EXPLORATIONS IN PROVENANCE IN THE INFORMATION SCIENCES

BY

MICHAEL ROBERT GRYK

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Library and Information Science
in the Graduate College of the
University of Illinois Urbana-Champaign, 2024

Urbana, Illinois

Doctoral Committee:

Professor Bertram Ludäscher, Chair
Assistant Professor Rhiannon Bettivia, Simmons University
Professor J. Stephen Downie
Professor Michael Twidale

Abstract

Provenance is important throughout Library and Information Science and is particularly important for the information infrastructures which support the computational aspects of the natural sciences. This is highlighted by the prominence of provenance as a plank in the FAIR principles for data stewardship (principle R1.2). While traditionally focused on the history/lineage of physical objects, provenance is now commonly accepted to apply to digital objects such as the results of computation as well as to the recipes for computing; in the case of recipes this prospective provenance is critical for reproducibility. This dissertation begins with background in provenance pertaining to data curation and computational reproducibility. The second part describes attempts to "FAIRify" the reporting and execution of workflows within a domain of natural science for better data stewardship to support data reusability. The next chapters argue that there remains a gap in our ability to fully document provenance as there are more story-telling tenses than just the past (retrospective) and future (prospective). There is also the subjunctive (conditional) and perhaps many others. Supporting new flavors of provenance requires new modeling constructs. The thesis concludes with novel information modeling techniques which exploit reification of sub-class relationships suitable for modeling these many sub-classes of provenance, as well as other domains.

This dissertation is dedicated to my loving parents, Joan and Leon.

Acknowledgments

A dissertation has a single author but this work would not have been attempted nor completed without the help of a host of people who offered guidance, support, friendship, and assistance leading up to and throughout my time at the University of Illinois.

First, I would like to acknowledge my friends and family for their lifetime of support. This thesis is dedicated to my parents who sacrificed themselves throughout their lives for their children. My family, Benjamin, Eva, David, Thomas, Lynette, Kevin, Daniel, Anthony, Katherine, Hank, Evie, Scott, Lori, Christopher, Cheyenne, Scott, Elizabeth, Justin and Jaime were extremely supportive of my second doctoral degree and the plights of a displaced person. The same thanks are given to my friends who are too large in number to name but who all helped support me in returning to student life.

Next, I would like to acknowledge the faculty at the University of Illinois iSchool (formerly known as GSLIS). The U of I is an incredible, amazing, wonderful academic institution and I learned much more than I expected in the few years I spent on campus. Drs. Alistair Black, Lori Kendell, Vetle Torvik, Bertram Ludäscher, Michael Twidale, David Dubin, Rhiannon Bettivia, Allen Renear and Elsa Gunter (Computer Science) are extraordinary teachers. I would also like to thank all of the various friends and colleagues who passed through GSLIS, with a special mention of Rhiannon Bettivia, Yi-Yun (Jessica) Cheng, Douglas Heintz and Jacob Jett. And of course, a great big "cheers" to the convener of symposium, J. Stephen Downie. Be there or be acknowledged somewhere else.

Next, I would like to acknowledge my colleagues at UCONN Health for supporting my efforts for a second doctoral degree. Many thanks to Dean Bruce Liang, and Professors Sandy Weller and Jeffrey Hoch for approving my leave of absence to return to school. I am also appreciative of my many collaborators in Connecticut who accommodated my transition, with particular mention of the NMRbox team (Jeffrey Hoch, Mark Maciejewski, Adam Schuyler, and Gerard Weatherby), as well as the CONNJUR team (Heidi Ellis, Timothy Martyn, Gerard Weatherby (again!), Jay Vyas, Matt Fenwick, and RJ Nowling). I owe special thanks to Heidi for encouraging me to pursue my second degree and her never-ending uplifting support as well as to Prof. Larry Rothfield for living the life he lived.

Finally, I would like to acknowledge my thesis committee, Drs. Bertram Ludäscher, Michael Twidale, J. Stephen Downie. and Rhiannon Bettivia for their many years of guidance and feedback and for their patience with an elderly doctoral student who finished later rather than sooner.

It may not be obvious to the reader, but as the author I see the threads of influence from everyone in these acknowledgments within this completed dissertation. They should assume no blame for my mistakes but take credit for helping foster this work, each in their own special way.

Table of contents

List of Abbreviations and Acronyms

ASIS&T	Association for Information Science and Technology
bioNMR	Biomolecular Nuclear Magnetic Resonance
BMRB	BioMagResBank
CONNJUR	Connecticut Joint University Research
COSY	Correlation Spectroscopy
CST	CONNJUR Spectrum Translator
CWA	Closed World Assumption
CWB	CONNJUR Workflow Builder
DBP	Driving Biological Project
DNA	Deoxyribonucleic Acid
EAV	Entity Attribute Value
EBI	European Bioinformatics Institute
ER	Entity Relationship
FAIR	Findable Accessible Interoperable Reusable
FCA	Formal Concept Analysis
FOAF	Friend of a Friend Ontology
FRBR	Functional Requirements for Bibliographic Records
FT	Fourier Transform
FTP	File Transfer Protocol
GSLIS	Graduate School of Library and Information Science
GTK	GIMP Toolkit
HDF	Hierarchical Data Format
HIMYM	How I Met Your Mother
HSQC	Heteronuclear Single Quantum Coherence Spectroscopy
HTTP	Hypertext Transfer Protocol

IDCC	International Digital Curation Conference
IGSN	International Geological Sample Number
IMLS	Institute of Museum and Library Services
INCHI	International Chemical Identifier
ISBN	International Standard Book Number
IUPAC	International Union of Pure and Applied Chemistry
JCDL	Joint Conference on Digital Libraries
LCSH	Library of Congress Subject Headings
MRI	Magnetic Resonance Imaging
NAN	Network for Advanced NMR
NCBI	National Center for Biotechnology Information
NIH	National Institutes of Health
NMR	Nuclear Magnetic Resonance
NSF	National Science Foundation
OCLC	Online Computer Library Center
OOP	Object-oriented Programming
OPM	Open Provenance Model
OWA	Open World Assumption
OWL	Web Ontology Language
PMEST	Personality Matter Energy Space and Time.
PREMIS	Preservation Metadata Implementation Strategy
PRIMAD	Platform, Research Objective, Implementation, Method, Actors, Data
PROV	W3C Provenance Standard
PUID	Persistent Unique Identifier
PVW	Preserving Virtual Worlds
RDF	Resource Description Framework
RLG	Research Library Group
sigProps	Significant Properties (Digital Preservation Term)
SPARQL	SPARQL Protocol and RDF Query Language
TOCSY	Total Correlation Spectroscopy
TRAC	Trustworthy Repositories Audit & Certification
UCONN	University of Connecticut
URL	Universal Resource Locator

VM	Virtual Machine
W3C	World Wide Web Consortium
WODB	Which One Doesn’t Belong
XML	Extensible Markup Language

Chapter 1

Introduction

1.1 Abstract

Provenance is important throughout Library and Information Science and is particularly important for the information infrastructures which support the computational aspects of the natural sciences. This is highlighted by the prominence of provenance as a plank in the FAIR principles for data stewardship (principle R1.2). While traditionally focused on the history/lineage of physical objects, provenance is now commonly accepted to apply to digital objects such as the results of computation as well as to the recipes for computing; in the case of recipes this prospective provenance is critical for reproducibility. This dissertation begins with background in provenance pertaining to data curation and computational reproducibility. The second part describes attempts to "FAIRify" the reporting and execution of workflows within a domain of natural science for better data stewardship to support data reusability. The next chapters argue that there remains a gap in our ability to fully document provenance as there are more story-telling tenses than just the past (*retrospective*) and future (*prospective*). There is also the *subjunctive* (conditional) and perhaps many others. Supporting new flavors of provenance requires new modeling constructs. The thesis concludes with novel information modeling techniques which exploit reification of sub-class relationships suitable for modeling these many sub-classes of provenance, as well as other domains.

1.2 Dissertation Roadmap

Chapter 2 provides an overview of workflows and provenance in the computational natural sciences focusing on issues of reproducibility in the field of biomolecular NMR. Virtual machines and PRIMAD are discussed as tools to help assess and support computational reproducibility.

Chapters 3 and 4 provide a more formal description of provenance in general as defined by the W3C PROV standard. These two chapters explore various facets of provenance – retrospective and prospective – as well as the various metadata concerns in recording provenance using a standard such as PROV. These concerns include class-level versus instance-level as well as partitioning the important attributes of provenance into the appropriate container – entities, activities and agents.

Chapters 5 and 6 describe the recording of both prospective and retrospective provenance for bioNMR spectroscopy within the NMRbox computing platform using the software tool called CONNJUR Workflow Builder (CWB). CWB was developed over the span of 2009 through 2014 and used an application-specific XML schema to record prospective and retrospective provenance. This allowed for the sharing of workflows within the CWB community but did not adhere to the FAIR principles of supporting provenance documentation using a "broadly applicable language for knowledge representation" (FAIR principle I1).

Chapter 5 describes the refactoring of CWB to use the PREMIS standard for provenance documentation. PREMIS is also serializable as XML and is supported by the Library of Congress primarily for digital preservation[1]. PREMIS version 3.0 allows for domain-specific extensions to the top-level provenance objects. In this chapter, CONNJUR-ML is described, which along with PREMIS 3.0 provides for a broader metadata standard for documenting provenance. The PREMIS record is included with the final computational output of a CWB workflow through a "zipped" container file containing the binary data and the PREMIS metadata.

Chapter 6 extends this work by including analytics on the intermediate data through the execution of the workflows. Typically, bioNMR spectra are reconstructed using multistep workflows (10-20 steps) throughout which the intermediate data are discarded. While the final data is of primary importance, it is also useful to make measurements on certain properties of the data at each stage along the workflow for quality assessment and comparison of various workflows. PREMIS/CONNJUR-ML supports such analytics in accordance with FAIR.

While standards such as PREMIS and PROV are capable of recording many aspects of retrospective and prospective provenance, there are still important aspects of provenance which seem to be either overlooked or beyond a simple capture with these standards. The "marrying" of prospective and retrospective provenance represents a case of this in which various bridge models have been proposed (PROV-ONE, P-Plan, OPM-W). However, retrospective (past) and prospective (future) only represent two tenses of story-telling which may be important for provenance documentation. The next two chapters present use cases and argumentation for the need for at least a third provenance modality, subjunctive provenance, which is needed for documenting conditional provenance – the provenance of computational results that could have come to be but may or may not have been realized.

Chapter 7 introduces the need for subjunctive provenance through a fictionalized use case. It also begins to more forcibly argue that provenance is not inherent to objects but rather is a story told about an object. The accuracy, persuasiveness and trustworthiness of that story may be hindered by our consideration of only the past and future without the conditional tense.

Chapter 8 continues the discussion to include consideration of *identity* in provenance records, such as when the same thing is both an agent and object of the provenance documentation and the thorny issues of when a thing referred to throughout a provenance story is substantively changing throughout the story timeline.

Chapter 9 discusses information models which differ by the reification of concepts within the model. This is a lead-in chapter to the following chapter on Concept Keys which exploits reification to solve a problem with the proliferation of sub-classes found in multi-parent hierarchies (also referred to as specialization lattices).

Chapter 10 discusses an old problem in object-relational mapping concerning the representation of sub-classes (sub-types) in a relational database system. While there are four standard methods of treating sub-classes, it is argued in this chapter that they can become unwieldy for large specialization lattices (which in object-oriented programming leads to multiple inheritance). A fifth method is introduced - termed Concept Keys - which is an attempt to provide a more flexible manner for managing multi-parent hierarchical information models. It is argued that Concept Keys provide a "normal form" which enriches the existing standards (W3C PROV and PREMIS) to allow for expressing the various temporal aspects of prospective, retrospective, subjunctive provenance and beyond.

Finally, Chapter 11 provides a summary of this dissertation along with limitations and future directions.

1.3 Chapters as Manuscripts

1.3.1 Chapter 2: Workflows and Provenance

Chapter 2 is an article written for a special issue of *Library Trends* which focused on the doctoral research occurring at the Illinois iSchool circa 2016. This article was co-authored with my advisor, Bertram Ludäscher. I did the vast majority of the writing. The chapter uses the PRIMAD model (co-developed by Ludäscher [2]) to explore different dimensions of reproducibility.

This chapter provides an overview of workflows and provenance in the computational natural sciences focusing on issues of reproducibility in the field of biomolecular NMR. A brief survey of workflows and workflow thinking is provided, followed by a discussion of the distinction between retrospective and prospective provenance and the connection between them and workflows.

Next the chapter discusses the development of a cloud-based, virtual research environment for bioNMR called NMRbox which provides virtual machines provisioned with hundreds of software tools for bioNMR computation. Some of the tools support workflow management and execution and in combination with the VMs themselves, provide an opportunity to explore the various aspects of reproducibility articulated in the PRIMAD model. The role of this chapter in the thesis is for background and context.

Gryk, Michael R. and Bertram Ludäscher. "Workflows and Provenance: Toward Information Science Solutions for the Natural Sciences." *Library Trends*, **vol. 65** no. 4, 2017, pp. 555-562. Project MUSE, https://doi.org/10.1353/lib.2017.0018.

1.3.2 Chapters 3 and 4: w3c PROV

Chapters 3 and 4 correspond to Book Chapters 2 and 3 of *Documenting the Future: Navigating Provenance Metadata Standards* which was co-authored by Rhiannon Bettivia and Jessica Cheng. This book was solicited by series editor Gary Marchionini after he learned of our series of workshops on provenance metadata standards. Rhiannon, Jessica and I offered workshops at the International Digital Curation Conference, ASIS&T and iConference from February of 2020 through March of 2021. The book is a combination of materials presented at the workshops and what we learned through them. Chapters 3 and 4 were written primarily by me with some editing by Rhiannon and Jessica prior to publication of the book.

Bettivia, R., Cheng, YY., Gryk, M.R. (2022). *Documenting the Future: Navigating Provenance Metadata Standards*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Springer, Cham. https://doi.org/10.1007/978-3-031-18700-1

Chapter 3: Introduction to PROV

This chapter introduces the PROV standard along with the key concepts of provenance. Wine making is used as an example to explore the various concepts important to provenance and how they can be documented using the W3C PROV standard. Throughout these examples, provenance concepts are described using illustrations of the model along with PROV snippets using the PROV-N, PROV-XML and PROV-O notations as well as how they can be encoded using Prov Python.

Key concepts include entities, activities, agents, collections, entity-types, process-types, and locations.

Bettivia, R., Cheng, YY., Gryk, M.R. (2022). Introduction to PROV. In: *Documenting the Future: Navigating Provenance Metadata Standards.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Springer, Cham. https://doi.org/10.1007/978-3-031-18700-1_2

Chapter 4: PROV Advanced Topics

This chapter continues exploring the PROV standard along with more advance provenance concerns. Still using wine making examples, the various relationships (generation, usage, derivation) of provenance are defined and the complexities of the semantics of these relationships is explored. This chapter also discusses issues of class-level descriptions vs. instance level ones, alternates and specialization, and documenting prospective versus retrospective provenance in standard PROV. Finally, the chapter introduces the *Open World* and *Closed World Assumptions* and the ramifications for interpreting provenance documentation with a focus on those who create provenance documents.

Bettivia, R., Cheng, YY., Gryk, M.R. (2022). Introduction to PROV. In: *Documenting the Future: Navigating Provenance Metadata Standards.* Synthesis Lectures on Information Concepts, Retrieval, and Services. Springer, Cham. https://doi.org/10.1007/978-3-031-18700-1_2

The role of these two chapters are to explain the basics of W3C PROV and then to conclude by identifying the shortcomings of the PROV standard for fully documenting provenance. These shortcomings led to the development of other provenance standards such as PROV-ONE and p-Plan; however, it is argued that these solutions also fail to deliver.

1.3.3 Chapters 5 and 6: PREMIS for Provenance Recording in NMR

The work in Chapters 5 and 6 are related to the NMR software tools, CONNJUR Workflow Builder and CONNJUR Spectrum Translator. These two tools are the result of several years of effort (2004-2014) of many colleagues at UCONN Health with NIH support from an R21 (2004-2008) and R01 (2008-2014) grant. The development and use of these applications were published in various journals up until 2015 (see Fenwick, et al. [3]).

Chapter 5: Curating Scientific Workflows

Chapter 5 describes a retooling of CONNJUR Workflow Builder (CWB) to use the PREMIS framework for documenting workflows and reconstructions. (In CONNJUR parlance, a workflow is a prospective recipe for processing data and a reconstruction is a retrospective accounting of the execution.) Prior to this work, workflows and reconstructions were recorded in CWB using an application-specific XML schema.

Recognizing that one of the goals of CWB is to support the FAIR principles within scholarly communication, the topic of this chapter is the refactoring of the XML schema built into CWB from being entirely closed and application-specific to using PREMIS along with domain-specific extensions embedded within the extension points. The PREMIS framework is designed and maintained by the Library of Congress as a mechanism of recording the provenance of digital preservation pipelines. Version 3.0 of PREMIS allows for the embedding of custom, domain-specific schemas within the PREMIS framework via extension points with the Object, Agent and Event entities.

PREMIS provides the general provenance scaffold through the cross-referencing of Objects (NMR datasets), Events (data processing steps) and Agents (instruments, people and software). In PREMIS 3.0, these records are recorded as XML files with XML elements corresponding to each of these top level entities along with cross-references between the various elements which allow for the reporting of a directed acyclical graph.

A final important deliverable of this work is the specification of a file format for processed NMR spectra which allows for the FAIR reporting of provenance. Technically, the format is simply a tar file. However, the specification provides for a particular representation of the NMR spectral data – the file format used by the third party software tool, the Rowland NMR Toolkit – along with a PREMIS record documenting the provenance of the spectral reconstruction. As it is simply a tar file, the so-called NBX format is completely extensible and it is envisioned that other data can be included and of other schema types, such as the NMR-STAR format used by the BioMagResBank. Bundling provenance converts the NMR data object into a transparent research object [4].

This chapter was presented at the *13th International Digital Curation Conference* (IDCC) and was published in the International Journal of Digital Curation. The article was co-authored with Douglas Heintz. Douglas and I both took iSchool courses in Digital Preservation and Metadata, Theory and Practice taught by Rhiannon Bettivia in the summer and fall of 2016 respectively. Our class project for the latter course was to construct CONNJUR-ML and demonstrate how it could be embedded within PREMIS to support the reporting of workflows and reconstructions. Douglas worked primarily on the "rights" portion of PREMIS while I worked in the three provenance related portions: agents, events and objects. A preliminary form of the work was presented as a poster at the 58th Experimental NMR Conference in Pacific Grove, California in 2017. The final paper was presented at the IDCC in Barcelona, Spain in 2018 and published that year.

Heintz, D., & Gryk, M. R. (2018). Curating Scientific Workflows for Biomolecular Nuclear Magnetic Resonance Spectroscopy. *International Journal of Digital Curation*, **13**(1), 286–293. https://doi.org/10.2218/ijdc.v13i1.657.

Chapter 6: Analytics in Workflows

Chapter 6 describes the recording of analytics along the execution of a CWB workflow and embedding these metrics with the CWB-PREMIS record. This work was inspired by the prior work including the keynote address at the IDCC. In the keynote, the speaker mentioned that while traditionally, data curation and data analytics are done separately, there is value in combining the two. With this as inspiration, I decided to add analytics to the standard operation of a spectral reconstruction workflow with CWB.

An additional contribution important to the Information Science community: PREMIS allows for recording software agents and linking them to events (which are typically transformations along a computational workflow). When embedding analytics within the workflow execution, there are now multiple software agents in play: the workflow execution system, the individual software tools which transform the data, and now the software tools which measure the properties of the transient dataset. These software tools can no longer simply be linked to an "object" or an "event" but the tools need to be linked to an object characteristic, namely the measured analytic.

Gerard Weatherby is a co-author on this paper and is a gifted software engineer. Gerard is responsible for building the latest version of CWB (as presented in the Fenwick paper of 2015) and implemented the metrics described in this chapter. One of my contributions was defining and refining the metrics – as is described, the metrics such as signal to noise ratio are proxies for qualities such as sensitivity and there was some refinement required after the first choices. I also expanded CONNJUR-ML to include the reporting of metrics and designed and executed the workflow shown in the paper. In this example, the metrics are used to illustrate and identify which processing steps are improving spectral quality as well as to highlight important implementation differences between the Fourier Transform as implemented within the NMRPipe software tool versus the FT implemented within the Rowland NMR Toolkit.

This chapter was accepted at the 15th International Digital Curation Conference (unfortunately, I was very ill that week and was unable to attend and present) and was published in the International Journal of Digital Curation later that year.

Weatherby, G., & Gryk, M. R. (2020). Embedding Analytics within the Curation of Scientific Workflows. *International Journal of Digital Curation*, **15**(1), 10.2218/ijdc.v15i1.709. https://doi.org/10.2218/ijdc.v15i1.709

1.3.4 Chapters 7 and 8: Subjunctive Provenance

Chapters 7 and 8 are part of a very productive, ongoing collaboration between Rhiannon Bettivia, Jessica Cheng and myself. This collaboration began with us offering provenance metadata workshops at IDCC, ASIS&T and iConference; continued as a book on provenance metadata standards from Springer; led to conference papers at iConference and iPres (these two chapters); continued with panel/forums on provenance at ASIS&T and iPres; and we currently have a planning grant under consideration at IMLS and a second book contracted with Cambridge University Press.

While the other chapters in this thesis are almost entirely my own writing, these chapters are a true collaboration with my co-authors. It would be difficult to tease apart each author's contributions – although some of the sections which refer to the biological sciences can be assumed to be led by me, while others such as those on taxonomy or video games were initiated by my colleagues. The overall trajectory of our exploration of subjunctive provenance is a group effort – and in fact, include the assistance, guidance, and suggestions from other colleagues such as Allen Renear, Bertram Ludäscher, our panelists, paper and grant reviewers, and eventually our forum participants if the IMLS grant is funded.

Chapter 7: Subjunctive Provenance

This chapter introduces the concept of subjunctive provenance as a missing piece of provenance documentation. The paper presents a fictionalized but realistic use case drawing from the television show How I Met Your Mother and the LACK table sold by Ikea furniture.

While partly an introduction / survey into retrospective and prospective provenance, it introduces a fictional case study which pushes at the edges of these concepts and illustrates the need for subjunctive provenance. It also firmly establishes the notion that provenance is not an intrinsic property of an object that can simply be discovered or unearthed. Rather, the argument is that provenance documentation is inevitably story-telling. How accurate, compelling and trustworthy that story is is dependent on provenance documentation and information science practitioners.

Bettivia, R., Cheng, YY., Gryk, M. (2023). What Does Provenance LACK: How Retrospective and

Prospective Met the Subjunctive. In: Information for a Better World: Normality, Virtuality, Physicality, Inclusivity. iConference 2023. *Lecture Notes in Computer Science*, vol **13972**. Springer, Cham. https://doi.org/10.1007/978-3-031-28032-0_6

Chapter 8: Provenance as Significant Properties

This chapter discusses issues of identity in managing provenance and particularly for objects which change over time. In these scenarios, the provenance documentation needs to both refer to the changing object as a whole as well as a specific time-point in the lifespan of the object. These temporally distinct referents are reflected in the various tenses in natural language and yet remain very thorny to deal with given the existing provenance standards of PROV and PREMIS. This chapter is also related to Chapter 10 in that it introduces a particularly thorny modeling issue related to biochemical samples. Modeling of biochemical samples is a driving use case for the development of the technique in Chapter 10.

Bettivia, R., Cheng, YY., Gryk, M. (2023). I Got A Letter From My Past Self: (Un)managed Change and Provenance. Accepted for *iPres2023*, Champaign, IL.

1.3.5 Chapter 9: Which Model Does not Belong

This chapter begins a discussion of information models which differ by the reification of concepts within the model. This is a lead-in chapter to the following chapter on Concept Keys which exploits reification to solve a problem with multihierarchical models.

This chapter was presented at a workshop[1] which took place at the virtual 2020 JCDL conference and is posted on the github website. This paper[2] was coauthored by Bertram Ludäscher and myself.

1.3.6 Chapter 10: Concept Keys

Concept Keys is my name for an implementation strategy for modeling specialization lattices. In standard relational database modeling, there are four different mechanisms for dealing with subclasses between entities. These four mechanisms have various pros and cons associated with them depending on the precise scenario they are used.

This chapter describes a situation in which there is a large proliferation of subclasses which form a specialization lattice (i.e. the subclasses derive via multiple inheritance from the superclass(es)). In this particular situation, all of the common approaches to modeling subclasses have weaknesses.

The chapter concludes by describing two novel ways of modeling a specialization lattice which are useful for the case study in the chapter and may also be useful for extending existing provenance models to subjunctive provenance.

1.3.7 Chapter 11: Conclusions

This chapter concludes this dissertation and summaries the motivations of my doctoral research as well as the results and contributions to Information Sciences. This chapter is a reflection of the introductory chapter and also a reflection on the work undertaken over the past decade of my career and studies at the University of Illinois.

[1] https://sig-cm.github.io/news/JCDL-2020-CFP/

[2] https://github.com/sig-cm/JCDL-2020/blob/master/jcdl_20_gryk_ludaescher.pdf

1.4 The FAIR Principles

Most of the work within this dissertation involves documenting provenance as a curation practice to support data reuse and repurposing. One important theme within the dissertation, is that provenance metadata and curating the provenance of digital objects should not be solely the responsibility of librarians and data repository practitioners. Provenance documentation will be more accurate if the data creators also play a role in data curation (see Chapter 6 for example.)

Nevertheless, common community practices as well as funding agency mandates dictate that research data be deposited in domain-specific or generalists data repositories. Data repositories, in turn, have sets of best practices under which they operate. One important set of practices which is an overarching theme to this dissertation is called the FAIR principles.

The FAIR principles were codified by the Force 11 group and published in 2016 [5]–[7]. These principles or guidelines are a set of recommendations for the implementation of scientific data repositories. Gleaned from existing data repositories, these 15 principles are presented under the acronym FAIR - as they are grouped under the broader concepts of ensuring that data are Findable, Accessible, Interoperable and Reusable.

The principles for ensuring that data are findable or discoverable include the suggestion that all data records use persistent, unique identifiers (PUIDs). Persistence is intended to prevent "file not found" errors while uniqueness prevents the need to disambiguate. This principle is great for situations where the searcher knows the identifier for which they search; the second principle is to provide rich and detailed metadata so the data record can be found by search for its properties: subject, author and title, for instance. The third principle is guarantee that the metadata record contains the actual PUID so the metadata and data are linked. The final 'F' principle is to submit the metadata to a searchable index (such as Google) so that the data is discoverable to the broadest possible audience.

Finding a data record is only part of the challenge; it is also important that potential users are capable of Accessing the data record. The next four principles state that a repository should use standard communication and transfer protocols (such as http or ftp) that include authentication protocols if necessary. The latter implies that the data itself need not be free and open in order to be FAIR, but that anyone with appropriate privilege to retrieve the data is able to access it. Finally, the fourth 'A' principle is that the metadata of a record should survive in perpetuity even if the data themselves are obsoleted or otherwise inaccessible.

The three Interoperable principles are intended to facilitate the machine actionability of data. For this purpose, the data and metadata should be represented using a broadly applicable knowledge representation language such as one using the Resource Description Framework (RDF). Following the vision of the semantic web, all data and metadata should provide links to other relevant data and metadata. In addition, it is suggested that the knowledge representation language and metadata standards should be treated the same as research data and also follow the FAIR guidelines.

The final four principles are to facilitate data Reuse or repurposing. These four principles cover different aspects important to reusing data but bundled under the unifying concept of including detailed metadata to support reuse (R1). One issue is the permission to reuse the data and R1.1 suggests including license information with the data. Another issue for reuse is understanding the meaning of the data. While the appropriate details of the data semantics is presumed to be left to the relevant scientific communities, whatever vital metadata is deemed important to those domain standards should be included with the data (R1.3). Finally, the other remaining principle for fostering data reuse is to include the detailed provenance of the data - which is the theme of this dissertation. While principle R1.2 does not elaborate on precisely what "detailed provenance" means, the go-fair website[3] mentions that this includes origin stories [8] as well as processing workflows (Chapters 2, 5, and 6).

1.5 Contributions to Information Science

There are many contributions to Information Science throughout this dissertation. They fall within a few broad themes which are described below. They are also itemized in Table 1.1.

1.5.1 Existing Standards

There are several existing standards for documenting provenance; ones which are explicitly discussed within this dissertation are the PROV model from W3C, the ProvOne and p-Plan extensions to W3C PROV, and PREMIS from the Library of Congress. A few contributions in terms of the existing standards include a deep-dive into the details of these standards and how well or poorly they are at modeling certain provenance stories (Chapters 3 & 4). Particularly for the cases of bridging prospective and retrospective provenance, the ProvONE and p-Plan extensions are critiqued. These critiques are enlightened by consideration of subjunctive provenance which is the topic of Chapters 7 & 8.

1.5.2 PREMIS for Workflows and Research Objects

PREMIS has been used extensively in the digital preservation community for documenting the preservation workflows. In order to support the FAIR principles for documenting both prospective and retrospective provenance, PREMIS has been used to capture provenance within a domain-specific workflow management system - CONNJUR Workflow Builder (Chapter 5). This required refactoring the existing CWB system and creating a domain-specific XML standard which is embedded within PREMIS. In addition to traditional provenance information, the PREMIS records were extended to include analytics during workflow execution (Chapter 6). This was more complicated than originally expected as a shortcoming of PREMIS is that software tools need to be linked more granularly than PREMIS allowed. Together, these developments allowed for the bundling of provenance with data to support Transparent Research Objects such as those described by McPhillips, *et al.*[4].

1.5.3 Subjunctive Provenance

Traditionally, provenance refers to the history or origins of objects. The provenance of artworks, for instance, is important for establishing its authenticity and valuation. Two decades ago, the term prospective provenance was introduced to refer to future-looking recipes on how to create something or execute a task. This introduction of a new term has led to dozens of research articles exploring this provenance concept which is not the same as the traditional, retrospective provenance, and yet shares commonalities.

In this thesis and the work I have done with my colleagues Rhiannon Bettivia and Jessica Cheng, we have introduced another qualifier for provenance: **subjunctive** provenance. While still a relatively new concept, introducing the term allows us to converse about another possible type of provenance. Things that can be or things that could have been but are not.

[3] https://www.go-fair.org/fair-principles/r1-2-metadata-associated-detailed-provenance/

This is an important contribution as before the term prospective provenance had been overloaded with both the concepts of what will be as well as what could be or what ought to be. By introducing a new term, it is now possible to start to fine-tune our thinking about this similar but distinct concepts. There are potentially more than two temporal tenses to provenance.

Explicitly defining what ought to be as opposed to what will be or what has been provides an opportunity for comparing these various workflow graphs for quality assessment and error detection. As discussed at the end of Chapter 7, this also can help document provenance during the execution of iterative workflows - computational scenarios where retrospective, prospective and subjunctive provenance are manifested together.

1.5.4 Reification and Concept Keys

Information (data) modeling is about making choices. The final section of this dissertation explores different reification schemes for modeling problem spaces. In the final chapter, I identify a problem of existing relational modeling techniques for specialization lattices with many layers of subclasses. Next, I introduce Concept Keys as a novel "normal form" to tackle this problem. By altering the reification of the model we can inter-convert between needing hundreds of subclasses to a half dozen concepts which discriminate the subclasses. As a contribution to modeling and documenting provenance, the Concept Key proposal can be a general solution for merging prospective, retrospective, subjunctive and other temporal forms of provenance documentation.

Part I

Workflows and Provenance

These three chapters provide an introduction to workflows and provenance, both from a perspective of provenance metadata required for reproducibility and trustworthiness as well as an in-depth description of the PROV metadata standard supported by the World Wide Web (W3C) Consortium.

Chapter 2 provides an overview of workflows and provenance in the computational natural sciences focusing on issues of reproducibility in the field of biomolecular NMR. Virtual machines and PRIMAD are discussed as tools to help assess and support computational reproducibility. This chapter was published in *Library Trends* [9].

Chapters 3 and 4 provide a more formal description of provenance in general as defined by the W3C PROV standard. These two chapters explore various facets of provenance – retrospective and prospective – as well as the various metadata concerns in recording provenance using a standard such as PROV. These concerns include class-level versus instance-level as well as partitioning the important attributes of provenance into the appropriate container – entities, activities and agents. These two chapters were included in the book *Documenting the Future: Navigating Provenance Metadata Standards* published in 2022 [8].

These chapters represent the following contributions to the field of Library and Information Science:

- Deep-dive into W3C PROV and the complexities in recording provenance.
- Identifying weakness of the PROV and PREMIS models
- Critiquing the PROV-ONE and p-Plan extensions

Chapter 2

Workflows and Provenance

2.1 Abstract

The era of big data and ubiquitous computation has brought with it concerns about ensuring reproducibility in this new research environment. It is easy to assume computational methods self-document by their very nature of being exact, deterministic processes. However, similar to laboratory experiments, ensuring reproducibility in the computational realm requires the documentation of both the protocols used (workflows) as well as a detailed description of the computational environment: algorithms, implementations, software environments as well as the data ingested and execution logs of the computation. These two aspects of computational reproducibility (workflows and execution details) are discussed in the context of biomolecular Nuclear Magnetic Resonance spectroscopy (bioNMR) as well as the PRIMAD model for computational reproducibility.

2.2 Introduction

The era of big data is upon us. Along with it, computers and computation have become ubiquitous in almost every human endeavor. It should come as no surprise that concerns have been raised about the reproducibility of computational methods in research and science [10]. Reproducibility is a cornerstone of the scientific method – addressing both the universality of the reported scientific claims and providing transparency such that the scientific results can be trusted. In general terms, a process can be reproduced if both what was done and how it was done are sufficiently documented. It is often beneficial to record who conducted the process as well as when it was done, but a truly reproducible process is independent of either of them. What then are the requirements for sufficient documentation and how do those requirements translate to computation?

Method sections in the natural sciences typically have two components: the protocol used and a detailed description of the reagents, equipment and calibrations. It is easy to assume computational methods self-document by their very nature of being exact, deterministic processes, whose outcomes are dictated by “the program”. However, leaving aside non-deterministic processes for the moment, by burying the computation within a software tool, it can be very difficult to reproduce the exact process without a detailed record of the software tools used, their configuration and their execution. This problem is amplified with each software tool added to the process stream. As emphasized by Stodden, et al. (page 1240)[10], “We thus focus on the ability to rerun the same computational steps on the same data the original authors used as a minimum dissemination standard, which includes workflow information that explains what raw data and intermediate results are input to which computations."

Mirroring the situation with laboratory experiments, to ensure reproducibility in the computational realm requires the documentation of both the protocols used (workflows) as well as a precise description of the computational environment: algorithms, implementations, software environments as well as the data ingested and execution logs of the computation. These two aspects of computational reproducibility (workflows and execution details) are discussed in the context of biomolecular Nuclear Magnetic Resonance spectroscopy (bioNMR) as well as the PRIMAD model for computational reproducibility [2].

2.3 Workflows and Provenance

A workflow is a model for complex processes in which the process is decomposed into discrete, sequential operations. Many formalized procedures fit this definition, culinary recipes for instance, as well as administrative workflows such as those exhibited by "the culture of the checklist" in the field of archival science [11]. The rationales for representing processes as workflows can be as varied as the processes themselves. The goal of recording a cooking recipe may be to assist in the organization and timing of the interdependent tasks, as well as to increase the reproducibility of the end product by applying uniform measurements of ingredients, cooking duration, and temperature. The goal of a TRAC checklist is to improve and document the quality of a digital archive, increasing its trustworthiness to the community to which it serves.

Scientific workflows are useful for a similarly diverse array of purposes. However, scientific workflows have an important distinguishing characteristic from business workflows. Typical business workflows focus on sub-processes and their validity within the context of an organizational infrastructure. (For instance, documenting proper oversight and approval for a requisition request). As pointed out by Bowers and Ludäscher (2005) [12], scientific computational workflows are distinguished from administrative workflows in their focus on data and dataflow [13] . The emphasis on data has two important aspects: one, managing the timing of a workflow (one step in a workflow may not be able to begin until a preceding step has completed), and two, managing the semantic data types used within the workflow (a process which requires oranges as input should not receive apples).

Apart from this simplistic description of the design and operation of scientific workflow systems, there are two significantly different perspectives from which one can consider the workflow itself. At a detailed or instance level, and in retrospective, it can be viewed as an execution log, capturing the exact sequence of events which occurred and their relationships with respect to each other (both in order and in data typing). Yet at a more abstract level, the workflow is disjoint from any particular execution event.

There are many such flavors of such an abstract workflow. It might still be detailed as in the former case but neglecting specifics of the precise execution, as in, computational step A of type α followed by computational step B of type β, without recording timestamps or execution details of the individual computations. In a more abstract case, it might be a broad sketch of general processing chunks: ingestion, cleaning, transformation, visualization, and result reporting, with very little detail on the underlying computation at all. Finally, the workflow may represent the idea for a future process or protocol which has not yet been executed – or it may even be the case that applications for conducting the individual steps in the putative workflow do not yet exist. At this abstract level, the workflow is more similar to a cooking recipe and less like a stack trace.

An important distinction between the former and latter workflow types is not just the level of abstraction, but this consequence that an abstract workflow is capable of describing events which have not yet occurred.

This is the inherent distinction between retrospective provenance (a representation of prior workflow execution) and prospective provenance (a description of how to execute a future workflow) (Lim et al., 2010). It can also be thought of as the distinction between what was done and what is intended to be done.

This underpins the importance of workflows and provenance in documenting how data is processed. These provenance stories can be truly retrospective and about past events as in an execution log, or they can be prospective and describe a process for future events and objects. Yet our language for storytelling has more than just two tenses, for past or future events. There are various other tenses and moods which are important for human discourse. In the context of provenance, there is considering prospective provenance retrospectively - or describing what was intended to happen. This is the perspective behind the "plan" in the provenance ontology of the World Wide Web consortium[14] (discussed in Chapter 4.) In Chapters 7 and 9, the term *subjunctive provenance* is introduced to distinguish pure, future-looking prospective provenance (what will happen or what is intended to happen) from hypothetical considerations of what could happen or what might happen. Finally, in addition to these many subtleties of provenance language, there are the notions of what should happen or what ought to happen. Those are of particular importance for protocols in which authorities recommend or mandate a particular method due to its reliability at achieving a particular outcome.

Despite many similarities, these two different workflow or provenance "worlds" are vastly different, both in their conceptualization of a workflow as well as with the underlying tools and approaches for managing workflows. Workflow management systems designed to operate at the execution level concentrate on the details of tool operation and interoperability. Systems such as Kepler [15] or HTCondor [16] must ensure that data of the correct type is being shuttled between individual actors (Kepler) or jobs (HTCondor). These tools may also manage the invocation and resource allocation of the individual jobs and check for completion and/or any errors.

Another approach to capture retrospective provenance is that of noWorkflow [17]. In this approach, the provenance is recorded from the execution of a standard processing script (e.g. Python), avoiding the learning curve and overhead of a specialized workflow system such as HTCondor. Prior to execution, noWorkflow parses the script and maps the dependencies between code blocks. Upon execution, noWorkflow relies on built-in Python utilities to extract the provenance and map the dataflow.

These former methods all have as common an interest in the actual execution – describing an event which occurred. The notion of "workflow thinking" is to pattern a process as a workflow regardless of the manner of its execution or whether it has been executed at all. Workflow thinking is more about conceptualizing processes as recipes and protocols, structured as dataflow graphs with computational steps, and subsequently developing tools and approaches for formalizing, analyzing and communicating these process descriptions. An important example of one approach is the YesWorkflow annotation and query system [18]. YesWorkflow provides a few simple syntactical annotations which can be embedded within code – or within a stand-alone file. These annotations describe the flow of data through the various processes such that many of the aspects of a workflow can be visualized, queried and understood, in the absence of any execution events.

2.4 Reproducibility in biomolecular NMR spectroscopy: NMRbox

The National Center for NMR Data Processing and Analysis is a recent initiative to help foster reproducibility in the field of biomolecular NMR spectroscopy. The Center has three overlapping research directions. First, the Center is provisioning virtual machines (VMs) with most of the common software tools used by bio-NMR

[19]. Provisioning VMs with the software helps ensure that both the software and underlying computing environment will persist into the future. Second, the Center is modeling and capturing the metadata required to replicate the computational workflow of a bio-NMR study. Third, the Center is providing Bayesian inference modules for consistent analysis of bio-NMR data. The research developments and directions can be examined within the context of the PRIMAD model for computational reproducibility.

PRIMAD is an acronym for six key variables of a computational system which must be controlled for reproducibility. Platform (P) refers to the entire computational environment of the underlying software tool. This contains the computer hardware as well as the operating system and any ancillary software components such as shared libraries. Research objective (R) refers to the scientific goal of the research – what hypothesis is being tested or what claim is being supported or refuted. Implementation (I) refers to the actual software code by which a particular Method (M) is being invoked. Method refers to the computational approach taken, e.g., for ordering a list or pruning outliers. Actors (A) refers to the human agents who conduct the experiment. Data (D) refers to the datasets under analysis during the computational study. The report from the working group outline this model in the context of a few examples of computation (bubble sorts and statistical analysis). The research endeavors of the Center for NMR Data Processing and Analysis will be examined in the context of this model for computational reproducibility.

2.4.1 Platform (P)

As discussed by Rauber *et al.* [2], computational results which are independent on platform are considered to be portable as well as reproducible. The field of bioNMR relies on dozens of software tools, most of which were developed in academic labs, and rely on antiquated operating systems, compilers and code libraries. A consequence of this is that most bioNMR studies are not portable. To address this issue, the Center for NMR Data Processing and Analysis is (a) provisioning VM's with all available bioNMR software, (b) maintaining a cloud-based Platform as a Service model of accessing these VM's, and (c) is in the process of establishing an archive of the various versions of the NMRbox VMs.

2.4.2 Research Objective (R)

Following the PRIMAD model, for a process to be considered reproducible, the research objective of the replicate process must be the same. This can be the most complicated barrier to reproducibility, for instance if two research groups do not agree on the overall purpose of the research, or if a subtle difference in objective is not fully explained. While this is difficult to address computationally, this is being addressed by the administrative structure of the Center. The research developments are driven by a so-called "push-pull" relationship with external investigators conducting research on "Driving Biological Projects". By focusing technological developments on established external research projects, these external DBPs will assist ensuring that the research objects are agreed upon by the various biomedical communities.

2.4.3 Methods / Algorithms (M)

NMRbox aims to include two hundred software packages used by the bioNMR community. There is a great deal of overlap in the functionality of this software smorgasbord. For instance, there are perhaps a dozen software tools capable of spectral reconstruction – the process of converting time domain data to frequency domain. While there is a great deal of overlap between the various packages, there are methods and algorithms which are unique to a given tool: for instance, the maximum entropy reconstruction algorithm within the

Rowland NMR toolkit. Maintaining all of the various software packages within one common VM aids in evaluating reproducibility, as the platform dependence inherent to any computational tool is eliminated.

2.4.4 Implementation / Source Code (I)

In some sense, it can be difficult to draw a decisive line between methods, algorithms, implementation and source code. For the purposes of this case study, we will assume that methods / algorithms are in an abstract sense (as in the Dagstuhl report by Rauber *et al.* of "bubble sort" vs "quick sort") while implementation and source code contains the possibility of performance tweaks and or bugs / side effects. As such, the implementation would contain the various versions of an implementation. This is also being addressed by NMRbox in that the various VM versions will also maintain a registry of the various software tool versions contained within each. Therefore, questions of reproducibility regarding a particular implementation of a method can be explored within the NMRbox VMs.

2.4.5 Actors (A)

In the context of the PRIMAD model, actors refer to human agents. Computational agents are considered to be a combination of methods and implementation (in the overall context of a platform). The role of actors in reproducibility is addressed in part by capturing annotations of human agents when performing manual analysis. An example of such an annotation strategy is that of the reproducibility extensions to the program Sparky [20]. In this example, the Sparky program was augmented with a few routines which (a) assist in version control of the assignment process using Git, and (b) provide a helpful conceptual model for NMR peak assignment to assist in providing meaningful snapshots along the assignment process. Ongoing research of the Center is to continue to expand the set of captured metadata along a bioNMR study to further foster reproducibility.

2.4.6 Data (D)

The final variable for computational reproducibility is the data sets used in the study. In the context of NMRbox, this is being addressed through the partnership with the BioMagResBank (BMRB) [21] hosted by the University of Wisconsin, Madison. The BMRB has been the national repository for bioNMR data for the past several decades. One additional goal of the Center is to assist in research reproducibility by tracking additional data/metadata within the virtual machine and providing additional software tools to assist the researcher in reporting this data/metadata to the BMRB. Thus, the BMRB will have richer data depositions ensuring that all of the data required for reproducing a study are made available to the community at large.

Workflows and Provenance within NMRbox Along with provisioning the standard bio-NMR software, NMRbox will also include utilities and resources to manage workflows and provenance. A workflow management system for bioNMR spectral reconstruction has already been developed [3]. Termed CONNJUR Workflow Builder (CWB), the tool allows the NMR spectroscopist to craft a spectral reconstruction process as a workflow utilizing any of three software tools: NMRPipe, the Rowland NMR Toolkit, and CONNJUR Spectrum Translator utilities. CWB stores the spectral metadata along with the reconstructions (workflow executions) within a MySQL database. Other workflow management systems such as HTCondor and YesWorkflow are also supported within NMRbox.

2.5 Conclusions

“Workflow thinking” can be a beneficial way of conceptualizing a computational process. By documenting the computational process as a workflow the computation is more transparent and more easily reproduced. When combined with retrospective provenance information, additional value can be derived from a workflow (Pimentel et al. 2016). The PRIMAD model describes additional variables which can be controlled to investigate the universality of the computational process. The new Center for NMR Data Processing and Analysis, while predating the Dagstuhl working group, provides a good case study for how these variable can be documented and controlled in the laboratories of natural scientists.

Chapter 3

Introduction to PROV

3.1 Abstract

We introduce the PROV family of standards and key ideas and concepts relating to PROV. The chapter describes PROV and scenarios in which a reader might use it. We introduce the core components in PROV, entities, activities, and agents, and we create simple diagrams and Python snippets using simplified wine making processes as an example.

3.2 A Provenance Story

Provenance is a description of how something has come to be. Right now you are reading my dissertation, perhaps relaxing on your couch. Think back to how this situation came to be. Perhaps you took the bus to your local library, searched for this book in the card catalog, jotted down the call number, found the book on a particular shelf, checked it out with the librarian, and finally returned home. This would be a description of provenance; the provenance of how you came to find yourself relaxing on your couch reading about provenance.

Let's explore this scenario a bit more carefully and ask ourselves a few questions about what we consider to be **provenance**. What is the underlying structure of this description of provenance, and what are its core components? The structure is a series of steps or events, some components which led to a particular outcome, in this case, our hypothetical reader enjoying a book. This leads to other questions. Can all provenance be described as a linear sequence of steps? *Should* provenance be described as a linear sequence of steps, even if it can be? Those are questions to keep in mind as we think more deeply about provenance.

Our provenance example also has physical things described in it, in addition to the steps. A bus, a library, a card catalog, a shelf and of course, this dissertation. It seems that objects are also core components of provenance. Finally, there is another class of physical things in this example: people who cause the sequence of events to occur. Those would include the bus driver, the librarian and our hypothetical reader.

These three concepts: *events*, *objects* and *agents* will be the foundation for any description of provenance we attempt. The goal of this chapter, and the goal of the PROV standard, is to be more precise in what we mean by these concepts, expand upon them as necessary, and explore a vocabulary for describing how objects, events, and agents interact in narrating how something has come to be.

3.3 What is PROV?

PROV is a standard for representing provenance that was introduced by the World Wide Web Consortium (W3C) in 2013 and reviewed by [22]. It is more appropriate to describe PROV as a set of standards, as the PROV initiative supports and maintains a data model along with multiple serializations for XML, RDF, OWL, and other uses. This reflects on the mission of the W3C in promoting interoperability on the web: the PROV standard(s) are meant to provide human readable but machine actionable representations of provenance.

The introduction to the PROV-Overview [23] illustrates the various components of the PROV standards and their relationship to the data model (PROV-DM) [14]. In this and the following chapter, we will concentrate on the content provided in Table 3.1. This includes PROV-DM as well as the three serializations: PROV-N (PROV notation), PROV-XML, and PROV-O (Ontology, in OWL2 format). There is also PROV-DC which is an effort to map the PROV standard onto the Dublin Core terms. The reader is encouraged to make use of the valuable resources provided by the W3C; their documentation is extensive and clear. The reader is also referred to the wonderful book by Moreau and Groth [24].

Table 3.1: Online resources available for PROV

Information	**Webpage**
Wikipedia	https://en.wikipedia.org/wiki/PROV_(Provenance)
Overview	http://www.w3.org/TR/2013/NOTE-prov-overview-20130430/
Data Model	http://www.w3.org/TR/2013/REC-prov-dm-20130430/
Notation	http://www.w3.org/TR/2013/REC-prov-n-20130430/
Ontology	http://www.w3.org/TR/2013/REC-prov-o-20130430/
XML	http://www.w3.org/TR/2013/NOTE-prov-xml-20130430/

3.4 Provenance with PROV

The PROV-Overview [23] describes provenance in this way:

> Provenance is information about entities, activities, and people involved in producing a piece of data or thing, which can be used to form assessments about its quality, reliability or trustworthiness.

As we see in this description, the developers of PROV focus on the three core concepts introduced at the beginning of this chapter: entities (objects), activities (events), and people (also referred to as agents).

We also notice in this description that the rationale or purpose for recording provenance is for assessing an item's "quality, reliability or trustworthiness". This aspect of provenance was discussed with respect to artwork in Chapter 1 of *Documenting the Future*[8]. However, quality, reliability, and trustworthiness are important in many contexts, as is provenance.

3.4.1 Making Wine, Making Provenance: The Basic PROV Model

For the remainder of this chapter, we will explore provenance and the PROV standard with the help of a *toy example*, wine making. Let us consider a hypothetical winery, *JeMiRi wines*, which manufactures a broad selection of wines. JeMiRi winery places great value in being open and transparent with its customers on the manufacturing processes of its family of wines. JeMiRi achieves this transparency by attaching the PROV standard to all of its activities. This public provenance serves as a testament to its wine quality.

Definition Toy Example: A toy example is a simple model that purposefully leaves out fine details, used for teaching and explaining. We will follow some basic wine making practices in the following example, but we know there is a lot more involved in making great wine!

Recall from the PROV-Overview that provenance is a bookkeeping of entities, activities, and people involved in the production of something. If our something is wine, what would those entities, activities, and people be?

To begin in the simplest of terms, vinification (wine making) is the process of turning grapes into wine. This provides us with two critical entities for our provenance record: grapes and wine. Our central activity has also been defined, namely vinification. And finally, we have the JeMiRi winery acting as the agent responsible for the vinification.

Figure 3.1 shows the general PROV data model for provenance along with our toy example.

On the left panel, PROV uses recursive relationships: each core concept has an arrow that loops back on itself. While the arrows appear to say that entities are derived from themselves or that agents are acting on behalf of themselves, the model is actually allowing for entities to be derived from other entities, activities to be informed by other activities, and agents acting on behalf of other agents when we add more core components. This is a *class diagram* for PROV. In our wine example on the right, the two entities, grapes and wine, are given distinct symbols and the relationships between the entities, activities, and agents become clearer. This is a *data diagram* using the PROV classes and relationships.

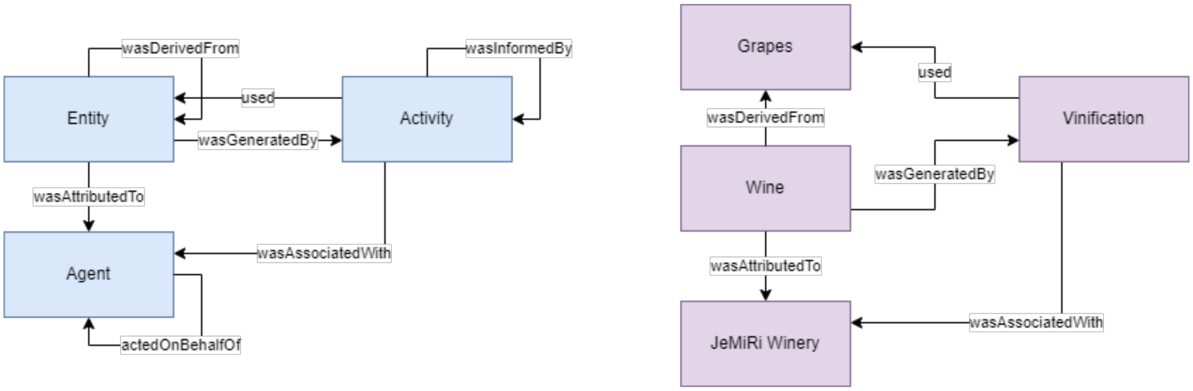


Figure 3.1: Basic PROV model for vinification. The left-hand figure is the class diagram provided as the W3C PROV standard [14]. The right-hand figure is a data view for ascribing provenance to wine making.

3.4.2 PROV-Notation

As mentioned in Section 3.3, the PROV standard supports three serializations of the DM: PROV-N, PROV-XML, and PROV-O. Table 3.2 illustrates how entities, activities and agents are written in the three serializations.

The XML serialization is primarily for machine representations. The OWL representation carries with it the Resource Description Framework which adds extra semantics. Note that all of the examples have PROV-XML and PROV-O serializations provided in the supplementary materials of *Documenting the Future*[1].

[1] https://metaprov.org

Table 3.2: PROV Entity, Activity and Agent in different serializations.

Serialization	**Format**
Prov-N	entity(wine)
PROV-XML	<prov:entity prov:id=“ex:wine”/>
PROV-O	:wine a prov:entity .
Prov-N	activity(vinification)
PROV-XML	<prov:activity prov:id=“ex:vinification”/>
PROV-O	:vinfication a prov:activity .
Prov-N	agent(JeMiRi-Winery)
PROV-XML	<prov:entity prov:id=“ex:JeMiri-Winery”/>
PROV-O	:jemiri-winery a prov:agent .

For the rest of this chapter we will work with PROV-N along with diagrams for understanding the PROV standard. Here, we will go over the basic structure of PROV-N. Simple components like entities, activities, and agents are succinctly described by their type with an identifier as shown in the examples above. Relationship names come from the PROV documentation and are described using a comma separated list of the entities to which the relationship holds. Note that the order in which the entities are listed within the relationship is important. For example:

```
used(vinification , grapes)
wasGeneratedBy(wine, vinification)
wasDerivedFrom(wine, grapes)      // wine was derived from grapes
wasAssociatedWith(vinification , JeMiRi−Winery)
wasAttributedTo(wine, JeMiRi−Winery)
```

In these examples, our identifiers[2] are broad, human-readable names for things along the process: wine, grapes, vinification, JeMiRi-Winery. In real world provenance recording, the identifiers are likely to be numeric-based IDs such as bar codes or ISBNs.

In addition to the identifiers, each of the core components has various attributes which can be associated with it, such as the time an activity occurred or for sub-typing activities, agents, or entities. These will be introduced to our wine example as they become relevant. Let's explore our winery some more.

3.4.3 Composite Entities or Collections

In Section 3.4.1 we defined vinification as the process of turning grapes into wine. Let's expand on that concept a little more. The first step to wine making is to crush the grapes. This produces a liquid (called *must*) which contains both the grape juice as well as the skins, seeds, and stems. The solid material is referred to as *pomace*.

We can add these steps to our vinification diagram as shown in Figure 3.2. PROV does not have an explicit concept for composite entities; however, there is something very suitable called *collections*.

```
entity(must, [ prov:type='prov:Collection' ])
hadMember(must, juice)
hadMember(must, pomace)
entity(grapes)
```

[2] There is another small but important detail about the identifiers. Notice that JeMiRi winery is hyphenated when used in the PROV-N. This is required as part of the formal grammar which makes the notation machine readable.

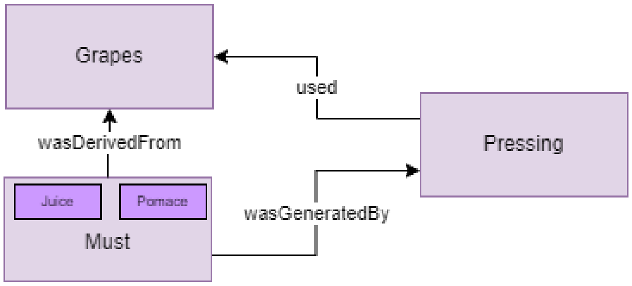


Figure 3.2: Composite Entities.

```
used(pressing, grapes)
wasGeneratedBy(must, pressing)
wasDerivedFrom(must, grapes)
```

This example illustrates a few things. One is the use of the entity attribute to define must as a collection of juice and pomace. Note that in the notation, the namespace of our type definition is PROV. The *prov*:type is of a *prov*:collection. Namespacing allows for the freedom to use types or relationships from other vocabularies / ontologies within a PROV record.

Definition Namespace: Namespaces provide a mechanism for scoping a term. This allows multiple vocabularies to be used together, as namespace1:type can be distinguished from namespace2:type. Namespaces are typically defined through the use of a URL to the schema or vocabulary, along with a custom abbreviation. In our example, prov would point to http://www.w3.org/ns/prov.
If no namespace is provided, the namespace of the parent document is assumed.

Another thing to note in this example is that by describing the must as a collection of juice and pomace, we can now define provenance to either the collection as a whole or using the individual pieces. Consider the following example shown in Figure 3.3 in which we can distinguish a red wine from a white wine depending on whether the pomace was used during fermentation or not.

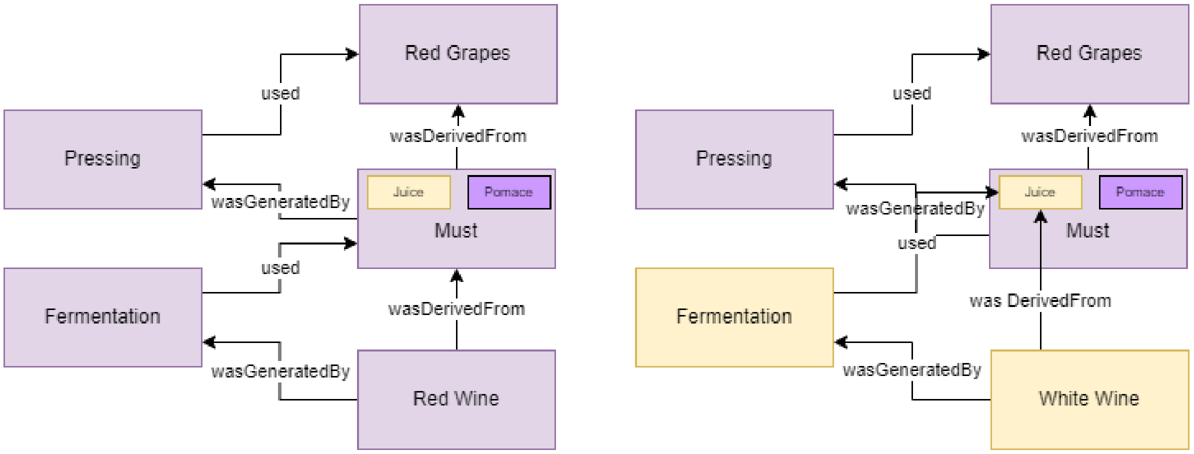


Figure 3.3: Provenance can be linked to a collection (Must in left panel) or a member of a collection (Juice in right panel).

These two scenarios could be written in PROV-N as follows:

```
entity(red-grapes)
activity(pressing)
used(pressing, red-grapes)
entity(juice)
entity(pomace)
entity(must, [ prov:type=‘prov:Collection’ ])
hadMember(must, juice)
hadMember(must, pomace)
wasGeneratedBy(must, pressing)
activity(fermentation-red)
used(fermentation-red, must)
entity(red-wine)
wasGeneratedBy(red-wine, fermentation-red)
activity(fermentation-white)
used(fermentation-white, juice)
entity(white-wine)
wasGeneratedBy(white-wine, fermentation-white)
```

3.4.4 PROV-Notation Revisited

In the previous sections, we visualized provenance with the help of diagrams and the PROV vocabulary. We then described that view using the PROV-Notation. PROV-Notation was developed to support both human-readability in communicating provenance information as well as machine-interpretability. To accomplish these dual goals, PROV-N was designed to have a simple, technology-independent syntax which can both be parsed according to a formal grammar as well as be read by humans.

As for human-readability, the syntax is a bit awkward: it does not reflect natural human language. However, it is certainly a simple enough task to rearrange the words a bit to interpret *wasGeneratedBy(wine, vinification)* as *wine was generated by vinification*. Similarly, for (the collection) must had juice as a member.

Another goal for PROV-N was to support machine-interpretability. Trung Dong Huynh has developed and continues to maintain a Python library for PROV-N [3]. The Python library treats the core components of the PROV data model and PROV-N as Python objects of their respective classes. This allows provenance to be created, manipulated, and queried the same as any other Python code.

A translation of our latest example into Prov Python is shown below:

```
d1 = ProvDocument()
e1 = d1.entity(‘eg:red-grapes’)
e2 = d1.entity(‘eg:juice’)
e3 = d1.entity(‘eg:pomace’)
e4 = d1.collection(‘eg:must’)
e4.hadMember(e2)
e4.hadMember(e3)
e5 = d1.entity(‘eg:red-wine’)
e6 = d1.entity(‘eg:white-wine’)
```

[3] PROV-N Python Library: https://prov.readthedocs.io/en/latest/

```
p1 = d1.activity(‘eg:pressing’)
p2 = d1.activity(‘eg:fermentation-red’)
p3 = d1.activity(‘eg:fermentation-white’)
e4.wasGeneratedBy(p1)
e5.wasGeneratedBy(p2)
e6.wasGeneratedBy(p3)
p1.used(e1)
p2.used(e4)
p3.used(e2)
```

This is just an abbreviated example. A fuller example which is executable as a Python notebook can be found in the supplementary materials of *Documenting the Future*[4].

Let us examine the Python snippet. The notation has been changed from the PROV-N syntax to fit with Python syntax. The developer(s) of Prov Python have mapped the PROV entities to Python objects which are accessed as standard Python objects using the Python ‘.’ conventions.

To accommodate this, there is a top-level object called a ProvDocument in which all the prov components are embedded. For simplicity, each element of provenance is given a unique variable name using the Python prov syntax for entities, activities, collections, etc. Apart from the slight syntactical modifications, it is hoped that the Python version is similar enough to PROV-N that it is not difficult to translate between the two.

The code above simply builds a provenance record (document) within the Python virtual environment. The benefit of having done this is that it can now be manipulated using Python. For instance, a simple command:

```
print(d1.get_provn())
```

outputs the provenance record in the standard PROV-N.

```
document
prefix eg <http://www.example.org/>
entity(eg:red-grapes)
entity(eg:juice)
entity(eg:pomace)
entity(eg:must, [prov:type='prov:Collection'])
hadMember(eg:must, eg:juice)
hadMember(eg:must, eg:pomace)
entity(eg:red-wine)
entity(eg:white-wine)
activity(eg:pressing, -, -)
activity(eg:fermentation-red, -, -)
activity(eg:fermentation-white, -, -)
wasGeneratedBy(eg:must, eg:pressing, -)
wasGeneratedBy(eg:red-wine, eg:fermentation-red, -)
wasGeneratedBy(eg:white-wine, eg:fermentation-white, -)
used(eg:pressing, eg:red-grapes, -)
used(eg:fermentation-red, eg:must, -)
```

[4] https://metaprov.org

```
used(eg:fermentation-white, eg:juice, -)
endDocument
```

The above summary of our provenance record is now written in well-formed PROV-N. Note, that in the above example, we have now provided a namespace for our terms: ‘eg’. The notation also provides dashes to identify those attributes which are optionally provided to the term, such as the start and end time attributes for activity.

Another benefit of this object-oriented representation for PROV-N is that we can build a graph diagram for PROV using an external package ‘GraphVis’.

The following code:

```
# visualize the graph
from prov.dot import prov_to_dot
dot = prov_to_dot(d1)
dot.write_png('article-prov.png')
```

produces the graph in Figure 3.4 shown below.

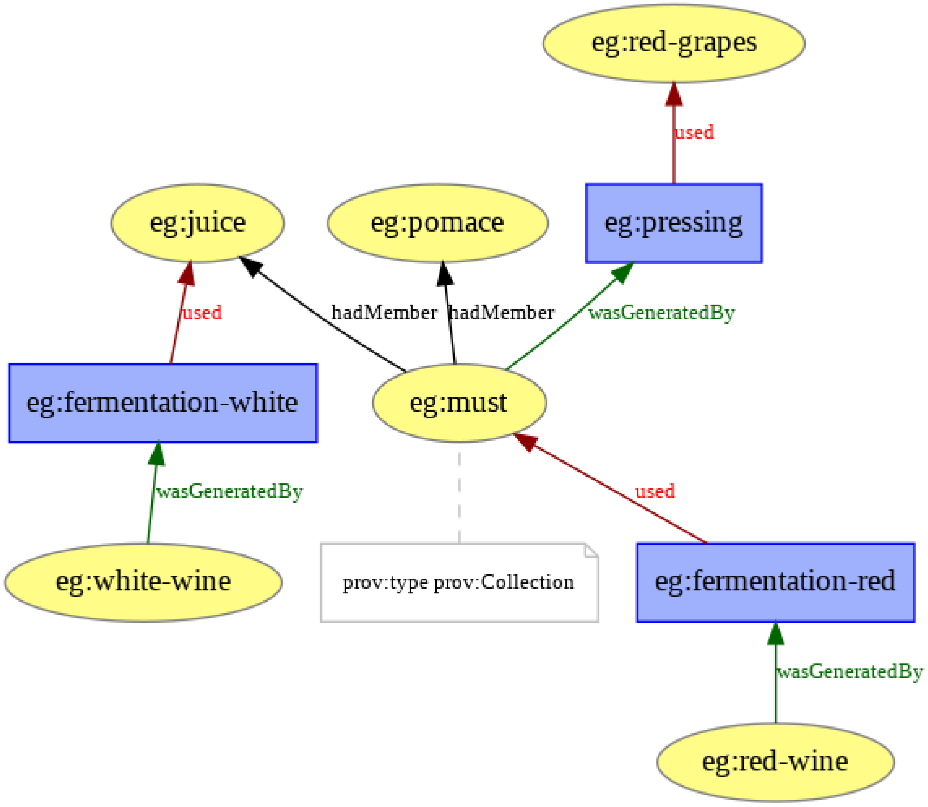


Figure 3.4: GraphVis representation of the Prov Python document.

Convention: Note that in Figure 3.4 entities are represented by yellow ovals and activities by blue rectangles. This is a recommended visualization scheme from the W3C (https://www.w3.org/2011/prov/wiki/Diagrams). There is a third shape for agents: an orange “pentagon house” which is illustrated later in Figures 4.2 and 4.3.

3.5 Core Components

For the remainder of this chapter, let's return to the three core components of PROV, activities, entities, and agents, and explore them in more detail. Particularly, we will focus on the role each of these components play in an object's provenance by considering variations in the entities, agents, or activities.

3.5.1 Entity View

Entity-centric provenance was already discussed in the context of composite entities (PROV collections). The general idea is that a difference in entities along a provenance chain may have different outcomes even if the activities and agents are unchanged. In the composite entity example, whether our fermentation process used the must or the juice changed the type of wine produced. Thus, that portion of the graph was critical for our provenance.

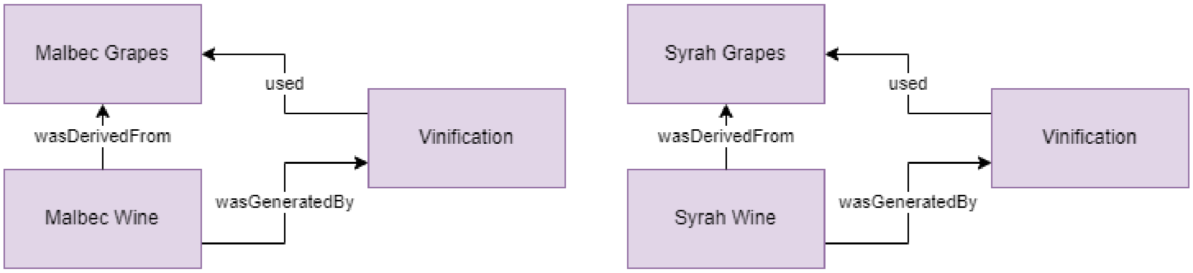


Figure 3.5: Provenance dependent on entity type.

Another similar example is shown in Figure 3.5 where we change the type of grapes used for vinification. When the type of grape changes, so does the type of wine. This can be written succinctly using PROV-N as below:

```
entity(grapes1, [ grape:type='Malbec' ])
entity(grapes2, [ grape:type='Syrah' ])
entity(wine1, [ wine:type='Malbec' ])
entity(wine2, [ wine:type='Syrah' ])
activity(vinification1, [ type='Vinification' ])
activity(vinification2, [ type='Vinification' ])
used(vinification1, grapes1)
used(vinification2, grapes2)
wasGeneratedBy(wine1, vinification1)
wasGeneratedBy(wine2, vinification2)
wasDerivedFrom(wine1, grapes1)
wasDerivedFrom(wine2, grapes2)
```

Notice that in this example, we need different identifiers for each of the entities and activities regarding grapes, wine, and wine-making. This is because we are now distinguishing between particular instances of things and not simply classes of things: not grapes in general, but *grape 1* for *Malbec* and *grape 2* for *Syrah*. This was true in the red wine / white wine example, but we were able to get away with the class names as *must* is a collection while *juice* is a member in the collection. This will be discussed more in the next chapter on Advanced PROV, including how we can use attributes like “type” shown above to group specific provenance instances into provenance classes.

3.5.2 Activity View

The previous section provides an example of how changing the entity in two otherwise identical processes changes the outcome. The same holds for changes in process. In the following example, the type of grapes remain the same, and we do not distinguish between must, juice, or pomace. Rather, the distinction between sparkling wine and a traditional chardonnay arises from a second fermentation process.

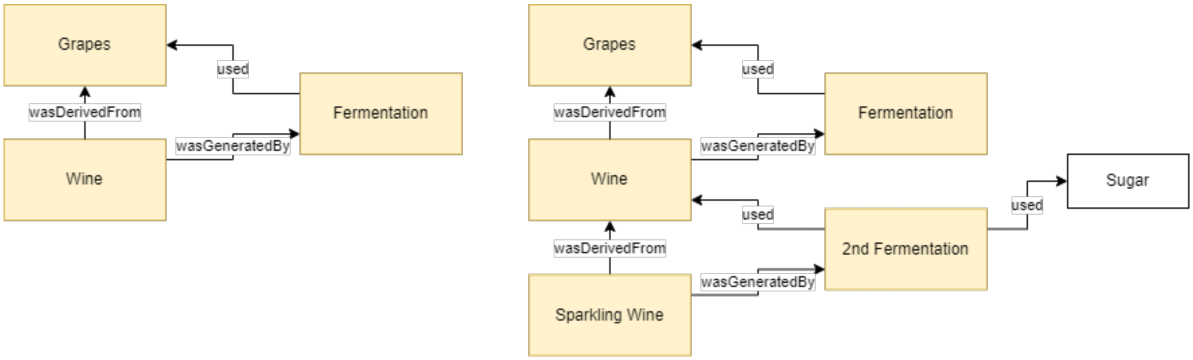


Figure 3.6: Provenance dependent on process type.

```
entity(grapes)
entity(wine)
entity(sparkling-wine)
entity(sugar)
activity(fermentation1, [ type='Fermentation' ])
activity(fermentation2, [ type='Fermentation' ])
used(fermentation1, grapes)
used(fermentation2, wine)
used(fermentation2, sugar)
wasGeneratedBy(wine, fermentation1)
wasGeneratedBy(sparkling-wine, fermentation2)
wasDerivedFrom(wine, grapes)
wasDerivedFrom(sparkling-wine, wine)
```

Another classic wine example where changing an aspect of the process is the difference between Champagne and other sparkling wines which use the méthode champenoise but are not produced in the Champagne region of France. This is illustrated in Figure 3.7 and introduces the optional attribute “loc” applied to the activity.

The PROV-N for this is example is shown below:

```
entity(grapes)
entity(champagne)
entity(sparkling-wine)
activity(methode-champenoise1, [ prov:location=‘‘Napa, California’’ ])
```

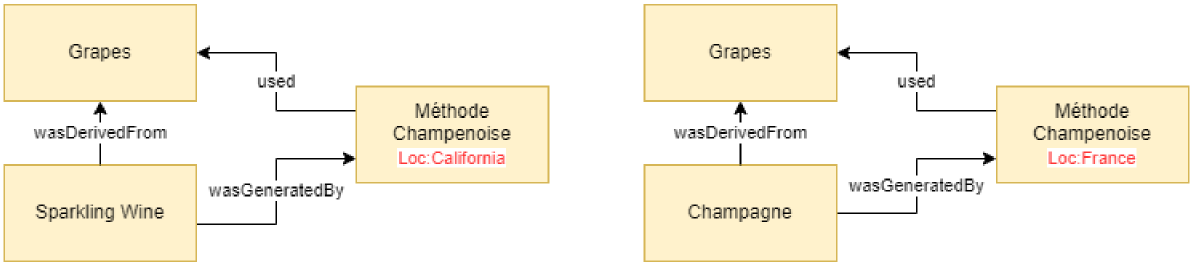


Figure 3.7: Provenance dependent on location.

```
activity(methode-champenoise2, [ prov:location=‘‘Champagne, France’’ ])
used(methode-champenoise1, grapes)
used(methode-champenoise2, grapes)
wasGeneratedBy(sparkling-wine, methode-champenoise1)
wasGeneratedBy(champagne, methode-champenoise2)
```

3.5.3 Agent View

To return to the last of our three core components, agents, the provenance of the final product also depends on the agents along the chain. This is a key component in distinguishing genuine products from counterfeits. Was that watch truly manufactured by Rolex or is it a timepiece which was built from the same parts and using the same process but by a different company?

The importance of agency when considering the provenance of products or merchandise is worth considering further. If the preceding question was actually being asked, would it truly matter if the agent was different as long as the parts (entities) and process (activities) were identical? This is a complicated question as product branding has its own perceived value irrespective of the final product. However, in addition to simple brand recognition, the brand can be used as a judgement of quality in itself. That is to say, a customer may inherently trust the process of manufacturer A over manufactured B. In a sense, the value judgement placed on the brand functions as a proxy for an examination of the provenance to genuinely attest to the quality. Provenance is a narrative whose storyline arises as much from the provenance recorder as from properties inherent to the object.

Lastly, there is another important reason for the recording of agents in provenance. Assuming the provenance record may be used for quality control, identifying key agents along the chain, whether they be people, software, robots, or organizations, can be used in identifying and controlling for mistakes throughout the production process.

3.6 Summary

This chapter covered the three core concepts of provenance – entities, activities and agents – in the context of the PROV standard(s) maintained by the W3C. Those three core components are foundational to provenance and provenance metadata and can also be found in the PREMIS standard (as objects, events and agents).

Chapter 4

Advanced PROV

4.1 Abstract

This chapter introduces more advanced concepts in PROV. We cover relationships between core concepts including generalization, usage, and derivations. Following the wine example from vat to special fermentation process to bottle, the chapter explores the PROV mechanisms of alternates and specializations that allow us to describe the same entities at different levels of abstraction. Bundles and plans lead to the introduction of the concept of prospective provenance, provenance of what can and/or will be in future. The reader is encouraged to consider differences in perspective between the Closed World Assumption (CWA) and Open World Assumption (OWA) when documenting provenance.

4.2 Introduction

In the last chapter, provenance was defined as the way something has come to be. This history or lineage of an object of interest was decomposed into the three core components of the PROV data model: entities, activities, and agents. A wine-making example explored these concepts and their relationships in more detail while illustrating how various serializations of the PROV-DM [14], such as PROV-N, can be used to record the provenance of something. The chapter concluded with some exercises on which aspects of provenance and those core components are useful for verifying or assuring the quality of some common household products.

This chapter will cover three more advanced topics of provenance, once again exploring their use within the W3C PROV family of standards [23]. The first will be the important semantic considerations required for defining *entities*, *activities*, and *agents*. One aspect of this, the distinction between classes and instances, was touched upon in 3.5.1 with regards to the example of distinguishing Malbec wine from Syrah. Keeping track of levels of abstraction as is the case for classes versus instances is one semantic concern; however, there will be many others. For instance, various entities which are related to activities can take on different roles in the activity. It requires care in building the provenance graph to preserve these distinctions. There are many other roles and semantic distinctions for which PROV also has terms that will be explored in this chapter. Once again, the reader is encouraged to explore the wonderful documentation provided by the W3C [23] as well as those by Moreau, *et al.* [22], [24].

The second topic is that of *prospective provenance*. The previous chapter defined provenance as the way something has come to be. That definition and the concepts associated with it are sufficient if we are concerned about the provenance of one specific object. For instance, in the case of works of art, we are interested in the chain of custody of a particular painting in order to assure its authenticity. This particular type of provenance is called *retrospective provenance.*

However, provenance is more expansive than just retrospective analysis. Perhaps we not only want to know how a particular object has come to be, but we also want a recipe for making more. This concept is referred to as prospective provenance. We introduce the distinction between retrospective and prospective provenance here and discuss them further in Chapter 4 and in Chapter 7.

The third topic will be considering the consequences of the *Open World Assumption* (OWA) versus the *Closed World Assumption* (CWA)[1]. With the CWA, it is assumed that everything that is "true" about a system is defined within it. For instance, when querying the employee table of a database, it is assumed that a record for every current employee exists within the database. This allows us to draw conclusions about the absence of items. With the OWA, it is assumed that the information in the system is true; however, there can be many other true facts which are not recorded within the system. When crafting a provenance record, care must be taken to explicitly define what is known to be true as well as what is known to be false, if that information is important for provenance.

This chapter will begin by revisiting the relationships between the three core PROV components and then move on to additional PROV terms and concepts. The three broad topics of semantics, prospective vs. retrospective provenance, and OWA vs. CWA will be highlighted throughout this discussion.

4.3 PROV Relationships

In the preceding chapter we illustrated how the core components are related to each other to produce a provenance chain. Specifically, we used the relationships *used* and *wasGeneratedBy* to connect entities with activities; *wasDerivedFrom* to connect entities with each other; and *wasAttributedTo* and *wasAssociatedWith* to associate agents with entities and activities, respectively.

Each of these terms for relationships has a specific definition provided in the PROV standard. The previous chapter did not specify these formal definitions, as common usage of the words was sufficient. Such human-readability for provenance was one of the design goals of PROV. It is time to consider the definitions from the PROV standard [14], starting with the entities, activities, and agents.

- An **entity** is a physical, digital, conceptual, or other kind of thing with some fixed aspects; entities may be real or imaginary.
- An **activity** is something that occurs over a period of time and acts upon or with entities.
- An **agent** is something that bears some form of responsibility for an activity taking place, for the existence of an entity, or for another agent's activity. An agent may be a particular type of entity or activity. This means that the model can be used to express provenance of the agents themselves.

These definitions allow for the uses described in the preceding chapter, along with the various observations of classes versus instances and concrete objects versus conceptual objects. Agents have not been discussed much up to now; however, the suggested usage has been for people or other groups such as organizations. The PROV standard allows for agents to be responsible for both entities and activities, but a subtle addition

[1] https://en.wikipedia.org/wiki/Closed-world_assumption

also allows for them to *be* either entities or activities [14]. An agent can be an activity, so in principle a process such as a *democratic vote*[2] could be responsible for another activity such as the election of an official.

The previous paragraph defined the PROV classes; the following will define the PROV relationships between these classes. Below are the PROV definitions [14] for *generation*, *usage* and *derivation*.

- **Generation** is the completion of production of a new entity by an activity. This entity did not exist before generation and becomes available for usage after this generation.
- **Usage** is the beginning of utilizing an entity by an activity. Before usage, the activity had not begun to utilize this entity and could not have been affected by the entity.
- A **derivation** is a transformation of an entity into another, an update of an entity resulting in a new one, or the construction of a new entity based on a pre-existing entity.

The definitions of generation and usage should be intuitive and agree with the examples in the preceding chapter. Activities can use entities and they can generate entities. The term derivation, however, requires some consideration. In the wine making example, vinification used grapes to generate wine and therefore wine is considered to be *derived* from grapes. In this context, it would be appropriate to say that the grapes were transformed into wine, or that the wine was based on the pre-existing grapes.

Usage is a more general term than the terms *transformation*, *updating* or *basing*. As an example, consider a more verbose provenance record for wine making in which case we wish to specify that the wine was fermented in a steel vat, and that it required refrigeration which used gasoline as a fuel for the compressor. In this more detailed example, we could create the provenance graph shown in Figure 4.1.

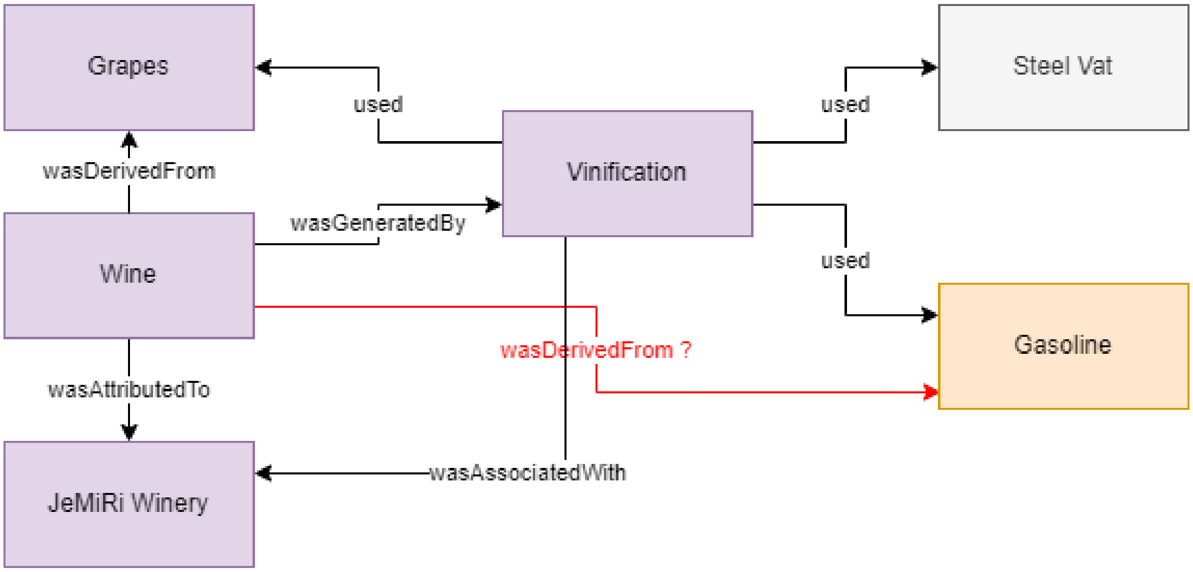


Figure 4.1: Illustration of different semantic roles for *usage*.

In this case, the activity of vinification is shown to have used grapes (as an ingredient for the wine), a steel vat (as equipment), and gasoline (as a fuel source for the activity).

This graph presents an obvious question / dilemma. In the basic PROV examples, the input (used)

[2] As a thought exercise, if one wanted to associate each of the individual voters to the process of the democratic vote, would all of the voters be associated with the final outcome or only the portion which voted for the winner?

entities and output (generated) entities of a single activity stood in a derivation relationship. What about in this case? Is it fair to claim that the wine was derived from a steel vat? Certainly the choice of a steel vat over an oak barrel will affect the wine – but is that effect a derivation relationship? Similarly, is it fair to claim that the wine was derived from gasoline?

Take Home: Derivation cannot be inferred from a usage and generation alone. It must be explicitly stated in the provenance record. Similarly, due to the OWA, failure to specify that a derivation relationship exists does not imply that one entity was not derived from the other. Care must be taken if this type of provenance information is deemed important.

This example can be defined using the following PROV-Notation which results in the provenance graph shown in Figure 4.2.

```
entity(wine)
entity(grapes)
entity(gasoline)
entity(steel-vat)
agent(JeMiRi-Winery)
activity(Vinification)
wasAttributedTo(wine, JeMiRi-Winery)
wasDerivedFrom(wine, grapes)
wasGeneratedBy(wine, Vinification)
used(Vinification, grapes, [ prov:role=‘‘ingredient’’ ])
used(Vinification, gasoline, [ prov:role=‘‘temp-regulation’’ ])
used(Vinification, steel-vat, [ prov:role=‘‘container’’ ])
```

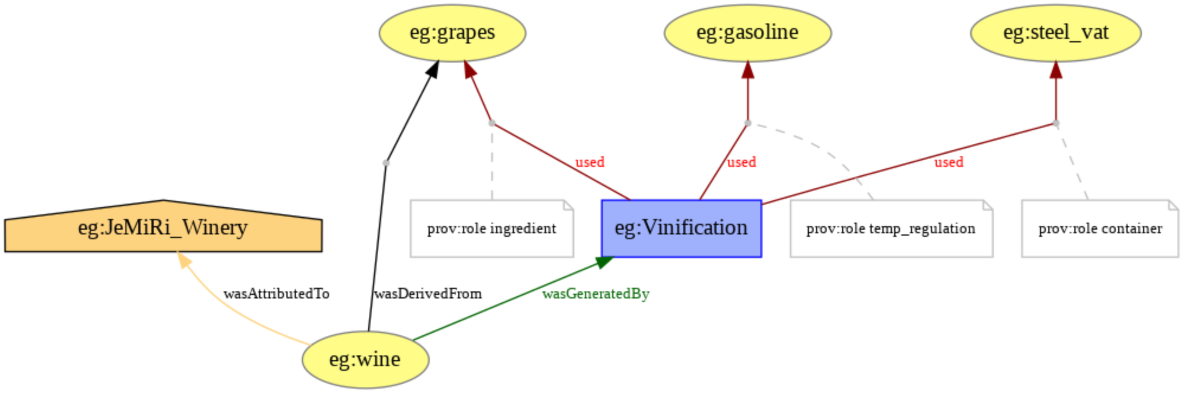


Figure 4.2: Python PROV figure of example of usage.

4.4 Alternate and Specialization

While the wine-making examples of PROV are designed to walk the reader through uses of PROV, the individual examples have jumped through different levels of abstraction. In some cases vinification refers to a general process (Fig 3.3) while in others it refers to a specific process applied to a specific set of ingredients

(Fig 3.3). In other real world cases, it is likely that this will be fine-tuned further such that individual lots or batches of wine would be tracked with suitable IDs and timestamps, and each individual run of the fermenter would also be tracked at a time-stamped instance level.

There is another overlapping issue regarding levels of abstraction: how to allow two different provenance recorders to refer to the same "thing" in a provenance record but at different levels of abstraction. For instance, perhaps at the organizational level, the management is only concerned with the different wine products of JeMiRi winery while at the quality control department, they are concerned about the individual lots and batches. PROV [14] provides two mechanisms to support this: *alternates* and *specializations*.

- Two **alternate** entities present aspects of the same thing. These aspects may be the same or different, and the alternate entities may or may not overlap in time.
- An entity that is a **specialization** of another shares all aspects of the latter, and additionally presents more specific aspects of the same thing as the latter. In particular, the lifetime of the entity being specialized contains that of any specialization.

As is true with much of the PROV standard, these items are designed for flexible use. For example, we can use alternate in the wine making example to distinguish wine that has finished fermenting and is stored in a vat from the same wine after it has been bottled.

```
entity(syrah-vat)
entity(syrah-bottle)
alternateOf(syrah-vat, syrah-bottle)
```

The implication here is that the two entities refer to the same thing, in this case the Syrah wine. The storage vessel is different and this distinction can be captured, when it is important, via the two distinct entities. The alternate relationship maintains that they are two different representations of the same thing. One can go a step further and create a single entity for Syrah which is an alternate of both.

```
entity(syrah)
entity(syrah-vat)
entity(syrah-bottle)
alternateOf(syrah, syrah-vat)
alternateOf(syrah, syrah-bottle)
```

The latter example might benefit from the use of the specialization relationship. Remember that in the specialization relationship, the specialized member has all of the same properties of the general entity, plus additional attributes which serve to sub-class it. In the case of the Syrah example, the specialized entities of syrah-vat and syrah-bottle would share all the same properties of the general Syrah wine, but they would have additional attributes for their storage location. The two specialized entities would be considered alternates of one another. Indeed, one could imagine that the wine is bottled from a vat and subsequently poured back into the vat without change to the wine itself[3].

```
entity(syrah)
entity(syrah-vat)
entity(syrah-bottle)
specializationOf(syrah-vat, syrah)
```

[3] At least with respect to this being a *toy* example!

```
specializationOf(syrah−bottle, syrah)
alternateOf(syrah−vat, syrah−bottle)
```

4.5 Provenance Levels

Provenance is a description of how something came to be. The previous chapter began with a hypothetical example of how you came to be reading this book. In that provenance description, a book was only a single entity in the larger provenance tale which contained a library, a bus, and other sundries. One could ask not only how you came to be reading the book, but also, how did the book come to be in the first place? This would require digging deeper into the provenance of a single entity within the larger provenance record. If such “book” provenance exists, it might look something like this:

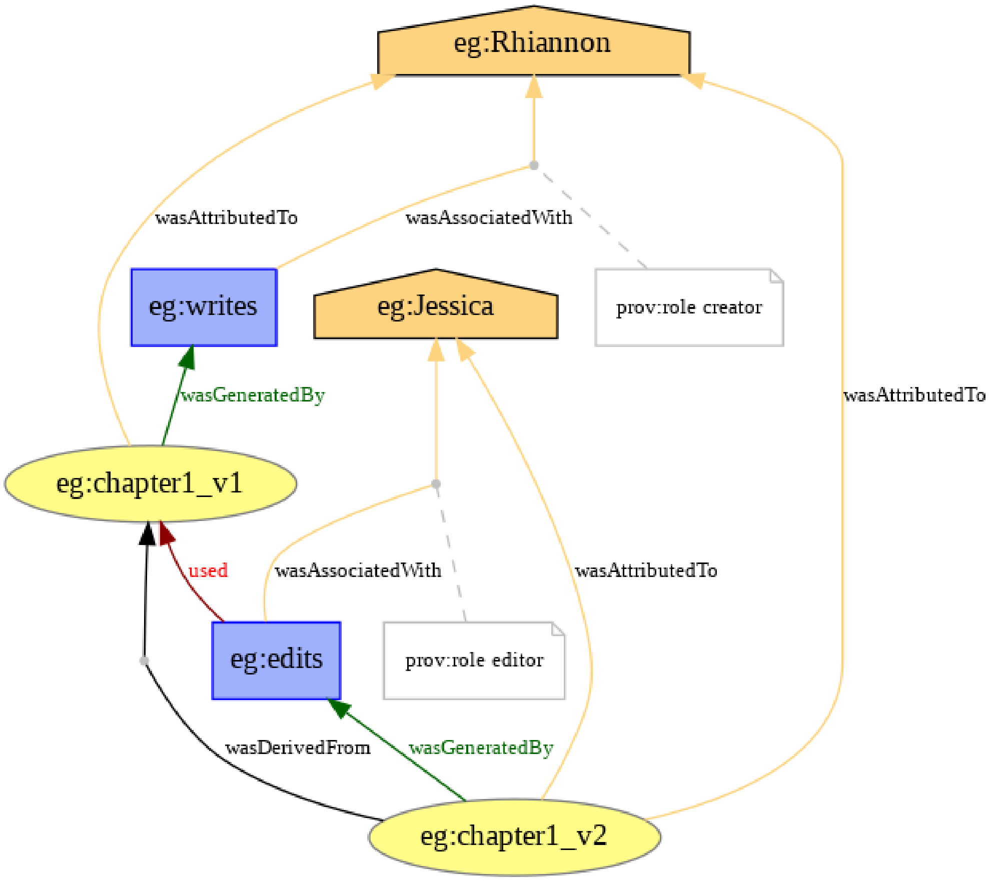


Figure 4.3: Hypothetical provenance of Chapter 1 of *Documenting the Future*[8] The original draft was written by Rhiannon (*creator*) and subsequently edited by Jessica (*editor*). The edited chapter (*chapter1_v2*) is attributed to both authors.

This provenance description is similar the W3C examples of PROV. In this fictitious example, provenance is used to record the history of how the book was compiled, which authors wrote which chapters, who created the figures, who edited the manuscript, and who compiled the final manuscript for submission to the publisher.

There is a different perspective between the provenance of how the book was created versus how a reader came to be reading the book. The domains of interest are different. A reader may not be interested in the different software tools required for writing text versus illustrating figures, or how to manage version control of various edits. The reader may only be interested in the book's availability (whether the local library has a copy), its accessibility (whether there is a hardcover version), and the book's content (ratings from other readers).

An important aspect of provenance is that irrespective of the differing perspectives of various provenance narratives, they can be stitched together. The important question for those creating a provenance record is which perspectives should be included, which should be omitted, and most importantly, which perspectives should be connected. Provenance needs its own appraisal process; This is discussed in Chapter 6 of *Documenting the Future*[8] in the context of environments in PREMIS.

Note that this is not simply a consideration for provenance, but for any modeling technique which allows multiple levels of abstraction. As an example, if one creates a relational database table for Books, every Book record in the table is of the same kind. They are each forced to share the same schema, the same attributes, and each book exists on the same semantic level even if the contents are vastly different. An example of this is shown in Table 4.1.

Table 4.1: Example database table for books.

Title	**Authors**	**Publisher**	**Publication Date**
Harry Potter and the Chamber of Secrets	J.K. Rowling	Bloomsbury	1998
Documenting the Future	R. Bettivia, Y. Cheng, M.Gryk	Springer	2022
A Brief History of Time	S. Hawking	Bantam	1988
Locke & Key	J. Hill	IDW Publishing	2008

Contrast this with an RDF description of the world. For RDF, there is complete freedom to connect objects via arbitrary relationships of various levels of abstraction. For instance, one could easily construct the following RDF[4]:

```
:Charlie_Brown :owns :Snoopy .
:Snoopy :befriends :Woodstock .
:Woodstock :hasColor :Yellow .
:Snoopy a :dog .
:Snoopy a :fictional_character .
:Charles_Schultz :created :Snoopy .
```

This set of RDF triples connects various aspects of the Peanuts work from different real-world and fictional perspectives, in some cases spanning both. Snoopy is a dog only in the fictional world; Charles Schultz created Snoopy only in the real world; yet Woodstock is yellow in both the fictional and real worlds.

For the remainder of this chapter, we will explore various perspectives of provenance. To start, we will explore the provenance of provenance.

4.6 Provenance of Provenance

One perspective on provenance which is often important is the provenance of the provenance record itself. If provenance is a proxy for trustworthiness or quality, then the provenance of provenance is a proxy for the trustworthiness of the provenance record, or a proxy for the proxy. There are two perspectives on the provenance of provenance which are supported explicitly by PROV, the concepts of *bundles* and *plans*.

[4] This is described using Turtle format (https://www.w3.org/TR/turtle/)

4.6.1 Bundles

Bundles are similar in a sense to collections in that they are entities which represent a combination of provenance items. The definition of collection [14] is provided below.

- A **collection** is an entity that provides a structure to some constituents that must themselves be entities. These constituents are said to be **members** of the collections. An **empty collection** is a collection without members.
- **Membership** is the belonging of an entity to a collection.

There are three critical differences between collections and bundles. One, a collection can only contain entities while a bundle can contain an entire provenance record. Two, collections have the *hasMember* relationship (as they contain only entities) while bundles use a bundle constructor to include the individual bundle contents. Three, and most importantly, bundles implicitly and explicitly contain provenance. In that respect, they are a specialized type of entity: while PROV entities can be anything - grapes, bottles, works of art - bundles are always a record of provenance. Therefore, any provenance related to bundles is the provenance of provenance. Here is the definition [14] of a bundle.

A **bundle** is a named set of provenance descriptions, and is itself an entity, so allowing provenance of provenance to be expressed.

Recall how in the previous chapter, the concept of must was created as a collection of juice and pomace. This collection entity was useful as it allowed provenance relationships to connect either to the individual components (members), juice or pomace, as well as to the must as a whole.

Bundles on the other hand, exist at a different level of abstraction. Bundles are ways of referring to portions of the provenance record, not as components, but as provenance itself. For instance, the winemaker may be in charge of the fermentation process, while an accountant or auditor is responsible for documenting individual fermentation runs. Recording provenance at multiple levels of abstraction is always allowed within PROV; bundles provide an explicit mechanism where the provenance of provenance is concerned.

4.6.2 Plans

Bundles are used for documenting the provenance of provenance, that is to say, documenting how the provenance document came to be. There is another aspect of the provenance of provenance: documenting the protocol or recipe that a person or agent intended to follow during a process. The PROV entity for referring to such a recipe is a plan [14].

A **plan** is an entity that represents a set of actions or steps intended by one or more agents to achieve some goals.

An activity **association** is an assignment of responsibility to an agent for an activity, indicating that the agent had a role in the activity. It further allows for a plan to be specified, which is the plan intended by the agent to achieve some goals in the context of this activity.

Note in the second bullet point that the plan entity is recorded as part of the activity association relationship. It is insufficient to say there was a recipe for making Syrah wine; it must be documented that the activity of vinification was associated to an agent (the vintner) as well as the plan (or recipe). That is not to say that plans can only be associated with people. Computer code could be considered a plan and associated with a software agent or even a robotic assembly line.

An important thing to note about plans is that they are the recipe for creating something; they are not the historical lineage of how something was created. Consider a hypothetical example of a chemistry student preparing a buffer (a solution able to neutralize small amounts of an acid or base). The protocol states that one mole of sodium chloride should be added to one liter of water. The student prepares the buffer and notes in their log book that 58.6 g of NaCl was added to 990 mL of water. In this case, the plan was for creating a 1 M solution but the recorded provenance demonstrates the solution is actually 1.01 M[5].The plan is a mechanism of recording the intent along with the outcome.

4.7 Prospective versus Retrospective

The distinction between plans and historical provenance in the preceding section is a distinction between prospective and retrospective provenance. Prospective provenance refers to a protocol for how to make something come to be. Retrospective provenance refers to the lineage of how something has come to be. There is an obvious connection between the two concepts, namely, that both types of provenance use the same underlying core components: entities, activities, and agents. The difference is one of perspective. Is the process something which happened at a previous date and time (retrospective) or is it a prescription for something which can be done in the future (prospective)?

At one level, prospective provenance can be considered a generalization of retrospective provenance. Recall the definition of specialization in PROV – the specialized version has the same attributes as the general version but also contains extra attributes. Those extra attributes in the context of retrospective provenance would be the date or time stamp included for a historic activity as well as items such as the actual recorded mass in the chemistry example. We will further explore this notion embodied in a workflow management system in Chapter 5. It is also the motivation and rationale for the ProvONE extension to PROV.

4.8 Summary

This chapter covered several advanced concepts of provenance in the context of the PROV standard(s) maintained by the W3C. The first were provenance relationships such as usage, generation, derivation, attribution, specializations, and alternatives. Finally, the provenance of provenance section covered retrospective provenance (in the context of bundles) and prospective provenance (in the context of plans). Retrospective and prospective provenance play very important roles in provenance capture and feature prominently in ProvONE and the use case in the next two chapters.

4.9 Extended Conclusions for Dissertation

Chapters 2 and 3 from *Documenting the Future*[8] conclude with the previous summary of advanced concepts in the W3C PROV standard. The subsequent book chapters in *Documenting the Future* introduce two other provenance standards: ProvONE and PREMIS. Since those chapters were primarily authored by the

[5] This may or may not make a difference for the uses of the solution.

co-authors of *Documenting the Future*, they are not included as chapters in this dissertation. However, the topics are important enough to be summarized in this chapter.

4.9.1 Prospective Provenance, P-Plan and ProvONE

The examples throughout chapters 3 and 4 are primarily about retrospective provenance - how something has come to be. The PROV standard is designed for retrospective provenance and this perspective is encoded in the tense of the PROV terms: *used*, *wasGeneratedBy*, *wasAttributedTo*. However, a prospective recipe could be defined in a very similar manner using the same core concepts - entities, activities and agents - by simply changing the tense to *uses*, *isGeneratedBy*, *isAttributedTo*. Note that while PROV has the term *plan* to refer to prospective provenance, it does not prescribe the nature of the representation of a plan. As such, the prospective components for activities and entities are not defined within the PROV standard. PROV bundles allow for enumerating provenance at a granular level, but bundles contain retrospective provenance rather than prospective.

This has led to the development of a few extensions to W3C PROV. One of these developed by Garijo and Gil[25] is called the P-Plan Ontology[6].

P-Plan Ontology

The P-Plan ontology provides additional elements for describing prospective provenance and for connecting those elements to the retrospective elements provided by W3C PROV. These new prospective elements are all intended to provide a structured representation for the plan entity defined by PROV. This is illustrated with a wine example in Figure 4.4. Note that this figure uses the recommended colors and shapes for distinguishing entities (yellow rounded boxes) and activities (blue rectangles).

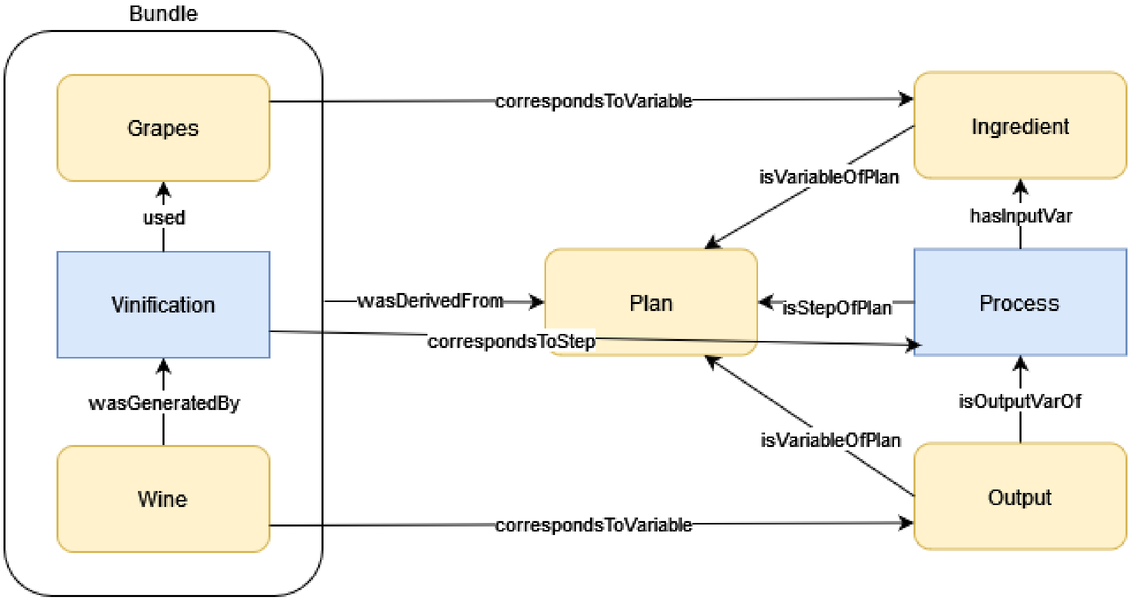


Figure 4.4: Illustration of the P-Plan Ontology for a wine example.

In this example, the left-hand portion of the figure shows a standard PROV representation. In this case,

[6] http://purl.org/net/p-plan

the final product of wine was generated by vinification which in turn used grapes as the source material. This is further elaborated with the use of a PROV bundle to associate these three PROV elements with each other as one provenance record.

It is this PROV bundle (which is a PROV entity) which is associated with the concept of a plan (both a PROV plan and P-Plan plan). It is illustrated that the provenance bundle itself was derived from the plan of how to make the wine.

The plan components are illustrated on the right-hand side of the figure. The plan consists of one step which receives an ingredient as input and produces an output. In order to clarify the level of abstraction, these are represented using generic terms (ingredient, process, and output) while the plan is likely specific for wine making in which case the names for the concepts on both the left and right sides would be the same. What would differ is the attributes associated with each element: the retrospective ones having details such as lot numbers and timestamps and the prospective ones having generic information such as which types of grapes are allowed and how the process should be controlled.

Note that from the use of the term Variable, it is expected that the P-Plan Ontology would be used for the prospective provenance commonly found in computational workflows. However, there is nothing structurally prohibitive about using it for physical processes such as wine making. The ingredients for a cooking recipe are analogous to variables in a computer program.

One final note about the P-Plan Ontology is that it provides a more formal mechanism for defining the PROV plan and associating it with the retrospective provenance. In PROV, a plan is simply an entity and so it could be part of the association of an agent with an activity, an entity from which other entities are derived (Figure 4.4), or an entity which was used by an activity. While the P-Plan Ontology recommends the relationships illustrated in the figure, in principle the other two options could be used. We shall see in the next section that ProvONE chooses to associate a plan with an activity.

ProvONE

The ProvONE[26] extension to PROV was introduced by the DataONE project[7] in 2016. ProvONE was developed to support so-called **hybrid** provenance, the linking of retrospective and prospective concepts as was seen in the previous section with the P-Plan Ontology.

The ProvONE data model is similar to P-Plan in that it is an effort to provide formal structure to the PROV plan entity. It is different from P-Plan in the precise manner in which it achieves this which requires a specialization of PROV elements for both the retrospective and prospective components. ProvONE is designed specifically for recording the provenance of computation; the hybrid nature of ProvONE allows the simultaneous specification of a computational workflow (prospective provenance) and its execution (retrospective provenance referred to as a 'trace').

Figure 4.5 illustrates the use of ProvONE once again using our wine example. As ProvONE is intended for computation, this example is a bit out of place (as it was for P-Plan); however, the general themes should still be apparent.

In terms of the retrospective components to the provenance, ProvONE uses the same general PROV terms (*used, wasGeneratedBy, wasDerivedFrom*, etc.); however, ProvONE subclasses the PROV entities into Data, Documents and Visualizations. The important aspect of this is that ProvONE will introduce new terms for the provenance components. The rationale and importance of Data versus Documents versus Visualizations

[7] https://dataone.org

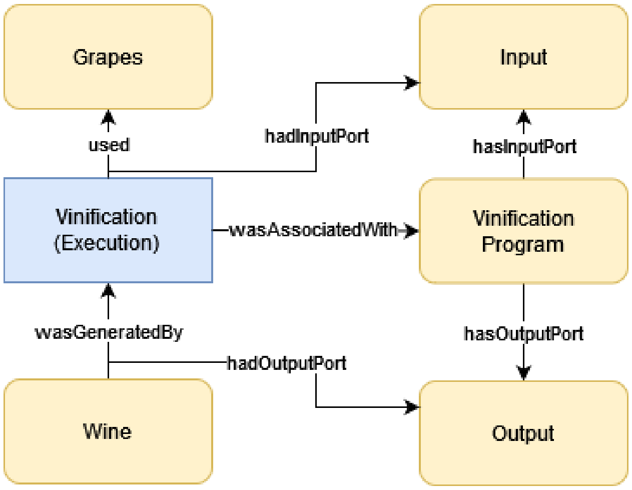


Figure 4.5: Illustration of the ProvONE for a wine example. The retrospective trace is shown on the left; the prospective workflow is on the right.

is related to its use in scientific computation and not of general relevance to provenance *per se*. In Figure 4.5, the retrospective entities are Grapes and Wine and would most closely associate with Data.

Activities in ProvONE are replaced with Executions which are meant for retrospectively realized computational runs. In our wine example, this signifies that a particular Vinification process was executed, in which a particular set of grapes were consumed and a particular batch of wine was produced. ProvONE includes PROV collections, and so the earlier examples of fermenting just the juice versus the pomace can be expressed in ProvONE the same as in PROV.

The major additions to ProvONE are the elements for Programs (including sub-programs and workflows), Controllers, and Ports (including channels). These are the prospective portions of the ProvONE model.

It is important to note that the ProvONE extension to PROV is attempting to accomplish two things simultaneously. One is to be able to document prospective provenance alongside retrospective provenance. The other is to document specific aspects of computational workflows and executions - many of which are not entirely applicable to provenance in general.

This leads to an interesting modeling decision by the creators of ProvONE. The prospective representation of an activity is an entity - a Program. Activities can have many steps, and Programs can have many subprograms, yet there is still an impactful disconnect between the retrospective and prospective. (Note in 4.4 that the Steps or Processes in P-Plan are activities.)

An interpretation of the semantics of this decision is that while the retrospective provenance documents activities which are executed to consume input entities and produce output entities, the prospective provenance is associated with a machine which performs the executions. That machine is an entity rather than an activity and the inputs and outputs are negotiated via the machine specifications (ports). In our wine making world, we could imagine the “Program” to be a steel vat with temperature controls to which grape pomace is added via an InputPort and wine is eventually made available via an OutputPort.

With this model in hand, it can easily be documented that a single vinification vat (Program) produced multiple batches of wine by recording the lot numbers of the grapes used for each execution, the batch numbers of the wine produced for each execution, as well as details for each run (such as datestamps, temperature readings, etc.). The metadata for vinfication machinery would be identical for all batches of wine produced by it.

Are the Provlets Complete?

Rhiannon Bettivia coined the term “provlet” for the various representations and extensions of W3C PROV which are needed for specific applications. The fact that so many provlets have been introduced begs inquiry into the potential deficiencies of PROV and whether the provlets fix all of the holes.

These questions have been in the forefront throughout my graduate studies at the University of Illinois, School of Information Sciences. My early work (which is the subject matter of the following two chapters) used PREMIS to document the provenance of scientific workflows. Colleagues at the iSchool and elsewhere questioned me as to why I chose PREMIS and not PROV. Attempts to answer that question led to work presented at the 2018 International Conference on Knowledge Management [27], several workshops on using PROV, ProvONE and PREMIS for documenting provenance metadata, and eventually to Rhiannon, Jessica and I authoring *Documenting the Future*.

As to the rationale for PREMIS rather than PROV, that mostly comes down to the fact that PREMIS from the very beginning was designed to support a particular provenance workflow in the field of digital preservation. As it was built with a specific practice and specific practitioners in mind, it is very straightforward to implement - *as long as the provenance you wish to document aligns with the digital preservation workflow.* PROV, on the other hand, is designed to tackle provenance in the abstract and while useful in a larger conceptual space, is a bit more challenging to use as so many implementation details are left to the practitioner.

As for P-Plan and ProvONE, there is something disconcerting about how additional elements need to be introduced to record both prospective and retrospective provenance. Throughout these two chapters, various examples of describing the provenance of wine making were provided. While they are inherently retrospective, as they illustrate PROV, there does not seem (to me) to be a very deep divide between the past and the future. A recipe for making a wine and the documentation for how a wine was made are conceptually very similar.

Yet in P-Plan, there are two copies to be tracked. A prospective recipe and a retrospective bundle of provenance with linkages between each corresponding element. Figure 4.4 shows that three PROV elements quickly become eight with the P-Plan ontology, and this is a very simple and small provenance record. What happens in the real world where processes have dozens or more steps and hundreds of entities? ProvONE suffers from the same duplication of elements with the added complexity of mapping retrospective activities to prospective entities and mapping retrospective entities into prospective ports.

If we instead consider retrospective and prospective to simply be additional levels of specialization for provenance, then there is no need for duplication at all. The same provenance structure could be used for both recipes and executions with the addition of attributes for those specialized features. As a short foreshadow to Chapter 10, rather than reifying the prospective and retrospective elements as distinct conceptual things, the temporal nature can be encoded in attributes of those things instead. This does not come without cost and is more fully investigated in the realm of relational models in Chapter 10.

PREservation Metadata: Implementation Strategy (PREMIS)

PREMIS is an implementation strategy embodied as a metadata standard to manage provenance information in the field of digital preservation. Managed by the Library of Congress, PREMIS is about 20 years old with the current version, 3.0, approximately age 10.

In comparing PREMIS with PROV, it is worth comparing the motive behind the two standards. PROV has the broad mandate of the World Wide Web Consortium, charged with managing provenance metadata on the World Wide Web and other electronic infrastructure. Owing to the diversity of content on the Web, nearly any object - physical or digital, real or imaginary - is within the purview of PROV. In contrast, PREMIS is focused on digital objects and maintaining their usability and accessibility into the future. This centers around issues such as the digital encoding of digital objects, as well as their normalization and migration as data formats evolve and are deprecated. One facet which distinguishes the use of PREMIS and PROV is that while PROV use cases are open-ended, the methods and manners for preserving digital objects are largely the same irrespective of the technical details of how the objects are encoded.

Reflecting upon the introduction to Chapter 3, it should come as no surprise that PREMIS defines essentially the same three, top-level concepts as PROV, albeit with slightly different terminology. PREMIS defines Objects (analogous to PROV entities), Events (analogous to PROV activities) and Agents (the same term as PROV). In addition to these three core concepts, PREMIS also has a fourth major category called Rights. Preserving digital objects also requires care and consideration about who owns the rights to access and view those objects, and those rights must be preserved as the digital object is preserved. While rights take center stage in PREMIS, it would be wrong to state that rights cannot be described within PROV. Rather, rights are just one of the many aspects of an object whose provenance can be defined within the PROV standard.

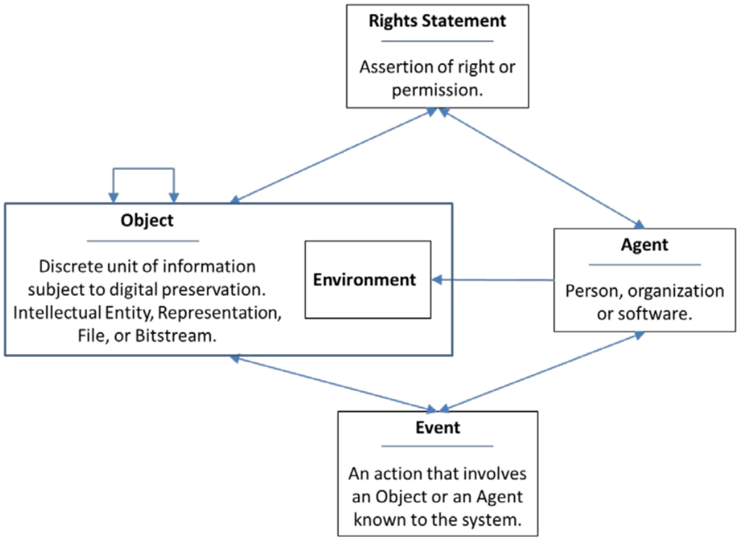

Rights Statement
Assertion of right or permission.
Object
Discrete unit of information subject to digital preservation. Intellectual Entity, Representation, File, or Bitstream.
Environment
Agent
Person, organization or software.
Event
An action that involves an Object or an Agent known to the system.

Figure 4.6: Schematic of the Top Level Entities of the PREMIS Data Model (reprinted from PREMIS website).

Figure 4.6 illustrates the four top-level entities of the PREMIS data model and their interactions. The relationships between the entities are not named as they are in PROV and are for the most part bi-directional. This is important for the XML representation of PREMIS where each top-level entity has its own XML root content block. A linked Object and Event would record the linked Event within the Object block and the linked Object within the Event block.

PREMIS has an XML representation as well as an OWL representation starting with version 3.0. In accords with linked data, PREMIS concepts are formally linked to PROV concepts. For instance, the PREMIS agent is defined as a subclass of both the PROV agent as well as the FOAF agent[8].

This overall structure is very well conceived for digital preservation, where the preservation of objects is expected to be conducted piecemeal over a time frame of months and years. Documenting the preservation activities as data blocks allows appending new changes to an existing PREMIS file as additional preservation steps are undertaken. Since the relationships are managed bidirectionally, the new objects, events, agents and rights can be appended with links to their predecessors and the predecessors augmented with the links to the new materials.

This is also a very useful manner for documenting the provenance of pipelined, scientific computations where each individual computational node can be responsible for adding the modification it added to a growing PREMIS record. This has been implemented for a workflow management system within the scientific domain of biomolecular NMR spectroscopy and is the subject of the next two chapters.

[8] Friend Of A Friend Ontology

Part II

NMR and Natural Science Perspective

Chapters 5 and 6 describe the recording of both prospective and retrospective provenance for bioNMR spectroscopy within the NMRbox computing platform using the software tool called CONNJUR Workflow Builder (CWB). CWB was developed over the span of 2009 through 2014 and back in 2014 was able to record prospective and retrospective provenance using an application-specific XML schema. This allowed for the sharing of workflows within the CWB community but did not adhere to the FAIR principles of supporting provenance documentation using a "broadly applicable language for knowledge representation" (FAIR principle I1).

Chapter 5 describes the refactoring of CWB to use the PREMIS standard for provenance documentation. PREMIS is also serializable as XML and is supported by the Library of Congress primarily for digital preservation. PREMIS version 3.0 allows for domain-specific extensions to the top-level provenance objects. In this chapter, CONNJUR-ML is described, which along with PREMIS 3.0 provides for a broader metadata standard for documenting provenance. The PREMIS record is included with the final computational output of a CWB workflow through a "zipped" container file containing the binary data and the PREMIS metadata.

Chapter 6 extends this work by including analytics on the intermediate data through the execution of the workflows. Typically, bioNMR spectra are reconstructed using multi-step workflows (10-20 steps) throughout which the intermediate data are discarded. While the final data is of primary importance, it is also useful to make measurements on certain properties of the data at each stage along the workflow for quality assessment and comparison of various workflows. PREMIS/CONNJUR-ML supports such analytics in accordance with FAIR.

These chapters represent the following contributions to the field of Library and Information Science:

- Supporting FAIR principles for workflow execution by refactoring application specific XML to PREMIS
- Development of CONNJUR-ML to be embedded within PREMIS extensions to support FAIR provenance tracking.
- Bundling provenance with data to support Research Objects
- Embedding analytics within curation / workflow execution
- Analytics report on the effectiveness of the workflow steps - is of general use for data cleaning to understand how ”clean” a certain step makes the data.
- Identified and mitigated a shortcoming of PREMIS in that software tools need to be linked more granularly, not to an object but an object characteristic. (This can be related to concept keys at the end)

Chapter 5

Curating Scientific Workflows

5.1 Abstract

This paper describes our recent and ongoing efforts for enhancing the curation of scientific workflows to improve reproducibility and reusability of biomolecular nuclear magnetic resonance (bioNMR) data. Our efforts have focused on both developing a workflow management system, called CONNJUR Workflow Builder (CWB), as well as refactoring our workflow data model to make use of the PREMIS model for digital preservation. This revised workflow management system will be available through the NMRbox cloud-computing platform for bioNMR. In addition, we are implementing a new file structure which bundles the original binary data files along with PREMIS XML records describing the provenance of the data. These are packaged together using a standardized file archive utility. In this manner, the provenance and data curation information is maintained together along with the scientific data. The benefits and limitations of these approaches as well as future directions are discussed.

5.2 Introduction

An acknowledged goal in the field of data curation is to move curation tasks upstream closer to the creation and origination of the data[28]. When considering a scientific study, it is often the case that the digital dataset is not the product of a single, experimental observation. Rather, multiple observations are collected, digitized, normalized, cleaned, and otherwise transformed. Several different and potentially disparate datasets are then analysed in concert along an involved computational pipeline or workflow [12].

As the production of computational data through the aforementioned workflows has become increasingly complicated, there has been a growing concern for the lack of a detailed reporting of the workflows along with the intermediate datasets, particularly as these are necessary for the reproducibility and/or repeatability of the computation [10]. In the case of such scientific workflows, data curation must be an ongoing process along the entire computational pipeline.

In this paper we discuss efforts to improve the reproducibility of scientific computation by adding curation tasks directly within the construction and execution of the computational workflows. This effort is being conducted within the context of the field of biomolecular nuclear magnetic resonance spectroscopy (bioNMR) using the NMRbox platform for bioNMR computation. The workflow management system used within NMRbox is CONNJUR Workflow Builder [3]. CONNJUR Workflow Builder (CWB) is currently being refactored such that curation metadata will be stored as XML using a hybrid of the PREMIS metadata schema for digital preservation [1] and a bioNMR specific metadata schema currently referred to as CONNJUR_ML[1]. The short-term goal is to package this PREMIS XML file together with the scientific dataset (typically a binary file) using standard file archive utilities.

5.3 Biomolecular Nuclear Magnetic Resonance Spectroscopy

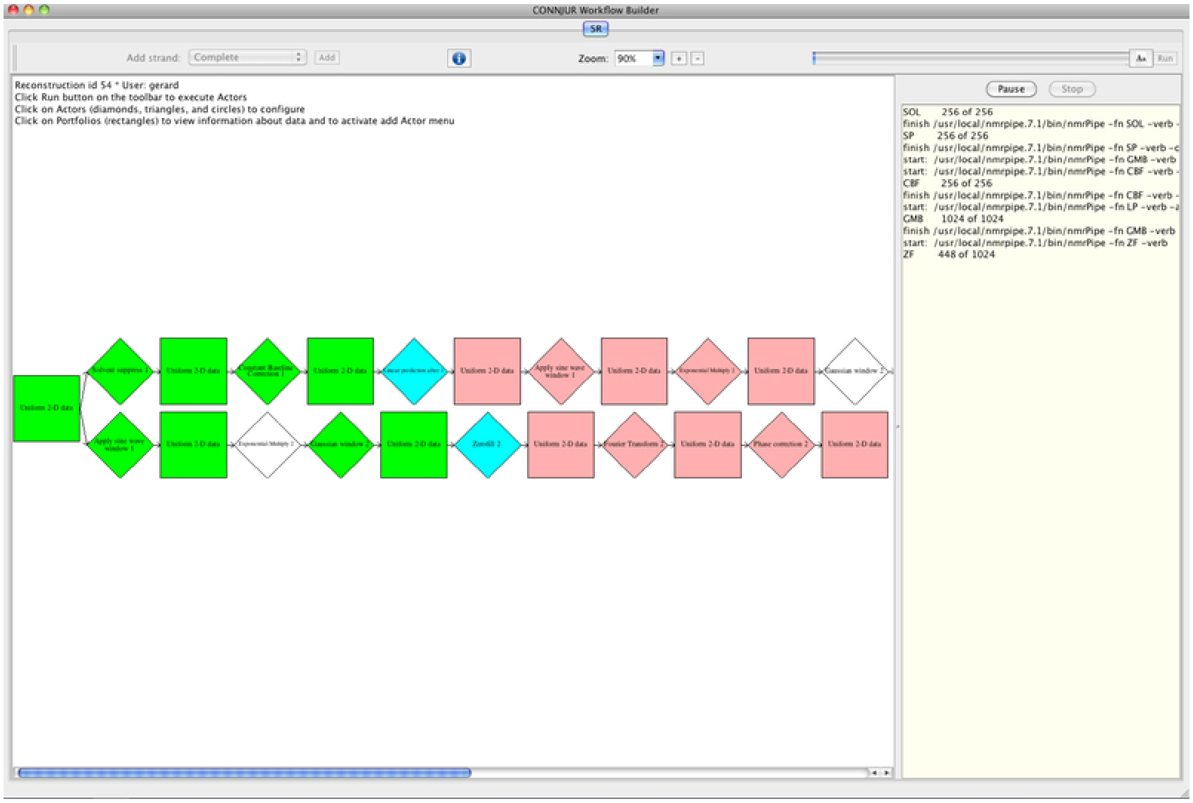
Biomolecular NMR spectroscopy is a biophysical technique which exploits the magnetic moments of the nuclei comprising the matter all around us. A close sibling of Magnetic Resonance Imaging (MRI) which uses this intrinsic magnetism to image human tissue, nuclear magnetism is used in bioNMR studies to explore the structure and dynamics of biological molecules at atomic detail. These studies include determining the three-dimensional structure of proteins and nucleic acids, drug discovery, kinetics and mapping the interfaces of protein-protein and protein-ligand interactions.

The computational workflow for modern bioNMR spectroscopy consists of three phases: spectral reconstruction, the process of converting time domain data into the frequency domain; spectral analysis, including peak identification and resonance assignment; and biophysical characterization, including all subsequent data analysis in which the spectroscopic data is used to draw biophysical inferences (e.g. structure determination) [29], [30]. The data semantics vary throughout these phases, from primary 'raw' data of the nuclear precessions to various levels of derived or interpreted data including resonance frequencies and interatomic distances. This computational workflow uses more than a dozen, academically-developed software tools with many file translation and data cleaning steps along the workflow. Proper curation of the bioNMR workflow is an ongoing challenge affecting data sharing, the archival of research results, and the reproducibility of prior studies [19].

NMRbox [19] is a recent initiative to foster computational reproducibility for the bioNMR community by (a) establishing an archive of the various software tools for bioNMR and (b) provisioning a virtual machine for bioNMR computation. NMRbox uses CONNJUR for semantic data management within these virtual machines. CONNJUR[2] is a long-standing project for developing a software integration environment for bioNMR, and currently supports CONNJUR Workflow Builder (Figure 5.1), a scientific workflow management system for bioNMR spectral reconstruction [3]. The benefits of using CWB are that the metadata are stored in a relational database making them available throughout the computational workflow, multiple software tools can easily be interleaved within the workflow, and the workflows can be exported as XML to facilitate reuse and sharing between researchers [3]. An example of CWB used for spectral reconstruction is found on our YouTube channel under the title 13C HSQC.

While useful in this context, the data model for CWB is restricted to the context of bioNMR and the implementation details for constructing and executing the workflow itself. As pointed out by [31], such customization has the drawback that the reuse and sharing enabled by workflow management systems such as CWB is limited in scope to the users of CWB. This can be more readily appreciated by examining the XML output from version 1 of CWB. (Figure 5.2). The XML for the exported workflow contains information about the Java classes required to configure and execute the workflow within the CWB program. With appropriate knowledge of the operation of CWB and the XML itself, it should be possible to translate such a workflow into a more generic description amenable to other workflow management systems. However, this knowledge

[1] https://github.com/CONNJUR/CONNJUR_ML
[2] http://www.connjur.org

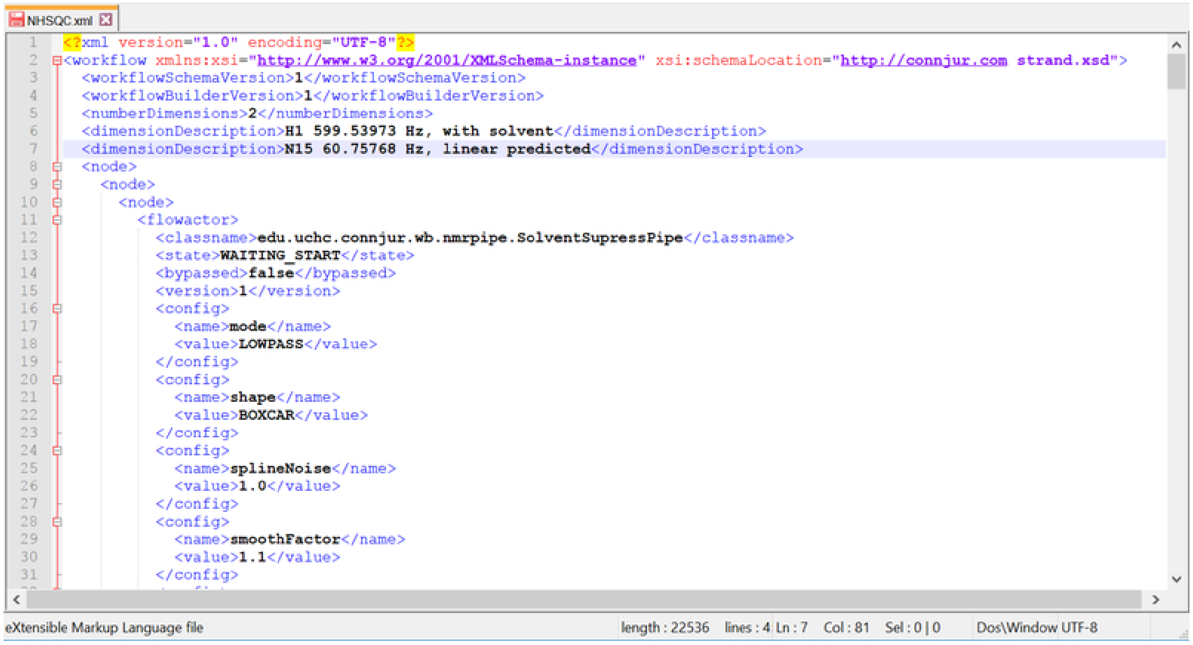


Reconstruction id 54 * User: gerard
Click Run button on the toolbar to execute Actors
Click on Actors (diamonds, triangles, and circles) to configure
Click on Portfolios (rectangles) to view information about data and to activate add Actor menu

```
SOL      256 of 256
finish /usr/local/nmrpipe.7.1/bin/nmrPipe -fn SOL -verb -
SP      256 of 256
finish /usr/local/nmrpipe.7.1/bin/nmrPipe -fn SP -verb -c
start:  /usr/local/nmrpipe.7.1/bin/nmrPipe -fn GMB -verb
start:  /usr/local/nmrpipe.7.1/bin/nmrPipe -fn CBF -verb -
CBF      256 of 256
finish /usr/local/nmrpipe.7.1/bin/nmrPipe -fn CBF -verb -
start:  /usr/local/nmrpipe.7.1/bin/nmrPipe -fn LP -verb -a
GMB      1024 of 1024
finish /usr/local/nmrpipe.7.1/bin/nmrPipe -fn GMB -verb
start:  /usr/local/nmrpipe.7.1/bin/nmrPipe -fn ZF -verb
ZF      448 of 1024
```

Figure 5.1: Screenshot of the graphical canvas for CONNJUR Workflow Builder. Squares represent datasets while diamonds represent actors. The above workflow is in the process of being executed. Green objects are those who have completed successively, blue are in progress, and pink actors have yet to be invoked. While actors are being bypassed and not executed in the above workflow.

requirement is quite extensive and has prompted us to examine a more generic workflow representation which is both useful for the bioNMR community it serves, as well as the broader audiences of workflows, provenance and digital curation. It is hoped that a broader knowledge representation language for workflows will make the provenance descriptions understandable to a larger community of researchers apart from the domain of NMR scientists. The broad XML architecture chosen for this purpose is that of PREMIS, a general framework used by the digital preservation community.



NHSQC.xml
eXtensible Markup Language file
length : 22536 lines : 4 Ln : 7 Col : 81 Sel : 0 | 0
Dos\Window UTF-8

```xml
<?xml version="1.0" encoding="UTF-8"?>
<workflow xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://connjur.com strand.xsd">
  <workflowSchemaVersion>1</workflowSchemaVersion>
  <workflowBuilderVersion>1</workflowBuilderVersion>
  <numberDimensions>2</numberDimensions>
  <dimensionDescription>H1 599.53973 Hz, with solvent</dimensionDescription>
  <dimensionDescription>N15 60.75768 Hz, linear predicted</dimensionDescription>
  <node>
    <node>
      <node>
        <flowactor>
          <classname>edu.uchc.connjur.wb.nmrpipe.SolventSupressPipe</classname>
          <state>WAITING_START</state>
          <bypassed>false</bypassed>
          <version>1</version>
          <config>
            <name>mode</name>
            <value>LOWPASS</value>
          </config>
          <config>
            <name>shape</name>
            <value>BOXCAR</value>
          </config>
          <config>
            <name>splineNoise</name>
            <value>1.0</value>
          </config>
          <config>
            <name>smoothFactor</name>
            <value>1.1</value>
          </config>
```

Figure 5.2: Notepad++ screenshot of original CONNJUR Workflow Builder (CWB) XML schema. While this XML document could be shared between CWB users, the metadata schema was specific for CWB and not intended for broader distribution. It contains information about the software state as well as specific Java classes of software code.

5.4 PREMIS

Maintained by the Library of Congress, Preservation Metadata: Implementation Strategies (PREMIS) has been developed by the archival and library communities to provide digital preservation systems with a framework to build reliable systems for sustainable information stewardship [1]. With the release of version 3 of PREMIS, it is possible to embed custom structural, descriptive and administrative metadata within a PREMIS XML record. We have developed a bioNMR XML for spectral reconstruction which is currently made available on GitHub. Called CONNJUR_ML, this bioNMR XML is intended to provide metadata suitable for the significant_properties field in PREMIS objects, as well as the extension field for PREMIS agents to provide metadata about the software and hardware environments.

Digital preservation is itself a type of workflow, albeit a very general workflow supporting a wide assortment of digital objects. CWB was re-assessed in the context of PREMIS as it became clear that the panoply of workflow components for spectral reconstruction are all variations on the data cleaning, data translation, data normalization and data transformation operations typical for digital preservation workflows. The important metadata being captured by CWB throughout the workflow are structural and administrative metadata important for both describing how the component processing steps were configured, and also for identifying classification properties of the intermediate datasets. Willoughby and Frey[31] call attention to the importance of the intermediate data generated during a scientific workflow. In the case of bioNMR, the intermediate datasets can be quite large, often gigabytes in size. We have found that a detailed metadata record can be a suitable surrogate to storing the actual intermediate files.

5.5 CONNJUR Workflow Builder

The data model for CONNJUR Workflow Builder is being refactored to support the more general concepts provided by PREMIS, as well as the bioNMR-specific classifications of the bioNMR data. The next version of CWB will import and export spectral reconstruction workflows as PREMIS digital preservation records, widening the audience for data sharing and reuse. As part of its integration within the NMRbox platform, CONNJUR/PREMIS workflows will also be translated into NMR-STAR, the file format supported by the primary biomolecular NMR data repository, the BioMagResBank[3] [21]. By supporting linked digital preservation events, the PREMIS metadata standard provides provenance of what operations were performed on any digital object (in our case, bioNMR datasets), when they were performed, by what human agents and using what software tools - up to and including the entire software environment. As part of the NMRbox platform, this entire software environment is archived and will be accessible as a virtual machine for several years to come.

Figure 5.3 shows a snippet of a CWB metadata record using PREMIS in conjunction with CONNJUR_ML (the actual metadata record for this workflow is over 1000 lines long). PREMIS is used to define the intellectual objects represented by each intermediate dataset along the computational pipeline. As these intermediates are only intended to be stored local to the execution environment, they are given local identifier types and values.

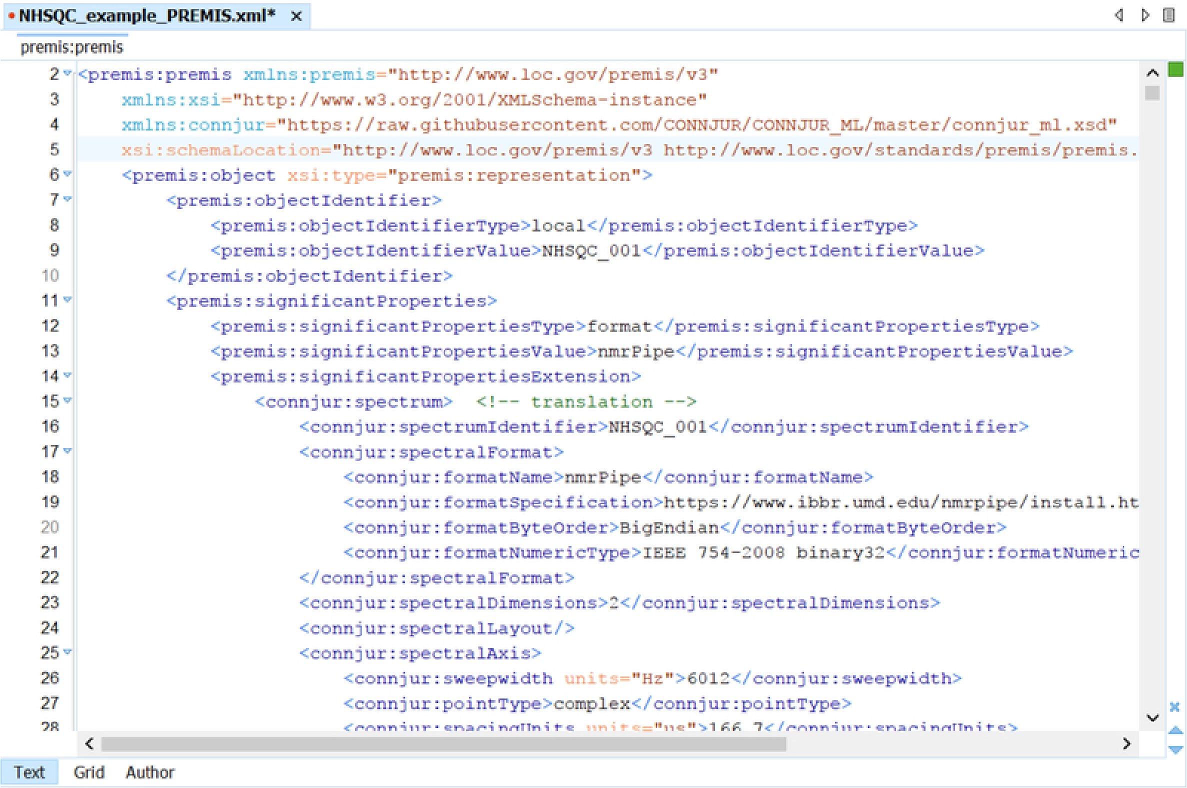
Each intermediate dataset is manifested (perhaps only transiently) in a file format which can be interpreted by the underlying software tool which will process the data. The format of the data is recorded using the PREMIS significant properties tag. All additional structural metadata about the intermediate dataset is contained in the significant properties extension section. This makes use of the bioNMR specific CONNJUR_ML to record important features of the intermediates – such as the levels of signal coverage, resolution, as well as various state information regarding the types of transformation which have been applied. This structural metadata can be used as a sort of fingerprint for (a) classifying which types of transformations can be made on which datasets as well as (b) for validating the workflow.

Each subsequent processing step, in which an actor ingests a dataset and produces another, is recorded as a PREMIS event with associated metadata describing and linking the events to the intellectual objects with which they are involved. Additionally, each PREMIS event is associated with a PREMIS agent which may be a human or a computer. For the computational workflows executed by CWB, additional metadata on the software and software versioning is added to the PREMIS agent extension.

5.6 Limitations and Future Directions

Limitations of this approach are related to that of virtual machines themselves. While the digital image of a virtual machine can be stored indefinitely, its operation is dependent on suitable computing hardware and

[3] https://bmrb.io/



```xml
<premis:premis xmlns:premis="http://www.loc.gov/premis/v3"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:connjur="https://raw.githubusercontent.com/CONNJUR/CONNJUR_ML/master/connjur_ml.xsd"
    xsi:schemaLocation="http://www.loc.gov/premis/v3 http://www.loc.gov/standards/premis/premis.
    <premis:object xsi:type="premis:representation">
        <premis:objectIdentifier>
            <premis:objectIdentifierType>local</premis:objectIdentifierType>
            <premis:objectIdentifierValue>NHSQC_001</premis:objectIdentifierValue>
        </premis:objectIdentifier>
        <premis:significantProperties>
            <premis:significantPropertiesType>format</premis:significantPropertiesType>
            <premis:significantPropertiesValue>nmrPipe</premis:significantPropertiesValue>
            <premis:significantPropertiesExtension>
                <connjur:spectrum>  <!-- translation -->
                    <connjur:spectrumIdentifier>NHSQC_001</connjur:spectrumIdentifier>
                    <connjur:spectralFormat>
                        <connjur:formatName>nmrPipe</connjur:formatName>
                        <connjur:formatSpecification>https://www.ibbr.umd.edu/nmrpipe/install.ht
                        <connjur:formatByteOrder>BigEndian</connjur:formatByteOrder>
                        <connjur:formatNumericType>IEEE 754-2008 binary32</connjur:formatNumeric
                    </connjur:spectralFormat>
                    <connjur:spectralDimensions>2</connjur:spectralDimensions>
                    <connjur:spectralLayout/>
                    <connjur:spectralAxis>
                        <connjur:sweepwidth units="Hz">6012</connjur:sweepwidth>
                        <connjur:pointType>complex</connjur:pointType>
                        <connjur:spacingUnits units="us">166.7</connjur:spacingUnits>
```

Figure 5.3: Oxygen screenshot of PREMIS XML record with CONNJUR_ML metadata embedded within the ‘significantProperties’ PREMIS tag. CONNJUR_ML is an ongoing modelling task and the XML can be found on GitHub.

hypervisor software capable of emulating the software environment. Virtual machines are currently ubiquitous and are expected to remain so for years to come; however, future migration to new hardware/software would seem unavoidable.

It is for this reason that many scholars concerned with reproducibility are advocating the use of container technology such as Docker. Docker containers provide a lighter footprint of virtualization and might be expected to remain stable farther into the future than full-fledged VM's. However, NMRbox currently supports over 100 individual software tools. To package each of these tools within an individual container and have the individual containers all interoperate is a significant task. Nevertheless, this usefulness of containers is being explored by the developers at NMRbox.

In this context of long-term sustainability, one of the added benefits of describing workflows within a digital preservation standard such as PREMIS is that the transformations along the workflow are described using a broader language, making the context and purpose of the workflows accessible to a wider audience. In this sense, our workflow management system is being repurposed as a data curation tool.

Our current metadata model and workflow implementation is for well-structured workflows using primarily mathematical transformations. Additional future directions will be to provide for more general data curation activities and more free-form annotation schemes such as done for spectral analysis using the reproducibility extensions of NMRFAM-Sparky [20].

Chapter 6

Analytics in Workflows

6.1 Abstract

This chapter reports on the ongoing activities and curation practices of the National Center for Biomolecular NMR Data Processing and Analysis[1]. Over the past several years, the Center has been developing and extending computational workflow management software for use by a community of biomolecular NMR spectroscopists. Previous work had been to refactor the workflow system to utilize the PREMIS framework for reporting retrospective provenance as well as for sharing workflows between scientists and to support data reuse. In this chapter, we report on our recent efforts to embed analytics within the workflow execution and within provenance tracking. Important metrics for each of the intermediate datasets are included within the corresponding PREMIS intellectual object, which allows for both inspection of the operation of individual actors as well as visualization of the changes throughout a full processing workflow.

These metrics can be viewed within the workflow management system or through standalone metadata widgets. Our approach is to support a hybrid approach of both automated, workflow execution as well as manual intervention and metadata management. In this combination, the workflow system and metadata widgets encourage the domain experts to be avid curators of the data which they create, fostering both computational reproducibility and scientific data reuse.

6.2 Introduction

The National Center for Biomolecular NMR Data Processing and Analysis (informally referred to as NMRbox) is an NIH-supported Biomedical Technology Research Resource [19]. The goal of the Center is to foster computational reproducibility in the field of biomolecular nuclear magnetic resonance (bioNMR) spectroscopy. In pursuit of this goal, the Center provides XUbuntu Virtual Machines (VMs) provisioned with more than one hundred fifty software packages used in bioNMR data processing and analysis. These NMRbox VMs are offered as downloadable images which can be run on the user's host computer with any suitable hypervisor, or from a freely accessible cloud computing environment which is connected to using the RealVNC[2] virtual desktop client. At the time of the writing of this chapter, NMRbox has more than 5000 registered users.

The NMRbox VMs help address the problems of software persistence and availability which hinder

[1] http://www.nmrbox.org

[2] http://www.realvnc.com

computational reproducibility. A second hurdle which must be overcome is the inadequate curation of the bioNMR datasets and subsequent computational analyses. The Center actively collaborates with the Biological Magnetic Resonance Data Bank[3] (BMRB), the international data repository for bioNMR data[21]. Most NMR-related journals require data depositions with the BMRB as a condition for publication; however, the time frame between data creation and publication can be several years limiting the richness of the depositions as much of the critical metadata to support reproducibility has been lost or forgotten. Most depositions only contain a small subset of usable and useful data[4].

The need for foregrounding data curation to support data reuse has long been emphasized [28], [32], [33] with proposed solutions involving software tools which assist in curation at the time of data creation [32], [33]. The NMRbox VMs present just such a solution by including the CONNJUR scientific workflow management environment (called CONNJUR Workflow Builder or CWB) for creating, sharing and executing bioNMR spectral reconstruction workflows [3] and assisting in the curation of the processing schemes at the source of the computation. The original release of CWB allowed import and export of reconstruction workflows as XML; however, the XML schema was custom-built for the CWB application, limiting its usefulness for scholarly communication in general [31]. This limitation has been mitigated by refactoring the provenance and workflow tracking to utilize the PREMIS[5] framework (PREservation Metadata: Implementation Strategies), a standard for curating digital preservation workflows maintained by the Library of Congress. Subsequent to this refactoring, CWB currently imports/exports reconstruction workflows using the PREMIS XML standard as the top-level framework, which is supported by a bioNMR XML[6] for recording domain specific metadata[34]. Each of the top-level PREMIS semantic units allow domain specific XML extensions. CWB and the associated bioNMR XML make use of extensions to object characteristics, event details, and agents.

The use of PREMIS as the scaffold for bioNMR workflow representation was presented at the 13th International Digital Curation Conference (IDCC) in Barcelona, Spain in 2018. One of the keynote lectures at the 13th IDCC was given by Luis Martinez-Uribe; the topic of this keynote was the notion of augmenting data curation by blending it with analytics. While the context of Martinez-Uribe's work was the DataLab at the Library of Fundación Juan March, the ideas presented appeared to be applicable to the curated workflows within NMRbox and inspired this work.

6.3 BioNMR Spectral Reconstruction

For various reasons important to the domain of NMR spectroscopy, bioNMR data are typically collected as hypercomplex multidimensional arrays of values, where each value represents a signal amplitude at a particular coordinate in a multidimensional space consisting of orthogonal time axes. The term hypercomplex refers to the property that at each multi-dimensional time point the data are complex (real, imaginary pairs) along each time axis. Spectral reconstruction is the process by which this time-domain data is converted into multi-dimensional frequency plots, where the individual bioNMR signals can be characterized by their respective frequencies, amplitudes, and line shapes. The workhorse for spectral reconstruction is primarily the Fourier Transform; however, there are several data cleaning steps applied during the process which are undertaken in order to improve the sensitivity (ability to identify signals), the resolution (ability to distinguish signals of similar frequencies) as well as the removal of artefacts or other unwanted signals. The spectral

[3] http://bmrb.io

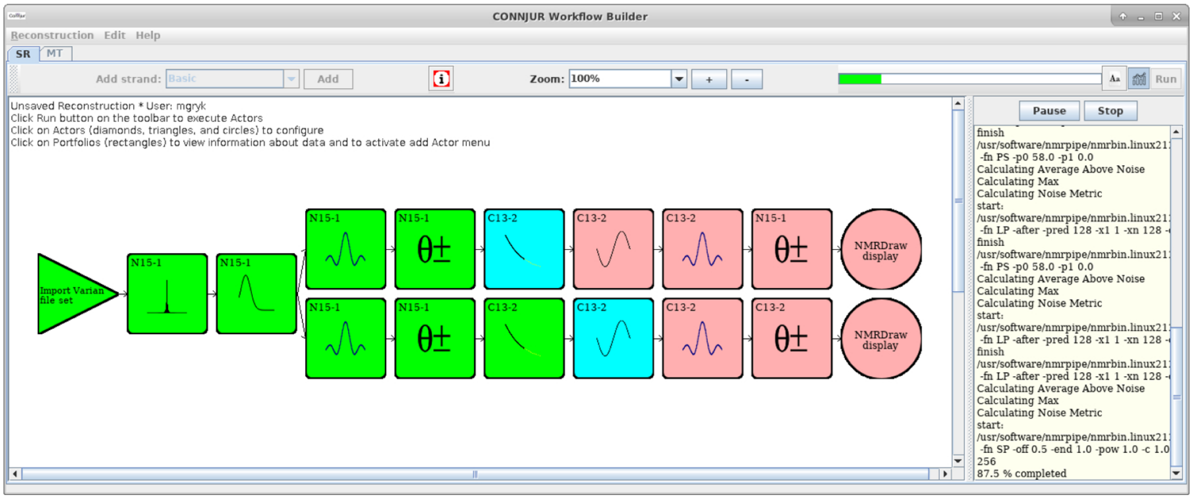
[4] https://bmrb.io/search/query_grid/overview.php

[5] https://www.loc.gov/standards/premis/

[6] https://github.com/CONNJUR/CONNJUR_ML

processing workflow is typically quite involved and uses approximately five to ten operations per spectral dimension.

CONNJUR Workflow Builder (CWB) is a software tool used for the creation, execution, curation and sharing of spectral reconstruction workflows for the scientific domain of bioNMR spectroscopy (see Figure 6.1). A complete description of the software architecture is given in Fenwick, et al. [3] but it is worth emphasizing that a core design decision of CWB was to leverage a previously developed data translation tool called CONNJUR spectrum translator (CST). CST is required for robust workflow execution as there are several software tools used along the processing pipeline which require different formats for the input data. CST was designed to convert between the various existing software tool formats efficiently by translating through a data model common to all formats [35]. While passing the data through this common CST model, it is possible to perform a host of analytics on the bioNMR spectral data.



Unsaved Reconstruction * User: mgryk
Click Run button on the toolbar to execute Actors
Click on Actors (diamonds, triangles, and circles) to configure
Click on Portfolios (rectangles) to view information about data and to activate add Actor menu

```
finish
/usr/software/nmrpipe/nmrbin.linux21
 -fn PS -p0 58.0 -p1 0.0
Calculating Average Above Noise
Calculating Max
Calculating Noise Metric
start:
/usr/software/nmrpipe/nmrbin.linux21
 -fn LP -after -pred 128 -x1 1 -xn 128 -
finish
/usr/software/nmrpipe/nmrbin.linux21
 -fn PS -p0 58.0 -p1 0.0
Calculating Average Above Noise
Calculating Max
Calculating Noise Metric
start:
/usr/software/nmrpipe/nmrbin.linux21
 -fn LP -after -pred 128 -x1 1 -xn 128 -
finish
/usr/software/nmrpipe/nmrbin.linux21
 -fn LP -after -pred 128 -x1 1 -xn 128 -
Calculating Average Above Noise
Calculating Max
Calculating Noise Metric
start:
/usr/software/nmrpipe/nmrbin.linux21
 -fn SP -off 0.5 -end 1.0 -pow 1.0 -c 1.0
256
87.5 % completed
```

Figure 6.1: Screenshot of CONNJUR Workflow Builder showing the configuration and execution of a two-dimensional spectral reconstruction workflow. The triangle represents the import of the original dataset; subsequent boxes each represent a separate processing step. CONNJUR Spectrum Translator handles data translation between the steps and also performs analytics on the data. The circles invoke an integrated visualization tool. There is an extensive color mapping for indicating the status of actors along the workflow. Green represents actors which have completed, blue actors are in process, and pink actors are configured but have not been executed yet. Finally, the workflow shown above is forked after the third actor, allowing a single dataset to be processed in two different ways.

Just as many of the object characteristics for bioNMR workflows are domain-specific requiring a custom data model, so are many of the analytics which are useful to the bioNMR community. Metrics for generalized concepts such as sensitivity, signal intensity and noise level require domain-specific implementations, as will be discussed in the following section.

Despite the specificity of computing these metrics, once they have been measured during the workflow execution, the values are easily stored within the larger PREMIS structure. These augmented provenance records contain elements of data curation (metadata mappings and annotations as to the how and why of actor configuration), workflow execution (retrospective provenance of the computation) and analytics (metadata recording both dataset characteristics as well as metrics of the underlying data). Once recorded within a PREMIS XML document, the individual metrics and characteristics of any processing intermediates can be retrieved. It is also possible to visualize how a particular metric changes throughout the processing workflow, as illustrated with the signal intensity and noise level as shown in Figure 6.2.

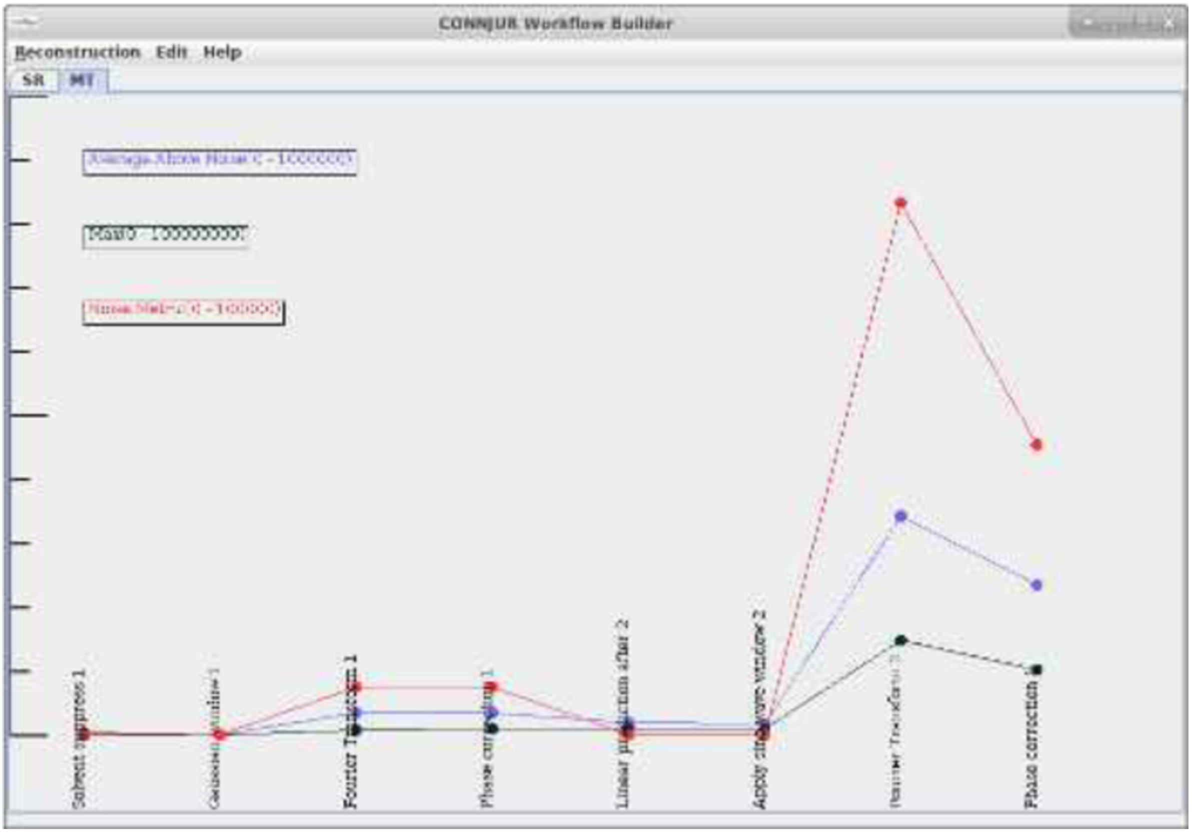


Figure 6.2: Analytics display tab of CONNJUR Workflow Builder. The graph shows a measurement of the overall noise observed in the data along the workflow (red line) as well as two metrics designed as proxies for sensitivity (average signal intensity above the noise (blue line) and the maximum signal intensity (black line)). Note that the scale is different for each graph making the noise appear larger than the signal. The insight from these metrics is discussed in the following section.

6.4 Embedded Analytics

As mentioned in the preceding section, spectral reconstruction is the computational process by which bioNMR data which were originally acquired as sets of amplitudes varying in time (milliseconds) are converted into amplitudes of corresponding frequencies (Hertz). The primary computation for this is the Fourier Transform; however, it is useful to apply mathematical data cleaning steps during this process, resulting in a fairly sizeable workflow (Fig 6.1). The various types of cleaning operations and their purpose are shown in Table 6.1. Also of note is that NMR signals are most often analysed relative to one another. For that reason, the absolute magnitude and units of the observational data is neglected and often given in arbitrary units (a.u.). A consequence of this will be illustrated with the help of the embedded analytics.

Figure 6.1 illustrates the configuration and execution of a forked workflow which will be used as a case study of the embedded analytics now available for CWB. In this spectral reconstruction workflow, the following sequence of data cleaning / data transformation operations are applied: (Dimension 1) Import Data, Solvent Suppression, Apodization, Fourier Transform, Phase Correction; (Dimension 2) Linear Prediction,

Table 6.1: Common data cleaning operations in NMR reconstructions.

Operation	**Purpose**	**Domain Applied**
Solvent Suppression	Remove unwanted signal	Time
Linear Prediction	Augments / extends existing signal	Time
Apodization	Suppress noise	Time
Fourier Transform	Converts time to frequency	Time
Phase Correction	Improves appearance of signal	Frequency

Apodization, Fourier Transform, Phase Correction, Export Data. The top-level organization allows for sequential processing of the two dimensions of the dataset (corresponding to hydrogen and nitrogen nuclei, respectively). The very first step following import attempts to remove unwanted solvent signals which would complicate further analysis. The very first step along the second dimension artificially extends the observational data with "predicted" data which are calculated from the existing data. Finally, the common elements to each dimension allow for reduction of noise through apodization, transformation to frequency, and phase correction of the frequency components.

The workflow in Figure 6.1 is forked, meaning that the same original dataset is processed in two ways. The first three operations are shared, while the remaining ones are distinct. For the purpose of this case study, the entirety of both forks in the workflow are identical with two exceptions. The two Fourier Transform actors (positions 4 and 8) in the workflow use two different implementations of the Fourier Transform. One feature of CWB is that it wraps multiple third-party tools and can use either the Rowland NMR Toolkit [36] for processing or NMRPipe [37]. CST handles all of the data translation so it is seamless to create and execute such hybrid workflows.

At each step along the workflow, CWB is capable of measuring properties of the intermediate data and these are reported in the PREMIS record which records the workflow provenance. For this case study, two metrics related to sensitivity will be shown as well as a rough measurement of the noise.

Sensitivity metrics are calculated by first identifying the quadrature of each dimension. For complex data, the real and imaginary values are replaced by the square root of the sum of the real and imaginaries values squared, yielding a power spectrum. Next, all floating points values in the remaining data are sorted. The tenth percentile value is used as an estimate for the noise level of the signal. The top ninety percent of floating values are considered signal and are used to calculated both the average signal and maximum signal values. The limitations of these metrics will be discussed further in the final section.

The values of the metrics for each processing fork are plotted for each step in their respective reconstructions in Figure 6.3. Unfortunately, due to the nature of the processing workflows it is not possible to plot all three metrics for each fork on one graph at the same scale. As is seen in Figure 6.3, there are several orders of magnitude difference between the various metrics and workflows. However, a few key properties of the workflow can be gleaned from inspecting these metrics.

As shown in Figure 6.3 panel B, the overall signal in the spectra tends to decrease from the beginning of the reconstruction through the end. Why should this be the case? As described in Table 6.1, the Solvent Suppression actor (SOL) is applied specifically to remove unwanted solvent signals. This naturally leads to a drop in the overall signal strength. Additional processing along dimension 1 has little effect on the overall signal. However, the Linear Prediction actor (LP) also appears to reduce overall signal. This observation may appear strange – the purpose of linear prediction is to add signal, not reduce it. However, when all actions

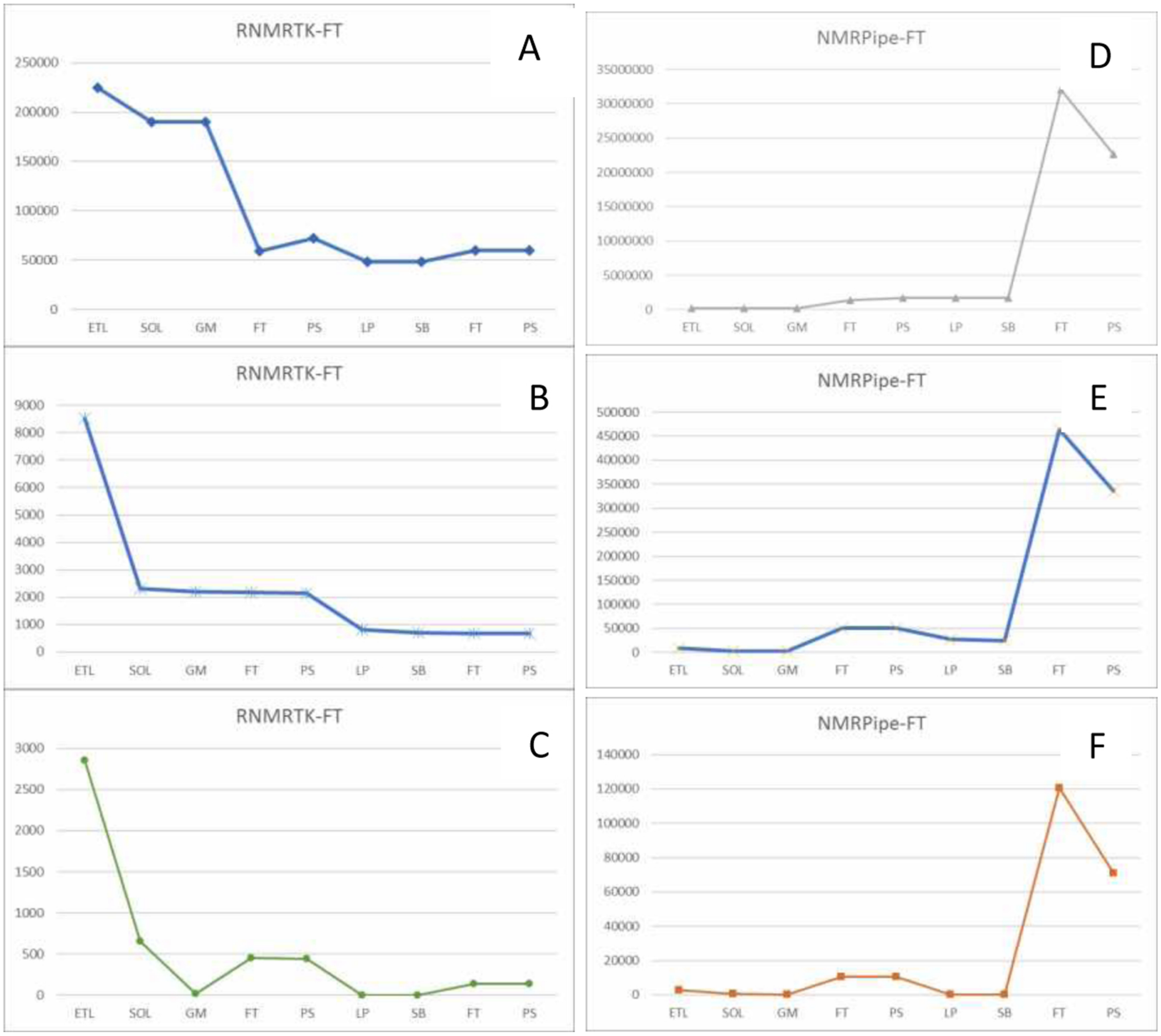


Figure 6.3: Embedded Analytics reported by CONNJUR Workflow Builder. There are three metrics for each step in the workflow, maximum signal (panels A and D), average signal (panels B and E) and approximate noise level (panels C and F). The left-hand panels (ABC) refer to one fork of the workflow in Figure 6.1 in which the Fourier Transform is handled by the Rowland NMR Toolkit (RNMRTK) software, while the right-hand panels (DEF) use the NMRPipe FT. All metrics are measured in arbitrary units for the vertical axis. Please note that due to the nature of the metrics and the data, the scales are very different for each figure, where the maximum value varies by several orders of magnitude. BioNMR data are typically analysed relative to one another and therefore it is the step-to-step changes which are of interest. The horizontal axis is labelled in abbreviations for the steps listed in Table 6.1.

on a particular dimension have been completed (at the point of the LP actor in the workflow), the imaginary components of the dataset are removed resulting in a slight drop in measured signal.

The noise metric is shown in Figure 6.3 panel C. It appears that the solvent suppression algorithm removes both unwanted signal as well as some noise. The two apodization steps (GM and SB) also have the result of reducing the noise component, which is their intended purpose.

An intriguing result is observed when comparing panels B and E. Panel B shows the reconstruction using the RNMRTK implementation of the Fourier Transform while Panel E shows the NMRPipe implementation. (Again, please note the difference in scale.) What is striking is that signal increases by an order of magnitude for each invocation of the NMRPipe FT. This is illustrative of a difference in the way the Fourier Transform is calculated, or more specifically, the way it is normalized. In the case of RNMRTK, the normalization of the FT is such that the average signal intensity remains the same after the transform. This is clearly not the case for the NMRPipe implementation. However, as mentioned earlier, since bioNMR data tend to be analysed in arbitrary units, such arbitrary scaling has no effect on the end analysis; it only complicates exploration of the workflow using these metrics.

6.5 Conclusions and Future Directions

CONNJUR Workflow Builder has been extended to provide analytics during workflow execution. The values of these measurements are recorded within the larger provenance document using a combination of the PREMIS framework along with domain-specific metadata. This provenance record can be exported separately or can be bundled with the processed dataset to encapsulate a proper research object. This research object not only contains the data and the provenance required to replicate the computation, it also contains important analytics on all transiently-existing, intermediate datasets. This can be probed and visualized as shown in Figures 6.2 and 6.3.

The exploration of analytics within this workflow management environment is still ongoing. These original metrics have been designed to be useful in a broad context. That is, the metrics are meaningful whether the data is hypercomplex, complex or real, and whether the data is in completely time domain, completely frequency, or a mixture of the two (termed an interferogram to the domain experts). The PREMIS provenance record is augmented with a domain specific XML which is linked through the extension fields built into PREMIS 3 [34]. This domain specific XML allows for reporting either global metrics as presented in this paper or axis / dimension specific metrics (in progress). Future directions also include exploring context sensitive metrics, meaning metrics which are only useful dependent on the precise type of the dataset (for example, frequency data only). It is anticipated that these metrics will also be useful to capture; however, perhaps for the purpose of global comparison between different processed spectra (as in the FT implementation example above) rather than monitoring changes along a single reconstruction workflow.

Operating within the NMRbox VM environment provides more opportunities to engage the data creators in becoming data curators. The contributions within this dissertation are efforts to simplify (and automate) provenance recording to satisfy the FAIR principles without forcing the data creators to be experts in the nuances of the FAIR principles. As the NMRbox Center customizes the VMs, we also embed custom data visualization and curation widgets[7] within the Xfce desktop manager [38]. The extensibility of PREMIS, the CONNJUR-ML mark-up language, and the GTK+ widget library will allow a gradual increase in the level of curation achievable as well as in increase in the usefulness of the metadata captured. Finally, the ongoing

[7] https://github.com/CONNJUR/CONNJUR_widgets

collaboration with the BMRB will ensure that this important metadata will be included within the public repository to support scientific data reuse. This strategy of measuring relevant properties of objects during a curation workflow and embedding the analytics with the provenance record should be of general use for quality control and provenance tracking in other application domains.

6.6 Extended Conclusions for Dissertation

These two chapters on PREMIS with CONNJUR Workflow Builder resulted from work undertaken from 2016 through 2019. Early results were presented at the Experimental NMR Conference (Douglas Heintz, presenter) and the Gordon Research Conference on Computational Aspects of Biomolecular NMR (myself, presenter) in 2017. The papers were published in the *International Journal of Digital Curation* in 2018 and 2020. Looking back on this work, there are a few additional considerations regarding the choice of using the significant properties extensions within PREMIS, documenting the analytics within PREMIS, and the creation of the .nbx file format which was only cursorily mentioned within Chapter 5.

6.6.1 Significant Properties

As mentioned within Chapter 5, PREMIS 2.0 and 3.0 introduced the capability of including domain-specific dictionaries within the PREMIS framework. There are connection points for these extensions in each of the main entities: Objects, Events, Agents and Rights. For the scientific workflow case study of these two chapters, and in the context of users processing data within NMRbox, rights statements are not of paramount concern as the users operating within the system have the rights over their own data. Rights would be handled when depositing the data to a repository such as the BMRB, although the NMR community expects scientists to provide such data openly without restrictions. That said, the bioNMR XML schema does support Rights statements which may find renewed usefulness with the advent of the Network for Advanced NMR[8] project which is harvesting large swaths of NMR data and will eventually make this data available within NMRbox.

The bioNMR XML schema implemented within CONNJUR Workflow Builder has detailed aspects of bioNMR experimentation / computation covering Objects, Events and Agents. This was presented as a poster at the 2018 iPres conference in Boston. Agents within bioNMR include the person who conducts the experiment as well as the software used for computation and the scientific instrument which collects the data. The latter is quite important within the context of the PREMIS-CONNJUR record as details of the instrumentation (such as the strength of the magnetic field) are critical for understanding the data.

Events within bioNMR data collection refer mainly to the orchestration of the pulsed NMR experiment, which is intricate and complicated and has been referred to as "spin gymnastics"[39] as the pulsed NMR experiment manipulates the state of the quantum spins of the nuclei under investigation. Metadata for events would include the so-called spectral width of NMR experiments, which essentially defines which signals can be observed and/or how they are manifested in the resulting data.

Finally, the objects within the CONNJUR-ML schema refer to the born-digital objects which contain the empirical data as well as the metadata from the experiment. Important domain-specific metadata for these digital objects include the ordering of the numbers in the multiple-dimension arrays as well as the layout of how the hypercomplex values are represented.

[8] https://usnan.org

As for how these domain-specific metadata were packaged within the PREMIS record, for Agent data there was only one choice - the AgentExtension sematic unit of PREMIS. Events on the other hand had two possible extension points: EventDetailInformation (which was the chosen extension point) and EventOutcomeDetail. (Event Outcome is meant more for describing whether an experiment was a success or failure and so was not considered for the Event Detail information).

It is the Object extension which - upon reflection - may have been a mistake. In crafting the metadata for NMR datasets, the metadata seemed naturally to describe the significant properties of the dataset. For instance, distinguishing a one-dimensional dataset from a two-dimensional dataset certainly is a property of the data and a very significant one.

However, this is not the accepted meaning of “significant property” in the digital preservation community from which PREMIS derives. In that context, a significant property is a property which should not change throughout the digital processing pipeline. That is very different from the metadata in the CONNJUR Object data model where our “significant properties” are the properties which *do* change over the course of the processing workflow and whose change we consider significant. Of course, this is the entire focus of this chapter on embedding analytics within the processing platform and recording those results in the PREMIS record.

In retrospect, these data would be better suited to the ObjectCharacteristics extension within PREMIS so as to avoid confusion with the accepted meaning of sigProps within digital preservation. ObjectCharacteristics is already home to the file format and size and so the layout of the data within the digital object would be appropriate in this location. If given the opportunity to refactor the existing application, this change is something I would like to make.

Of note, while including Events, Agents and Objects, PREMIS has no notion of Subjects - as in the subject matter of an experiment. Chapter 8 will consider significant properties of biochemical samples and the conundrum of defining properties which cannot change for things which cannot avoid changing. It will be argued that in such a situation, provenance itself can be a significant property.

6.7 Analytics: Metadata About Metadata

As mentioned in the summary of PREMIS at the end of Chapter 4, PREMIS provides XML blocks for each of the four top-level entities of Objects, Events, Agents and Rights. A provenance record can be stitched together by linking these blocks to form a graph. For instance, if Object 1 was processed by Event A to produce Object 2, Object 1 would link to Event A, Event A would link to Objects 1 and 2, and Object 2 would link to Event A. In this way, the provenance graph can be queried in any direction from any node.

Recall also that in the preceding section it was mentioned that the software which processes the bioNMR data is documented as a PREMIS Agent. So if Event A was a Fourier Transform of Object 1 - Agent FT is linked to Event A to document that relationship.

This works fine for computational workflows in digital preservation as well as the spectral reconstruction workflow of bioNMR data processing. However, the introduction of analytics in this chapter introduces a problem with this linking structure.

For the analytics, the metric which is reported is a characteristic of the object (or a significant property of the object, if you wish). Different aspects of the same object can have different measurements which are made by different software applications. In this scenario, how do we link the software agent with the analytic which is reported?

One possibility is to expand the representation of the workflow. For instance, if characteristic X is measured for our objects at each step along the workflow, the new pipeline would be Object 1 → Measurement M(X) → Object 1' → Event A → Object 2 → Measurement M(X) → Object 2'. (In this example, the ' signifies that the object itself has not changed but there is an addition to the metadata record.) In unrolling the workflow this way, the agents which do the measurements can become bone fide agents in the PREMIS record and they can be linked to these additional events. However, this results in a complexity in the reporting of the workflow, particularly for the user who wishes to focus on Event A and not on the extra Events M(X) which are added by the software.

A second possibility is to leave the workflow as the user defined it but to allow linking of software agents not only at the level of Events and Objects, but also to the particular piece of metadata (the analytic measurement) within an Object. This can potentially be done formally within PREMIS as Objects can have associated Environments. However, since the analytic metadata is defined within the CONNJUR schema embedded within ObjectCharacteristicsExtension - the linkage can be supported by the domain-specific language.

6.8 The "NMRbox" format: .nbx

The final section in this expanded conclusion is about packaging the PREMIS documentation with the NMR dataset. On the one hand, this might be considered superfluous. The PREMIS XML record stands on its own - as long as there is a reference ID to the dataset to which it refers, it can exist as a separate file.

However, it is also clear that keeping a set of files together is not always guaranteed and if the reference ID is to something local - like a file path in a directory - then if the files are moved and the PREMIS provenance record is separated from the data, it becomes useless.

NMR data exists in many different specialized file formats, either from the spectrometer vendors (Bruker, Agilent, JEOL, TecMag) or from various software developers who created ancillary formats for the data to support their applications (Roland NMR Toolkit, NMRPipe, NMRFx, Sparky, to name a few).

Prior to my studies at Illinois, I led a project which built a software application for translating between these various formats[35]. In the course of that software development, we surveyed the various file formats which are all charged with storing multi-dimensional arrays of numbers. In short, some formats maintain the data and the metadata in a single binary file (where the metadata is encoded using some header schema), other formats maintain the data as a binary object with the metadata in one or more ancillary text files with custom schemas, and still others use a hybrid approach where there can be multiple text metadata files with some of the metadata being encoded as headers or footers with the actual data.

There is an XKCD comic (927: Standards) which jokes about how a universal standard was created to resolve the 14 existing, competing standards and now there are 15 existing, competing standards. Well, in Chapter 5 we were confronted about how to include the provenance metadata with the NMR data which conformed to the existing standards. The existing file formats which embedded metadata in binary headers and footers were not an option as those formats are not extensible.

One format however, the RNMRTK format, stores the NMR data in two files: one which only includes the binary data without headers or footers and the other which is text with a small amount of metadata, including the layout of the binary file. This seemed perfect for our purposes as the PREMIS record could simply be a third file which sits alongside the other two.

Yet this brings us back to the earlier problem which is what if either the PREMIS record or the RNMRTK text file are separated from the binary file? In the case of the former, the provenance is lost. In the case of the latter, the data may be uninterpretable.

We decided to formally codify a solution for the NMR community which is not novel in any way. Rather, it is the common solution to this problem. The specification for the .nbx (NMRbox) file format is to use the RNMRTK format (one data file, one metadata file), add as many self-documenting metadata files as you wish (infinitely extensible) and tar them all up so the files stay together. This is essentially the strategy for docx, xlsx, pptx, etc. except we chose tar rather than zip for packaging the files together.

While this is not a novel solution, I do think this is a very important behavior change for the scientific community to try to adopt. We want metadata solutions which are extensible but also not prone to failure and this strategy is effective. Also, since there exist libraries for extracting sub-files from within zip and tar containers, it is not a burden on software developers who wish to write code which interacts with the underlying data. Also, zip and tar are not the only technological solutions for this; Hierarchical Data Formats (HDF5) could also be used.

This topic made it into the talk given at the International Digital Curation Conference in 2018 but due to space restrictions, was not described within the paper (Chapter 5).

Part III

Information Modeling

While standards such as PREMIS and PROV are capable of recording many aspects of retrospective and prospective provenance, there are still important aspects of provenance which seem to be either overlooked or beyond a simple capture with these standards. The marrying of prospective and retrospective provenance represents a case of this in which various bridge models have been proposed (PROV-ONE, P-Plan, OPM-W). However, retrospective (past) and prospective (future) only represent two tenses of story-telling which may be important for provenance documentation. The final chapters present use cases and argumentation for the need for at least a third provenance "tense", subjunctive provenance, which is need for documenting conditional provenance – the provenance of computational results that could have come to be but may or may not have been realized.

Chapters 7 introduces the need for subjunctive provenance through a fictionalized use case involving a scene from the American sitcom, How I Met Your Mother and a piece of IKEA furniture. It also begins to more forcibly argue that provenance is not inherent to objects but rather is a story told about an object. The accuracy, persuasiveness and trustworthiness of that story may be hindered by our consideration of only the past and future without the conditional tense.

Chapter 8 continues the discussion to include consideration of identity in provenance records, such as when the same thing is both an agent and object of the provenance documentation and the thorny issues of when a thing referred to throughout a provenance story is substantively changing throughout the story lifespan.

Chapter 9 discusses information models which differ by the reification of concepts within the model. This is a lead-in chapter to the following chapter on Concept Keys which exploits reification to solve a problem with specialization lattices.

Chapter 10 discusses an old problem in object-relational mapping concerning the representation of subclasses (subtypes) in a relational database system. While there are four standard methods of treating subclasses, it is argued in this chapter that they become unwieldy for specialization lattices (which in object-oriented programming leads to multiple inheritance). A fifth method is introduced - termed Concept Keys - which is an attempt to provide a more flexible manner for managing multi-parent hierarchical information models. It is argued that Concept Keys provide a "normal form" which enriches the existing standards (W3C PROV and PREMIS) to allow for expressing the various temporal aspects of prospective, retrospective, subjunctive provenance and beyond.

These chapters represent the following contributions to the field of Library and Information Science:

- Subjunctive provenance. Identified the need for more temporal expressions of provenance than simply past and future tense. Need the conditional or subjunctive tense as well.
- Subjunctive provenance could be used to compare what could have been to what should have been for quality assessment and error detection.
- Subjunctive provenance can also be used for documenting provenance during the execution of an iterative workflow.
- Introducing provenance as a significant property of an object as related to digital preservation. How can we properly deal with objects which cannot be preserved - that are destined to change over time. How can we meaningfully apply identity constraints to mutable objects?
- Identifying a problem of existing relational modeling techniques for multihierarchical models with many layers of subclasses

- Introduced concept keys as a novel ”normal form” to tackle this problem. By altering the reification of the model we can interconvert between needing hundreds of subclasses to a half dozen concepts which discriminate the subclasses.
- Proposed concept keys as a more general solution for merging prospective, retrospective, subjunctive and other temporal forms of provenance documentation.

Chapter 7

Subjunctive Provenance

7.1 Abstract

Provenance is the story of objects: how they have come to be, what they could have been, what they will be. This paper explores the temporal complexity of provenance and suggests the need for the concept of *subjunctive provenance*. Using the example of building an IKEA LACK table, the authors explore the concepts of retrospective and prospective provenance to highlight gaps and the potential for subjunctive provenance.

7.2 Introduction

Documenting provenance is a core concern in information science. In archives, functional provenance documents the origins of materials [40]. In metadata, provenance elements and properties document process information about digital objects to ensure long-term accessibility [41]. In data curation, provenance helps establish trust in data as it is (re)used [42]. This wide-ranging body of information science research has yielded several provenance documentation models. An OCLC/RLG working group developed PREMIS, metadata for digital preservation. The eScience community developed various provenance-related models, namely PROV and its subsequent *PROVlets*.

Different subfields center different temporal aspects of provenance. History-oriented subfields like archives and museum studies focus on reconstructing the story of how an object came to be (or *retrospective provenance* [43]). In such cases, provenance describes how an object was created and how the object has traveled through space and time to the present. Showing unbroken chains of custody from an artist's hands to a museum curator's, for example, helps establish that a painting is real rather than a forgery. In curation and e-science, models and tools foreground a set of stories about what can or will happen (or *prospective provenance*[43]). Here, provenance documentation speaks to potential futures, what steps could be taken to rerun, recreate or extend upon an experiment. The relationship between retrospective and prospective provenance is a complicated one: provenance documentation travels backward and forward in time. This paper was inspired by the following provocations::

- When do the links between prospective and retrospective provenance break? What can be done to prevent/identify such separation?
- How can retrospective and prospective provenance be distinguished and applied to different areas?

We argue that these two tenses, the retrospective and prospective, are not sufficient to describe the temporal richness of provenance. We propose the term *subjunctive provenance* to speak about non-actual events that are contingent and dependent. This study addresses these questions using the all-too-common challenge of building IKEA furniture as a toy model for exploration and analysis.

7.3 Context

7.3.1 Retrospective Provenance

Retrospective provenance is the story of how an object has come to be– its history in the world. In art history, researchers are tasked with constructing a narrative about what happened to artworks in the past, sometimes stretching back hundreds of years [44]. In archives, retrospective provenance operates in relation to concepts like the 19th century French principle of *respect des fonds*. Because many archival records derive from quotidian practices of institutions, in many cases the need to find ownership traces is less relevant than in art history. In its earliest iterations, provenance in archives was tied to the standardization of organization in European archives that privileged original order, as opposed to a library-like organization that sorts materials according to external categories like Dewey and LCSH designations [45]. Challenges in documenting provenance arise where original order is disturbed, when changes to the organizational structure of the institution means the order of the archives also must change, or when provenance documentation for an object is missing.

Cook argues that we need to know about the creator of the archival records; the creation of the record itself becomes bound up with the larger provenance story of an object [46]. Archives scholars introduced secondary provenance to expand the definition of provenance further, to encompass layers of context surrounding objects [47], [48]). Nordland [47] examines how archival records change over time as they are reinterpreted, and Conway employs secondary provenance to address the need to re-imagine provenance in a world of digital surrogates where the original objects might no longer be present. Conway's work [48] also invokes the work of digital humanists such as Kirschenbaum and Drucker who raise concerns about digital materiality [49], [50]. Kirschenbaum's work [49] on digital traces brings retrospective provenance into conversation with e-materials: the speed of creation and volume of e-materials pose a challenge to older archival approaches to provenance based on *fonds*. Digital humanities work also shares concerns with media history and media archaeology work. While this field rarely labels their work as provenance work, the construction of the context in which media objects are made and deployed, such as Robertson's [51] work on filing cabinets and Sterne's work [52] on MP3s combine the contextual approaches of archives, the material traces of digital humanities, and the historical research of art history.

Yet while most of the literature about retrospective provenance comes from history domains, the germinal literature in these areas rarely uses the *retrospective* qualifier. Provenance in its original use cases is, by dictionary definition, retrospective[1]. This terminological distinction originates with scholars in the e-sciences in particular, who need to differentiate between the recipe (prospective provenance) and the runtime (retrospective provenance) in scientific workflows [18].

[1] https://www.oed.com/search/dictionary/?q=provenance

7.3.2 Prospective Provenance

While retrospective provenance is about what happened, prospective provenance can be thought of as a recipe for how to make something happen, or how an object will come to be. Formal recipes, in the form of cookbooks, date back to 1700 B.C.; formalizing recipes into workflows became popular in the computer age, stemming from punch cards in the 1700s and expanding into computational sciences in the twentieth century. The advent of scientific workflow management systems gave rise to a need to standardize workflow specifications. The Open Provenance Model for Workflows (OPMW) and ProvONE were responses to this need [53], [54]. These models allow for the descriptions of the data, processes, and agents that will be used to perform computations.

Computational models link prospective and retrospective provenance. The workflow specification is a description of how a computation can be performed in the future. Retrospective provenance is the description of how a particular computation has actually been performed. The PROV and ProvONE standards center the importance of maintaining connections between these tenses, and each standard has elements for expressing both retrospective and prospective concepts [53], [55].

Prospective provenance refers to workflows, plans, or recipes [53], [54]. In doing so, it sometimes elides the space between what will happen and what could happen. Some recipes include a finite set of linear steps: in this case, barring equipment failure, the workflow documents what *will* happen. Prospective provenance is also used to refer to situations with branching steps: workflows where agents, human or otherwise, can choose Path A, leading to one set of steps, or Path B, leading to a separate set of steps. These branches might be formalized: a program can include two protocols and it might run each a number of times to generate data for comparison. This process is a simplification of what happens frequently in computational or laboratory experiments. In everyday examples, like cookery recipes, written instructions might not present multiple optional protocols, but people cooking in a kitchen informally add steps, diversions that *could* happen. These conditional changes to the plan occur when, for example, substituting gluten-free flour for wheat because of an allergy or forgetting to add fresh parsley at the end.

While computational sciences tend to combine the *could happens* with the *will happens*, we argue that these two tenses, the conditional and the future, merit separation. If prospective provenance, by definition, is about what will happen in the future, then we propose the term *subjunctive provenance* to speak about what could happen. In the following section, we explore this full temporal spectrum of past, conditional, and future using a simple toy model: building an IKEA table.

7.4 Use Case: IKEA as a Microcosm

> "I'm supposed to attach a brackety thing to the side things using a bunch of these little worm guys. I have no brackety thing. I see no worm guys whatsoever. And I cannot feel my legs."
>
> –Ross Geller in the pilot of *Friends*, building IKEA furniture

Provenance concerns abound in many disciplines, and part of what impacts the diversity of definitions and approaches is the objects to which these provenance stories attach. The provenance of wine may look quite different to the provenance of seeds or the provenance of a nuclear magnetic resonance spectroscopy analysis process, even while we argue that all these conceptions share a foundational definition: provenance is a story of how objects come to be [8].

We offer a tangible example for exploring the temporality of provenance: building IKEA's LACK coffee table. The scenario is derived from the sitcom *How I Met Your Mother* (*HIMYM*), when the characters Marshall and Ted duel with swords to determine who gets to keep tenancy of their shared apartment (Figure 7.1). Marshall hops onto a coffee table to gain an advantage. A flashback interrupts, showing the characters building the coffee table when they first moved in. Unable to locate the final support screws, they used wood glue to complete the table's construction. Post flashback, the improperly constructed table collapses under Marshall's weight. He falls as his partner, Lily, walks in and she is stabbed by the sword.

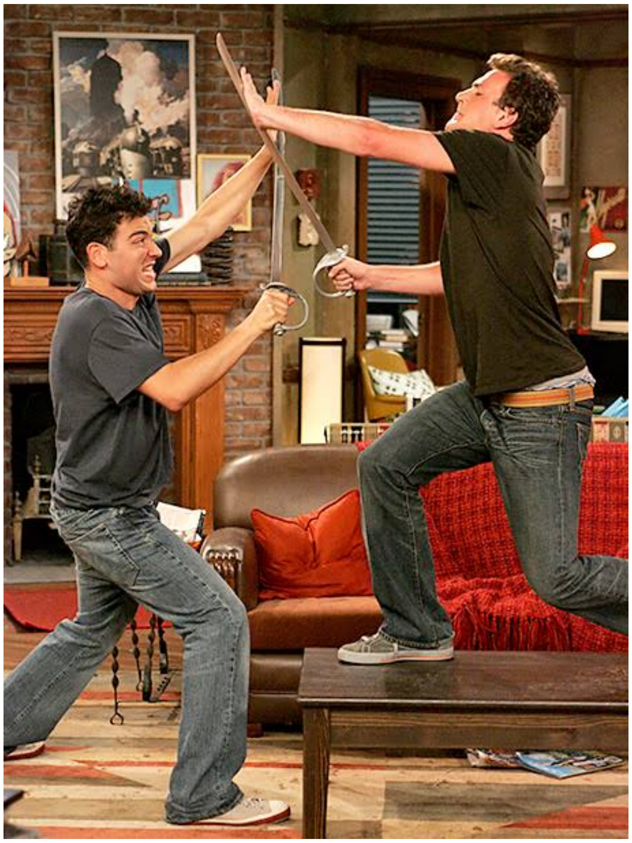


Figure 7.1: Marshall and Ted's sword fight in season 1 episode 8 of *How I Met Your Mother: The Duel*. Retrospective provenance documents that they did not precisely follow the prospective instructions for assembling the table. Exploring the consequences of deviations between prospective and retrospective provenance is the realm of subjunctive provenance.

Through this scenario, we will explore retrospective and prospective provenance. By digging into the flashbacks and flash-forwards in *HIMYM*, we will begin to unpack the full temporal spectrum of provenance.

7.4.1 Retrospective Provenance

The retrospective provenance of an IKEA table is an account of the steps taken when assembling it. Whether the suggested instructions were followed does not matter: the task is to adequately document what actually happened. If the builder follows each step in the manual without deviation, documenting provenance is likely informal. It might include ticking off steps on the manual or posting an online review stating that the table was successfully built as instructed.

In *HIMYM*, the retrospective provenance story is recounted in the flashback: Marshall followed the written instructions up until the last step, whereupon he used wood glue in place of the final screw. However, much like reconstructing the history of a mysterious painting, if we had to tell this story in the absence of the flashback, it would take some detective work. In this scenario, asking Marshall how he built the table might elicit a response much like the flashback: Marshall would explain about the manual, the screw, and the glue. However, the remembrance of the wood glue was predicated on exceptional circumstances that occurred when the table collapsed: the structural failure of the table and resulting stabbing brought to mind the table's history. Had the table held Marshall's weight or had the sword fight not occurred, the fact that the table was short one screw might have been forgotten in time. Marshall might tell a friend that he built the table in accordance with the manual; in the absence of a memorably difficult journey building the table, like Ross's negative experience from *Friends*, it's entirely possible that the glue might be left out of the provenance story. The hypothetical is an important one in the realm of retrospective provenance: when reconstructing the past, even the relatively recent past, small details that might later become important can be overlooked, intentionally or unintentionally.

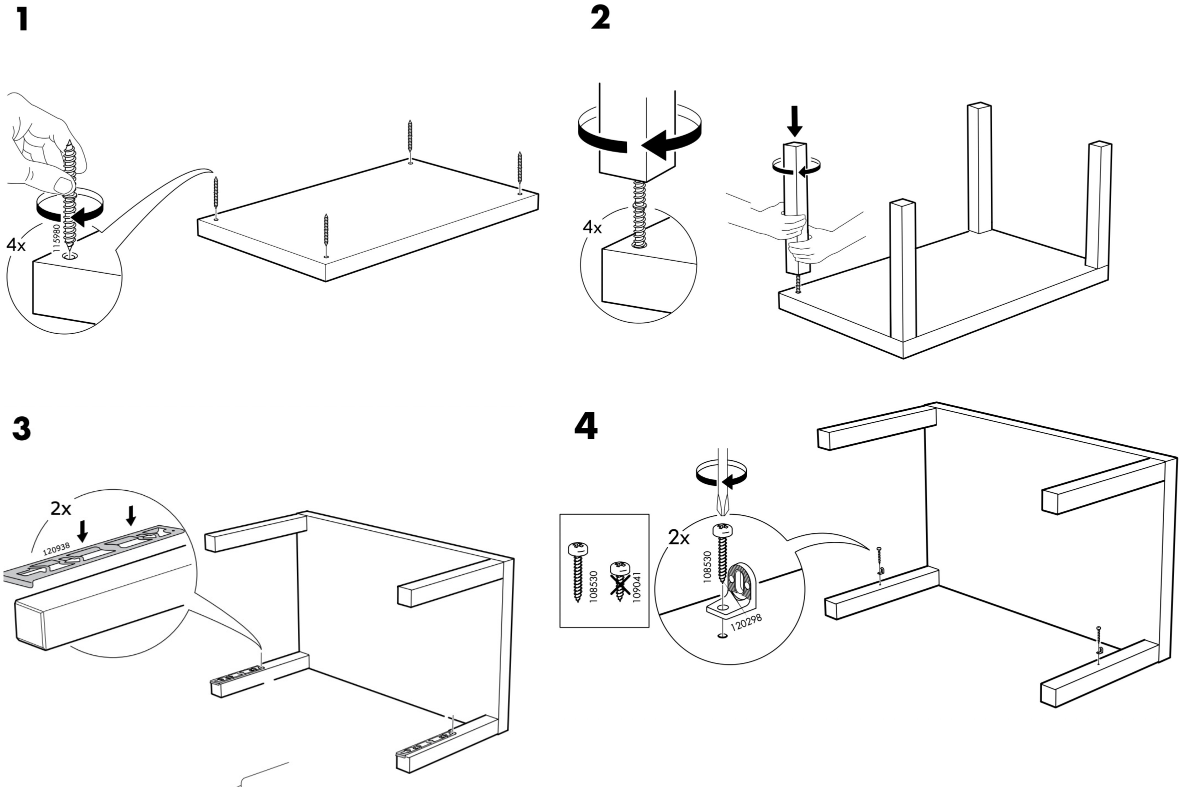


Figure 7.2: Four of the eight steps for building an IKEA LACK coffee table. Source: https://www.ikea.com/us/en/p/lack-coffee-table-black-brown-40104294/

Prospective provenance deals with how things will come to be. In the *HIMYM* scenario, we can flash even further back to when Marshall and Ted begin to build the coffee table. They use a manual like that of the IKEA LACK table (Figure 7.2). There are eight steps involved. Each step demonstrates which furniture pieces are involved, the literal nuts and bolts to complete that step, and the actions required (e.g. rotate the dowels). The steps to building are a simple form of *prospective provenance*.

Yet what seems simple rarely is: what happens when everything doesn't go to plan? A common example is the "tools not included" problem, in which someone doesn't have a Phillips-head screwdriver and so uses a flat-head, or even a butter knife. These deviations to the prospective provenance impact the retrospective provenance: the original prospective provenance in the form of the manual was not inclusive of everything in the real-life workflow. Without documentation, these deviations can be forgotten in the retrospective provenance. Replacing the screwdriver is a process alteration. It is also common to alter underlying materials. Ross lacks brackety things and worm guys; Marshall cannot find screws. Lost screws are omitted; replaced with spare screws or entirely different screws; or replaced by another fastener like wood glue.

Recording real world provenance like this gets messy. PROV has Prov:activities with associated plans, but how do we document deviations to the plan at all the various levels of abstraction between entities (objects), agents (builders), and activities (events)? Is there a mechanism for crafting and future-proofing prospective provenance to allow for common variations, like the simple culinary case of roasting time being proportional to turkey size? Reviews of products on furniture websites commonly feature users who document challenges specific to building a particular piece of furniture, like needing a hammer to insert dowels or a drill when pre-drilled holes do not line up even when the instructional manuals do not call for these tools. Responses to these comments demonstrate that other users adopt these process and material alterations as part of their own prospective provenance.

When building IKEA furniture, many users collect multiple plans before proceeding: this might include the official manual, YouTube videos, a toolbox, recommendations from reviews, and their own building experience. Models like PROV and the PROVlets account for plans; reality is an amalgamation of plans that *could* be executed in many overlapping configurations. How are these multiple possibilities documented? How can we model more comprehensive prospective provenance that better resembles real-world scenarios?

7.4.2 Subjunctive Provenance

Had Marshall and Ted followed the steps for building the table correctly, as depicted in Figure 7.2, Lily might not have been stabbed. However, most of the time, discerning the exact time-slice that led to an error is difficult. While the flashback details what went wrong with the table's construction, in reality finding the precise step that lead to its collapse would be challenging.

Retrospectively identifying "what could have been" from a step performed in a *prospective* plan is what we call *subjunctive provenance*. We use the term subjunctive in reference to a verb tense that describes non-actual scenarios: subjunctive provenance is the documentation of the plans that *could have been* taken. It enables stories about what we *were* going to do. Separating the possible from the prospective provides a means to examine the temporal complexity of provenance. For example, subjunctive provenance can be useful for identifying errors by investigating possible paths a step could have led to. Stabbing Lily makes Marshall and Ted remember what they did wrong (*retrospective*), which calls into being the imaginings of what could have been (*subjunctive*): perhaps they made an error as early as step 1 in the manual, or they used shorter screws when they should have used longer ones. When prospective provenance is not followed properly, it is possible to look back when something goes wrong, which in turn brings up arguments about what would have happened if the plans were followed correctly.

It is also possible that a well-built table is still breakable. Had Marshall found that final screw, the table still might not have been able to hold Marshall without breaking – he's nearly two meters tall and 95 kilos. The subjunctive is helpful in discerning what provenance needs to be documented. The lives of objects can be long. At some juncture, custodians must be judicious about what to include in metadata. Changes to process and materials that do not materially alter the outcome may be rightly deemed unimportant in telling the story; even without the wood glue, Lily was going to be stabbed once Marshall hopped on the table with a sword.

The *HIMYM* example illustrates how subjunctive provenance is a useful concept dealing with situations that have already come to be, like error detection; selecting pertinent information to include in provenance documentation; and documenting the real and perceived affordances of digital technologies. Subjunctive provenance can also be forward-looking. It provides a framework to think about the complexity of the possible paths that could be taken, which can be helpful in scenarios like calculating risk and version control. Like a Bayesian inference where hypotheses based on past events are updated as new information becomes available, subjunctive provenance covers the temporal space between the past and the future. It bridges decision-making based on what has already happened with expectations about what will happen in future.

7.5 Discussion and Conclusion

Documenting provenance is a form of storytelling: the future is talking to the past when prospective plans become retrospective history. Simultaneously, retrospective provenance must persuade into the future: the current misinformation crisis demonstrates that provenance that seems convincing to one person may not be to another. The challenge for the person telling stories to the future is that they are speaking to a group of unknowns: it is not real-time communication. Subjunctive provenance enables a broader picture of the audiences who might be listening to the provenance story in the future. Taking these audiences into consideration when making documentation opens the possibility of better communication across time. If provenance is a story of how something comes to be, subjunctive provenance is a method of future-proofing: it can help reconcile the future and past object.

We began with provocations about the breakdown of links between prospective and retrospective provenance. We've shown that the challenges of persuading future listeners and documenting plans in the inherent complexity of the real world reveal that the temporal spectrum of provenance is rich and insufficiently defined by retrospective and prospective concepts. We identified subjunctive provenance as a potential bridge between these existing conceptions. We also applied this concept to the IKEA scenario to distinguish what subjunctive provenance can accomplish. The separate articulation of this concept suggests that future avenues of research are needed to explore how to document subjunctive provenance in ways that aid broader provenance practice.

7.6 Extended Conclusions for Dissertation

This chapter presented an argument for the utility of considering subjunctive provenance through a whimsical use case of constructing an IKEA coffee table. It is useful to contemplate the subjunctive nature of provenance when working with computational workflows as well - such as those found in Chapters 5 and 6 in the context of provenance recording for CONNJUR Workflow Builder.

The CWB program manages workflow execution by simultaneously wrapping third party tools through an abstract object layer within the Java application as well as checking that the output of a preceding actor is suitable as input to the next actor. Beyond following the direction laid out in the workflow constructed by the user, CWB does not provide any processing logic, such as examining the data mid-workflow to automatically determine the configuration of subsequent processing steps.

Yet this limitation is not intrinsic to workflow management systems but simply the current implementation of CWB. In a previous version of CWB, a single processing step could be incremented; that is to say, the user could select a parameter for an actor and rather than choose a single value for that step, instruct CWB to run the same calculation multiple times with different values. This feature was analogous to a for loop in conventional programming, with the caveat that multiple datasets were produced rather than having the results summed or averaged in a single output dataset.

In addition to incrementation, one could envision a workflow where one or more actors are optimized during a workflow execution. In this scenario, a downstream analysis actor would measure properties of the computed dataset and apply some business logic to reconfigure an earlier actor. This is illustrated schematically in Figure 7.3.

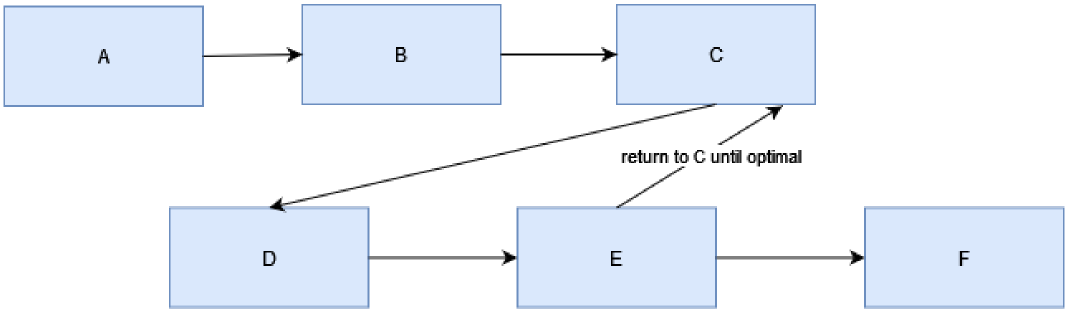


Figure 7.3: Directed Cyclic Graph for an iterative workflow. In this scenario, subjunctive provenance could be used to document cycles which were tried and eventually the results abandoned as part of an optimization.

In this conceptual example, a dataset is processed using six steps labeled A through F. The first two processing steps, A and B, are executed only once. The prospective provenance is the direction to the workflow manager on how to perform these operations; the retrospective provenance is an execution trace of how those steps were performed.

The next three operations, however, can be executed multiple times. In this scenario, the output from steps A-E is analyzed to determine if the output is optimal or at least good enough. If the output meets expectations, it receives the final processing step of F and the workflow is complete. However, if the output of E is not sufficient, the results of E are used to reconfigure processing step C in order to try again. This loop can be rerun dozens or hundreds of time in search of a good result to send forward to step F.

Consider that such a cyclical workflow is executed and at some point the results are sitting at step E. Let's assume this is the 50th iteration and we can call it process E(50) to distinguish it from the first 49 attempts. At this point, there is retrospective provenance of how the data were generated - namely the execution traces of A, B, C(50) and D(50). There is also prospective provenance of how E can be transformed to F and the workflow completed.

In addition to the retrospective and prospective, there is also the provenance of the 49 failed attempts. These attempts happened in the past and so it is tempting to consider them retrospective. However, they also represent options not pursued. Just as Ted and Marshall opted against the final screw and instead used glue, in this scenario the other 49 options represent what could have been. And this subjunctive provenance includes metrics provided by E describing what is to be expected of those failed attempts.

The point of this argument is that there are two viewpoints to this type of iterative workflow. The standard approach would be to view everything as retrospective or prospective. We unroll the loop and state that the retrospective provenance is the execution of A-B-C(1)-D(1)-E(1)-C(2)-D(2)-E(2)-C(3) ... C(50)-D(50). From a temporal perspective the earlier iterations did occur earlier in time than the later ones. Also, the later runs were at some level dependent on the earlier ones as they helped in reconfiguring the successful parameter for C.

However, from a data flow perspective, the data from all of the earlier loops is discarded. There is no data from C(1)-C(49) which makes it into D(50). If C(1) was configured using a random number generator, it is possible that many possible values (if not every possible value) of C(1) would eventually lead to the same C(Final).

The value of subjunctive provenance in this hypothetical example is that it treats the first 49 runs of C-D-E as possibilities rather than as necessary precursors to the 50th. Returning to the HIMYM example, it would be as if 50 coffee tables were constructed using various parts - such as screws, nails, glue, sticky tape - and then stress tested. While the outcome of each individual test would be retrospective, when considering the prospective nature of assembling a new table, these tests provide subjunctive provenance for possible alternatives along with considerations of the outcome.

Computation thrives on abstraction. Workflows are an abstraction. Language is an abstraction. The goal of this chapter is not to prove that subjunctive provenance is critical for some specific set of outcomes. Rather, the goal is to introduce a new abstraction, or combination of abstractions, which may be useful in contemplating and documenting provenance: the stories of how things have come to be, will come to be, and could come to be.

Chapter 8

Provenance as Significant Properties

8.1 Abstract

Significant properties (sigProps) research often focuses on the preservation targets. Yet research consistently shows that what is significant about an object is not necessarily inherent to objects. Simultaneously, sigProps research does not adequately attend to temporality. Time is built into the concept of sigProps: they are about what ideally should not change over time. This paper centers temporality in relation to sigProps to explore challenging case studies.

8.2 Introduction

> Calvin: My past self is corresponding with my future self.
>
> Hobbes: Too bad you can't write back.
>
> –Watterson, 1995

Digital preservation recognizes that long-term preservation entails managed change. Managing change is necessary to ensure that users understand the overarching conceptual object as one and the same over time [18]. The need to imagine and plan for the future is one of the inherent challenges of digital preservation: digital preservationists must think like futurists [17]. Yet the relationship between identity and change is a quotidian concern. The cartoon character Calvin, of Watterson's Calvin and Hobbes series, constantly engages in time travel wherein he interacts with his future and past selves (Fig. 1). This comedic device points to the very real ways in which a person is, at different points in their life, both the same person and a fundamentally different person.

The challenges of identifying that which must change over time has impacts on digital preservation work across disciplinary spaces. In this short paper, we explore two research themes:

Theme 1: In what ways is Past Calvin the same and different than Future Calvin?

Theme 2: How do the nuances that distinguish people over time change when applied to physical and digital objects?

These themes have practical applications for digital preservation. Significant properties (sigProps) are "[t]he characteristics of an Information Object that must be maintained over time..." [9]. The concept of sigProps is both crucial and challenging: the need is acknowledged but the practice is hard. SigProps refer generally to the properties of a conceptual object that are required for its ability to establish its authority in the world. SigProps hinge on two key aspects: objects and time. In this paper, we focus on the temporal aspects and provenance in order to advance the scholarly conversation around the wicked problem of sigProps.

I GOT ANOTHER LETTER FROM MY PAST SELF.
WHAT'S IT SAY?
"DEAR FUTURE CALVIN, I WROTE THIS SEVERAL DAYS BEFORE YOU WILL RECEIVE IT. YOU'VE DONE THINGS I HAVEN'T DONE. YOU'VE SEEN THINGS I HAVEN'T SEEN. YOU KNOW THINGS I DON'T KNOW. YOU LUCKY DOG! YOUR PAL, CALVIN."
* SNIFF * I FEEL SO SORRY FOR MYSELF TWO DAYS AGO.
POOR HIM. HE WASN'T YOU.
© 1995 Watterson/Dist. by Universal Press Syndicate
WATTERSON 4-20

Figure 8.1: Calvin and Hobbes, [April 20, 1995]

8.3 Literature Review

8.3.1 Temporal Provenance

SigProps support inherent change over time. Documenting these changes is part of telling the stories of objects, or provenance. Literature on temporal provenance focuses primarily on the e-sciences domains. Temporal provenance in scientific data is framed as (1) an ordered process based on causal relationships; (2) independent time slices; (3) circular processes.

Provenance models usually express time in an ordered fashion. For instance, in the Open Provenance Model (OPM), a second sequential process can only be initiated after a first process has occurred [15]. This suggests that the processes are directional, forming a directed acyclic, provenance graph.

In defining temporal provenance, Chen et al. discussed the potential of partial ordering of provenance graphs, and how one might be able to partition events into distinct time slices [5]. Similarly, Beheshti et al. proposed the Temporal Provenance Model (TPM) that puts time at the core in provenance documentation, as opposed to other event- or object-oriented provenance models [1]. In the TPM, time in provenance is captured not as a causal event, but as individual time-stamps to allow for versioning control of the same data objects. In this sense, time is an independent variable that partitions data objects into snapshots.

McPhillips et al. developed YesWorkflow, a scientific workflow management system built on the foundational concepts of retrospective and prospective provenance [13]. While retrospective provenance documents the execution, or past occurrences of a program, prospective provenance records the scripts, or the forward-looking recipes that enable a program to run.

Discussions of the temporal dimensions of provenance often center on metadata documentation, not on the data objects per se. Further investigation is needed on the use of temporal provenance to understand how data objects evolve over time.

8.3.2 Necessary Change

The Digital Preservation Coalition defines digital preservation as "...the series of managed activities..." [8]. In discussing artifactual objects, Owens [16] writes, "... what makes Mount Vernon Mount Vernon? Like all physical objects, it is changing at every moment" (p. 16). What are the sigProps of objects that are constantly changing? Historical contiguity is maintained through changes that comport with physical changes already happening: preservationists, digital or physical, roll with the changes that are going to come and make conservation decisions accordingly.

The question here, what makes a thing *that thing*, is central to the foundational understandings of the field of digital preservation. Thibodeau (2002) contributes terminological structure to the idea of the things that are the preservation targets: *that thing* is a conceptual object, supported by a pyramid of logical and physical objects. Preservationists make changes that can alter, re-order, substitute, or otherwise move the logical and physical pieces, while the top-level conceptual object must remain the same for the user in question. This approach mirrors models like the Functional Requirements for Bibliographic Records (FRBR), where the overarching conceptual work has various manifestations, expressions, and items that represent it [10]. The PREMIS metadata model also mirrors this structural approach to delineating that thing with its top-level intellectual entity object type [17]. In the computational workflows of Chapters 5 and 6, the intermediate and final results of the computations are all considered separate PREMIS objects, but they could be considered different FRBR manifestations of a common FRBR work - the original empirical data.

Because of the foundational approaches digital preservation takes to that thing and managed change over time, it is a field that is poised to make broader impacts on issues at the intersection of the identity of objects and time. The following section employs case studies, biochemical research samples and video game franchises, to explore the themes stated at the outset.

8.4 Use Cases

8.4.1 Biochemical Research Samples

There is a renewed push to adopt persistent unique identifiers for samples in the natural sciences [4]. Biochemical samples are often altered, degraded or consumed in the process of a study, introducing the question of whether a persistent identifier is warranted for objects which themselves are not persistent.

In a biochemical laboratory, these ephemeral samples are typically given local identifiers, for instance with controlled experiments on multiple samples which vary in the concentration of a reagent or some other preparation step. This local identifier fulfills two simultaneous purposes: (1) it identifies the physical sample which is part of the experimental workflow and (2) it identifies the significant attributes[1] of this particular sample with respect to the other samples which will be part of the study. In the latter case, a sigProp of the sample is its provenance - what it contains, how it was prepared, how it was treated, how it was stored, as well as temporal issues such as how long it has been since it was treated. Each of these concerns manifest itself on both the physical and concept level. It might be of importance whether a sample was stored at 4°C or at -20°. Alternatively, it might matter that a sample was stored in the 3rd floor freezer because there was a power outage in that room.

[1] The term *Significant Properties* (sigProps) is a digital preservation term referring to the specific properties of a digital object which should not be altered for the preservation to be considered successful. I write significant attributes here to contrast sigProps with this more general concept - the attributes of a sample which the researcher deems to be significant.

All of this is compounded by the fact that biochemical samples degrade over time. Samples age just as Calvin does, yet often on a timescale where the controlled variation between samples may be smaller than the variation within a single sample over time. This leads to some particularly tangled provenance stories when one wants to document the provenance of a sample and the methodology of an experiment in sufficient detail that it can be reproduced by others.

8.4.2 Super Mario

The previous case looked at the mechanics of organic change and the implications for identifying biochemical research samples over time. This section explores a socio-cultural example of the same phenomenon in the evolution of popular media figures over time, drawing from the work of McDonough and the Preserving Virtual Worlds grants [2,11,12]. That thing is Super Mario (Fig. 2), the Nintendo character who features in many media, starting with the Donkey Kong arcade games in the early 1980s. The work of Preserving Virtual Worlds (PVWI and PVWII) is foundational to video games preservation. Two key findings that arose from PVWI are that (1) preserving interactive digital media requires a more systemic approach to determining sigProps even while acknowledging that (2) the preservation of popular games defies universal solutions.

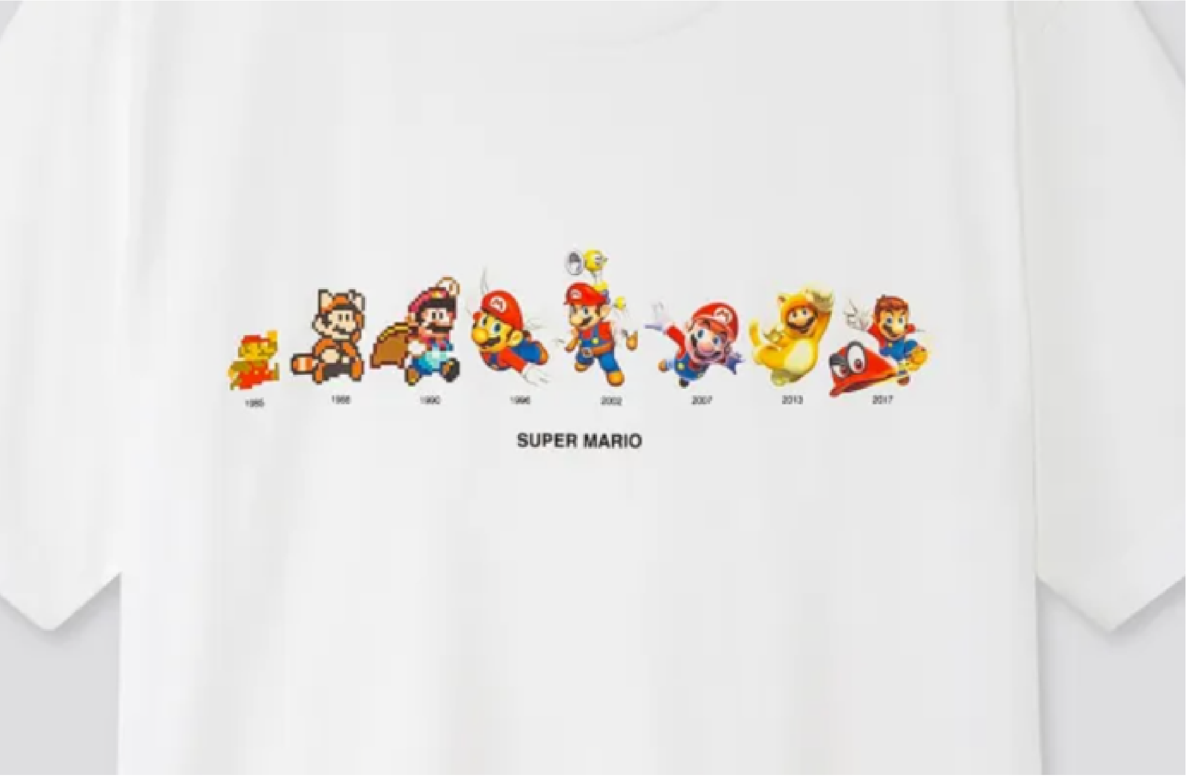


Figure 8.2: Uniqlo Super Mario 35th Anniversary T-Shirt depicting iterations of the character spanning the years 1985-2017, released in 2020

PVWII identified the technical layers that make up a digital game as part of locating those sigProps. These layers include: the hardware/processor; the firmware; the software support; the physical; the application; and the experience layer [2]. Technological capabilities play a role in character design. Early design was frequently defined by the pixels and colors that fit within the storage and processing limits. Early Mario is pixelated in red, brown, and peach in 1988's Super Mario Bros. (Mario 1). 2022's Mario + Rabbids Sparks of Hope is three-dimensional and brightly colored, wearing the iconic blue and red outfit (Fig. 3).

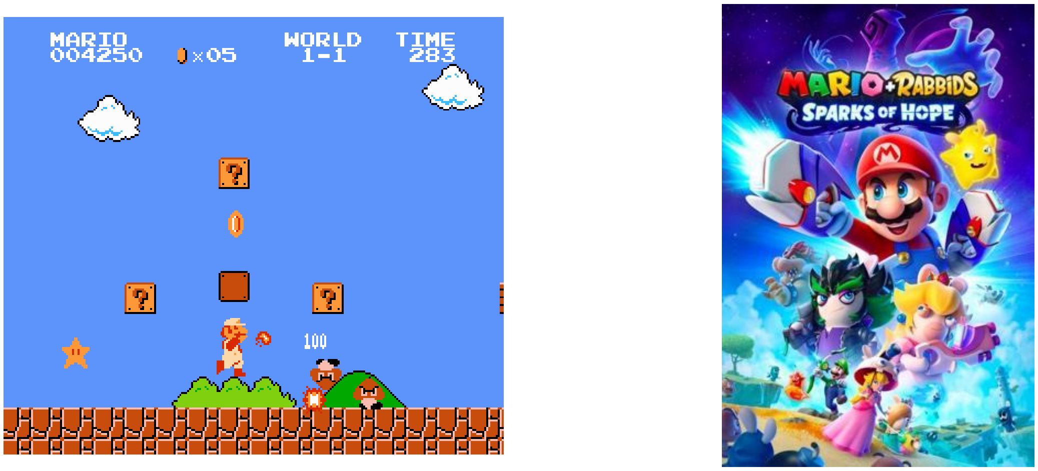


Figure 8.3: Fig 3. Super Mario Bros. (1985) and Mario + Rabbids Sparks of Hope (2022); images drawn from Wikipedia, image rights belong to Nintendo and Ubisoft.

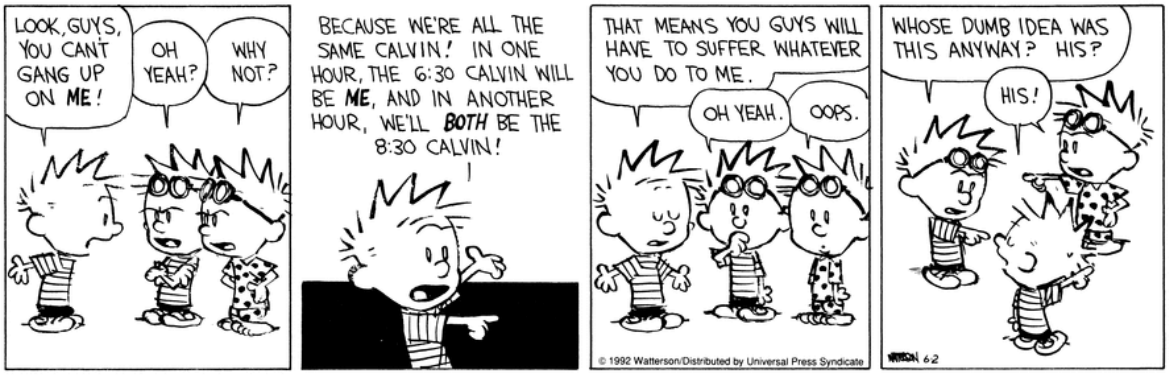
At every layer of the technical stack, these versions of Mario are vastly different across a span of 37 years, including the processors, peripherals, displays, and experiences. Experiential differences are important, because this is where many users find the conceptual object in gaming. That it is possible to take the technological stack of the Switch and approximate the experience of Mario 1 via Nintendo's emulator indicates that underlying physical and logical pieces can change while the experience of that thing remains largely intact: this is a manifestation of sigProps in practice.

This case study is about the relationships between various manifestations of Mario (Fig. 2). Much as biochemical samples and Mount Vernon change over time, so has Mario over nearly four decades. When biochemical samples change in a lab context, the experiential differences might arise from their behavior in experiments. Marios differ in many ways over time. How and why do players recognize Mario as Mario? Part of the answer lies in how people make meaning of information. Clement traces how meaning is included in early information theories and she argues that users make meaning with information, rather than it being inherently meaningful [6]. Marios remain Mario not just because of inherent characteristics like his blue and red costume, but because of meanings that come with interaction. The colors of Marios' costumes evoke a Mandela Effect: even when his outfit isn't actually red and blue, like in Mario 1 or 1988's Super Mario Bros. 3, players remember Mario as red and blue.

McDonough notes that, "... [the p]reservation of computer games is in many ways a knowledge management problem, and without adequate metadata, managing the knowledge necessary to keep a game accessible and understandable is an insurmountable task." [14] This metadata is a form of provenance, and it must incorporate time: temporal framing for the objects and the temporal provenance that documents change in a way that enables objects to establish and maintain authority.

8.5 Discussion and Conclusion

In a series of 1992 strips, Calvin attempts to avoid homework by time traveling to find a future Calvin who has already done it (Fig. 4). Unlike the arc where Calvin had a one-way conversation with himself via snail


LOOK, GUYS, YOU CAN'T GANG UP ON ME!
OH YEAH?
WHY NOT?
BECAUSE WE'RE ALL THE SAME CALVIN! IN ONE HOUR, THE 6:30 CALVIN WILL BE ME, AND IN ANOTHER HOUR, WE'LL BOTH BE THE 8:30 CALVIN!
THAT MEANS YOU GUYS WILL HAVE TO SUFFER WHATEVER YOU DO TO ME.
OH YEAH.
OOPS.
WHOSE DUMB IDEA WAS THIS ANYWAY? HIS?
HIS!
WATTERSON 6-2

© 1992 Watterson/Distributed by Universal Press Syndicate

Figure 8.4: Calvin and Hobbes, [June 2, 1992]

mail, here the Calvins literally find themselves in a room, communicating across time from 6:30-8:30, from homework time to bedtime. Ultimately, the 3 temporally differentiated Hobbes mediate the situation and do the homework. The aim of provenance documentation is to move beyond the one-way communication that comes from the past leaving missives for the future to something that resembles mediated conversations where past, present, and future can collaborate to form the best solutions. In previous work, we suggest that subjunctive provenance may improve provenance practice, acting as a mediator like the Hobbeses [3].

SigProps are inherently related to identity and time: they are the characteristics which determine whether the thing remains that thing over time. These cases demonstrate that significance is not necessarily inherent to an object: vastly different Marios are still experienced as Mario, the 3 Calvins are still just Calvin. Authenticity doesn't occur in a vacuum: meaning comes from experiences with objects rather than objects being inherently meaningful. Authenticity is a product of a relationship between objects and stakeholders [2,7].

The fundamental question remains: is the thing that thing? The answer is partly domain-dependent: in data management, it would be culturally common to see a change in a dataset resulting in a new data set, △ dataset, even if the contents remained largely the same. However, a visibly obvious change in Mount Vernon, like the loss of a roof during a hurricane, does not result △ Mount Vernon: it is still Mount Vernon. When Calvin tells himself, "You know things I don't know," he's talking about his own provenance: what differentiates the Calvins is what they've experienced. This raises the question: can provenance itself be employed as that which distinguishes a thing both as and from that thing?

These challenges are not academic. Practitioners manage diverse object and data types that behave differently enough that preservation and provenance practices are hard to universalize. Persistent identifiers that work for moon rocks do not work for biosamples. It is not that moon rocks don't change, but that the speed at which they do so is slower than a human life span, while biochemical samples might change more through natural organic decay in a few days than they do in an experiment which is meant to alter them. Simultaneously, documentary processes that were done by hand for artifactual objects are impossible in computational environments: humans cannot document nanoseconds by hand. Incremental change is also a temporal facet that challenges documentary practices: there is a saying that it takes 7 years for every cell in the human body to be replaced with a new one. This saying points to three things: (1) that biological matter is always in a state of flux and change; (2) that humans assign symbolic meaning to this type of change; and (3) that humans understand incremental changes differently than other types of alteration. This type of biological incremental change is analogous to the Ship of Theseus story; it's the same kind of scenario that digital preservationists face when trying to track the knowledge base of a designated community.

This short paper presents a progressive idea: that digital preservation has not yet dealt sufficiently with the temporal aspects of sigProps. Time is always there in preservation work, but often at the periphery, where the changes of the object are documented and not the change of time itself. When that happens, difficult scenarios challenge existing models– Marios, Calvins, biochemical samples. This leads to a proliferation of standards and extensions, like the provlets of PROV, without solving the underlying issues. SigProps research often focuses on the preservation targets. Yet research consistently shows that what is significant about an object is not necessarily inherent to objects. Simultaneously, sigProps research does not adequately attend to temporality. Perhaps because time is part of the definition of sigProps, and part of digital preservation overall, it has been taken for granted and its role has been underexplored.

8.6 Acknowledgments

MRG acknowledges partial support from National Institutes of Health grants GM-111135 and GM-109046 and National Science Foundation grant 194670.

Chapter 9

Which Model Does Not Belong: A Dialogue

9.1 Abstract

Conceptual models can serve multiple purposes: communication of information between stakeholders, information abstraction and generalization, and information organization for archival and retrieval. An ongoing research question is how to formally define the fit-for-purpose of a conceptual model as well as to define metrics or tests to determine whether a given model faithfully supports a designated purpose.

This chapter summarizes preliminary investigations in this area by presenting toy problems along with different conceptual models for the system under study. It is argued that the different models are adequate in supporting a sophisticated query and yet they adopt different normalization schemes and will differ in expressiveness depending on the implied purpose of the models. As the subtitle suggests, this work is intended to be primarily exploratory as to the constraints a formal system would require in defining the "usefulness", "expressiveness" and "equivalence" of conceptual models.

9.2 Introduction

A monk was summoned to the Buddha's chamber. Upon a table were four pots: three gold and one silver. The first gold pot was large with ornate handles. The second could have been its younger sibling, sharing the same shape and handles yet of a diminutive size. The third was as big as the first but had no handles. The silver pot was large with handles.

B: Tell me my student, which of these pots is unlike the others?

M: The silver one of course. The others are made of gold.

B: Please hoist each one above your head.

M: Ah, master. I beg forgiveness. The one without handles is truly unique.

B: Would you please use them to milk the cow?

M: Once again master, I have changed my mind. The smaller one is inferior.

B: Several times I have given you a task and each time you have made a different choice. What task could I assign which would convince you to choose the large, handled pot of gold?

M: There is no such task, master. That pot is not unique in any way.
B: That is what makes it truly unique.

9.3 B's Story

The koan in the introduction contains what is referred to as a "Which One Doesn't Belong" (WODB) problem (Danielson, 2016). This type of problem typically challenges one to identify the object which is not like the others. In educational contexts (e.g., Sesame Street, K-12) the puzzle is sometimes not primarily about finding a single "right" answer, but the fact that there might be several answers, and the point becomes to articulate a justification for the chosen answer. A feature in the koan is that the chosen object (the first pot) differs from all others via a "meta-property": It is the only object without a unique property. In this section we introduce a relational model that can be used to systematically analyze WODB problems and that also sheds some light on the underlying conceptual modeling issues.

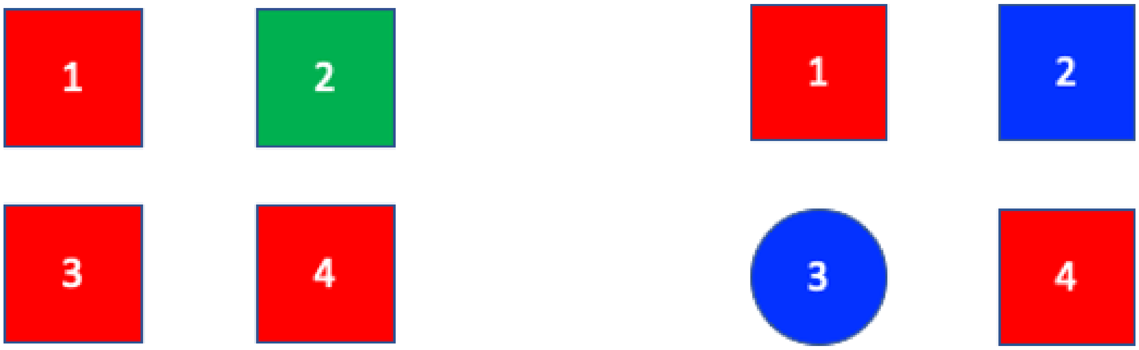


Figure 9.1: Two simple WODB problems: Examples 1 and 2.

The four objects o1, o2, o3, and o4 depicted on the left in Figure 1 constitute a trivial WODB problem: the boxes differ only in a single property: their color. Clearly o2 is not like the others, since it is the only object that is green. Another argument is that o1, o3, and o4 are indistinguishable, i.e., we cannot single out any one of them without also "retrieving" the other two. Note that it is implicitly understood that the object-ID or the relative position is not part of the model: e.g., we can't say o4 is special because it's the only red box located in the South-East quadrant of the puzzle.

Consider now the WODB puzzle on the right of Figure 1: Which object is not like the others? In this second example, o1 and o4 are indistinguishable and we might be tempted to answer either o2 ("it's the only blue box") or o3 ("it's the only blue circle"). To resolve the ambiguity, we can employ a relational database D and devise a formal justification using a query Q that picks the object we deem to be unlike all others. D consists of a set of facts prop(X,P) stating that object X has a property P:

```
prop(o1,box).  prop(o1,red).  prop(o2,box).  prop(o2,blue).
prop(o3,circle).  prop(o3,blue).  prop(o4,box).  prop(o4,red).
```

For Q we choose: "Object X is unlike the others if X has a property P that no other object has". Formalized in Datalog, we have:

```
 unique(X,P) :- prop(X,P), not another(X,P).                  (1)
another(X,P) :- prop(X,P), prop(X2,P), X != X2.               (2)
```

Rule (1) says that X is unique w.r.t. property P, if X is the only object with property P. The auxiliary rule (2) finds objects X with property P for which another object X2 exists that has the same property value P as X. Therefore if another(X,P) holds, it means that X is not unique w.r.t. P. Evaluating query Q on database D yields a unique answer A = Q(D) for the right puzzle in Figure 1:

```
unique(o3,circle)
```

Why is the object o2 not among the answers here? After all, o2 is the only blue box! However, we have modeled color and shape as independent properties: o2 is a box (and there are other boxes), and o2 is blue (and there are other blue objects). In contrast, o3 is the only object having the circle property. The same rules (1) and (2) can also be applied to the database instance D for Example 1 on the left of Figure 1. In that case, we get the expected answer:

```
unique(o2,green)
```

Now consider the slightly more challenging examples in Figure 2: Example 3 on the left in Figure 2, yields the following answer when running Q, i.e., rules (1) and (2), on the model database D:

```
unique(o1,small)
unique(o3,circle)
```

This means that two objects are now "equally special": o1 because it has the unique property of being small, and o3 (as in Example 2) because it is the only circle.

An interesting aspect of these "equally special" solutions is that a minimal change will turn the argument for selection of o1 into a justification for o3 and vice versa: The property small is to o1 what the property circle is to o3. Formally, the justification (i.e., derivation in terms of a Datalog proof tree) of the first solution can be changed into one for the second solution (and vice versa), simply by swapping the properties small and circle.

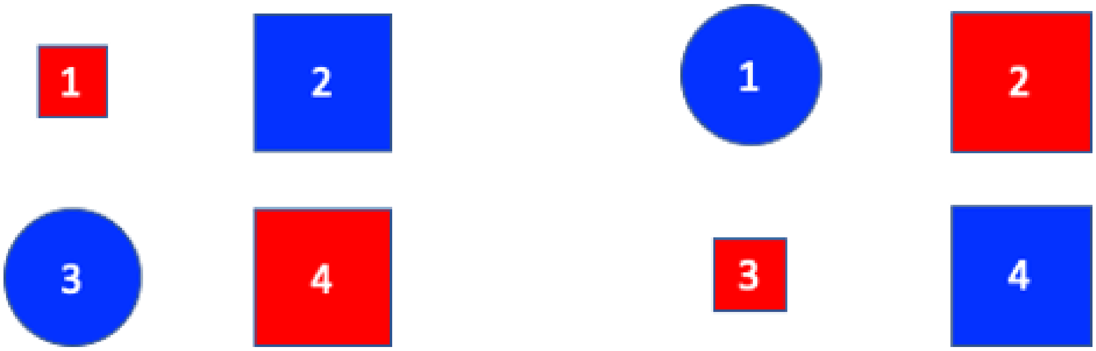


Figure 9.2: Two ambiguous WODB puzzles: Examples 3 and 4.

This last aspect also depends on the choice the query Q to select the object that doesn't belong. Interestingly, there are aspects of the model that are independent of any choice for Q, as can be seen from the following argument: Consider any model M1 of the problem on the left in Figure 2: it associates with each object exactly three properties (for shape, color, and size). If we swap the properties small ⇔ circle, blue ⇔ red, and large ⇔ box, we obtain a new model M2 that describes the WODB situation on the right of Figure 2. Interestingly, the two models are isomorphic under this permutation of domain elements, as show in Figure 3.

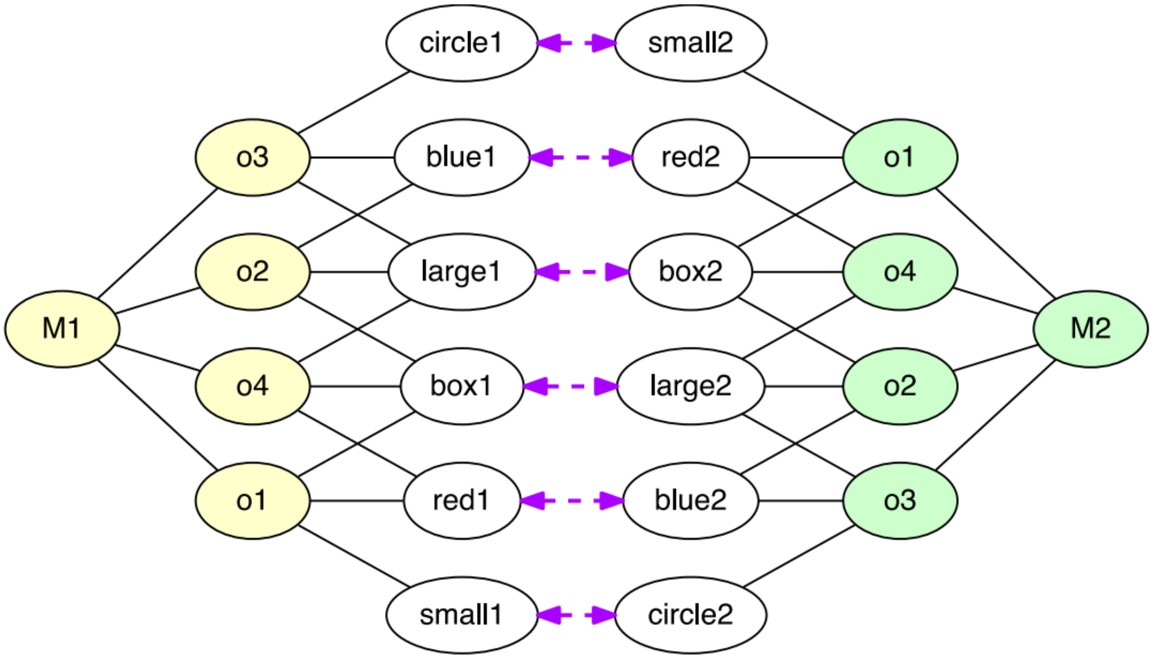


Figure 9.3: Models M1 and M2 are isomorphic under the permutation small ⇔ circle, blue ⇔ red, and large ⇔ box.

This also means that for any query Q (e.g., the one above), the answers will be isomorphic, too: the answer for M1 (Example 3) was unique(o1,small) and unique(o3,circle), which becomes, via the isomorphism, unique(o1, circle) and unique(o3,small), i.e., the answer for M2 (Example 4 in Figure 2).

Last not least, consider Example 5 in Figure 4: Here, o1, o2, and o3 are each unique in a different way:

```
unique(o1,small)
unique(o2,green)
unique(o3,circle)
```

But this is odd: If every object is unique in its own way, except for one object that is "normal" (i.e., not unique in any way), then shouldn't that object be the one which doesn't belong!? Indeed, we can model such "meta" arguments formally:

```
special(O)  :-  unique(O,_).
 normal(O)  :-  prop(O,_),  not  special(O).
```

Here we just declare objects special that are unique w.r.t. some (unnamed) property and declare objects normal if they are not special. When running this query, we obtain a single normal object o4 (and three special objects o1, o2, and o3), and we can declare the single normal object to be the one which doesn't belong. Note that in this example, like in the previous example, each argument for one of the special objects o1, o2, and o3, can be turned into an argument for any other special object, simply by swapping the "chosen property". In contrast, the way in which o4 is (meta-)special is different from the way o1, o2, and o3 are special. As in the opening koan, o4 is stands out, since it is the only object that has no unique property.

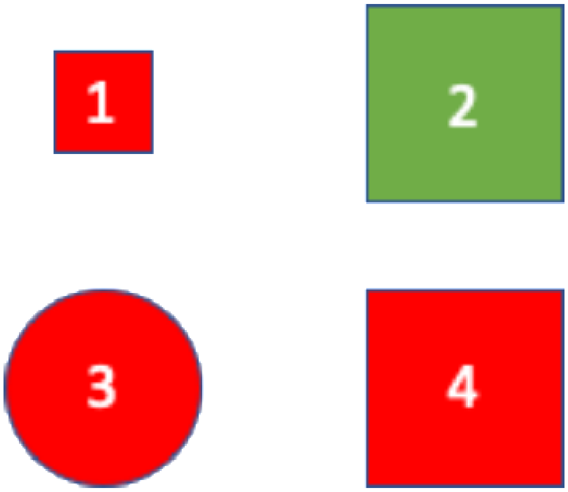


Figure 9.4: A confusing WODB puzzle: Example 5.

9.4 M's Story

One of the central themes in last year's workshop paper (Gryk, 2019) was the assertion of George Box (1979) that "all models are wrong, some are useful." While this statement seems to be generally accepted by most scientists, the authors have noticed anecdotally that there are various degrees of specificity to which people believe the adage applies. Puzzled by this limited acceptance, MRG began playing a "modeling game" with his peers in which he would quickly transform a peer's conceptual model into something similar but different in order to highlight both subtle differences implicit in different modeling choices as well as to demonstrate there are often multiple equivalent models (as defined by a purpose.) It was during this timeframe that BL presented his model and solution to the WODB problem during a weekly research meeting. What follows is an alternate model and solution to the problem

9.4.1 M's Model

It is important to note that there is a "trick" involved with the meta-level of uniqueness contained in this WODB problem. That "trick" is that each property for an object has only two possible values. A pot can be big or small, gold or silver, handled or not. This binary nature of the properties is what allows for the identification of the unique-by-not-being unique object.

With this is mind, one can easily transform the model from Section 2 into a model in which each object has a series of True/False attributes. For each Pot in our table, there is a binary attribute corresponding to the attributes gold, handled and large (Table 9.1).

Table 9.1: Character Matrix for the WODB problem.

POT	IS GOLD	HAS HANDLES	IS LARGE
1	X	X	X
2	X	X	
3	X		X
3		X	X

The unique pot is very easy to spot with this conceptual model; it is the one in which all attributes are True.

This new model succeeds by changing the reification of the properties of the object from values in a table to named attributes. For a simple toy problem, this model is an acceptable alternative to the one in Section 2. However it could be argued that such a model will not scale. What if the WODB problem was extended to support objects with dozens or hundreds of attributes?

Yet this model is both useful and used by an entire domain of scientists. One early iteration of the toy problem used animals with various properties: Predators vs. Prey, Stripes vs Uniform Coats, etc. This version of the problem had a zebra and a tiger as examples and was an obvious example of the character matrix used by taxonomists (as shown in Table 9.2.)

Table 9.2: Character Matrix for taxonomists. Adapted from [56].

TAXON	HAS 5 FINGERS	HAS FUR	LAYS EGGS
LION	X	X	
LIZARD	X		X
PLATYPUS	X	X	X
ZEBRA		X	

Character-taxon matrices as shown above are used to identify significant traits of organisms in order to help build a phylogenetic or evolutionary tree of life. Building such trees assumes that organisms with similar traits are more closely related than ones with dissimilar traits. While a simple concept, the use of such matrices requires much care (Vermeij, 1999).

The similarity between the WODB model (Table 9.1) and the character-taxon matrix (Table 9.2) is undeniable. Yet, there is still a question as to whether these two models are equivalent, regardless of the obvious similarity. In the case of character matrices, the similarity of traits are used to define evolutionary relationships. For instance, one might draw the conclusion from Table 9.2 that the platypus is the common ancestor to lions, lizards and zebra . Is that conclusion a defined purpose of the model? Or are we reading into the model something which it was not intended to convey.

Similarly, one might assume that all of the pots in the toy problem were derived from the first pot. While it is left as a rhetorical question whether this is a defined role of the model or not, the first pot was actually the common ancestor from which the other ones were derived.

9.5 Conclusions

A combined conclusion about George Box: all models are wrong, some are useful and how queries can be used to probe the usefulness of a model. Final question: if two different models can answer the same questions, are they equally useful despite their differences?

9.6 Acknowledgements

The authors thank Sahil Gupta who has created analysis tools for studying WODB problems, using his Possible Worlds Explorer tool (Gupta, 2020). In particular, his latest tool can identify the automorphisms of WODB instances.

Chapter 10

Concept Keys

10.1 Abstract

Concept Keys is my term for an implementation strategy for modeling specialization lattices. In standard relational database modeling, there are four different mechanisms for dealing with subclasses between entities. These four mechanisms have various pros and cons associated with them depending on the precise scenario they are used.

This chapter describes a situation in which there is a large proliferation of subclasses which form a specialization lattice (i.e. the subclasses derive via multiple inheritance from the superclass(es)). In this particular situation, all of the common approaches to modeling subclasses have weaknesses.

The chapter concludes by describing two novel ways of modeling a specialization lattice which are useful for the case study in the chapter and may also be useful for extending existing provenance models to subjunctive provenance.

10.2 Introduction

This chapter continues the conversation concerning choices of reification when creating an information model. Whereas the previous chapter explored reification in the context of the "Which One Doesn't Belong" problem, this chapter focuses on a different situation which arises in many data modeling contexts. Broadly speaking, the issue to be tackled is one in which various conceptual entities are referred to in the model with varying degrees of specificity.

It is often easy to accommodate specialization and generalization via a hierarchical model; however, there are cases where a strict hierarchy is not reflective of the real world system. One prominent example is the multihierachical rock classification system introduced by the Geological Survey of Canada [57] in 2002. Provenance is also a domain that has varying levels of specificity. The examples in Chapters 3 and 4 provide many examples of specificity concerns in documenting provenance. The distinctions between wine, white wine, red wine, grapes, white grapes, red grapes, Malbec grapes, Shiraz grapes, Malbec wine, Syrah wine, sparkling wine, Champagne, and JeMiRi-branded wines. Throughout those various provenance stories, these types of distinctions may or may not impact the documented provenance, either the retrospective or the prospective provenance. In addition, Chapter 7 introduced the concept of *subjunctive* provenance and the possibility of other temporal forms of provenance. Not only does specialization enter into provenance via the specialization of the entities, activities and agents, it also enters in the form of the type of provenance: retrospective vs. prospective vs. subjunctive vs. potentially others. At this meta-level, the different forms of provenance documentation might warrant different provenance models - as provided by ProvONE [53]. Yet, as more temporal forms of provenance are deemed important, future extensions to the PROV model risk becoming untenable and unsustainable.

A primary motivation for the work in this chapter is the modeling of biochemical sample metadata for the Network for Advanced NMR (NAN) (which was also mentioned in Chapter 8). This data modeling problem is also complicated by the need for a complicated specialization lattice within the information system. A novel implementation strategy for dealing with specialization lattices within a relational model (or XML model) was developed and is described in this chapter. Finally, this implementation strategy (termed Concept Keys) will be discussed within the context of other similar classification and modeling approaches such as faceted classification, formal concept analysis, the entity-attribute-value model, and the resource description framework (RDF).

10.2.1 Biochemical Samples

The Network for Advanced NMR is a recent NSF-funded project that hopes to automatically harvest data collected on NMR spectrometers at three institutions within the United States. The vision of NAN is to provision a lightweight application on each of the spectrometer computers which is capable of monitoring data acquisition and shuttling data when an experiment is complete. This software is intended to be small and simple, as the various instruments across the three sites of NAN are running older versions of both Windows and Linux operating systems. Importantly, those computers cannot be upgraded due to constraints of the scientific instruments they support.

The second component of NAN is a gateway computer which is installed in each NMR facility at the three sites. The gateway computers run a modern operating system and have custom-designed software for gathering the datasets provided by the instruments, monitoring instrument status, and queuing the accumulated datasets for transport to a central system at the UCONN Health site. Once transported to UCONN Health, the datasets are parsed and archived: the experimental data is written to a write-once-read-many file system and the metadata are stored in a relational database so the experiments can be searched via a website and associated web services.

The experimental data have been modeled previously (see chapters 5 and 6 and the associated work of Nowling, *et al.* 2011 [35] and Fenwick *et al.* 2015 [3]). However, there is one additional set of metadata which is critical for using the experimental data. That is metadata on the sample which was in the spectrometer and to which the data are related.

There have been several attempts to model physical samples as well as providing them with persistent unique identifiers. The International Geological Sample Number (IGSN) system is used in the geosciences; both the National Center for Biotechnology Information (NCBI) and its European counterpart, the European Bioinformatics Institute (EBI), have established metadata standards for biosamples. However, in the case of the geoscience metadata - the focus is mostly on who collected the sample, when the sample was collected, and where. These are the important aspects of a geological sample such as a rock or an ice core. However, in the case of NMR experiments, the sample often represents not a part of the Earth but rather the sample is a manufactured system which is designed to recapitulate some aspect of biochemistry which is the actual target of the investigation. For biochemical samples, the who, when and where are of lesser importance to what is in the sample to be studied.

The biosample standards of NCBI/EBI fall into two classes[1]. The first class is for cell and tissue samples in which a certain phenotype or disease state is associated with the biological material. In this case, organismal metadata takes center stage - items such as the species of the tissue, the organ from which it came, and the gender of the person or animal. These metadata fields are not normally applicable to biochemical samples[2].

The second class of biosamples modeled by NCBI an EBI refers to genetic sequence data in which case the primary data is the source organism / species within the taxonomy of life. The "tree" of life is a challenging problem on its own with millions of species on Earth and the added complexity of horizontal gene transfer in evolution such that no single "tree" can explain the connectedness of life on Earth[58].

Despite the existence of these various models and standards for biological samples, none of the previously described models was suitable for the NAN project. A newly defined model will be described in the next section.

10.2.2 Conceptual Model

In December of 2022, I was tasked with crafting a metadata model to accommodate any NMR sample that could be part of the NAN archive. This includes biochemical samples in solution, the solid state or other physical states (such as gels or aggregates); metabolomic samples such as urine, blood plasma and cell extracts from various organisms from bacteria to mammals; as well as material samples such as bone fragments, petroleum, wood, honey, stones and crystals.

The sheer breadth of sample types in NMR made this a daunting challenge. This is challenging not only in developing a one size fits all model for NMR samples, but also in the eventual design of a web-accessible user interface for scientists to record the relevant metadata about the particular sample they are studying.

Two criteria for this model were recognized on the onset. The model would need to record provenance - one commonality across all sample types (see Chapter 8 [59]). The second criteria was that the metadata model would need to be an extensible framework, as it would not be possible to identify *a priori* all of the potential fields individual researchers might deem important for their studies. A researcher studying a reconstituted ribosome might care about the biochemical preparation steps, storage and handling of the material, as well as links to relevant protein and DNA databanks. On the other hand, a researcher studying various honey samples gathered across the country might be more of the mindset of the IGSN's and be concerned about where and when samples were collected, while also needing to document the species and strain of bee.

The prototype conceptual model for NMR samples is shown in figure 10.1.

There are few things to note about this model. One, it utilizes the top level concepts of PROV (and PREMIS). Materials are prov:entities (premis:objects), Processes are prov:activities (premis:events) and Vendors/People are prov:agents. The PROV relationships between these entities are maintained, such that processes use materials to generate other materials, and in that manner materials can be derived from other materials.

Along with derivation (which is how a material was made) is the concept of composition (which is what a material is made of). This distinction is very important in chemistry as one could create a saline solution by

[1] NCBI Biosample contains genetic sequence data from GenBank and the Sequence Read Archive and cell/tissue information from ATCC, Coriell, and ICLAC. EBI Biosamples contains genetic sequence data from the Functional Annotation of Animal Genomes and cell information for induced pluripotent stem cells and covid-19 isolates.

[2] One exception would be the field of metabolomics in which NMR is used to quantify the amount of metabolites found in a bodily fluid such as plasma or urine.

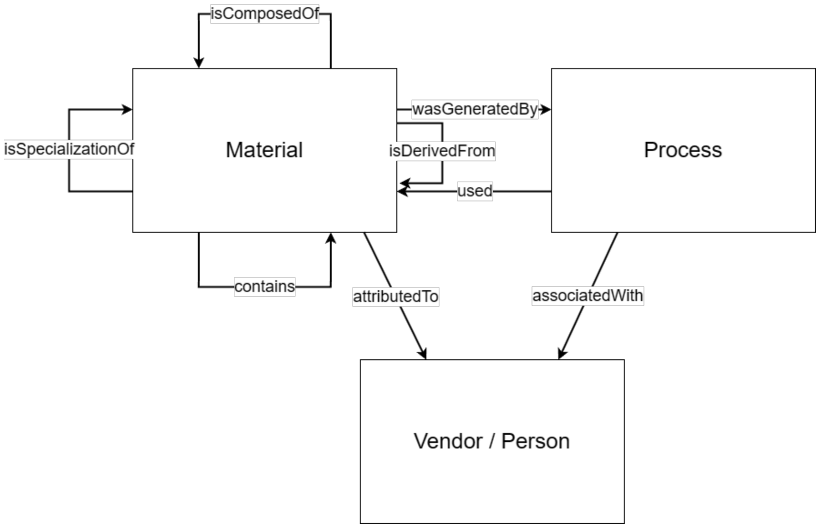


Figure 10.1: Conceptual model for materials which are stuied as NMR samples.

dissolving salt in water or conversely by mixing hydrochloric acid with a sodium hydroxide solution. In those two cases, the composition could be the same while the derivations are different.

A somewhat tangential feature of this model is that a material can also contain other materials. This is important in spectroscopic studies as the results of the experiment can be affected by the material composition of the vessel which holds the sample during the experiment.

The final important portion of this conceptual model is that materials can be specializations of other materials. In other words, they are sub-classed. It is quite common to think of a particular molecule under study in a hierarchical manner. Human DNA Polymerase β is a type of DNA Polymerase β which is a type of DNA Polymerase which is a type of polymerase which is a type of protein which is a type of biological macromolecule which is a type of molecule which is a type of material.

While the above representation is strictly hierarchical, to fully represent NMR samples we need to support a multiple parent hierarchy also known as a specialization lattice and for that we need to utilize a reification strategy which I refer to as Concept Keys.

10.3 Subclasses in Relational Databases

There are four documented strategies in *Fundamentals of Database Systems* for representing subclasses in a relational model and database[60]. Subclasses themselves require the use of the so-called Extended ER diagram for describing them. The example used in this section is a variation on one provided by Elmasri and Navathe in their textbook[60].

Figure 10.2 illustrates a conceptual model in which case there is an entity for Vehicles and that Vehicles can be subclassed into Cars and Boats.

In this modeling notation, a circle is used to denote a subclass relationship between the vehicle entity and the car and truck entities (the ∪ near the circle denotes the directional of the subclass). Each of these three entities have attributes associated with them. All vehicles (including cars and boats) have a VIN number, a price and license plate number. Cars have attributes of the maximum speed and number of passengers which boats do not have. Boats have fields for the hull length and number of sails which cars do not have.

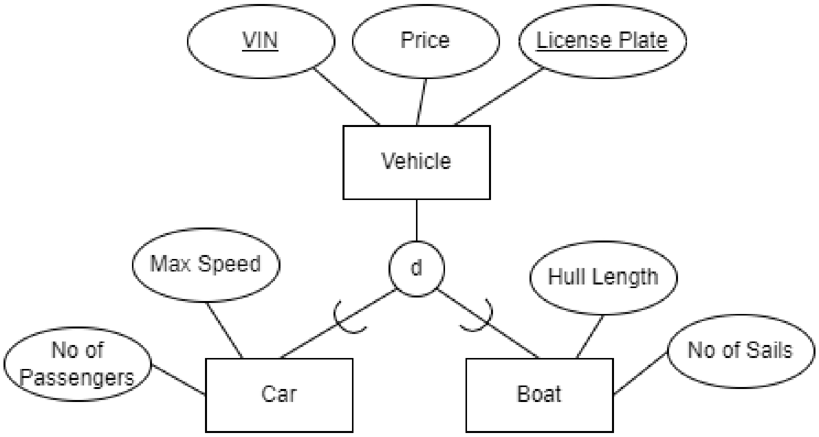


Figure 10.2: Enhanced ER modeling illustrating subclasses.

Apart from the keys (underlined attributes) this figure does not inform us as to which of the attributes are mandatory versus optional. The possibility and consequences of nulled attributes is something we will revisit later in this chapter.

The circle in the diagram has the letter 'd' inside of it. This letter is used to denote one of four possible properties of the subclass relationships. The subclasses are said to be **Disjoint** (d) if a member of the car class cannot also be a member of the boat class. If a vehicle can be both a car and a boat, then the subclasses are said to be **Overlapping** (o). Another pair of properties for the subclass relationship are partial versus total. A **Total** (t) subclass is one in which all vehicles are either cars or boats (or both in the case of overlapping). A **Partial** (p) subclass is one in which there exist vehicles which are neither cars nor boats.

10.3.1 Four Approaches

With these possibilities in mind, there are four approaches to handling subclasses outlined in Elmasri and Navathe. The first two approaches require multiple relations (database tables) which include the subclasses. The other two use a single relation for the superclass. Below are the formal definitions verbatim from Elmasri and Navathe.

Approach 1: Multiple relations - superclass and subclasses

> Create a relation L for C with attributes Attrs(L) = $\{k, a_1, ..., a_n\}$ and PK(L) = k. Create a relation L_i for each subclass $S_i, 1 \leq i \leq m$, with the attributes Attrs(L_i) = $k \cup$ attributes of S_i and PK(L_i) = k. This option works for any specialization (total or partial, disjoint of over-lapping).

Approach 2: Multiple relations - subclass relations only

> Create a relation L_i for each subclass $S_i, 1 \leq i \leq m$, with the attributes Attr(L_i) = attributes of S_i $\cup$ $\{k, a_1, ..., a_n\}$ and PK(L_i) = k. This option only works for a specialization whose subclasses are *total* (every entity in the superclass must belong to (at least) one of the subclasses). Additionally, it is only recommended if the specialization has the *disjointedness constraint* (see Section 4.3.1). If the specialization is *overlapping*, the same entity may be duplicated in several relations.

Approach 3: Single relation with one attribute type

Create a single relation L with attributes Attrs(L) = $\{k, a_1, ..., a_n\}$ ∪ attributes of S_1 ∪...∪ attributes of S_m ∪ $\{t\}$ and PK(L) = k. The attribute t is called a **type** (or **discriminating**) attribute that indicates the subclass to which each tuple belongs, if any. This option works only for a specialization whose subclasses are *disjoint*, and has the potential for generating many NULL values if many specific (local) attributes exist in the subclasses.

Approach 4: Single relation with multiple type attributes

Create a single relation schema L with attributes Attrs(L) = $\{k, a_1, ..., a_n\}$ ∪ attributes of S_1 ∪...∪ attributes of S_m ∪ $\{t_1, t_2, ..., t_m\}$ and PK(L) = k. Each $t_i, 1 \leq I \leq m$, is a **Boolean type attribute** indicating whether a tuple belongs to the subclass S_i. This option is used for a specialization whose subclasses are *overlapping* (but will also work for a disjoint specialization).

10.3.2 The Four Approaches Unpacked

It is perhaps useful to demonstrate with a small example database how these four approaches differ. The first approach creates a table for the superclass, Vehicle, as well as each of the subclasses, Car and Boat. The Vehicle-specific attributes are included in the Vehicle table and the Car and Boat attributes are included in their respective tables.

Table 10.1: Approach 1: Multiple relations - superclass and subclass

Vehicle			
Primary Key	VIN	Price	License Plate
V1	874-0983-45677	\$25,000	XVG-23
V2	ZY04877395.3	\$75,000	BJ7289
V3	987-9877-5630	\$49,500	PPR-84
V4	JF38720411.7	\$175,000	CK0023

Car		
Primary Key	Max Speed (mph)	Passenger Count
V1	100	4
V3	135	2

Boat		
Primary Key	Hull Length (ft)	Sail Count
V2	17	0
V4	40	2

Note that the primary keys for the subclasses are the same as that of the superclass allowing the selection of all attributes for any car or boat with a simple equi-join on the primary keys.

This approach may also be thought of as a “Codd approved” approach in that the model is normalized. Car attributes are in the Car relation, Boat attributes in the Boat relation, and Vehicle attributes in the Vehicle relation. This affords maximum control on integrity and enforcing non-NULL values as appropriate. As pointed out by Elmasri and Navathe, this approach is applicable regardless of whether the subclasses are disjoint, overlapping, partial or total.

Approach 2 is shown in Table 10.2. In this approach, there are only tables for the subclasses, Car and Boat. All of the vehicle attributes are placed within both the Car and Boat tables. This is of little consequence if

Table 10.2: Approach 2: Multiple relations - subclasses only

Car					
Primary Key	VIN	Price	License Plate	Max Speed (mph)	Passenger Count
C1	874-0983-45677	$25,000	XVG-23	100	4
C2	987-9877-5630	$49,500	PPR-84	135	2

Boat					
Primary Key	VIN	Price	License Plate	Hull Length (ft)	Sail Count
B1	ZY04877395.3	$75,000	BJ7289	17	0
B2	JF38720411.7	$175,000	CK0023	40	2

the subclasses are disjoint. However, if they are overlapping, than a vehicle can exist in both tables in which the vehicle attributes are duplicated and susceptible to inconsistency between the two tables. Care would also need to be taken if querying for a count of all vehicles as Car-Boats would exist in both tables.

Finally, as pointed out by the authors, this only works for total subclasses. If the subclasses are partial, there no longer exists a relation to store vehicles which are neither cars nor boats.

While Approach 2 pushes the superclass attributes into the subclasses, the next two approaches will pull the subclass attributes up into the superclass. Approach 3 and 4 only have a single relation, the superclass Vehicle.

Table 10.3: Approach 3: Single relation with one type attribute

Vehicle								
PK	VIN	Price	License	Speed	Passengers	Length	Sails	Vehicle Type
V1	874-0983-45677	$25,000	XVG-23	100	4	NULL	NULL	Car
V2	ZY04877395.3	$75,000	BJ7289	NULL	NULL	17	0	Boat
V3	987-9877-5630	$49,500	PPR-84	135	2	NULL	NULL	Car
V4	JF38720411.7	$175,000	CK0023	NULL	NULL	40	2	Boat

In this approach, the subclass attributes are included in the superclass relation. A consequence of this is that these subclass-specific attributes must be NULLable in the database as cars don't have boat attributes and vice versa. Therefore, some integrity control on the data is relinquished.

The final column of the table includes an attribute for vehicle type. This allows us to keep track of which vehicles are cars and which are boats. Note that if the subclass attributes were not allowed to be NULL, then the vehicle type could be inferred by querying for the existence of Max Speed or Hull Length. However, if the subclass attributes can be NULL - and for the database table itself they must be - then the vehicle type is necessary.

The authors note that this limits Approach 3 to disjoint subclasses. In actuality, overlapping subclasses could be accommodated by including a vehicle type of Car-Boat; however, such a solution would become unwieldy quite quickly as the number of subtypes increase.

The final approach is similar to Approach 3 in that there is only a single table for all Vehicles. The difference is that rather than requiring a single field for the vehicle type, a set of Boolean attributes are provided for defining if the Vehicle is a Car and/or a Boat.

Note that in this case, overlapping and partial situations are accommodated as both attributes can be True for Car-Boats or False for Vehicles.

Table 10.4: Approach 4: Single relation with multiple type attributes

Vehicle									
PK	VIN	Price	License	Speed	Passengers	Length	Sails	IsCar	IsBoat
V1	874-0983-45677	$25,000	XVG-23	100	4	NULL	NULL	True	False
V2	ZY04877395.3	$75,000	BJ7289	NULL	NULL	17	0	False	True
V3	987-9877-5630	$49,500	PPR-84	135	2	NULL	NULL	True	False
V4	JF38720411.7	$175,000	CK0023	NULL	NULL	40	2	False	True

10.3.3 Need for a Fifth Approach

While these four approaches provide technically sound solutions to modeling subclasses in a relational model, a fifth approach is warranted. This fifth approach is useful when there are large numbers of subclasses which need to be accounted for (see the previous section describing polymerases which are proteins which are macromolecules which are molecules and so on) and for which the subclass specific attributes can be NULL and therefore the subclass cannot be inferred from the attributes. This fifth approach is also particularly useful when the subclassing is not strictly hierarchical but form a multi-parent hierarchy also known as a specialization lattice. The limitations of the current four approaches are not easily presented in the context of the Vehicle-Car-Boat model. The next section introduces a more detailed specialization lattice for consideration.

10.4 Toy Problem

This toy problem is introduced to give an example of a fairly involved specialization lattice which will be challenging to support using the four approaches described earlier. A key feature of this particular toy problem is that this is a specialization lattice that is familiar to the average person and one in which the various subclasses all have well-known names attached to them. That is not always the case. In fact, as we dive into the example we shall see some natural subclasses which do not have common names.

This toy problem will stipulate a hypothetical information system and its requirements. The premise of this system is that I wish to store information about all of my relatives to assist in selecting gifts for them on holidays and for other events. It is further stipulated that I have a very large (and complicated) family and that while I will build the information system for managing the data, I wish to hire someone else to actually populate the system with information about my relatives. Therefore, care needs to be taken about how an arbitrary person will be able to knowledgeably, easily and correctly populate the database.

The primary superclass for the model is the concept of a Relative, denoted by a relation R with attributes $\text{Attrs}(R) = \{k, a_1, ..., a_n\}$. For this problem, it is stipulated that any and all of the attributes for a given relative may be NULL.

It is worth noting that Relative as used here is a unary relation. The term “relative” would typically imply a binary relation, as there are two people who are relatives of each other. However, in this example one of the two people is a constant, me, which effectively renders Relative (and all eventual subclasses) as unary relations.

10.4.1 Subclasses of Relatives

While each of my relatives has primary attributes which are needed for determining potential gifts, for instance, the attribute of whether I like the relative or not, there will be attributes that are specific to particular subclasses of relatives, akin to the vehicle-car-boat example. For this system, I wish to subclass my relatives into grandparents, parents and siblings. This is shown in Figure 10.3.

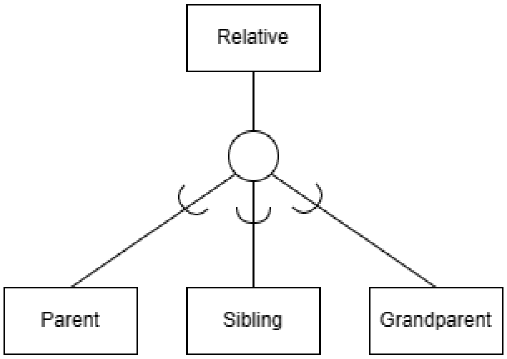


Figure 10.3: First layer of relative subclasses.

This diagram is completely analogous to the vehicle subclass introduced earlier. Note that the type of subclass is not defined, although for this layer it is certainly disjoint. An important aspect for this toy problem and the class of problems that Concept Keys are designed to address is that the subclassing is partial. When my employee eventually starts populating this database, there will be some relatives who are specified as parents, siblings and grandparents but there will also be some which are not specified to that level of detail. This immediately rules our the second approach of Elmasri and Navathe which only provides database tables for the subclasses, not the superclass.

Another important aspect of this model is that there exist attributes specific to each of the subclasses which should be included if the relative is specified as being a parent, grandparent or sibling, but which also can be NULL in cases where the subclass is known but the attribute is not. In an effort to enrich this toy problem, we can introduce a few such attributes. For parents, I wish to record how many times they nagged me to do my homework, for siblings, how many times they took the biggest piece of dessert, and for grandparents, how many hugs they gave me.

10.4.2 Subclasses of subclasses

How does this model change as we subclass the subclasses? For the sake of simplicity, Figure 10.4 only subclasses siblings for now but one could imagine what this hierarchy would look like if we did the analogous subclassing of parents and grandparents.

Once again, the brother/sister (or mother/father, grandmother/grandfather) subclasses all also have unique attributes that should be populated if known. And also, once again, the subclassing is partial in that some relatives will be defined as brothers and sisters while others are only specified as siblings or relatives. Figure 10.5 takes this example one step further.

In this extension to the model, Brothers and Sisters are subclassed further into Half-brothers, Step-brothers, Brothers-in-law, etc. As mentioned throughout this toy problem, each of these subclasses has attributes specific to the subclass, the attributes can be NULL, and the subclassing is partial.

At this point there are twelve total entities including the superclass and subclasses. This would increase further if the subclasses for the parent and grandparent branches were included (although while there are concepts of mother-in-law and step-father, there is no such thing as a half-mother).

The subclassing is partial so relatives without complete specificity can be accommodated. Yet in this

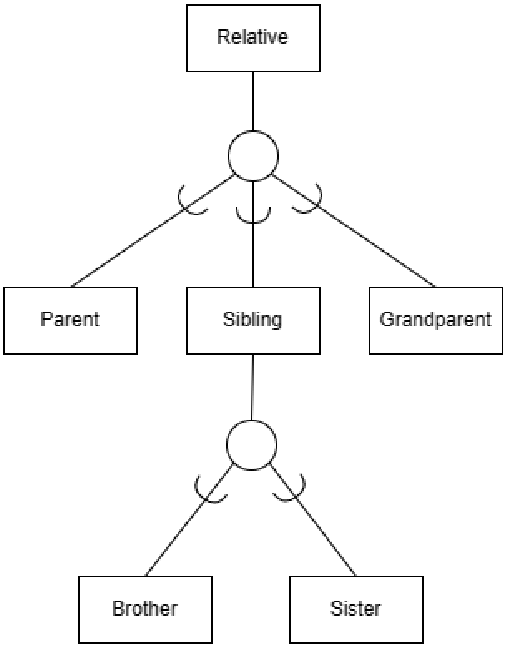


Figure 10.4: Second layer of relative subclasses.

example there are no relations for sibling-in-laws, half-siblings or step-siblings. A requirement for this database is that those subclasses exist as they also have attributes specific to them. The result is shown in Figure 10.6.

10.4.3 Specialization Lattice from Multiple Inheritance

The model in Figure 10.6 is no longer a strict hierarchy as some nodes of the hierarchy have multiple parents. This is referred to as a multiple-parent hierarchy or a specialization lattice. The latter term is favored throughout this chapter as it will be a feature of the lattice structure that will be exploited for this implementation strategy.

With Figure 10.6 the entity count is now up to 15 not including the branches for Parent and Grandparent which would more than double the number of entities (subclasses and superclass) which need to be attended to by the information system and the employee hired to populate this database.

10.4.4 The Four Approaches Revisited

Which of the four approaches are suited to a specialization lattice such as this? To start with it is important to note that there is no rule that the same approach must be used along each tier of a specialization lattice. However, the combinatorics of exploring all combinations of approaches across each subclass discriminator is prohibitive for this exercise.

Since a requirement for this problem is that the subclassing is partial, we have already ruled out the second approach of only providing relations for the specialized subclasses without the generalized superclasses.

Approach 3 is also problematic. Elmasri and Navathe state that this approach requires disjoint subclasses while in matter of fact, they could be accommodated but with major technical difficulties. The type attribute would need to be specialized for all possible overlaps. Note that in this toy problem, if one of my sisters was to marry my step-brother, he would be both a step-brother and a brother-in-law. That would require

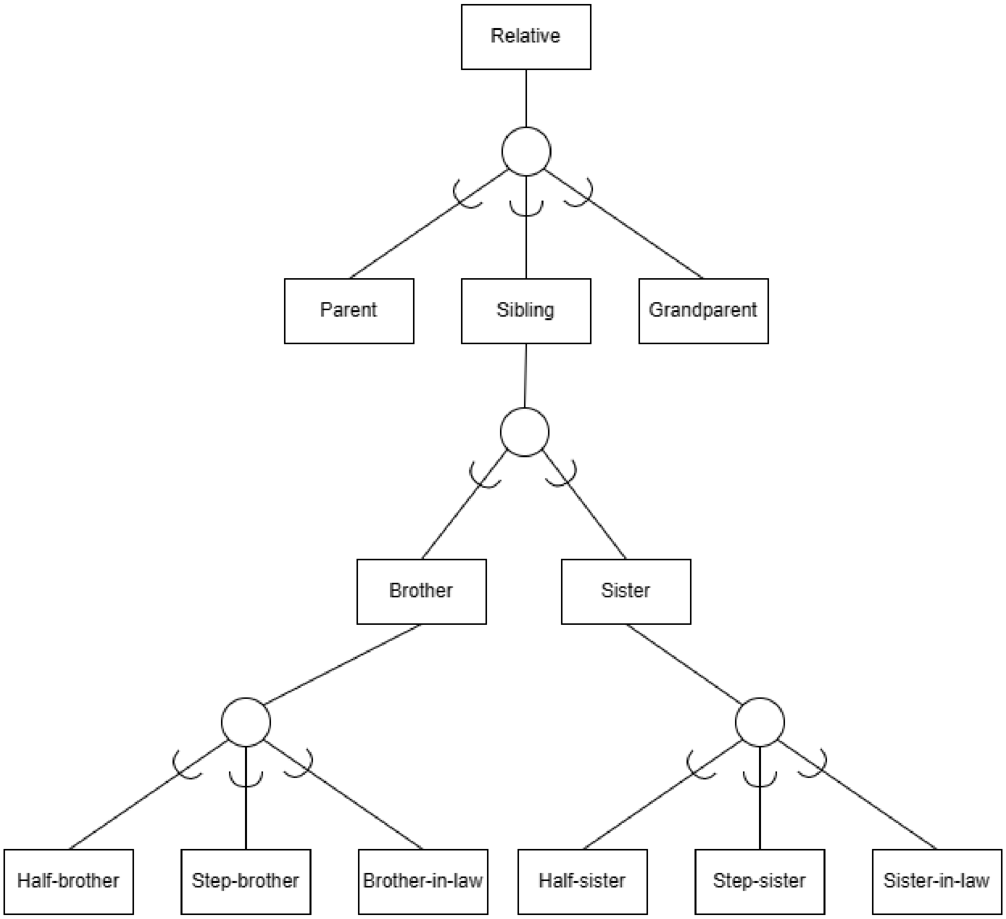


Figure 10.5: Third layer of relative subclasses.

introducing type attributes of not solely ’step-brother’ and ’brother-in-law’ but also ’step-brother-in-law’. This would become very unwieldy in a general case.

This leaves Approaches 1 and 4. Approach 1 is the “Codd approved” approach of a fully normalized database with separate tables for each entity - superclass or subclass - in the model. Approach 4 uses a single relation for Relative with a series of Boolean attributes as place-holders for all of the subclasses.

It is argued here, without proof, that both of these solutions become untenable for use by my employee. In order to populate a database of this sort, the employee would need a codebook or rulebook defining all of the possible entities and the rules for which subclass is applicable for each relative. Note that if the subclass attributes were not able to be NULL, then the data entry would be simpler as one could infer which superclass or subclass was relevant based on the attributes themselves.

Another cause for this difficulty is the existence of mid-model classes (those which are a subclass of the root class but a superclass of the leaf classes). This means we cannot use a hierarchical wizard system to guide the employee to the correct subclass. That is used in other real-world systems like Starbucks beverages (which will be discussed later in this chapter) in which the final product must eventually be a leaf class.

It is also argued that Approaches 1 and 4 are equivalent in complexity. It is equally difficult to identify which node(s) in the lattice a relative belongs to as it is to enumerate the Boolean attribute(s) which point to the subclass(es).

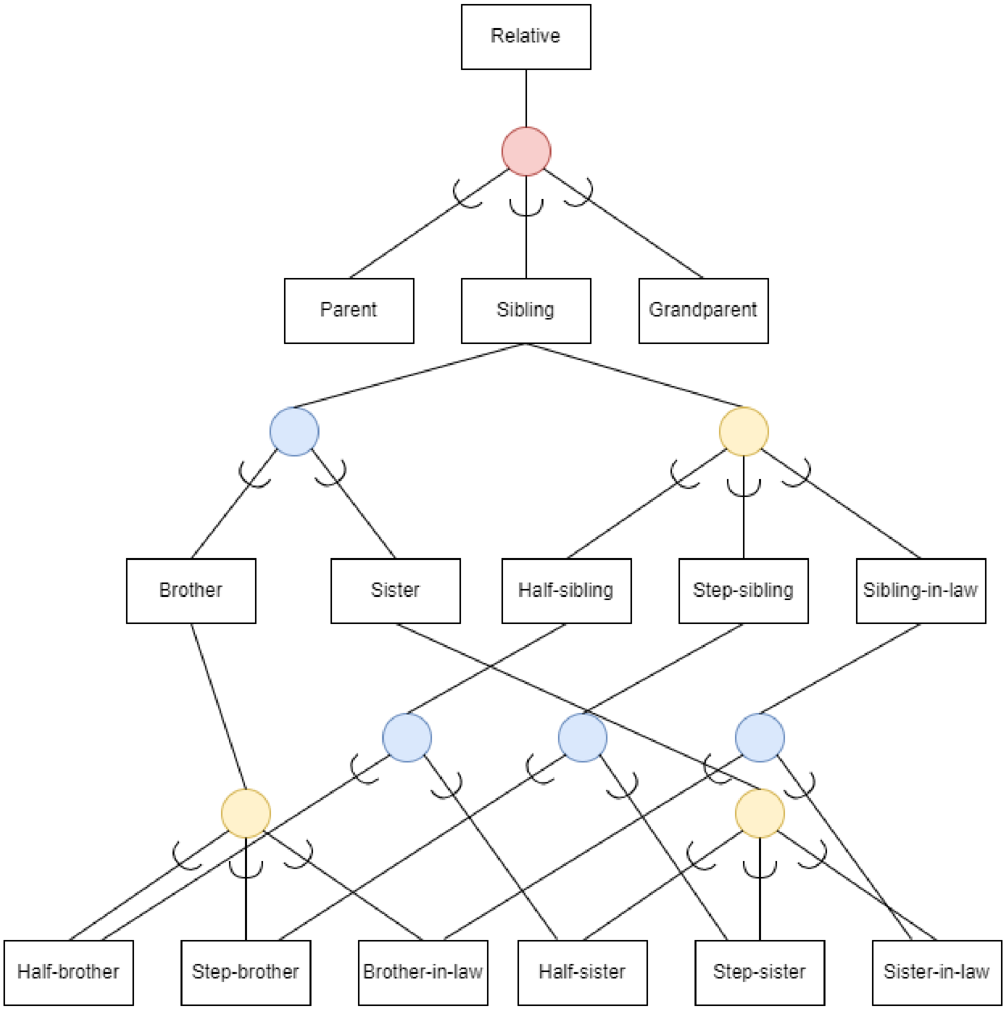


Figure 10.6: Third layer of relative subclasses with multiple inheritance. The three types of discriminators being used for subclassing are shown in different colors. See 10.5 for more information.

A proposed solution to this toy problem, and in fact to the problem of NAN material metadata management and perhaps provenance in general, is outlined in the following section. It is referred to as Concept Keys.

10.5 Concept Keys

There is a fifth (and sixth) approach to modeling a specialization lattice within a relational database. The key to this approach is to recognize that while there may be a dozen subclasses in the lattice shown in Figure 10.5 (and 40+ if we expanded the lattice to include parents, grandparents and other paths through the lattice), the discriminators for the subclasses are only of three types. They are shaded yellow, blue and red in the diagram.

The red discriminator subclasses the parent class based on **Generation** and has four options: sibling, parent, grandparent and NULL. The blue discriminator subclasses the parent class based on **Gender** and has three options: male, female and NULL. The yellow discriminator is a bit more complicated to define but it could be referred to as **Marriage** and has four options corresponding to half, step, in-law and NULL. Note that given these three keys and those options we can infer that there are 48 possible entities in the lattice.

$\prod(k_i)$ where k_i is the number of options (including NULL) for each key (discriminator).

For this lattice that yields 4 x 3 x 4 = 48 total classes. It is also interesting to note that while many of the subclasses have common names associated with them (one of the reasons for selecting this example for the toy problem) there are some which do not. There exist subclasses of relative for he-in-laws and she-in-laws which do not have a common name. (This is a common occurrence with faceted classification as will be discussed at the end of this chapter).

10.5.1 Formal Description of Concept Key Approach

Approach 5: Single relation with Concept Keys linking to an Entity-Attribute-Value style relation. The distinction between Approaches 5 and 6 are that one is for disjoint ONLY in which case the Concept Keys can be modeled as columns vs. the overlapping situation in which case they must be row-based. This approach requires 4 relations: a relation for the superclass, a relation between each record in the superclass and the concepts (discriminators) which it contains, a relation enumerating the properties available for each concept set, and finally an EAV-style relation for the actual values of the properties/attributes.

> 1. Create a single relation schema L with attributes Attrs(L) = $\{k, a_1, ..., a_n\}$ where k is the primary key and attributes a_i are attributes common to all members of the superclass. 2. Create a single relation schema CK with two attributes: k and c where k is the primary key of relation L and c enumerates the concept embodied by the instance of L. 3. Create a relation P with two attributes: c which are possible Concept Keys and p which are properties associated with the concept key set. 4. Create a relation EAV with three attributes k, p, v where k is the primary key for the superclass (the Entity), p is an available property from the P relation (the Attribute) and v is the Value for this property. This option can be used for all specialization types.

Approach 6: Single relation with Concept Keys linking to an Entity-Attribute-Value style relation.

> 1. Create a single relation schema L with attributes Attrs(L) = $\{k, a_1, ..., a_n\} \cup \{c_1, ..., c_n\}$ where k is the primary key, attributes a_i are attributes common to all members of the superclass and c_i are concepts which discriminate the subclasses. 2. Create a relation P with attributes Attrs(P) = $\{c_1, ..., c_n\} \cup p$, the latter which is a property associated with the concept key set. 3. Create a relation EAV with three attributes k, p, v where k is the primary key for the superclass (the Entity), p is an available property from the P relation (the Attribute) and v is the Value for this property. This option can be used for the disjoint case.

10.5.2 Unpacking Concept Keys

It is useful to illustrate these approaches with some hypothetical data as was done previously for the four textbook approaches. Concept Keys are most useful with a model that has a specialization lattice as provided by the toy problem, and the following example will provide a few relatives for a step-sibling, a sister-in-law and a parent. Concept Keys requires four relations.

All entities in the lattice are contained in a single relation that has all of the attributes for the superclass. In this example, that will include a name.

Table 10.5: Approach 5: Single relation with Concept Keys as rows

Relative	
PK	Name
R1	Alice
R2	Bob
R3	Charlie

Relative - Concept Keys	
Relative	Concept
R1	Step
R1	Sibling
R2	Sibling
R2	Female
R2	In-law
R3	Parent

The second relation is a binary relation between the superclass (relative) and concepts (the discriminators of the subclasses along the lattice).

Alice is both a Step and a Sibling (Step-sibling) but we do not know Alice's gender. Bob is a Sibling, a Female, and an In-law making Bob a sister-in-law. Charlie is a Parent of unknown gender or step/in-law relationship.

The third relation for this approach is a binary relation which defines the available attributes for combinations of Concept Keys.

Table 10.6: Approach 5: Concept Attributes Table

Concept Attributes	
Concept	Attribute
Parent	Scold Number
Sibling	Big Dessert Number
Step	Foo
Female	Bar
In-law	Baz
Step	Corge
Sibling	Corge
In-law	Garply
Sibling	Garply
In-law	Thud
Female	Thud
Sibling	Quz
In-law	Quz
Female	Quz

This table is a bit more complicated than the previous. In this case, a concept can be associated with more than one attribute. This reflects the fact that for any one subclass (like sister) there can be more than one attribute associated with that class. The reverse is also true in that an attribute can be associated with more than one concept. This allows for attributes to be associated farther down the specialization lattice. For instance, the attribute Quz applies to sisters-in-law as it requires all three Concepts: Sibling, Female and In-law.

The final relation is a standard Entity-Attribute-Value style table where the first column represents a row in the first relation (in this model, a relative), the second column represents an attribute for that relative (which are defined via the Relative-Concept Set-Attribute Set relations), and the third column represents the value for the attribute.

Table 10.7: Approach 5: Entity Attribute Value table

Entity Attribute Value		
Relative	Attribute	Value
R1	Foo	42
R1	Big Dessert Number	5
R1	Corge	True
R2	Big Dessert Number	0
R2	Bar	NULL
R2	Baz	NULL
R2	Garply	Blue
R2	Thud	Strong
R2	Quz	3.4
R3	Scold Number	2

It is worth noting that when storing all the values in a single EAV table, the schema for the value must be generic. Notice that there are Boolean, Integer, Real, and String value types in the third column. (When implementing EAV in a real-world system, multiple EAV tables are often provided, one for each type, allowing schema validation of the type.) Also, as part of the requirements, the attributes can be NULL.

10.5.3 Trade Offs

This implementation strategy has downsides. As mentioned above, this choice of reification moves most of the model structure from the schema level into values. Subclasses which would normally be represented as relations are represented as combinations of concepts. Attributes which would normally be columns are now represented as rows in the database. And the values are now denormalized and reside in a single table. A consequence of these choices is that the standard integrity checking built into relational databases would be of little use for this database structure.

I argue that the positive trade off is significant. In attempting to traverse the specialization lattice to find precisely which entity belongs to an individual record, it is no longer a matter of sifting through 48 possible entities but rather defining 3 concepts. This does imply some front-loaded difficulty in modeling the subclass discriminators - they must accurately represent the lattice while being able to be understood by the user.

The former issue of the denormalized state could be remediated through an appropriate use of database views. It would be possible to query the relations shown in Approach 5 and reconstruct any individual entity in the lattice as a table of its own (as done in Approach 1) as long as the concepts were also linked to the discriminators in the third relation. Each entity in the lattice is a unique tuple of the allowed concepts for each discriminator.

10.5.4 Approach 6: Partial Normalization if subclasses are disjoint

The need for pushing so much of the model structure into rows in the database is due not just to reframing the subclasses from entities to attributes, but also in managing overlapping subclasses at the same time.

(Recall the potential issue of step-brother-in-laws in which two concepts for the same discriminator apply to the same relative.) If the subclasses were all disjoint, the discriminators could be treated as columns rather than rows as in the following approach.

Table 10.8: Approach 6: Single relation with Concept Keys as columns

Relative				
PK	Name	Generation	Gender	Marriage
R1	Alice	Sibling	NULL	Step
R2	Bob	Sibling	Female	In-law
R3	Charlie	Parent	NULL	NULL

Concept Attributes			
Generation	Gender	Marriage	Attribute
Parent	NULL	NULL	Scold Number
Sibling	NULL	NULL	Big Dessert Number
NULL	NULL	Step	Foo
NULL	Female	NULL	Bar
NULL	NULL	In-law	Baz
Sibling	NULL	Step	Corge
Sibling	NULL	In-law	Garply
NULL	Female	In-law	Thud
Sibling	Female	In-law	Quz

Entity Attribute Value		
Relative	Attribute	Value
R1	Foo	42
R1	Big Dessert Number	5
R1	Corge	True
R2	Big Dessert Number	0
R2	Bar	NULL
R2	Baz	NULL
R2	Garply	Blue
R2	Thud	Strong
R2	Quz	3.4
R3	Scold Number	2

10.6 Connections to Related Works

While this implementation strategy for representing subclasses of a specialization lattice within a relational system is novel (as far as I am aware), there are many connections to related works which will be described in this section. They include faceted classification, formal concept analysis, composition over inheritance, Entity-Attribute-Value modeling as well as RDF and OWL.

10.6.1 Faceted Classification and Search

Faceted Classification

Faceted classification is an alternative to hierarchical and enumerative classification schemes. Enumerative classification simply refers to enumerating all items in the domain or collection and defining where the item belongs within the classification scheme. Hierarchical classification refers to the classification being strictly hierarchical (as with the taxonomy of life on Earth). For instance, a bat is a mammal which is an animal. The fact that a bat can fly does not make it a descendant of birds (most of which also fly).

Colon Classification

Faceted classification is often attributed to Shiyali Ramamrita Ranganathan, a mathematician turned librarian, who developed Colon Classification[61] in the early twentieth century. Ranganathan was dissatisfied with the hierarchical Dewey Decimal Classification System[62], particularly as there were books which could have been classified in more than one manner with Dewey's system yet existed in only one location in the hierarchy. The Colon Classication system, on the other hand, classifies books along several different facets which are delineated by : symbols giving the system its name.

The facets within colon classification have evolved over time but a guiding principle of Raganathan was that there should be only 5 fundamental categories to colon classification. Those five categories are known by the acronym PMEST referring to Personality, Matter, Energy, Space and Time. Space and Time are simple to contemplate, a book about Paris in the 19th century has the location (Space) of Paris and the era (Time) of the 1800's. The other three categories are a not as easily defined, in part because they differ based on the context. As an example used by Satija[61], gold would be consider Matter to an a numismaticist but Personality to a chemist or mineralogist.

An interesting and important quote from Raganathan[61]:

> Design work of any kind has to draw largely from intuition unmediated as far as possible by the intellect or by rules framed by intellect. In its general makeup, a scheme of library classification will have to come out whole as an egg from the intuition of a classificationist of the creative variety. The intellectual classificationist can only polish it with the aid of a theory germane to it.

This same issue is in play with Concept Keys as was mentioned in section 10.5.3 regarding the challenge in modeling the subclass discriminators correctly. The method of discriminating subclasses or faceting a classification is of vital importance but something of a creative act. However, Formal Concept Analysis is a method which can be used as a guide as will be discussed in section 10.6.2.

The end result of colon classification is a string of characters separated by colons along the major categories of classification. To use the example provided by wikipedia[3]:

$$L, 45; 421 : 6; 253 : f.44'N5$$

would refer to a classification of:

Medicine,Lungs;Tuberculosis:Treatment;X-ray:Research.India'1950

Note that this classification is a combination of orthogonal facets: Medicine, Treatment, and Research along with hierarchical specialization within the facets as in Medicine → Lungs → Tuberculosis. This is

[3] https://en.wikipedia.org/wiki/Colon_classification

reminiscent of the identifier system recently developed by the International Union of Pure and Applied Chemistry (IUPAC).

InChI

The International Chemical Identifier (InChI) system was developed to tackle a problem in reporting chemical compounds. That problem is that there are multiple names for the same chemical substance, either synonyms or translations to other languages. IUPAC has historically provided guidance for standardized naming conventions for chemical compounds, which unfortunately can be very long and tedious to write/read. The InChI system is an analogous effort to provide unique machine-interpretable identifiers for chemical compounds. An example InChI identifier for glucose is:

$$1S/C6H12O6/c7-1-2-3(8)4(9)5(10)6(11)12-2/h2-11H,1H2/t2-,3-,4+,5-,6+/m1/s1$$

There is a striking similarity between the InChI string and a colon classification. In the InChI system there are multiple *layers* of representation of a molecule which are separated by a slash, '/', rather than by a colon. The first characters of 1S refer to the current version of the InChI identifier system, version 1. The next segment is the molecular formula for glucose, $C_6H_{12}O_6$. A model of the molecular structure of glucose is shown in Figure 10.7.

The next few sections are more involved but provide the molecular connectivity of the heavy atoms (non-hydrogen atoms), the connectivity for the hydrogens, and the sterochemistry of the chiral centers.

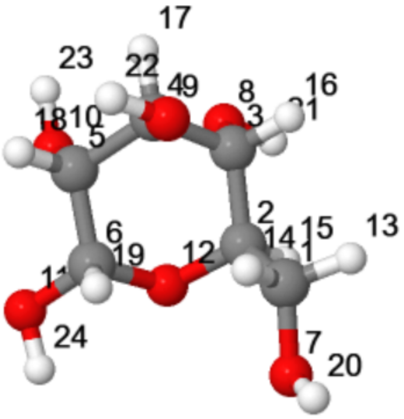


Figure 10.7: Ball and stick representation of glucose along with InChI numbering of the respective atoms.

This is very similar to colon classification as there are several layers, many of which are optional, but within a given layer there is a hierarchical level of specialization. However, unlike true facets, the layers themselves are hierarchical. As stated by Heller, *et al.*[63]:

> Each layer in an InChI representation contains a specific type of structural information. These layers, automatically extracted from the input structure, are designed so that each successive layer adds additional detail to the Identifier. The specific layers generated depend on the level of structural detail available and whether or not allowance is made for tautomerism. ... This layered structure design of an InChI offers a number of advantages. If two structures for the same

> substance are drawn at different levels of detail, the one with the lower level of detail will, in effect, be contained within the other.

This makes the InChI system very similar to a specialization lattice which would benefit from Concept Keys. If a particular level of detail is not available, it is left out.

One final word on InChI, the data model for materials which is a motivation for this chapter will include chemical substances using InChI strings, however, biological macromolecules use other identifiers. (As might be expected, there is a size limitation with the InChI string that as the number of atoms in the molecule increase the string length increases dramatically.

Faceted Search

One final related point regarding facets is their use in faceted search. Faceted search is a technique for searching for items on the web or within an information system in which the user is not forced into hierarchical thinking. A commonplace example of this would be searching for a product on Amazon. When searching for a particular item, a panel is produced on the left-hand side of the screen in which are provided multiple refinements to the search. For instance, if searching for a shirt, the refinements might include price, material, color, style and size. These separate facets are auto-generated for the parent selection of “shirt”.

While exact implementation details are in the purview of Amazon, this can be accomplished on the back-end by simply assembling a list of attributes for any product sold by Amazon. When a parent search term is entered, all of the attributes associated with any product of that search term are queried and the top 5-10 attributes are displayed as facets which are capable of further refinement. This technique is known as faceted search and is ubiquitous on the web both for online retailers as well as for online catalogs in libraries, archives and museums [64].

10.6.2 Formal Concept Analysis

Formal Concept Analysis (FCA) is a technique in information science for developing an ontology from a set of items. The approach is to categorize all of the identifying attributes for each item in the set or collection similar to the use of character matrices in developing taxonomies of life. Once the attributes have been identified, each item is recorded as to a true or false claim regarding the attribute. For instance, one might note that a bird has feathers and wings, a dog has fur and legs, and a bat has fur and wings. These attributes would allow for the creation of a concept lattice where each combination of attributes would specify a unique concept.

FCA differs from the approach in this chapter in that Concept Keys already presume that a specialization lattice is known to exist and the problem is how to implement a relational database solution for a lattice with a large number of subtypes. FCA, on the other hand, is a formal way of defining a lattice given a set of known items.

FCA is complementary to Concept Keys as the formal analysis can be used to assist in defining or analyzing the discriminators chosen for the subtypes along the lattice. In the toy problem, for instance, FCA could be used to explore the legitimacy of grouping *step*, *half*, and *in-law* together under one discriminator when there do not exist half-parents or half-grandparents.

10.6.3 Object Oriented Programming Design Pattern

It might seem strange to consider object-oriented programming (OOP) as a related work but a specialization lattice is a form of multiple inheritance. (In the toy problem, a sister-in-law *inherits* the attributes from sister as well as from in-law and also has new attributes specific to the subclass).

The seminal book on object-oriented design patterns[65] by the so-called Gang of Four stresses that best practices in OOP are to forgo inheritance for composition. The difference between the two approaches is that inheritance models dependence between classes as *is A* relationships, such as a bat is a mammal is an animal, while composition models dependencies as *has A* relationships, such as a bat has wings and a bat has fur.

The major connection between Concept Keys and OOP composition is the approach of changing reification. Switching between *is A* and *has A* relationships is similar to switching between attributes and values. It is sometimes useful to consider this in the choice of vocabulary. If a Person class is subclassed into Man and Woman classes, this distinction is reified at the entity or table level. If however, the Person class records an Boolean Attribute for isMale or isFemale, than the distinction is reified at the schema level through a table column. If the Person class has an attribute which is maleness or femaleness, then the distinction is reified as a value and is subtly altering the meaning from Woman is a Person or Woman is a Female Person to Woman is a Person who has Femaleness.

That said, it is worth pointing out that the reason for encouraging composition over inheritance is due to the problems associated with multiple inheritance - especially with overlapping subclasses which is one of the important situations in which Concept Keys are useful.

10.6.4 Entity-Attribute-Value Modeling

The final step in implementing Concept Keys via either approach 5 or 6 is to create an Entity-Attribute-Value (EAV) style table. EAV is considered an anti-pattern as it goes against the properties and benefits for which relational databases are usually chosen.

EAV is used for sparse tables, that is tables which have a large number of attributes but for any individual row most of the fields are NULL. In such situations, there is a cost for carrying around large numbers of columns and NULLs both in storage and in queries. Converting such a system to EAV allows one to store only the non-NULL attributes. It also eliminates the need to rigorously define the data model and schema at the onset - the schema is simply the triple of entity (table record), attribute (table column) and value.

The consequence as discussed for Concept Keys is that the integrity checking built into relational databases systems is thwarted, including the most simplest of defining the schema for values. If all values exist in the same table, than Booleans, integers, reals and strings must all be coerced into strings. Alternate implementations of EAV restore schema checking by creating several EAV tables, one for each type.

EAV modeling is also known as a type of row-modeling which refers to reifying the data structure as individual rows of EAV triples rather than modeling the attributes as columns.

The Concept Key approaches described in this chapter suffer from the same shortcomings as EAV in general, although it is not suggested that the entire model be EAV, only the portion referring to the specialization lattice. The specialization lattice is what gives rise to the potentially sparse relation (large number of attributes which can be NULL); however, the emphasis with Concept Keys is understanding how to populate the attributes rather than on space savings or query efficiency on sparse table.

10.6.5 Ontologies: RDF and OWL

The Resource Description Framework (RDF) is a standard developed by W3C for representing metadata about resources (and was discussed briefly in Chapter 4). It is structurally similar to EAV modeling in that RDF statements contain a triple of fields. However, unlike EAV which stores entities (table records), attributes and values, RDF triples represent subjects, predicates and objects.

The distinction is more semantic than structural. RDF allows statements such as

```
:Woodstock :isA :bird
```

which indeed is subject-predicate-object. However, it also allows statements such as

```
:Woodstock :hasColor :yellow
```

which is similar to an EAV record.

The structural similarity between EAV and RDF means the same similarity exists between the Concept Key approaches and RDF. However, the power of RDF is in extending the graph as in linked data approaches such as by reasoning/querying over pairs of statements such as

```
:Woodstock a :bird
:Snoopy :befriends :Woodstock
```

implies

```
:Snoopy :befriends :bird
```

RDF is very powerful and both RDF and OWL could certainly be used to model specialization lattices as they are designed for graphs and networks. That said, there are many real-world systems in which a specialization lattice is part of the domain and yet the information system requirements call for a relational database implementation rather than OWL, RDF and SPARQL. For those situations, Concept Keys can be useful.

10.7 Other Use Cases

10.7.1 Starbucks Beverages

Specialization lattices exist in commercial product lines. For instance, the Coca Cola Freestyle vending machine[4] boasts providing over 200 beverages which it can accomplish because many distinct beverages (Classic Coke, Diet Coke, Caffeine-free Coke, Diet caffeine-free Coke, Diet Cherry Coke, etc.) differ by discriminators of *sugar type*, *caffeine*, *added flavor* along a specialization lattice. Another similar example can be found with Starbucks coffee.

Figure 10.8 is the the top menu page for ordering beverages at Starbucks. Similar to Coca Cola, Starbucks offers a large number of distinct beverages which go by specialized names like Mocha, Latte, Coffee, Iced Tea, etc. Also similar to Coca Cola, behind the large variety is a specialization lattice in which the facets of the various beverages are the type of coffee or tea, whether the beverage is caffeinated or not, whether milk or chocolate flavor is added, and whether the beverage is served hot or cold.

Unlike the toy problem in which a design requirement was that relatives could exist at an intermediate level of specificity (like sibling), in the Starbucks store every beverage is eventually realized as an actual

[4] https://www.coca-colacompany.com/media-center/coca-cola-freestyle-brings-variety

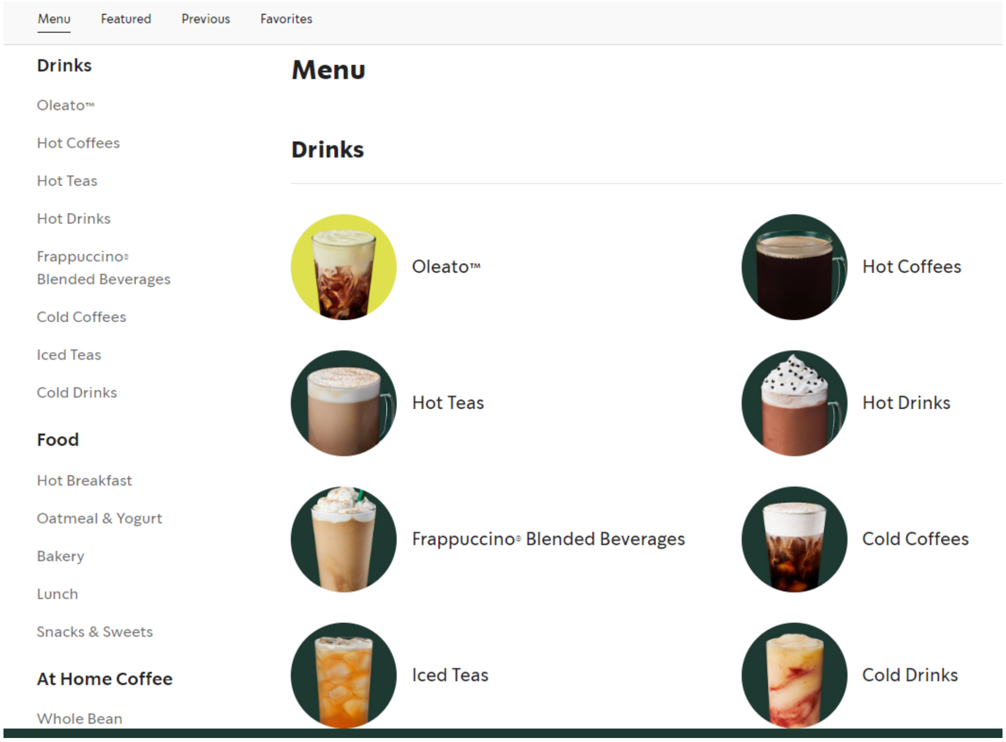


Figure 10.8: Top level beverage selection window for Starbucks online ordering.

delivered item and so Starbucks enforces that all beverages must exist at the maximum level of specificity. (This is similar to the next example on NMR Ambiguity Codes).

Yet, there is still a problem of navigating through the lattice to find the appropriate beverage. However, since all beverages exist as terminal leaves, the relative path through the lattice is unimportant as long as the final beverage is found. For that reason, Starbucks implements their product search through a hierarchy of choices. The top page shown in Figure 1 forces the customer to choose coffee vs. tea vs other as well has hot vs cold. There is no option for caffeine at this point (which is much to my chagrin as that is my primary personal discriminator.)

Only after selecting a top level category (hot coffee for example) and a mid level category (decaf coffee for example) does the customer get to the final screen shown in Figure 10.9.

At this point, the customer can choose a large assortment of customizations which could have been listed as distinct entities (as in sister-in-law, etc.) but presumably the combinatorics would be too large for a customer to remember. Therefore, the customer can simply customize. An open question is whether the customization at this point could actually replicate an identical beverage which exists at a different node in the Starbuck's hierarchy.

10.7.2 NMR Ambiguity Codes

In the field of NMR, one of the first steps in data analysis is to assign each of the observed signals to individual atoms/nuclei in the molecule under interest. The process of this "chemical shift assignment" is to start with a set of ambiguous experiments and by collecting more information attempt to hone in on the precise nucleus giving rise to each signal.

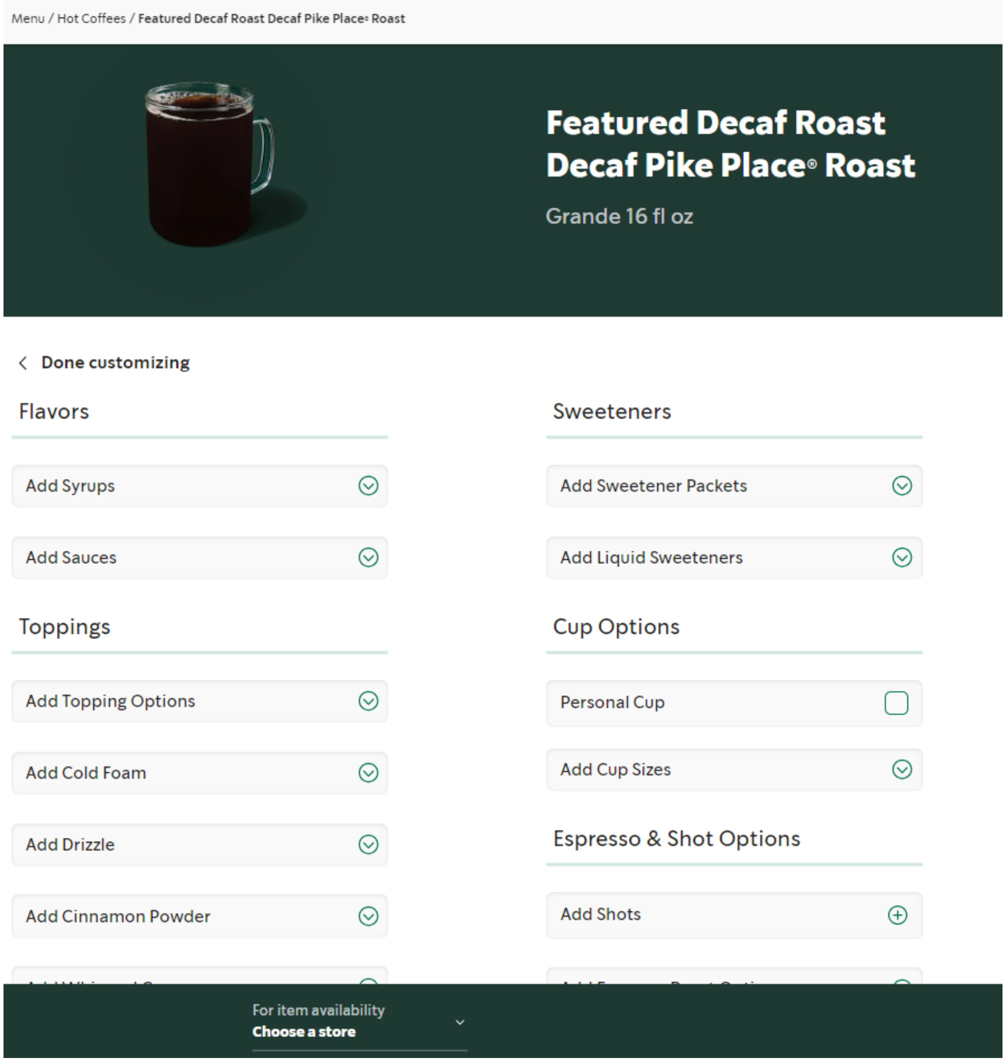


Figure 10.9: Bottom level beverage selection window for Starbucks online ordering.

As an example, a spectroscopist might begin by collecting a 1D ^{1}H experiment on a protein and records a specific chemical shift resonance frequency. The details of the experiment predetermine that this signal must arise from a hydrogen atom (^{1}H). Recognizing that atoms in proteins resonate within characteristic frequency ranges, one might be able to infer from the frequency that this hydrogen atom is covalently bound to a carbon atom.

Next the spectroscopist runs a series of TOCSY and COSY experiments which establish through-bond connectivities and determines that this particular signal arises from a β-hydrogen of the amino acid leucine. At this point it is known it is from a leucine but not which leucine as the protein has several.

Next the spectroscopist records sequential assignment experiments and after examining these determines the signal arise from a β-hydrogen of leucine 44. However, there are two β-hydrogens for this residue and at this point it is not known which. Sometimes more experiments are collected and the ambiguity between the two β-hydrogens resolved. However, often this ambiguity is not resolved as it may be unimportant since the two hydrogens are so close together in space.

In effect, the process of chemical shift assignment is traversing a specialization lattice of atom types -

from the most general as in ^1H to the most specific as in $^1H\beta_3$-L44. Upon deposition of these assignments in the BMRB, the spectroscopist is **required** to report the assignments at the most specialized identifier available. However, as mentioned earlier, sometimes this is not known. The BMRB's solution to this issue is to provide ambiguity codes which are entered to indicate that although the name of this atom is compeltely specialized, in reality it is a superclass higher up the specialization lattice.

In effect, the BMRB has chosen to employ the second approach[5] to modeling subclasses with all of the assignments being at the most specialized subclasses. The fact that some assignments are in reality at a more general level of specialization is conveyed through a codebook of ambiguity codes[6] for the most common issues where this arises in NMR.

10.7.3 Provenance Revisited

The topic of this dissertation is provenance in the information sciences. Concept Keys can also be applied to modeling and documenting provenance. As discussed in the first few chapters, provenance can refer to the past (retrospective provenance) or the future (prospective provenance). The PROV and PREMIS models are designed for retrospective provenance; however, PREMIS can be used for both retrospective and prospective provenance as was done with CONNJUR workflow builder in chapters 5 and 6. The examples in chapters 3 and 4 illustrate this for PROV as well.

That said, there have been efforts to provide a more detailed accounting of both prospective and retrospective provenance on the same computational workflow which have resulted in various extensions to PROV including ProvONE[53]. The ProvONE model allows for a method of connecting entities which exist in the prospective (workflow) perspective with those which exist in the retrospective (trace) perspective. In a sense, one can consider ProvONE as a method for subclassing generic provenance into retrospective and prospective. To take this analogy one step further, one can consider that ProvONE reifies the subclasses at the entity level, similar to the BMRB with chemical shift assignments.

To return to the original argument of this chapter, what happens if the subtypes proliferate? ProvONE is much more complicated than PROV and has approximately twice the number of entities. If other researchers deem it important to model subjunctive provenance as in chapters 7 and 8, or other temporal aspects of provenance, a future model will become increasingly more complicated.

Concept Keys provide a potential solution to the problem of unmanaged and unmanageable complexity. If provenance is indeed a specialization lattice, the solution is not to reify all aspects of provenance as additional tables with complicated relationships between each other. The solution is to allow any level of specialization to be specified with a common model and allow context-dependent attributes to be included for the various levels of specialization (for instance, temporal concerns ala retrospective, prospective, or subjunctive).

Explorations on this front will have to remain future directions at this point, but it is hoped that Concept Keys will be useful in developing subjunctive provenance further.

[5] Note that while the BMRB does use a relational database in its operations, the data format is actually in a format called STAR which is not relational

[6] https://bmrb.io/deposit/shifts_example_help.shtml

Chapter 11

Conclusions

11.1 Abstract

This chapter concludes this dissertation and summaries the motivations of my doctoral research as well as the results and contributions to Information Sciences. This chapter is a reflection of the introductory chapter and also a reflection on the work undertaken over the past decade of my career and studies at the University of Illinois.

11.2 The Journey

> Life is a Journey, not a Destination - Ralph Waldo Emerson

This dissertation is entitled "Explorations in Provenance in the Information Sciences." This was not a title I had expected when I began my pursuit for a second doctoral degree at GSLIS[1] almost a decade ago. Although provenance was on my mind when I began my studies, I was originally motivated by the "reproducibility crisis" in the natural sciences[10], [66] and how to resolve such a crisis.

While my original motivation was reproducibility, my academic pursuits were never far removed from provenance. Just as I believe that prospective and retrospective provenance are just additional layers of specificity to the broader concept of provenance (a theme throughout this dissertation), I also believe that reproducible computation can be thought of as just another provenance subclass similar to prospective workflows and retrospective provenance. Reproducibility demands documenting how something has come to be in enough detail that someone else can bring that same thing into being as well. In a sense, reproducible experiments are a merger between the prospective and retrospective.

This topic is discussed in Chapter 2 on Workflows and Provenance (which really could be entitled Workflows, Provenance and Reproducibility). This chapter introduces the concepts of workflows and how they are related to provenance along with various methods for documenting and executing workflows. Reproducibility is broached in the context of the PRIMAD model for testing the various aspects of computational reproducibility - namely the dependencies on Platforms, Research Objectives, Implementations, Methods, Actors and Data. As in much of this work, the information sciences problems are framed in the context of the scientific discipline

[1] GSLIS was a very popular acronym for the Graduate School of Library and Information Science at the University of Illinois - Urbana/Champaign. Now renamed as the School of Information Sciences, the term GSLIS is defunct; however, it is pleasing to see the acronym in print one last time.

of biomolecular NMR spectroscopy. UCONN Health has provided provisioned Virtual Machines for the bioNMR community for over ten years and those VMs are a great test bed for the various aspects of PRIMAD.

While reproducibility in the sciences is never far from the surface for the rest of the dissertation, it stopped being the central issue not too far into my dissertation research. One reason for this was the debate over the definition of reproducibility, in particular on its relationship to replicability. This was a fairly heated discussion as different domains of scientists swap the definitions of the two words. So as confusing as it might be to disambiguate American chips from British chips, it would be doubly more complicated if the term *crisps* was also swapped. There was an effort led by Tim McPhillips[4] to help resolve the issue by proposing to namespace the terms and also to simply switch the focus from reproducibility to transparency - as in transparent research objects.

Crafting transparent research objects is what CONNJUR Workflow Builder was refactored to do in Chapters 5 and 6. In these chapters, recording/documenting both prospective and retrospective provenance is discussed in the context of computational aspects of bioNMR. In that community, there are many mathematical operations which are applied to the data prior to analysis. These operations are akin to data cleaning and their parameterization is connected to attributes of the underlying data.

Prior to my studies at Illinois, I oversaw the development of both the CONNJUR Workflow Builder and CONNJUR Spectrum Translator programs. The goal of those projects were software interoperability which intrinsically requires data interoperability (I is one of the FAIR components). CWB is a custom, domain-specific workflow management system which reported both prospective provenance (workflows) and retrospective provenance (termed reconstructions) in a custom XML schema. The fact that the schema was custom to CWB was a violation of the FAIR principles - that data and metadata should be reported using knowledge representation languages which are broadly applicable. While XML fit that bill, the custom schema did not.

Chapters 5 and 6 report on refactoring the provenance metadata documentation using a combination of the PREMIS standard used in digital preservation along with a domain-specific data model to record metadata pertinent to the biomolecular NMR experiment. PREMIS is a broadly used knowledge representation language (FAIR principle I1) and is flexible enough to accommodate bioNMR workflows for spectra reconstruction. In order to support provenance and the three top-level concepts of Objects, Events and Agents - the metadata for bioNMR was decomposed into these three concepts. Metadata about the users, software as well as the spectrometer instrumentation were packaged as Agent extensions. Metadata about the experimental data collection event as well as data cleaning activities were packaged as Event extensions. Finally, metadata about the data which results from the NMR experiment were packaged as Object extension. Finally, a new standard file format for NMR spectra was proposed which includes both the data files as well as metadata files (including the PREMIS record) in a tar container. This is akin to the Office file formats, docx, xlsx, etc.

By refactoring CWB in this manner, we hoped to empower NMR spectroscopists to be curators of the data they created. Embedding a standard like PREMIS into the CONNJUR application means that these domain scientists don't have to be versed in information science standards and can concentrate on their own disciplinary tasks while still curating data for broader consumption. This was expanded on in Chapter 6 by embedding analytics within CWB and including those metrics within the PREMIS provenance record. While superficially simple, there were a few challenges. Primarily, PREMIS is designed to link software agents to Object blocks or Event blocks - while for recording the provenance of the analytical measurements, it is more appropriate to link the Agent to the granular metadata reported rather than the entire Object block.

While the CONNJUR work used PREMIS as the provenance metadata standard, there is another widely used knowledge representation language for provenance - W3C PROV. Throughout my doctoral study at the University of Illinois, several colleagues asked me why I chose to record provenance using PREMIS rather than PROV. The superficial answer is that PREMIS was easier to conceptualize and implement as PREMIS is designed for practitioners in digital preservation while PROV is documented at a more abstract level. Not content with the superficial response, I conducted a deep read into the W3C PROV standard with an objective to compare and contrast the two standards in practice. Part of this early work was presented at the 2018 International Conference in Knowledge Management [27]. It also led to several workshops given by my colleagues Rhiannon Bettivia, Jessica Cheng and myself at the International Data Curation Conference in 2019, ASIS&T in 2019, and the iConference in 2020. Interest by Gary Marchianini at ASIS&T led to the three of us writing a book on PROV, ProvONE and PREMIS entitled Documenting the Future[8] - the details of PROV which are covered in Chapters 3 and 4.

Documenting the Future was intended to be a written version of our workshops which started as in-person workshops with cross-walking exercises using index cards and strings but morphed into online versions during the pandemic using Miro collaboration software. The book includes the didactic material used in the workshops as well as lessons we had learned in offering the workshops.

It turns out writing the book left us with more unanswered questions. One of those exploring the temporal aspects of provenance. This dissertation has delved into both prospective and retrospective provenance along with the divide between the two which is spanned by standards like the P-Plan Ontology and ProvONE. However, if we consider that provenance metadata is storytelling about an object's past and future, we notice that there are other tenses which can be used to narrate these events. In the concluding chapter of *Documenting the Future* we considered the subjunctive mood as a way of distinguishing prospective provenance of what will happen from the conditional provenance of what could happen.

This is discussed in Chapter 7 in the context of building IKEA furniture. It is argued that having a manner to discuss subjunctive provenance alongside prospective and retrospective could allow for comparisons and quality control and assessment. In the context of computational workflows, iterative workflows or Directed Cyclical Graphs could be defined in which case when a computational node is reached, the node retrospectively knows what has happened prior to its execution, the node prospectively knows how to execute the remaining workflow, and the node subjunctively knows about what the outcome would be for parameterizations which were deemed sub-optimal.

In this context, retrospective, prospective and subjunctive provenance are simply three different specializations or sub-classes of provenance more broadly defined. Yet how can we model these various specializations? Do we need another provlet or set of provlets to connect Executions to Programs to Hypotheses? If so, won't this explode on us if we discover more provenance tenses are useful or necessary?

Chapter 8 discusses how to manage temporal changes in the context of significant properties of metadata - specifically for preserving objects which by their very nature change over time. Provenance itself can be one of those significant properties - young Calvin and old Calvin share some provenance (that of the young Calvin), but differ in the extra experience that old Calvin has gained. Finally, when it comes to documenting and modeling provenance, if these various temporal aspects of retrospective, prospective, subjunctive and others are to be modeled on a level playing field - how can that be accomplished?

Chapter 9 uses the childhood problem of Which One Doesn't Belong to introduce and investigate how the manner in which a model is reified impacts both the structure of the data which can be maintained in the information system as well as which questions one can ask. The concept of reification and particularly of re-reification is front-and-center in Chapter 10 which describes a novel method for modeling a specialization lattice within a relational database. While this problem is particularly thorny for a relational model, it is also difficult for hierarchical representations such as XML.

Life is a journey, not a destination. If we think about life in the tenses of provenance, retrospectively it is how we came to be here right now at the conclusion of doctoral studies. Prospectively, it is the recipe for how research is conducted to learn something new about the world (however small). Subjunctively, it is about all the paths not taken, the research which might have been done, the things which could have been discovered. But those paths are still available post graduation and I look forward to converting more of the "what could have been" to "what will be" and finally to "what have come to be".

References

[1] R. Denenberg, "Premis: Preservation metadata xml schema version 3.0," in *Library of Congress, Washington DC*, 2014.

[2] J. Freire, N. Fuhr, and A. Rauber, "Reproducibility of data-oriented experiments in e-science (dagstuhl seminar 16041)," in *Dagstuhl Reports*, Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, vol. 6, 2016.

[3] M. Fenwick, G. Weatherby, J. Vyas, *et al.*, "Connjur workflow builder: A software integration environment for spectral reconstruction.," *Journal of biomolecular NMR*, vol. 62, no. 3, pp. 313–326, 2015. DOI: 10.1007/s10858-015-9946-3. [Online]. Available: https://doi.org/10.1007/s10858-015-9946-3.

[4] T. McPhillips, C. Willis, M. R. Gryk, S. Nunez-Corrales, and B. Ludäscher, "Reproducibility by other means: Transparent research objects," in *2019 15th International Conference on EScience (EScience)*, IEEE, 2019, pp. 502–509.

[5] M. D. Wilkinson, M. Dumontier, I. J. Aalbersberg, *et al.*, "The fair guiding principles for scientific data management and stewardship," *Scientific data*, vol. 3, no. 1, pp. 1–9, 2016.

[6] A. Jacobsen, R. de Miranda Azevedo, N. Juty, *et al.*, *Fair principles: Interpretations and implementation considerations*, 2020.

[7] B. Mons, C. Neylon, J. Velterop, M. Dumontier, L. O. B. da Silva Santos, and M. D. Wilkinson, "Cloudy, increasingly fair; revisiting the fair data guiding principles for the european open science cloud," *Information services & use*, vol. 37, no. 1, pp. 49–56, 2017.

[8] R. Bettivia, Y.-Y. Cheng, and M. Gryk, *Documenting the Future: Navigating Provenance Metadata Standards*. Spring Nature, 2022.

[9] M. R. Gryk and B. Ludäscher, "Workflows and provenance: Toward information science solutions for the natural sciences," *Library trends*, vol. 65, no. 4, p. 555, 2017.

[10] V. Stodden, M. McNutt, D. H. Bailey, *et al.*, "Enhancing reproducibility for computational methods.," *Science (New York, N.Y.)*, vol. 354, no. 6317, pp. 1240–1241, 2016. DOI: 10.1126/science.aah6168. [Online]. Available: https://doi.org/10.1126/science.aah6168.

[11] G. Bak, "Trusted by whom? tdrs, standards culture and the nature of trust," *Archival Science*, vol. 16, pp. 373–402, 2016.

[12] S. Bowers and B. Ludäscher, "Actor-oriented design of scientific workflows," in *International Conference on Conceptual Modeling*, Springer, 2005, pp. 369–384.

[13] B. Ludäscher, M. Weske, T. McPhillips, and S. Bowers, "Scientific workflows: Business as usual?" In *International Conference on Business Process Management*, Springer, 2009, pp. 31–47.

[14] P. Groth and L. Moreau, *Prov-dm: The prov data model*, [Online; accessed 11-August-2023], 2013. [Online]. Available: https://www.w3.org/TR/2013/REC-prov-dm-20130430/.

[15] I. Altintas, C. Berkley, E. Jaeger, M. Jones, B. Ludascher, and S. Mock, "Kepler: An extensible system for design and execution of scientific workflows," in *Proceedings. 16th International Conference on Scientific and Statistical Database Management, 2004.*, IEEE, 2004, pp. 423–424.

[16] D. Thain, T. Tannenbaum, and M. Livny, "Distributed computing in practice: The condor experience," *Concurrency and computation: practice and experience*, vol. 17, no. 2-4, pp. 323–356, 2005.

[17] L. Murta, V. Braganholo, F. Chirigati, D. Koop, and J. Freire, "Noworkflow: Capturing and analyzing provenance of scripts," in *Provenance and Annotation of Data and Processes: 5th International Provenance and Annotation Workshop, IPAW 2014, Cologne, Germany, June 9-13, 2014. Revised Selected Papers 5*, Springer, 2015, pp. 71–83.

[18] T. McPhillips, S. Bowers, K. Belhajjame, and B. Ludäscher, "Retrospective provenance without a runtime provenance recorder," in *7th USENIX Workshop on the Theory and Practice of Provenance (TaPP 15)*, 2015.

[19] M. W. Maciejewski, A. D. Schuyler, M. R. Gryk, *et al.*, "Nmrbox: A resource for biomolecular nmr computation," *Biophysical journal*, vol. 112, no. 8, pp. 1529–1534, 2017. DOI: 10.1016/j.bpj.2017.03.011. [Online]. Available: https://doi.org/10.1016/j.bpj.2017.03.011.

[20] M. Fenwick, J. C. Hoch, E. L. Ulrich, and M. R. Gryk, "Connjur r: An annotation strategy for fostering reproducibility in bio-nmr-protein spectral assignment.," *Journal of biomolecular NMR*, vol. 63, no. 2, pp. 141–150, 2015. DOI: 10.1007/s10858-015-9964-1. [Online]. Available: https://doi.org/10.1007/s10858-015-9964-1.

[21] E. L. Ulrich, H. Akutsu, J. F. Doreleijers, *et al.*, "Biomagresbank.," *Nucleic acids research*, vol. 36, no. Database issue, pp. D402–8, 2007.

[22] L. Moreau, P. Groth, J. Cheney, T. Lebo, and S. Miles, "The rationale of prov," *Web Semant.*, vol. 35, no. P4, pp. 235–257, Dec. 2015, ISSN: 1570-8268. DOI: 10.1016/j.websem.2015.04.001. [Online]. Available: https://doi.org/10.1016/j.websem.2015.04.001.

[23] P. Groth and L. Moreau, *An overview of the prov family of documents*, [Online; accessed 11-August-2023], 2013. [Online]. Available: https://www.w3.org/TR/prov-overview/.

[24] L. Moreau and P. Groth, *Provenance: An Introduction to PROV.* (Synthesis Lectures on the Semantic Web: Theory and Technology). Morgan & Claypool, 2013.

[25] D. Garijo and Y. Gil, "Augmenting prov with plans in p-plan: Scientific processes as linked data," Jan. 2012.

[26] Y. Cao, C. Jones, V. Cuevas-Vicenttin, *et al.*, "Provone: Extending prov to support the dataone scientific community," *PROV: Three Years Later*, 2016.

[27] M. R. Gryk, P. Shrivastava, and B. Ludaescher, "A rosetta stone for provenance models," 2018.

[28] T. C. Chao, M. H. Cragin, and C. L. Palmer, "D ata p ractices and c uration v ocabulary (dpcv ocab): An empirically derived framework of scientific data practices and curatorial processes," *Journal of the Association for Information Science and Technology*, vol. 66, no. 3, pp. 616–633, 2015.

[29] H. J. Ellis, R. J. Nowling, J. Vyas, T. O. Martyn, and M. R. Gryk, “Itng - iterative development of an application to support nuclear magnetic resonance data analysis of proteins,” *Proceedings of the ... International Conference on Information Technology: New Generations. International Conference on Information Technology: New Generations*, vol. NA, no. NA, pp. 1014–1020, 2011. DOI: 10.1109/itng.2011.215. [Online]. Available: https://doi.org/10.1109/itng.2011.215.

[30] K. K. Verdi, H. J. C. Ellis, and M. R. Gryk, “Conceptual-level workflow modeling of scientific experiments using nmr as a case study,” *BMC bioinformatics*, vol. 8, no. 1, pp. 31–31, 2007. DOI: 10.1186/1471-2105-8-31. [Online]. Available: https://doi.org/10.1186/1471-2105-8-31.

[31] C. Willoughby and J. G. Frey, “Documentation and visualisation of workflows for effective communication, collaboration and publication@ source,” *International Journal of Digital Curation*, vol. 12, no. 1, pp. 72–87, 2017.

[32] C. L. Palmer, A. K. Thomer, K. S. Baker, *et al.*, “Site-based data curation based on hot spring geobiology,” *PloS one*, vol. 12, no. 3, e0172090, 2017.

[33] C. Willoughby and J. G. Frey, “Documentation and visualisation of workflows for effective communication, collaboration and publication@ source,” *International Journal of Digital Curation*, vol. 12, no. 1, pp. 72–87, 2017.

[34] D. Heintz and M. R. Gryk, “Curating scientific workflows for biomolecular nuclear magnetic resonance spectroscopy,” *International journal of digital curation*, vol. 13, no. 1, p. 286, 2018.

[35] R. J. Nowling, J. Vyas, G. Weatherby, M. W. Fenwick, H. J. Ellis, and M. R. Gryk, “Connjur spectrum translator: An open source application for reformatting nmr spectral data,” *Journal of biomolecular NMR*, vol. 50, pp. 83–89, 2011.

[36] J. C. Hoch and A. S. Stern, “Nmr data processing,” *(No Title)*, 1996.

[37] F. Delaglio, S. Grzesiek, G. W. Vuister, G. Zhu, J. Pfeifer, and A. Bax, “Nmrpipe: A multidimensional spectral processing system based on unix pipes,” *Journal of biomolecular NMR*, vol. 6, pp. 277–293, 1995.

[38] M. R. Gryk, “Widget design as a guide to information modeling,” *iConference 2019 Proceedings*, 2019.

[39] L. E. Kay and L. Frydman, “A special” jmr perspectives” issue: Foresights in biomolecular solution-state nmr spectroscopy-from spin gymnastics to structure and dynamics,” *Journal of Magnetic Resonance*, vol. 241, pp. 1–2, 2014.

[40] J. Niu, “Provenance: Crossing boundaries,” *Archives and Manuscripts*, vol. 41, no. 2, pp. 105–115, 2013.

[41] A. Dappert, R. Squire Guenther, and S. Peyrard, “Digital preservation metadata for practitioners,” *Cham*, 2016.

[42] M. S. Mayernik, T. DiLauro, R. Duerr, E. Metsger, A. E. Thessen, and G. S. Choudhury, “Data conservancy provenance, context, and lineage services: Key components for data preservation and curation,” *Data Science Journal*, pp. 12–039, 2013.

[43] S. B. Davidson and J. Freire, “Provenance and scientific workflows: Challenges and opportunities,” in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, 2008, pp. 1345–1350.

[44] G. Feigenbaum, I. Reist, and I. J. Reist, *Provenance: An alternate history of art*. Getty Publications, 2012.

[45] J. Douglas, “Origins and beyond: The ongoing evolution of archival ideas about provenance,” *Currents of archival thinking*, vol. 2, pp. 25–52, 2017.

[46] T. Cook, “What is past is prologue: A history of archival ideas since 1898, and the future paradigm shift,” *Archivaria*, pp. 17–63, 1997.

[47] L. P. Nordland, “The concept of” secondary provenance”: Re-interpreting ac ko mok ki’s map as evolving text,” *Archivaria*, pp. 147–159, 2004.

[48] P. Conway, “Digital transformations and the archival nature of surrogates,” *Archival Science*, vol. 15, no. 1, pp. 51–69, 2015.

[49] M. G. Kirschenbaum, *Mechanisms: New media and the forensic imagination*. mit Press, 2012.

[50] J. Drucker, “Performative materiality and theoretical approaches to interface.,” *DHQ: Digital Humanities Quarterly*, vol. 7, no. 1, 2013.

[51] C. Robertson, *The Filing Cabinet: A vertical history of information*. U of Minnesota Press, 2021.

[52] J. Sterne, *MP3: The meaning of a format*. Duke University Press, 2012.

[53] V. Cuevas-Vincenttin, B. Ludascher, P. Missier, *et al.*, *Provone: A prov extension data model for scientific workflow provenance*, 2016. [Online]. Available: https://purl.dataone.org/provone-v1-dev.

[54] D. Garijo and Y. Gil, “A new approach for publishing workflows: Abstractions, standards, and linked data,” in *Proceedings of the 6th workshop on Workflows in support of large-scale science*, 2011, pp. 47–56.

[55] P. Groth and L. Moreau, *Prov-overview: An overview of the prov family of documents*, 2013. [Online]. Available: https://www.w3.org/TR/prov-overview/.

[56] A. K. Thomer, M. B. Twidale, and M. J. Yoder, “Transforming taxonomic interfaces: “arm’s length” cooperative work and the maintenance of a long-lived classification system,” *Proc. ACM Hum.-Comput. Interact.*, vol. 2, no. CSCW, Nov. 2018. DOI: 10.1145/3274442. [Online]. Available: https://doi.org/10.1145/3274442.

[57] L. Struik, M. Quat, P. Davenport, A. Okulitch, *et al.*, *A preliminary scheme for multihierarchical rock classification for use with thematic computer-based query systems*, 2002.

[58] W. F. Doolittle and E. Bapteste, “Pattern pluralism and the tree of life hypothesis,” *Proceedings of the National Academy of Sciences*, vol. 104, no. 7, pp. 2043–2049, 2007.

[59] R. Bettivia, Y.-Y. Cheng, and M. R. Gryk, “I got a letter from my past self:(un) managed change and provenance,” *iPRES 2023*, 2023.

[60] R. Elmasri and S. B. Navathe, *Fundamentals of database systems*. Addison-Wesley, 2011.

[61] M. P. Satija, “Colon classification (cc),” *KO KNOWLEDGE ORGANIZATION*, vol. 44, no. 4, pp. 291–307, 2017.

[62] M. Dewey, *Dewey decimal classification: Centennial 1876-1976*. Forest Press Division, Lake Placid Education Foundation, 1876.

[63] S. Heller, A. McNaught, S. Stein, D. Tchekhovskoi, and I. Pletnev, “Inchi-the worldwide chemical structure identifier standard,” *Journal of cheminformatics*, vol. 5, no. 1, pp. 1–9, 2013.

[64] K. La Barre, “Faceted navigation and browsing features in new opacs: Robust support for scholarly information seeking?” *Knowledge Organization*, vol. 34, no. 2, pp. 78–90, 2007.

[65] Gamma and Erich, *Design patterns*. Pearson Education India, 1995.

[66] J. Ioannidis, "Why most published research findings are false.," *PLoS Med*, vol. 2, e124, 2005. DOI: https://doi.org/10.1371/journal.pmed.0020124.