, scholarly transcriptions record a number of things about the reading of the text itself. If letters of a word cannot be made out on the object, but you can interpolate from context what they likely were, you should indicate that your restoration is a result of that process. This happens so often when editing papyrological texts that a shorthand for indicating them in the text itself was developed and, in a 1931 meeting at the University of Leiden, standardized as a set of rules called the Leiden conventions. Epigraphers also adopted these conventions, as they faced many of the same challenges and, along with papyrologists, have used them for publishing printed transcriptions ever since (as in Figure 5).

[ἔτους α (?) Αὐτοκράτορος] . . [.]. . του
[- ca.12 -] Σεβαστοῦ
[εἴργ(ασται) ὑ(πὲρ) χω(ματικῶν) ἔ]ργ(ων) τοῦ αὐτοῦ πρώτου (ἔτους)
[-ca.?-] κ κϛ ἐ[ν] τῇ ʾΕπα -
[γαθιαν]ῇ διώ(ρυγι) Βακχιά(δος)
[-ca.?-] Πατκ(όννεως) τοῦ Θεαγένους
[. . . . .] μη(τρὸς) Ταύρεως
[-ca.?-] (hand 2) σεση(μείωμαι)

Figure 5: Typical print transcription following Leiden conventions (P.Sijp., 41a)

1. [ἔτους] [<#α=1#> (?)] [Αὐτοκράτορος] .2[.1].2του
2. [ca.12] Σεβαστοῦ
3. [(εἴργ(ασται)) (ὑ(πὲρ) χω(ματικῶν))] ([ἔ]ργ(ων)) τοῦ αὐτοῦ πρώτου ((ἔτους))
4. [.?] <#κ=20#> <#κϛ=26#> ἐ[ν] τῇ ʾΕπα
5.- [γαθιαν]ῇ (διώ(ρυγι)) (Βακχιά(δος))
6. [.?] (Πατκ(όννεως)) τοῦ Θεαγένους
7. [ca.6] (μη(τρὸς)) Ταύρεως
8. [.?] $m2 (σεση(μείωμαι))

Figure 6: Leiden+ representation of the P.Sijp., 41a text

EpiDoc is a TEI-based XML encoding standard for marking up the same sort of textual semantics represented in Leiden, with additional standardized markup for other features typically needed when encoding ancient texts. For example, numbers which are written as Greek text can be marked semantically as numbers with their value, orthographic corrections can have both the normalized and original word linked, and so forth. Thus, the key advantage of EpiDoc is that it acts as a superset of Leiden with explicit, computationally actionable semantics.

Because we have our transcriptions already encoded in EpiDoc, we wanted to surface these facets of it to users, without burdening them with the full verbosity of XML markup. We also wanted them to be able to explicitly mark up new texts in the same environment.

We contemplated trying to use the contentEditable HTML attribute to provide a sort of 'rich text', what-you-see-is-what-you-get (WYSIWYG), text entry form. However, browser implementations of this feature vary wildly in behaviour and often confound user expectations, and what exactly the meaning of 'WYSIWYG' is when semantically marking up things such as numbers is debatable. As a result, we decided to use a simple, plaintext, form element, utilizing a transformation of the XML to make the text more legible and easier to edit quickly. Of course, to update the modified plaintext would require a transformation back to XML to save it in our system. One way to do this would be to write two separate transform processes, one from XML to plaintext, and one from plaintext to XML. However, verifying and maintaining such a process would be difficult.

Instead, we use a tool called XSugar to perform both directions of the XML transformation.[20] This utility allows you to define a single, context-free grammar, where each rule has both an XML representation and a non-XML representation. Thus it can parse either representation into an intermediate form, and then use the same ruleset to output the opposite representation. Additionally, the tool can check that this transformation is reversible – that is, round-trips of a given input do not alter it.[21] Due to the immense size of the *DDbDP* corpus (over 55,000 transcriptions), we use automated nightly runs of transformations on the entire corpus to both verify and improve our definition of the Leiden+ grammar, as well as reduce encoding errors in the source XML (many being difficult, edge cases left over from the transitioning of the *DDbDP* from SGML to EpiDoc). We also use an XML normalization process to reduce the amount of 'thrashing' in the version history – small changes to the text should not alter unrelated parts of the XML and make it hard to spot the actual change when looking through the file's history.

We call our non-XML representation of EpiDoc markup 'Leiden+', as it attempts to use the same symbols as Leiden where possible, but is also able unambiguously to represent the additional markup enabled by EpiDoc encoding. Figures 5 and 7 illustrate how a traditional print transcription might be marked up in EpiDoc XML, with figure 6 being the Leiden+ transformation of that XML. As you can see, things like the Greek letter 'κ' on line four being the number '20' are implicit in print, but explicit in both EpiDoc and Leiden+.

Though Leiden+ must in some cases be more verbose than traditional Leiden, this is due to the fact that Leiden+ must be able to be transformed into unambiguous, valid EpiDoc XML. For example, on line 3, abbreviations expanded by an editor use nested parentheses instead of a single pair around just the expansion, because the unit of text which is being expanded must be marked up as well. Because multiple, standalone Unicode combining underdots (indicating vestiges of illegible characters, as in line 1 of the example text) can be confusing to type and count by themselves, Leiden+ simply uses a period followed by the number of characters. However, for characters which are unclear but can be inferred from context (as in 'ῦ' of 'αὐτοῦ' at the end of line 3), Leiden+ preserves the combining underdot for readability, and we provide a JavaScript helper for inserting the character (a screenshot

[20] XSugar – Dual Syntax for XML Languages: <http://www.brics.dk/xsugar/>.

[21] C. Brabrand, A. Møller, and M. I. Schwartzbach, 'Dual syntax for XML languages', *Information Systems* 33.4-5 (2008) 385-406.

of Leiden+ as it appears with helpers in the editing environment is shown in Figure 8). Users can, of course, still edit the XML directly, with a button provided to copy the entire content of each text area to their clipboard so they can paste it into their own editor. In either case, submissions are validated against the EpiDoc RELAX NG schema before saving in order to ensure that invalid XML does not make its way into the system.[22]

```
<div xml:lang="grc" type="edition" xml:space="preserve">
 <ab>
  <lb n="1"/><supplied reason="lost">ἔτους</supplied> <supplied
   reason="lost" cert="low"><num value="1">α</num> </supplied> <supplied
   reason="lost">Αὐτοκράτορος</supplied> <gap reason="illegible" quantity="2"
   unit="character"/><gap reason="lost" quantity="1" unit="character"/><gap
   reason="illegible" quantity="2" unit="character"/>του
  <lb n="2"/><gap reason="lost" quantity="12" unit="character"
   precision="low"/> Σεβαστοῦ
  <lb n="3"/><supplied reason="lost"><expan>εἴργ<ex>ασται</ex></expan>
   <expan>ὑ<ex>πὲρ</ex> χω<ex>ματικῶν</ex></expan></supplied>
   <expan><supplied reason="lost">ἔ</supplied>ργ<ex>ων</ex></expan> τοῦ
   αὐτο<unclear>ῦ</unclear> πρώτου <expan><ex>ἔτους</ex></expan>
  <lb n="4"/><gap reason="lost" extent="unknown" unit="character"/> <num
   value="20">κ</num> <num value="26">κ ς </num> ἐ<supplied
   reason="lost">ν</supplied> τῇ Ἐπα
  <lb n="5" type="inWord"/><supplied reason="lost">γαθιαν</supplied>ῇ
   <expan>διώ<ex>ρυγι</ex></expan> <expan>Βακχιά<ex>δος</ex></expan>
  <lb n="6"/><gap reason="lost" extent="unknown" unit="character"/>
   <expan>Πατκ<ex>όννεως</ex></expan> τοῦ Θεαγένους
  <lb n="7"/><gap reason="lost" quantity="6" unit="character"
   precision="low"/> <expan>μη<ex>τρὸς</ex></expan> Ταύρεως
  <lb n="8"/><gap reason="lost" extent="unknown" unit="character"/>
   <handShift new="m2"/><expan>σεση<ex>μείωμαι</ex></expan>
 </ab>
</div>
```

Figure 7: EpiDoc XML fragment equivalent to the preceding Leiden+

[22] RELAX NG (REgular LAnguage for XML Next Generation) allows the creation of complex XML validation rules not possible with traditional XML DTDs (Document Type Definitions).

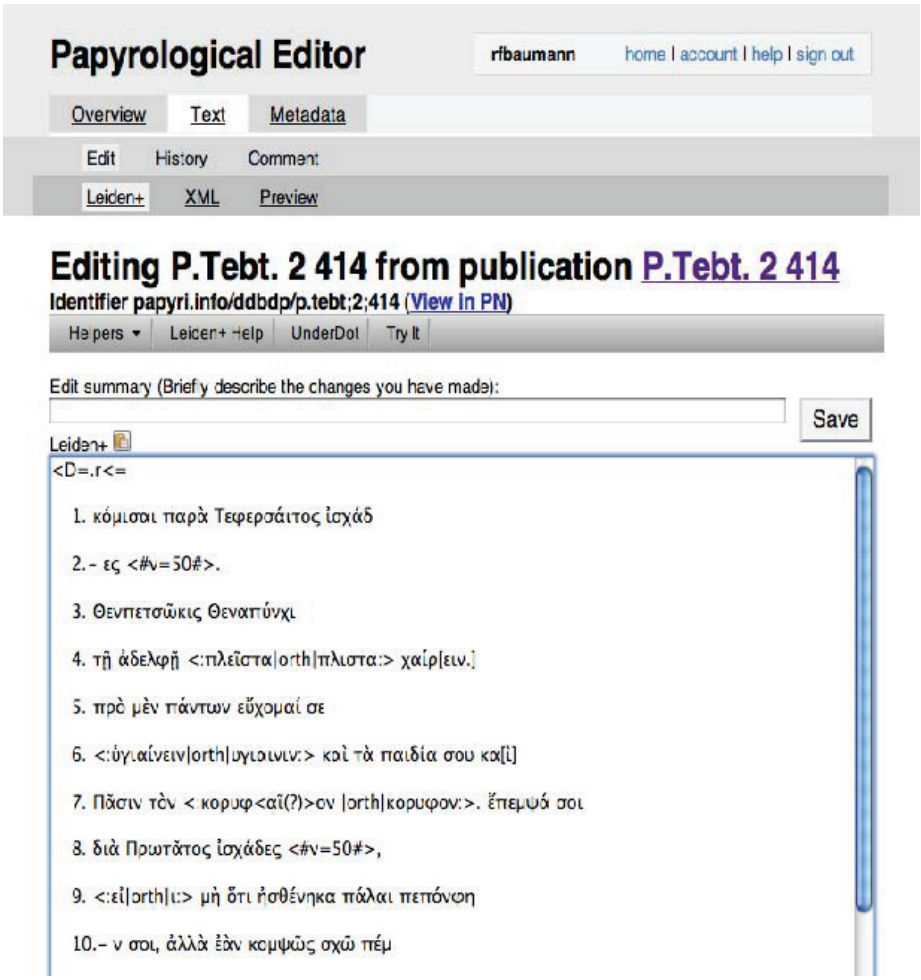


Figure 8: Editing Leiden+ in SoSOL

Conclusions

While Leiden+ does take some experience to get used to, in EpiDoc training seminars where we have introduced students and papyrologists to using it as an alternative for editing XML, the response thus far has been very positive. Users have entered and submitted thousands of new texts for inclusion in the *DDbDP*, using the production version of *SoSOL* running on papyri.info, dubbed the Papyrological Editor.[23] As of this writing, almost 25,000 commits have been made by over 200 different authors since the transition of *IDP* to using Git for its data backend, the majority of these through *SoSOL*. New, electronic editions of texts can now be made available much more quickly, and the potential for born-digital editions with vetting and version control has been enabled by the system. In addition, the *Perseus Digital*

[23] Papyrological Editor: < http://www.papyri.info/editor/> the public, running instance of *SoSOL* for *IDP*.

Library has recently announced that they plan to use *SoSOL* 'to decentralize the curation, annotation, and general editing of the TEI XML texts that it hosts'.[24]

The model of loosely-coupled tools operating on their interests over standard interfaces (for example: *SoSOL*'s interactions with Git, *SoSOL*, and PN using standard RDF to interact with the Numbers Server, *etc*.) has allowed for flexible and unexpected uses with very little additional work. For example, because the Leiden+ grammar definition and transformation code is completely separate from *SoSOL* and only included externally, the same code should be easily reusable by any project wishing to use it in conjunction with their standard EpiDoc XML texts. The Numbers Server being implemented as an RDF triplestore has also had great utility, allowing arbitrary SPARQL queries to be written, which reveal useful information about complex relationships in the data.[25]

We believe that exposing our complete dataset and its history to the community is the best approach to enabling true community ownership of the data. Using a DVCS for our data backend in the editing environment, instead of a traditional relational database backend, is what allows us to do this with very little friction, and facilitates direct interaction with the data repository without necessitating going through our project-specific editing environment. This is an approach which could be adopted by other Digital Humanities projects, even if not using *SoSOL* itself. That the system transparently preserves attribution at every step will, we hope, foster a sense that users do have some investment and ownership in their contributions. Additionally, it is hoped that this will enable academic institutions to recognize individuals' work in the system as scholarly activity, equivalent to work with traditional print publications.

This is also a way of moving away from the rigidity and implied authority of print publications. The *DDbDP* is not a fixed resource, finished at some date, unwavering and confident that it knows all; rather, it is a collection of conjectures, now easily capable of being revisited, revised, and improved. The technological framework now in place aims to reduce the overhead of these activities, to speed the expansion of knowledge and detection of error. It invites in the Popperian spirit: 'if you are interested in the problem which I tried to solve by my tentative assertion, you may help me by criticizing it as severely as you can'.[26] By publishing our data as a resource which is easily capable of decentralization and reuse, we hope also to apply this ideal to the system itself. While submissions which pass our system of editorial review derive their authority from the composition of the editorial board and the submitter, others may independently modify and publish our data under their own authority.

Ryan Baumann (*Harvard Centre for Hellenic Studies*) rfbaumann@gmail.com

[24] *Perseus Digital Library* news item, March 20th 2012: <http://www.perseus.tufts.edu/hopper/>.

[25] SPARQL (SPARQL Protocol and RDF Query Language) is the language used to query and discover relationships in RDF triplestores. (See also n. 19 above).

[26] K. Popper, *Conjectures and refutations* (London 1963) 30–36.

Bibliography

Bagnall, R., 'Integrating digital papyrology' in *Online humanities scholarship: the shape of things to come*, ed. F. Moody and B. Allen, Mellon Foundation (New York 2010): <http://hdl.handle.net/2451/29592>.

Brabrand, C., A. Møller, and M. I. Schwartzbach, 'Dual syntax for XML languages', *Information Systems* 33.4-5 (2008) 385-406.

Crane, G., 'Give us editors! Re-inventing the edition and re-thinking the humanities', in *Online humanities scholarship: the shape of things to come*. ed. F. Moody and B. Allen, Mellon Foundation (New York 2010): <http://cnx.org/content/m34316/latest/>.

Elliott, T., *Background and funding: integrating digital papyrology* (2008). <http://idp.atlantides.org/trac/idp/wiki/BackgroundAndFunding>.

Finkel, R., W. Hutton, P. Rourke, R. Scaife, and E. Vandiver, 'The *Suda On Line*', *Syllecta Classica* 11 (2000) 178-90: <http://www.stoa.org/sol/about.shtml>.

Mahoney, A., 'Tachypaedia Byzantina: the *Suda On Line* as collaborative encyclopedia', *Digital Humanities Quarterly* 3.1 (2009).

Popper, K., *Conjectures and refutations* (London 1963).

Sirks, A. J. B., and K. A. Worp, ed., 'Papyri in memory of P. J. Sijpesteijn', *American Studies in Papyrology* 40 (2007) 275-76.

Sosin, J. D., 'Digital papyrology', in *26th Congress of the International Association of Papyrologists* (Lexington KY 2010): <http://www.stoa.org/archives/1263>.