Enhancing Traceability in Clinical Research Data Through an Information Product Framework

Samuel Hume
Dakota State University

Follow this and additional works at: https://scholar.dsu.edu/theses

Recommended Citation

This Dissertation is brought to you for free and open access by Beadle Scholar. It has been accepted for inclusion in Masters Theses & Doctoral Dissertations by an authorized administrator of Beadle Scholar. For more information, please contact repository@dsu.edu.
ENHANCING TRACEABILITY IN CLINICAL RESEARCH DATA THROUGH AN INFORMATION PRODUCT FRAMEWORK

A dissertation submitted to Dakota State University in partial fulfillment of the requirements for the degree of

Doctor of Science

in

Information Systems

June 23, 2017

By

Samuel Hume

Dissertation Committee:
Dr. Dorine Bennett – Co-chair
Dr. Cherie Noteboom – Co-chair
Dr. Surendra Sarnikar
Dr. Zixing Shen
This dissertation is approved as a credible and independent investigation by a candidate for the Doctor of Science in Information Systems degree and is acceptable for meeting the dissertation requirements for this degree. Acceptance of this dissertation does not imply that the conclusions reached by the candidate are necessarily the conclusions of the major department or university.

Student Name: Samuel Hume
Dissertation Title: Enhancing Traceability in Clinical Research Data through an Information Product Framework

Dissertation Co-chair: [Signature] Date: 08-01-2017
Dissertation Co-chair: [Signature] Date: 8/2/17
Committee member: [Signature] Date: 8/2/2017
Committee member: [Signature] Date: 8/1/2017

Original to Office of Graduate Studies and Research
Acid-free copies with written reports to library
ACKNOWLEDGMENT

Thank you to Dr. Cherie Noteboom, Dr. Dorine Bennett, Dr. Surendra Sarnikar, and Dr. Zixing Shen, my dissertation committee, for guiding me through this process and providing your insights and encouragement. I would particularly like to thank my dissertation co-chairs Dr. Dorine Bennett and Dr. Cherie Noteboom for the extra time and help they put into my dissertation work.

A special thank you to Dr. Surendra Sarnikar for his guidance and mentoring during this entire journey, as my academic advisor, initial dissertation committee chair and now committee member, and as a teacher. I appreciate you making yourself available to help me on a regular basis.

Thanks to CDISC for their interest in advancing Trace-XML, and particularly to Lauren Becnel and Wayne Kubick for their input and support. Also, thanks to the CDISC community and XML Technologies team for their willingness to participate in the research and provide useful input.

I would like to thank my parents, Jim and Claudia Hume, for their encouragement and support throughout my extended educational endeavors. I would also like to thank my kids (Sarah, James, and Matt) for their patience while I was buried in my laptop and less entertaining than I should have been.

Finally, I would like to thank my wife, Kathy Hume, for her unwavering support during this long process and for occasionally proof reading articles that she undoubtedly found utterly uninteresting.
ABSTRACT

Data quality is foundational to clinical research and each process through which the data passes must be documented such that the published tables and listings can be traced back to the original, raw data (Nahm, 2012). In regulated clinical research, traceability is a requirement and indicates that the content of the clinical study report can be traced back to the original source data. Today’s clinical research solutions have significant traceability limitations. This design science research (DSR) project creates a traceability framework for clinical research that builds on the existing industry data standards and improves a reviewer’s understanding of the data for data quality and regulatory compliance purposes. The framework consists of 3 layers: (1) the Information Product Map (IP-Map) model: a high-level view of the manufacturing process for creating an information product (IP); (2) the CDISC standards metadata: metadata describing the IPs, data elements, and computations at a detailed level of granularity; and (3) a graph model: traceability throughout the clinical research data lifecycle that supports traceability visualization, validation, and queries. The primary benefits of the Trace-XML Framework are (1) to identify and resolve traceability gaps in clinical study metadata, (2) validate metadata traceability in a clinical study, and (3) query and visualize traceability metadata. The Trace-XML Framework maximizes the use of the existing standards models and technology to minimize barriers to implementation. This framework contributes a new extension to the industry standard metadata that addresses traceability gaps uncovered during the development of the Trace-XML software. The Trace-XML software contributes the algorithms needed to remedy and validate traceability in the Trace-XML graphs, in addition to adding support for traceability queries.
DECLARATION

I hereby certify that this dissertation constitutes my own product, that where the language of others is set forth, quotation marks so indicate, and that appropriate credit is given where I have used the language, ideas, expressions or writings of another.

I declare that the dissertation describes original work that has not previously been presented for the award of any other degree of any institution.

Signed,

[Signature]

Samuel Hume
# TABLE OF CONTENTS

| DISCUSSION APPROVAL FORM | ................................................................. II |
| ACKNOWLEDGMENT | ............................................................ III |
| ABSTRACT | ............................................................. IV |
| DECLARATION | ............................................................... V |
| TABLE OF CONTENTS | ............................................................. VI |
| LIST OF TABLES | ................................................................. IX |
| LIST OF FIGURES | ................................................................. X |
| INTRODUCTION | ................................................................. 1 |
  | GENERAL BACKGROUND ON CLINICAL RESEARCH | ................................................................. 1 |
  | BACKGROUND ON CLINICAL RESEARCH DATA | ................................................................. 2 |
  | THE TRACEABILITY PROBLEM: LIMITED COMPUTABLE TRACEABILITY IN CLINICAL RESEARCH | ........................................... 5 |
  | RESEARCH OBJECTIVES OF THE PROJECT | ................................................................. 9 |
| LITERATURE REVIEW | ................................................................. 10 |
  | HEALTH INFORMATION TECHNOLOGY STANDARDS GENERAL TO CDISC | .................................................. 10 |
  | XML IN HEALTHCARE DATA STANDARDS | ................................................................. 13 |
  | INTRODUCTION TO THE CDISC STANDARDS FOR CLINICAL RESEARCH | .................................................. 14 |
  | OVERVIEW OF THE CDISC FOUNDATIONAL STANDARDS | .................................................. 16 |
  | TRACEABILITY WITHIN CLINICAL RESEARCH DATA | .................................................. 20 |
  | TRACEABILITY WITHIN INFORMATION PRODUCT MAPS | .................................................. 23 |
  | TRACEABILITY AS DIMENSION OF PROVENANCE | .................................................. 27 |
  | PROVENANCE REQUIREMENTS IN HEALTHCARE SYSTEMS | .................................................. 41 |
  | COGNITIVE LOAD AND VISUALIZING TRACEABILITY INFORMATION | .................................................. 45 |
| RESEARCH METHODOLOGY | ................................................................. 47 |
  | TRACEABILITY FRAMEWORK HIGH-LEVEL REQUIREMENTS | .................................................. 49 |
  | ARTIFACT DESIGN TABLE | ................................................................. 50 |
  | TRACE-XML DEVELOPMENT | ................................................................. 52 |
  | TRACE-XML LAYERED FRAMEWORK IMPLEMENTATION | .................................................. 52 |
  | TRACE-XML ACCESSIBILITY AND LICENSE | ................................................................. 53 |
  | EVALUATION | ................................................................. 53 |
  | ANALYTICAL EVALUATION | ................................................................. 54 |
DISCUSSION AND CONCLUSION .................................................................................................................. 114

PRIMARY BENEFITS OF Trace-XML ........................................................................................................... 114
IMPLEMENTATION FEASIBILITY .................................................................................................................. 115
Trace-XML as a Validation Tool .................................................................................................................. 117
Trace-XML as a Tool to Enhance Data Quality .......................................................................................... 119
Trace-XML as a Visualization Tool ........................................................................................................... 120
Trace-XML Framework Utility .................................................................................................................... 121
Impact on Regulatory Submissions ............................................................................................................. 126
Future Work .................................................................................................................................................. 127
Research Contributions ............................................................................................................................. 130
Conclusion .................................................................................................................................................... 130

APPENDIX A: GLOSSARY OF TERMS ........................................................................................................ 132

APPENDIX B: Trace-XML User Documentation ......................................................................................... 135

Trace-XML .................................................................................................................................................... 135
Running a Java Program ............................................................................................................................. 135
Installing Trace-XML ................................................................................................................................... 135
Configuration File ......................................................................................................................................... 135
Running Trace-XML ..................................................................................................................................... 137
Trace-Query .................................................................................................................................................... 138
Running Trace-Query ................................................................................................................................... 139

APPENDIX C: Trace-XML Define-XML Extension ..................................................................................... 140

APPENDIX D: Trace-XML Source Code Documentation ............................................................................. 144

APPENDIX E: Trace-XML Qualitative Assessment Informed Consent ......................................................... 145

APPENDIX F: Trace-XML Qualitative Assessment IRB Approval Letter ...................................................... 147

References ...................................................................................................................................................... 149
LIST OF TABLES

Table 1. Example healthcare data standards .......................................................... 11
Table 2. Define-XML v2.0 *def:Origin* types taken from (CDISC, 2013a) ............. 22
Table 3. Define-XML v2.0 traceability rules taken from (CDISC, 2013a) ................. 23
Table 4. IP-Map block types ................................................................................. 25
Table 5. Clinical study metadata types or sources from (Zozus & Bonner, 2017) ...... 42
Table 6. Interoperable provenance recommendations from (Curcin et al., 2014) ...... 42
Table 7. Based on Hevner et al. (2004) design science research guidelines (P. 83).... 47
Table 8. Traceability framework high-level requirements ........................................ 49
Table 9. DSR artifact design table ......................................................................... 50
Table 10. Trace-XML Framework layers .................................................................... 58
Table 11. Main Trace-XML Framework software applications ................................. 65
Table 12. Example Trace-XML query results for the Pooled Site Group 1 variable... 72
Table 13. Summary of Trace-XML specific traceability rules .................................... 77
Table 14. Summary of Trace-Query display rules .................................................... 80
Table 15. Prototype implementation source code inventory ...................................... 81
Table 16. Clinical data expert subject counts by category ....................................... 101
Table 17. Number of Categories and Codes ............................................................. 105
Table 18. Coding Frequency Analysis ................................................................... 107
Table 19. Salient themes for each research question .............................................. 109
Table 20. Glossary of Terms / Acronyms ................................................................. 132
LIST OF FIGURES

Figure 1. Clinical research data lifecycle ................................................................. 3
Figure 2. Example clinical trial data flow (Iberson-Hurst, 2008) ................................. 5
Figure 3. Clinical research data lifecycle example .................................................... 6
Figure 4. Metadata definitions for datasets include Origin for traceability ................. 7
Figure 5. Variable traceability through a PDF-based annotated CRF ......................... 8
Figure 6. CDISC foundational standards ................................................................. 16
Figure 7. The ODM metadata and data hierarchies .................................................. 18
Figure 8. Provenance taxonomy (Simmhan et al., 2005a) ........................................ 28
Figure 9. CDISC ODM audit trail support (CDISC, 2008) ........................................ 30
Figure 10. Design science research cycles (Hevner, 2007) ....................................... 49
Figure 11. Trace-XML adds cross-lifecycle, study-level metadata traceability ........ 57
Figure 12. 3-layered Trace-XML Framework conceptual diagram ............................ 58
Figure 13. Example layer 1 partial IP-Map for ADSL ............................................. 60
Figure 14. Using Alias to link a Define-XML domain to an IP-Map block ............... 61
Figure 15. Full lifecycle Trace-XML graph using the tree layout .............................. 63
Figure 16. Trace-XML Unreachable Nodes report fragment ..................................... 63
Figure 17. Example Trace-Query output for the ADSL SEX variable ....................... 64
Figure 18. Trace-XML predecessor traceability graph generated using yEd ............. 66
Figure 19. Trace-XML Framework creates new dimensions of traceability ............ 67
Figure 20. Example Trace-XML extension shown with the trc namespace prefix ..... 68
Figure 21. Full lifecycle Trace-XML graph fragment in a hierarchical layout .......... 69
Figure 22. A predecessor graph showing the metadata details for a node ............... 70
Figure 23. Trace-XML edges added to Define-XML using SHARE content .......... 70
Figure 24. Links to trace queries added to the Define-XML Source column .......... 73
Figure 25. Report showing unexpectedly unreachable nodes in a Trace-XML graph .. 74
Figure 26. Full lifecycle Trace-XML graph in a directed tree layout ....................... 74
Figure 27. Using graph visualization to identify traceability gaps ......................... 76
Figure 28. Simplified IP-MAP for creating a submission-ready dataset ................. 86
Figure 29. DFS marks reachable nodes in the first 3 stages of IP-Map ..................... 90
Figure 30. Identifying the metadata missing from the attribute graph ....................... 91
Figure 31. Algorithm to create the missing metadata in the attribute graph ............. 92
Figure 32. Example of the Trace-XML Define-XML extension .............................. 94
Figure 33. Re-testing for reachability from DS$_1$ to P$_1$ using the updated graph ........ 95
Figure 34. Testing shows reachability from DS$_1$ and DS$_2$ to P$_1$ in the updated graph. 96
Figure 35. The Trace-XML thematic hierarchy ...................................................... 108
Figure 36. Code category frequency percentage totals .......................................... 111
CHAPTER 1

INTRODUCTION

General Background on Clinical Research

Clinical trials test the safety and efficacy of new medical treatments for human use. The medical field uses clinical research to determine if a new therapy provides the expected benefits to patients based on its ability to treat the disease and improve the patient’s overall quality of life. The Code of Federal Regulations (CFR) defines a clinical trial as a clinical investigation in which a drug is administered, dispensed to, or used by one or more human subjects (21 CFR 312.3). As a form of scientific research, clinical trials follow exacting scientific standards to ensure reliable and repeatable results.

Federal regulations apply to clinical trials involving the investigation of a new drug, device, biologic, or cases where an existing therapy is used for a new indication. Industry often sponsors such trials to gain the regulatory approval needed to market the therapy for the specified indication. In 2013 the biopharmaceutical industry sponsored 6,199 clinical trials of medicines in the United States (US) involving 1.1 million subjects (Battelle, 2015). The clinical trials used to generate medical evidence in support of new therapies have grown increasingly lengthy, expensive, and risky (English, Yeonwoo, & Griffin, 2010). The high cost of prescription drugs in the US has become a point of national concern. Prescription drugs now account for 10% of all healthcare spending in the US and remain the fastest growing segment of healthcare costs increasing at a rate of 12.2% in 2014. The average development time for a new medicine now exceeds ten years at a total cost of $2.6 billion, including the cost of capital and failed R&D projects (DiMasi, Grabowski, & Hansen, 2014). The costs of a single large, multi-center trial can range from $300 - $600 million, depending on factors such as the duration, number of subjects, and the number and location of the research sites (English et al., 2010).
Although total spending on biopharmaceutical research and development (R&D) has continued to increase annually, rising costs, complexity, risk, and process inefficiencies have limited corresponding increases in productivity (Lamberti & Getz, 2015). Success rates for commercializing medicines have been dropping with only 11.3% of those medicines that enter human trials making it through to Food and Drug Administration (FDA) approval (DiMasi et al., 2014). The length of the clinical trials phase of drug development has increased 15% over the last ten years to 6.8 years. The complexity of the clinical trials has also been increasing with more sophisticated study designs involving more study sites and countries. The lengthy timelines to complete the clinical trials needed to gain regulatory approval both increase cost and reduce revenue due to the limited patent life granted to new compounds. “Improving the efficiency and effectiveness of medical product development is a national priority,” according to the US FDA (FDA, 2013a).

The expanding complexity and scope of clinical research coupled with the pervasiveness of technology has yielded a corresponding increase in associated data volumes (Hume, Aerts, Sarnikar, & Huser, 2016). From 2002 to 2012, the number of endpoints in a typical phase III clinical trial increased by 86%, the number of procedures by 58%, and the number of data points collected per patient to 929,203. FDA submissions from the top 10 pharmaceutical companies over the period from 1997 to 2005 increased in sized from an average of 70,000 to 222,000 pages. From 2005 to 2012 FDA submission sizes increased 1,423% to 3.4 million pages (Getz, 2013). At this scale, operational inefficiencies add considerable time and expense to the clinical development cycle time. A lack of supporting clinical trials infrastructure in the US has been identified as a contributing cause of the inefficiencies (English et al., 2010). The Center for Information Technology Leadership estimated that the implementation of the national standards for interoperability for the exchange of health information would save the US approximately $77 billion annually. Innovative new tools and processes are needed to improve the efficiency of clinical research to reduce the cost of new medicines while maintaining high quality.

**Background on Clinical Research Data**

Clinical research data can be expressed as a lifecycle where data and information are created and transformed through the various phases of the clinical research process based on a variety of planning documents including the study protocol, statistical analysis plan, and data management plan. Regulated clinical trials include additional requirements mandated by federal regulations, or strongly recommended by FDA guidance. Figure 1 shows a representation of a regulated clinical research data lifecycle that generates a submission to a regulatory agency such as the FDA. The lifecycle displayed in Figure 1 represents a clinical data perspective at a level of granularity that suits the needs of this research project, but many valid alternative representations exist. Acronyms used in the diagram and throughout this document are defined in the Glossary in Appendix A.

![Clinical research data lifecycle](image)

Figure 1. Clinical research data lifecycle

The lifecycle shown in Figure 1 highlights the states of the clinical data throughout a trial including: EHR electronic source data, data collection, standardized data tabulations, analysis datasets, analysis results, and data archival. Data transformations throughout the lifecycle transition the data from state-to-state. Data transformation has remained largely a manual development effort despite the availability of more modern alternatives. The current processes for clinical research data transformation are error prone, inefficient, lack traceability, and are not portable. While partially interoperable data standards exist for protocol through data analysis, there are no standard mechanisms for capturing and executing the transformations that convert data from state-to-state in the clinical research data lifecycle.
A typical data transformation implementation in a biopharmaceutical organization involves manually crafting SAS macros to transform SAS datasets from one state to the next, possibly using metadata captured in spreadsheets. These bespoke data transformation applications make maintainability and traceability difficult, especially when the clinical data lifespan is often 50 years or longer. This process is particularly time consuming and costly due to the software validation requirements imposed by federal regulations.

The clinical trial data lifecycle maximizes the use of structured data. Alternatively, in routine healthcare the diversity of diseases and conditions has necessitated a higher level of narrative text. In clinical research, subjects are pre-screened for specific clinical conditions and treatment follows a pre-specified protocol allowing for more structured data during collection (Nadkarni, Marenco, & Brandt, 2012). Unstructured narrative text presents significant challenges during data analysis. Clinical trial data collection instruments seek to leverage structured data and controlled terminologies to improve the ability to accurately analyze the data. The use of data standards, including standard terminologies, further benefit clinical research by improving data quality, easing data integration, facilitating data exchange between organizations, facilitating regulatory reviews, and enabling the development of software tools that work with the data (Downey & Olson, 2013).

A typical clinical trial sponsored by industry involves a growing number of participating organizations and is increasingly conducted in network structures where seamless data exchange is essential to operational efficiency (Hume et al., 2016). Figure 2 shows a simplified data flow for a regulated clinical trial. It highlights the type of participants that generate and process data, as well as the data interchange scenarios necessary to support data collection during the trial. Over the last decade the total number and diversity of data sources used in all clinical research has increased ten-fold (Zozus et al., 2017). As clinical trials continue to grow in complexity, the network of participants contributing to the collection, processing, analysis, and integration of clinical data across systems and organizational boundaries has grown (Hume et al., 2016). The progression of biopharmaceutical in-licensing and partnerships have made common place new data sharing scenarios that add to the challenge of maintaining a sound chain of custody.
The data flows in Figure 2 highlight the need for efficient data exchange. Most systems that support data interchange during a clinical trial do so via a file export and import mechanism (CDISC, 2017; Jiang, Solbrig, Iberson-Hurst, Kush, & Chute, 2010; Wolfgang Kuchinke, Wiegelmann, Verplancke, & Ohmann, 2006), but some use a more modern web services-based approach (Deserno, Haak, Samsel, Gehlen, & Kabino, 2013; Haak, Gehlen, Jonas, & Deserno, 2014; Tröger, Wilke, Prokofyeva, & Zrenner, 2008). In the context of a regulated trial, the complete audit trail and data traceability must be maintained throughout the data exchange process.

The Traceability Problem: Limited Computable Traceability in Clinical Research

While bioinformatics and other computationally oriented e-science domains have embraced computable traceability, clinical research informatics, routine healthcare and other domains that rely on distributed, heterogeneous information systems have not yet achieved the same level of computable traceability (Curcin, 2016). Within the domain of regulated clinical research, submissions to the FDA and Japan’s Pharmaceuticals and Medical Devices Agency (PMDA) are required to include the metadata needed to establish traceability for the variables
included in the submission datasets. Traceability should provide the means to trace a variable in a dataset back to its source by linking to content in each of the previous steps in the clinical research data lifecycle. A representation of the clinical data lifecycle is shown in Figure 3. Traceability in this example would link content in the analysis results back as far as original data captured in an EHR to include any derivations and transformations needed to support the transition between each state in the lifecycle.

Figure 3. Clinical research data lifecycle example

Figure 4 shows an example of the current state-of-the-art for establishing traceability within regulated clinical research. The table in Figure 4 provides the metadata definitions for the demographics dataset and includes the Origin column to support traceability for each demographics variable.
Figure 4. Metadata definitions for datasets include Origin for traceability.

Origin values shown in Figure 4 represent content originating from the protocol, an assigned value, case report form (CRF) variables, and derivations. The CRF Page origin value provides a link to an image of an annotated CRF as shown in Figure 5. Referencing an annotated CRF in portable document format (PDF) provides traceability that only works one step backwards. In this case, it traces a variable in the demographics standardized tabulation dataset back to the demographics data collection CRF. There is no capability to show if the CRF content originated from an EHR. The annotated CRF is a visual representation of the data capture instrument in PDF, but it is not queryable as it is not available in a computable format.
Three key limitations hinder the effectiveness of today’s traceability solution: (1) gaps exist in the computable traceability provided by the standard metadata models, for example the trace only represents a non-computable description of the previous step in the data lifecycle; (2) gaps exist in the semantics and metadata needed to express computable traceability in the standard metadata models, for example no mechanism exists to explicitly reference the metadata for a CRF variable for traceability purposes; and (3) no traceability query capability exists limiting the ability to interrogate the traceability metadata, for example a reviewer cannot execute a query to view the entire lifecycle of a variable. These limitations are a significant hindrance to the in-depth and thorough analysis of the available evidence in the regulatory decision making process (van Valkenhoef, Tervonen, de Brock, & Hillege, 2012). The missing metadata restricts the possible automation alternatives and limits the extent to which automation can be applied (Zozus et al., 2017). Despite considerable existing research, determining the appropriate analytic and query mechanisms to answer traceability questions remains an open research opportunity (Carata et al., 2014; Davidson & Freire, 2008). Provenance in healthcare and science is currently receiving substantial attention as a research topic (Curcin et al., 2013).
Research objectives of the project

The primary research objective of this design science research (DSR) project is to create a clinical research traceability framework that builds on the existing industry data standards. The model created within this framework will enable the clinical study metadata to be represented as a graph displaying the full, interconnected history of each data element. This research project extends the Information Product Map (IP-MAP) research to use information products (IP) as the means to establish computable traceability within standardized clinical research metadata.

The specific research objectives addressed by this project include: (1) develop metadata models to support computable traceability and traceability visualizations that are compatible with industry data standards for the regulated clinical research domain, (2) adapt graph traversal algorithms to make them capable of identifying traceability gaps and validating traceability across the clinical research data lifecycle, and (3) develop a traceability query capability for retrieval and visualization of traceability information. This DSR project seeks to demonstrate the feasibility of the traceability framework by implementing and evaluating it. This project will also evaluate the traceability framework’s utility to clinical research data experts as a means for determining the integrity of the data based on the information manufacturing processes used to create it. The IP-Maps that exist within the framework will be extended to enable traceability at the data element level as a dimension of data quality.
CHAPTER 2

LITERATURE REVIEW

Health Information Technology Standards General to CDISC

“Standards are not only technical questions. They determine the technology that will implement the Information Society, and consequently the way in which industry, users, consumers and administrations will benefit from it” (Jakobs, 2003).

Vertical information systems (VIS) standards provide technical specifications that promote information exchange and other coordinated activities within an industry vertical (Markus, Steinfield, & Wigand, 2006). VIS standardization consists of (1) standards development and (2) standards diffusion. Both are collective action dynamics, and solving the first does not guarantee success with the second (Markus et al., 2006). Open standards such as the world wide web standards have a significant potential impact on information systems (IS) theory and practice (Markus et al., 2006). Technology standards are an essential driver of industrial productivity, but their value can only be maximized with broad standards diffusion (Zhu, Kraemer, Gurbaxani, & Xu, 2006). In an increasingly global economy with diverse value chains, inter-organizational systems provide both operational and strategic benefits to the organizations and industries that adopt them (Boh, Xu, & Soh, 2008). From a practitioner perspective numerous industries have a standards consortium (Nelson & Shaw, 2003), and standardization has significant practical implications for information technology (IT) management. From an economic perspective, standardization provides a means for organizations to increase the overall size of their market (Nickerson & Muehlen, 2006).

According to Erickson, Wolcott, Corrigan, and Aspden (2003) data standards are “formally accepted or endorsed definitions and rules regarding the format, meaning, and transmission of data elements.” Creating standardized data elements involves the establishment of standards that: (1) define what to collect, (2) determine how to represent the collected data, and (3) delineate how to encode the data for transmission to another party
(Erickson et al., 2003). **Table 1** lists a sub-set of the standards development organizations (SDO) in healthcare and the data standards they develop.

<table>
<thead>
<tr>
<th>SDO</th>
<th>Example Standards</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>HL7 (Health Level 7)</strong></td>
<td>• HL7 v2.x</td>
<td>HL7 develops healthcare information standards for exchange, integration and sharing of electronic health data. Although HL7 is based in the United States and accredited by ANSI (American National Standards Institute) the HL7 organization and the standards it develops are international in scope.</td>
</tr>
<tr>
<td></td>
<td>• FHIR (Fast Healthcare Interoperability Resources)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• CDA (Clinical Document Architecture)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• CCD (Continuity of Care Document)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• SPL (Structured Product Labeling)</td>
<td></td>
</tr>
<tr>
<td><strong>CDISC (Clinical Data Interchange Standards Consortium)</strong></td>
<td>• ODM (Operational Data Model)</td>
<td>CDISC develops healthcare information standards for clinical research, including regulated clinical research.</td>
</tr>
<tr>
<td></td>
<td>• SDTM (Study Data Tabulation Model)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• CDASH (Clinical Data Acquisition Standards Harmonization)</td>
<td></td>
</tr>
<tr>
<td><strong>NEMA (National Electronics Manufacturers Association)</strong></td>
<td>• DICOM (Digital Imaging and Communications in Medicine Committee)</td>
<td>DICOM is a standard for transmitting, storing, and viewing medical images.</td>
</tr>
<tr>
<td><strong>WHO (World Health Organization)</strong></td>
<td>• ICD-9 (International Classification of Diseases-9)</td>
<td>ICD-9 and ICD-10 provide diagnosis and disease codes widely used for billing and claims.</td>
</tr>
<tr>
<td></td>
<td>• ICD-10</td>
<td></td>
</tr>
<tr>
<td><strong>Regenstrief Institute for Health Care</strong></td>
<td>• LOINC (Logical Observation Identifiers Names and Codes)</td>
<td>Concept-based terminology for medical laboratory orders and results.</td>
</tr>
<tr>
<td><strong>IHTSDO (International Health Terminology Standards Development Organization)</strong></td>
<td>• SNOMED CT (Sytematized Nomenclature of Medicine – Clinical Terms)</td>
<td>A standardized, multilingual vocabulary of clinical terminology widely used for exchanging and storing clinical healthcare information.</td>
</tr>
<tr>
<td><strong>MSSO (MedDRA Maintenance and Support)</strong></td>
<td>• MedDRA (Medical Dictionary for Regulatory Activities)</td>
<td>MedDRA is a medical terminology used to classify adverse events that is</td>
</tr>
</tbody>
</table>
Common Data Elements, or CDEs, are a common way of standardizing content for data collection in clinical trials. A number of healthcare metadata repositories (MDR) have been developed to maintain and publish CDEs, many with ties to national governments, as can be seen on the US National Institutes of Health (NIH) CDE Resource Portal (NLM, 2013). CDEs are metadata that can describe any type of measurement or concept. CDEs represent the “smallest meaningful data container in a given context” (Sinaci & Erturkmen, 2013). They provide an informal meaning and usage for the data element, as well as synonyms, alternative definitions, valid units of measurement, and code lists (Papatheodorou et al., 2009). CDEs, unlike items that derive their meaning from their position in a formal taxonomy or ontology, do not rely on context to derive their meaning. Such classification systems can be used to organize CDEs, but CDE semantics exist independent of the classification hierarchy (Papatheodorou et al., 2009).

CDEs can be combined by context into groups, such as for use in CRFs or datasets, and they can be grouped according to Data Element Concepts. A Data Element Concept links associated CDEs that derive their meaning from a common concept, a fundamental tenant of ISO/IEC 11179 (ISO/IEC, 2013; Ngouongo, Löbe, & Stausberg, 2013). A number of CDE models have been developed. The Health Information Technology Standards Panel (HITSP) has specified a set of standard CDEs called C154: Data Dictionary Component to harmonize their use across the various HITSP standards. For instance, HITSP C32 annotates the data elements in the HL7 CCD with their HITSP C154 CDE counterparts to harmonize the meaning of the CCD elements (Sinaci & Erturkmen, 2013). The Federal Health Information Model (FHIM) provides another set of CDEs in the form of an information model of healthcare data intended to promote interoperable Electronic Health Records (EHRs). The Transitions of Care Initiative (ToC) provides the Standards and Interoperability (S&I) Clinical Element Data Dictionary (CEDD) for CDEs aimed at improving the exchange of core clinical

<table>
<thead>
<tr>
<th>SDO</th>
<th>Example Standards</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Services) and JMO (Japanese Maintenance Organization)</td>
<td>commonly used in regulated clinical trials.</td>
<td></td>
</tr>
</tbody>
</table>
information among healthcare entities in support of meaningful use and improved quality of care (ONC, 2015).

Although some have classified the CDISC standards as CDEs for clinical research, the CDISC approach differs from the typical CDE philosophy. The CDISC standards do not follow the CDE approach since they specify the permissible context for each Data Element, and do not support the use of CDEs that function as independent units outside of their prescribed context (CDISC, 2015). CDISC data elements may be required in a specific domain or CRF, and they may support other data elements that function together to provide the full meaning of a research concept. CDEs focus on the characteristics of an individual data element often at the expense of relationships with other data elements. Many individual data elements cannot represent a complete clinical observation on their own, and require the context of other data elements to make the CDE meaningful. In other cases, CDEs may include content in the CRF question, such as units, that would be better captured by a separate data element.

**XML in Healthcare Data Standards**

XML (extensible markup language) has been widely used to create standards or models for healthcare data representation and data exchange (Domínguez et al., 2007; Forster & Vossen, 2012; Hume et al., 2016; Shabo, Rabinovici-Cohen, & Vortman, 2006; Vergara-Niedermayr, Wang, Pan, Kurc, & Saltz, 2012). Both CDISC and HL7 have produced XML-based standards that are mature and broadly implemented. The XML-based CDISC ODM has become the standard exchange format for case report form (CRF) data and metadata (Forster & Vossen, 2012). ODM has been successfully integrated with EHR-based HL7 CDA, another XML-based healthcare standard, to support the use of routine healthcare data in clinical research (Bernhard, Axel, Carsten, Fleur, & Martin, 2011; Daniel et al., 2013; El Fadly et al., 2007; R. Kush et al., 2007).

HL7 CDA defines the semantics and structure of clinical documents in XML format for the exchange of patient data ("Health Level Seven International," 2012). CDA semantics are based on the HL7 Reference Information Model (RIM), and referenced healthcare terminologies (Hodge, 2008). CDA documents consist of an unstructured textual section and a
structured section where standard coding systems, such as LOINC and SNOMED, establish machine-readable semantics (Crichton et al., 2009). The HL7/ASTM CCD specification is a constraint of the HL7 CDA containing US specific requirements. HITSP has specified HITSP C32 describing the HL7/ASTM CCD for interchange purposes (Sinaci & Erturkmen, 2013). CCD promotes interoperability of clinical data between healthcare organizations without the loss of meaning (Sinaci & Erturkmen, 2013). FHIR is the latest HL7 standard to use XML. FHIR’s use of RESTful web services and emphasis on conformance and reference implementations has made it a popular with implementers. H. Leroux, Metke-Jimenez, and Lawley (2015) made a comparison and mapping between HL7 FHIR and CDISC ODM with the goal of achieving semantic interoperability between clinical research and healthcare. Their approach to integrating ODM with FHIR involved a mapping of hierarchical ODM ClinicalData elements to a set of FHIR resources (H. Leroux et al., 2015).

XML’s acceptance as a technology for implementing healthcare standards has been aided by the broad number of tools and technologies that support it. UML (Unified Modeling Language), another well-accepted formalism for healthcare standards and models, can be converted into an XML schema, and can be represented in a standard XML format called XML Metadata Interchange (XMI) for model interchange (Crichton et al., 2009). Once standards or models are available as XML they can be transformed using XSLT or other XML processing technologies (Sendall & Kozaczynski, 2003). Other XML technologies that have been used to build ODM XML-based solutions include XML databases, XQuery, XForms, and schematron (Bruland et al., 2014).

Introduction to the CDISC Standards for Clinical Research

CDISC is an SDO and industry consortium creating standards for clinical research, including regulated clinical research. CDISC provides standards that intend to cover the complete clinical research data lifecycle from protocol through archival. The CDISC standards (CDISC, 2016) play a key role in clinical research informatics, including areas such as data exchange, archival, regulatory submissions, and interoperability with healthcare data. Use of the CDISC data standards has increased over the last several years with increasing interest from regulatory authorities such as the US FDA (FDA, 2010, 2013b) and the Japanese
PMDA, as well as significant resources being allocated to healthcare data interoperability (FDA, 2013a). In 2012, the fifth reauthorization of the FDA’s Prescription Drug User Fee Act (PDUFA V) became law and stated that the “FDA shall develop standardized clinical data terminology through open standards development organizations with the goal of completing clinical data terminology and detailed implementation guides by FY 2017” (FDA, 2014; Vadakin & Kush, 2014). PDUFA V further states that the “FDA will develop guidance for industry on the use of the CDISC data standards for the electronic submission of study data” (FDA, 2014).

Both the FDA and PMDA now require that new studies must use the CDISC standards if they will be part of a regulatory submission and have a strong preference for the use of CDISC standards in studies submitted today. The European Medicines Agency (EMA) has stated a preference for the CDISC standards and recommends their use as part of their effort to improve transparency and data sharing within clinical research (EMA, 2013). The Chinese FDA has also endorsed the CDISC standards.

Aided by the new regulatory mandates at the FDA and the PMDA, the CDISC standards required for submissions have seen very high adoption rates by industry. Standards diffusion has been difficult to measure in other industries, but metrics do exist within the regulated clinical research industry. The CDISC standards have been downloaded by users from over 90 countries, and adoption continues to increase, with the CDISC SDTM adoption as of 2014 estimated to be over 80% (Friggle, Li, Labout, & Kush, 2011; Vadakin & Kush, 2014). Getz (2007) showed that the primary benefit of standards usage was the exchange of data with other organizations in the industry with 59% of the respondents (and 78% of the CROs) identifying this as a key benefit. The primary reasons delaying the adoption of the standards within an organization were (1) the required levels of internal coordination, (2) the level of investment required to implement, (3) management resistance, (4) lack of ROI, and (5) change management or inertia (Getz, 2007). The 2014 CDISC business case claims that using CDISC standards from the beginning of the clinical research data lifecycle can save approximately $180 million per submission (CDISC, 2016). With over 420 member organizations, the CDISC standards are among the most widely used within regulated clinical research.
Different CDISC standards exist to address each phase of the clinical research data lifecycle, including: (1) Protocol Representation Model (PRM) for the protocol phase, (2) Clinical Data Acquisition and Harmonization (CDASH) for the data collection phase, (3) Study Data Tabulation Model (SDTM) for standardized data tabulations, and (4) Analysis Data Model (ADaM) for data analysis. Although the standards are all part of the CDISC portfolio and are intended to work together, each standard has its own pattern of development and diffusion dilemmas. Each standard has its own team of volunteers from industry participating in standards development. For example, the CDASH team consists primarily of those with data management backgrounds, while the ADaM team consists primarily of statistical programmers. Each standard has been implemented to varying degrees at different organizations within the industry. The regulatory agencies also impact each standard differently through their participation in its development or by requiring the standard as part of a regulatory submission. For example, SDTM has been more widely adopted than CDASH largely because SDTM is required by the regulators for submissions and CDASH is not.

**Overview of the CDISC Foundational Standards**

![Figure 6. CDISC foundational standards](image)

Figure 6 highlights the CDISC foundational content standards and identifies the corresponding ODM-based data exchange standards. The ODM XML-based standards provide the data exchange formats that represent the metadata and data for CDISC content standards such as CDASH and SDTM. The Protocol Products primarily include the PRM, a
UML-based model representing the main entities in a clinical research protocol, and the Trial Design Model (TDM) that has been instantiated as part of SDTM (de Montjoie, 2009). The CDASH standard describes the basic data collection fields on a CRF. Development of the CDASH standard was triggered by the (FDA)’s 2004 report, Innovation/Stagnation: Challenge and Opportunity on the Critical Path to New Medical Products and initially covered the common and safety domains (Richesson & Nadkarni, 2011). The Laboratory Data Model (LAB) is an XML representation model for laboratory data acquisition and interchange (CDISC, 2016; van Valkenhoef et al., 2012). The SDTM standard describes a standard structure for study data tabulations. SDTM provides standardized datasets that enable the development of software tools and provide the content for subject profiles and data listings (CDISC, 2016; Kubick, Ruberg, & Helton, 2007). The Standard for the Exchange of Non-clinical Data (SEND) standard is based on the SDTM model, but represents data from pre-clinical toxicology studies (CDISC, 2016; van Valkenhoef et al., 2012). The ADaM standard describes metadata models and examples for analysis datasets (CDISC, 2016; Kubick et al., 2007).

The CDISC Controlled Terminology (CT) is not referenced as part of the foundational standards, but plays a critical role in the effectiveness of the standards. The CDISC CT defines the value lists and associated terms, definitions, synonyms, submission values and concept codes used by the CDISC content standards. The CDISC CT is developed in partnership with the National Cancer Institute’s (NCI) Enterprise Vocabulary Services (EVS).

The CDISC Operational Data Model (ODM) standard is an XML schema-based document and exchange standard created specifically to support the needs of clinical research. As shown in Figure 6, the ODM standard (CDISC, 2016) plays an essential role in the transmission or exchange of clinical research data and metadata, including FDA submissions and interoperability with healthcare data. In general, ODM is considered the most relevant format for the exchange of metadata within the clinical research domain (Ngouongo et al., 2013; Stausberg et al., 2009). ODM is comprised of four major components: (1) Study containing the metadata that represents the structural definitions for the study, (2) ReferenceData providing information used to interpret the clinical data, (3) ClinicalData capturing the clinical data for each subject, and (4) AdminData storing user, location, and
electronic signature information (CDISC, 2013c; De Melo, Nagler-Ihlein, & Weber, 2006). **Figure 7** shows ODM’s hierarchical structure. The hierarchical structure is best suited to representing CRF-oriented data (Lefort & Leroux, 2013; Hugo Leroux & Lefort, 2012). ODM metadata can be used to generate data collection forms, decision support systems, and other applications (Crichton et al., 2009; De Melo et al., 2006; Domínguez et al., 2007; Forster & Vossen, 2012).

**Figure 7. The ODM metadata and data hierarchies**

In addition to supporting a core set of use cases centered around support for CRF data and metadata, ODM also provides an extension mechanism that enables both CDISC and vendors to augment the model as needed to support specific interchange scenarios. This makes it possible to extend the core ODM standard with the additional information needed to support other use cases as was done in Forster and Vossen (2012) and De Melo et al. (2006). CDISC has published a number of standardized extensions including: (1) Define-XML for dataset metadata (CDISC, 2013b), (2) Dataset-XML for dataset data (CDISC, 2014b), (3) SDM-XML for Study Design Model (CDISC, 2011a), (4) CT-XML for Controlled Terminology (CDISC, 2011b), and (5) Analysis Results Metadata (CDISC, 2014a) for Define-XML v2 (Hume et al., 2016).

The Define-XML v2.0 standard ODM extension provides the metadata to describe tabular datasets and plays a key role in establishing traceability in regulatory submission datasets (FDA, 2016a; Hume et al., 2016). When used to represent CDISC content standards Define-XML may be used to describe all the SDTM, ADaM, or SEND datasets for a study (Kubick et al., 2007; Hugo LEROUX, McBride, Lefort, Kemp, & Gibson, 2012; Lightfoot &
Define-XML was added to the FDA’s Study Data Specifications in March of 2005 (FDA, 2010; Kubick et al., 2007), and in December 2016 it becomes a requirement for all new studies included in FDA submissions (CDISC, 2013b; FDA, 2010). The FDA’s March 2017 Study Data Technical Conformance Guide (TCG) states that the Define-XML file "describes the metadata of the submitted electronic datasets, and is considered arguably the most important part of the electronic dataset submission for regulatory review" (FDA, 2017). The TCG further states that "an insufficiently documented data definition file is a common deficiency that reviewers have noted" (FDA, 2017). Validation of the Define-XML documents beyond mere schema validation has become a critical step in the regulatory submission process demanding the development of validation rules and validation engines to apply them (Aerts, 2014; Hume et al., 2016; "OpenCDISC," 2014). Currently, the FDA requires the use of SAS V5 XPORT format for submissions data creating the need to maintain the metadata in a separate Define-XML document (Kubick et al., 2007).

The CDISC standards metadata is maintained within the SHARE MDR (Jiang et al., 2012; Jiang et al., 2010; Sinaci & Erturkmen, 2013). The SHARE ISO/IEC 11179-based metamodel provides the foundation for a CDISC standards model and the means of specifying the semantics as well as the representation for the standards metadata (Davies, Harris, Crichton, Shukla, & Gibbons, 2008). As CDISC expands the scope of the standards to cover an increasing percentage of the clinical research data lifecycle (Jiang et al., 2010), the SHARE model has been used to explicitly express the relationships between each state in the lifecycle. These relationships are used to represent the data flow through the lifecycle (Souza, Kush, & Evans, 2007). Standards implementers desire more complete metadata to describe the transition, or mapping, from state-to-state within the data lifecycle. Providing metadata describing each state in the clinical research data lifecycle, as well as the transition from state-to-state is referred to as the end-to-end CDISC model. From a modeling perspective, there exists one end-to-end process that spans a number of activities and their associated data states (Curbera, Doganata, Martens, Mukhi, & Slominski, 2008). Mapping is the process of associating healthcare terms used in one system with the equivalent terms in another system (Hebda & Czar, 2013), and mapping metadata is often used to link one specific terminology, classification, or nomenclature scheme to another (Imel, Giannangelo, & Levy, 2004). In this
case, it maps one state in the clinical research data lifecycle to the next, such as CDASH to SDTM or SDTM to ADaM. When the standards are used together in this way the data transformations required to support the data lifecycle end-to-end are minimized. Today ODM and the other CDISC standards provide limited support for representing data mapping information (Richesson & Nadkarni, 2011). Lack of a published end-to-end CDISC standards model has limited the ability to represent complete traceability throughout the data lifecycle within a study.

**Traceability within Clinical Research Data**

Traceability is a fundamental element of data integrity. It represents both the knowledge of the data’s origin and the basis for belief in the data. Traceability indicates the preservation of an unbroken chain of data from its source to the point of consumption. Within the context of regulated clinical research, traceability indicates that the content of a clinical study report (CSR) is explicitly linked back to the source data to include any transformations, derivations, and imputations that have occurred in the process. R. D. Kush and Bleicher (2003) state that data integrity “means that the content of the submission database matches that found in source data.” Study data traceability provides an understanding of the relationships between the data in each state within the clinical research data lifecycle (FDA, 2016a). Traceability plays a critical role in supporting the clinical research analysis results since the strength of the study results depend on the source data and the processes used to create them (Curcin et al., 2014). Traceability, an essential component of provenance, helps other researchers to audit the clinical research process, verify the results, and ultimately can help reproduce the research results (Curcin et al., 2014). Traceability also enhances the secondary use of clinical research data. The trustworthiness of analysis conducted on aggregated data improves when the sources and methods of aggregation are clearly understood.

From a regulatory perspective the FDA has stated that the results presented in the analysis results must be traceable back to the original data elements (FDA, 2016a). FDA reviewers rely on traceability to: (1) determine the original observations as captured and the derivations used to transform them into other variables; (2) understand how the statistical
tests, such as confidence intervals or p-values, were determined; and (3) provide an overall understanding of the construction of the analysis datasets (FDA, 2016a). Traceability helps the reviewer to understand “the relationships between the analysis results, analysis datasets, tabulation datasets, and source data” (FDA, 2016a). The FDA has identified a lack of traceability as one of the top 7 data standards issues (Chhatre & Malla, 2012), and it has been cited as a key to the FDA’s ability to successfully review submission data (Peterson & Izard, 2010). “Messy data” that is difficult to understand can delay the FDA’s ability to complete the review of a New Drug Application (NDA) (Berkowitz, 2011) potentially costing the sponsor company millions of dollars in lost revenue. The PMDA has expressed traceability requirements similar to the FDA, and have identified traceability as a key element of an NDA submission (Takanami, 2014). The ICH GCP (International Conference on Harmonization of Technical Requirements for Registration of Pharmaceuticals for Human Use Good Clinical Practice) cites traceability as one its 13 core principles (ICH, 1996b).

Despite the importance of traceability requirements for regulated clinical research, the ability to easily trace data back to its source remains limited. The technology available to support the systematic review of submission datasets are limited in functionality when it comes to traceability (van Valkenhoef et al., 2012). The current situation precludes efficient and fully transparent decision making (van Valkenhoef et al., 2012). Today no tools exist capable of tracing a data element from the protocol through to the analysis results tables, listings, and figures (Dootson, 2011). Current federal regulations, such as Title 21 of the Code of Federal Regulations; Electronic Records; Electronic Signatures (FDA, 2003; Segalstad, 2008), better known as 21 CFR Part 11, describe traceability needs, but do not prescribe how traceability should be assessed. Due to these limitations regulatory decisions are not sufficiently based on the available evidence pointing to a lack of traceability and transparency (van Valkenhoef et al., 2012).

In a 2015 Accenture survey 80% of the respondents claimed “consistent data across the clinical data lifecycle” as a key reason for implementing the CDISC standards within their organizations (Accenture, 2015). The CDISC ODM v1.3.2 and Define-XML v2.0 standards provide the metadata models that represent the traceability across the data lifecycle, and Define-XML is currently required as part of a standards-compliant regulatory submission
(CDISC, 2016). The Define-XML `def:Origin` element provides metadata explicitly in support of traceability, and definitions for the `def:Origin Type` attributes are listed in Table 2.

Table 2. Define-XML v2.0 `def:Origin` types taken from (CDISC, 2013a)

<table>
<thead>
<tr>
<th>Type</th>
<th>Type Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>CRF</td>
<td><em>Data that was collected as part of a CRF and has an annotated CRF associated with the variable.</em></td>
</tr>
<tr>
<td>Derived</td>
<td><em>Data that is not directly collected on the CRF or received via eDT, but is calculated by an algorithm or reproducible rule defined by the sponsor, which is dependent upon other data values.</em></td>
</tr>
<tr>
<td>Assigned</td>
<td><em>Data that is determined by individual judgment (by an evaluator other than the subject or investigator), rather than collected as part of the CRF, eDT or derived based on an algorithm. This may include third party attributions by an adjudicator. Coded terms that are supplied as part of a coding process (as in --DECOD) are considered to have an Origin of “Assigned”. Values that are set independently of any subject-related data values in order to complete SDTM fields such as DOMAIN and --TESTCD are considered to have an Origin of “Assigned”.</em></td>
</tr>
<tr>
<td>Protocol</td>
<td><em>Data that is defined as part of the Trial Design preparation. An example would be VSPOS (Vital Signs Position), which may be specified only in the protocol and not appear on a CRF or transferred via eDT.</em></td>
</tr>
<tr>
<td>eDT</td>
<td><em>Data that is received via an electronic Data Transfer (eDT) and usually does not have associated annotations. An origin of eDT refers to data collected via data streams such as laboratory, ECG, or IVRS.</em></td>
</tr>
<tr>
<td>Predecessor</td>
<td><em>Data that is copied from a variable in another dataset. For example, predecessor is used to link ADaM data back to SDTM variables to establish traceability.</em></td>
</tr>
</tbody>
</table>

Table 3 describes the metadata traceability rules listed in the Define-XML v2.0 specification (CDISC, 2013b).
Table 3. Define-XML v2.0 traceability rules taken from (CDISC, 2013a)

<table>
<thead>
<tr>
<th>#</th>
<th>Rule Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>For regulatory submissions, <code>def:Origin</code> metadata must be provided for all SDTM or ADaM variables. It is at the sponsor’s discretion whether to provide <code>def:Origin</code> at the Variable or Value level.</td>
</tr>
<tr>
<td>2</td>
<td>If the <code>ItemDef</code> corresponding to a SDTM or ADaM variable includes a <code>def:ValueListRef</code> and all of the <code>ItemDef</code> elements referenced in the corresponding <code>def:ValueListDef</code> include a <code>def:Origin</code> element, the <code>def:Origin</code> is optional with the variable level <code>ItemDef</code>.</td>
</tr>
<tr>
<td>3</td>
<td>If the <code>ItemDef</code> corresponding to an SDTM or ADaM variable includes a <code>def:ValueListRef</code> and the <code>def:Origin</code> elements of <code>ItemDef</code> elements referenced in the corresponding <code>def:ValueListDef</code> are different, then the <code>def:Origin</code> cannot be provided with the variable level <code>ItemDef</code>.</td>
</tr>
<tr>
<td>4</td>
<td>If the variable or value is derived, the corresponding <code>ItemDef</code> must include a <code>MethodOID</code> attribute that references the corresponding <code>MethodDef</code>.</td>
</tr>
<tr>
<td>5</td>
<td>When <code>def:Origin/@Type</code>=&quot;CRF&quot;, there must be a <code>def:DocumentRef</code> child element and <code>def:DocumentRef/@leafID</code> must match the ID attribute of the <code>def:leaf</code> element corresponding to the <code>def:AnnotatedCRF</code> within the same <code>MetaDataVersion</code>. Otherwise, <code>def:DocumentRef/@leafID</code> must match the ID of a defined <code>def:leaf</code> element within the same <code>MetaDataVersion</code>.</td>
</tr>
<tr>
<td>6</td>
<td>If the variable is derived, a <code>MethodDef</code> must be provided. If the variable is assigned, a <code>MethodDef</code> is optional.</td>
</tr>
</tbody>
</table>

Where traceability is concerned domain-specific models and semantics are preferred (Curcin et al., 2014; Johnson, Kamineni, Fuller, Olmstead, & Wernli, 2014). The CDISC standards provide the models for traceability within the context of regulatory submissions. FDA regulations on electronic records and signatures, 21 CFR Part 11, came into effect in 1997 and required a computer generated, time-stamped audit trail (FDA, 2003). The 21 CRF Part 11 regulations were established prior to many of the recent advances in provenance and traceability.

**Traceability within Information Product Maps**

An Information Product Map (IP-Map) is an analysis model drawn from a top-down perspective that represents the data flows and processing that occur during the creation of an information product (IP) to support a decision maker’s understand of that IP (Shankaranarayanan & Wang, 2007). IP-Maps provide a conceptual visualization of an IP’s
manufacturing process that aids decision makers in identifying how data is being captured, transformed, stored, and utilized prior to becoming available to the decision maker (Chee, Yeoh, & Gao, 2011; Shankaranarayanan, Wang, & Ziad, 2000). IP-Maps provide a foundation for analyzing data quality (Shankaranarayanan & Wang, 2007). This research focuses on the use of IP-Maps to support traceability, a dimension of data quality, within the domain of regulated clinical research. In the context of a clinical trial, and particularly a regulatory submission, IPs include the tables, listings, and figures included in the analysis results that must be understood by a regulatory reviewer and possibly the analysis datasets themselves due to their essential role in traceability.

An IP-Map uses a set of modeling constructs to represent the process of creating an IP in a manner similar to the process for manufacturing a physical product (Shankaranarayanan et al., 2000). IPs share many of the same product quality-control processes with traditional manufacturing processes (Shankaranarayanan, Ziad, & Wang, 2003). IP-Maps allow information consumers to visualize, comprehend, and evaluate how the IP is assembled (Shankaranarayanan & Wang, 2007). MDRs have been referenced by IP-Maps as a means of providing access to the detailed data element-level metadata as well as information on data element mappings and transformations (Shankaranarayanan, 2005; Shankaranarayanan & Cai, 2006).

IP-Maps are designed top-down to describe an IP. They specify the raw data (RD) inputs that come from outside the IP-Map boundaries and the component data (CD) items that are generated as the IP is created. The RD and CD data flow into and out of one of eight different types of constructs called blocks. Each stage within the IP-Map is connected to show the flow of data through the process. Each stage consists of blocks that contain composition attributes, or a reference to those attributes in an MDR, that provide the metadata descriptions of the individual data elements. These attributes, or data elements, are necessary to provide the detail needed to understand the IP manufacturing process, as well as to measure data quality within the process. The top-down development of an IP-Map shows how these data elements flow through the process to include any transformations or derivations that alter the data as it moves through the process. In this sense, IP-Maps provide high-level support for the notion of traceability within the process for creating an IP. IP-Maps define traceability as the
ability to trace a sequence of one or more stages that precede a given block within the IP-Map (Shankaranarayanan et al., 2000). Thus, IP-Maps provide a high-level abstraction of the detailed data element-level traceability. Providing traceability at multiple levels of granularity can be used to improve a decision maker’s understanding of the process that creates the IP. Table 4 (Shankaranarayan et al., 2003; Shankaranarayanan et al., 2000) describes the block types used to construct an IP-Map.

### Table 4. IP-Map block types

<table>
<thead>
<tr>
<th>Block Name (Abbreviation)</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data Source (DS)</td>
<td>This block represents the source of the raw input data items as they enter into the IP-Map.</td>
</tr>
<tr>
<td>Processing (P)</td>
<td>This block represents any processing performed on data items including: transformations, calculations, imputations, derivations, concatenations, or other manipulations needed to produce the information presented in the IP.</td>
</tr>
<tr>
<td>Inspection (I)</td>
<td>This block represents pre-determined checks performed to test for data quality, appropriate authority, missing values, and adherence to other validity rules. Among other things this block ensures that the data items are free of defects.</td>
</tr>
<tr>
<td>Data Correction (DC)</td>
<td>This block is a specialized form of the process block that represents an exception process triggered by quality errors that require corrective action to resolve.</td>
</tr>
<tr>
<td>Decision (D)</td>
<td>This block represents a branching function that transfers data for a specific type of processing based on the values of specified data items.</td>
</tr>
<tr>
<td>Data Storage Block (STO)</td>
<td>This block represents data items that are stored in files or databases to make them available for further processing.</td>
</tr>
<tr>
<td>Information System Boundary (SB)</td>
<td>This block represents data items moving from one system to another, particularly data transfers from one type of system to another.</td>
</tr>
<tr>
<td>Business/Organization Boundary (BB)</td>
<td>This block represents data items moving from one business or organizational unit to another to highlight changes or quality issues that could arise from this transition.</td>
</tr>
<tr>
<td>Data Sink (Consumer) (CB)</td>
<td>This block represents the data items in a final IP that will be used by the IP consumer.</td>
</tr>
</tbody>
</table>
Blocks in an IP-Map are identified by unique, non-null names and are further described by a set of attributes (Shankaranarayan et al., 2003). A stage in an IP-Map represents a step in the IP manufacturing process. Each stage in the IP-Map is represented by one or more blocks. A start stage and an end stage in an IP-Map are represented by two blocks directly connected by a data flow (Shankaranarayan et al., 2003). IP-Maps can be used to determine reachability and traceability as elements of data quality within the IP manufacturing process (Shankaranarayanan & Wang, 2007). Shankaranarayan et al. (2003) defines IP-Map reachability as the “ability to identify all production stages of an IP that can be reached from a given stage” and traceability as the “ability to identify a sequence of one or more stages that precede any stage.”

Chee, Yeoh, Gao, and Richards (2014) implemented a layered metadata framework using IP-Map to support traceability in a Business Intelligence (BI) environment where aggregated data often comes from different sources and undergoes various transformations. The variety of data sources and transformations made it difficult for decision makers to understand the processes that created the data. Decision makers were unable to visualize the entire information manufacturing process of a given BI IP (Chee et al., 2011). Many of the data traceability needs in clinical research are similar to traceability needs in BI applications. The data warehousing view data lineage problem within a BI environment is another vein of research that shares many common objectives with the traceability and provenance research as covered in (Buneman & Tan, 2007; Cui, Widom, & Wiener, 2000; Missier, Paton, & Belhajjame, 2010; Tan, 2007; Woodruff & Stonebraker, 1997). The BI IP-Map solution was implemented using a 3-layered architecture: (1) business processes, (2) information processes, and (3) a detailed description of the metadata. While this BI framework supports traceability, the needs of clinical research data element-level traceability are more detailed and granular than those targeted by the BI IP-Map research.

Granularity and layering are two key concepts influencing the metadata model needed to support traceability and provenance within an application. From a granularity perspective, the semantics of individual data transformations are needed to ensure that a traceability query can determine the specific inputs that impact a derived data element. Such granularity decisions impact the types of questions that can be answered with the traceability metadata
Layering enables the model to represent traceability at multiple levels of abstraction, each layer supporting different views. The layered metadata frameworks described in (Chee et al., 2011; Chee et al., 2014) and (Shankaranarayanan, 2005) provide examples of layering based on IP-Maps.

**Traceability as Dimension of Provenance**

"Data provenance plays a critical role in advancing research data sharing nationally and internationally. Without provenance, it is difficult if not impossible for one researcher, separated by distance and decades of time from another, to trust the data of the latter" (Cheah & Plale, 2015).

**Definition of Provenance.** Provenance describes how a data item in a specific state was derived from a specified source (Moreau, Groth, et al., 2008). Broadly defined, provenance describes the origins of a data item and refers to the documented history of the process, including derivations, that the data item took to arrive in a dataset (Buneman, Khanna, & Tan, 2000). According to the W3C “provenance is information about the 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” (W3C, 2013). Traceability represents a key component of provenance that is essential to data integrity. Traceability within a provenance context may be referred to as (1) data provenance, (2) dataflow provenance, or (3) data lineage.

**Provenance Taxonomy.** The provenance literature has been categorized based on an extended Taxonomy of Provenance (Simmhan, Plale, & Gannon, 2005a) shown in Figure 8. The original Simmhan et al. (2005a) taxonomy was developed for a survey of provenance literature for e-science and is relevant for clinical research. Each of the following sections highlight the provenance research as it relates to both the regulated clinical research requirements and the current state-of-the-art.
**Data Quality.** Currently, an international consensus on how to assess data quality and its usability for research does not exist. Quality is defined by the International Standards Organization (ISO) as the "degree to which a set of inherent characteristics of an object fulfills requirements" (ISO, 2015). According to De Lusignan et al. (2011) six key concepts describe data quality for clinical research: (1) overall data quality which asks if "these data are fit for purpose?"; (2) data provenance, defined as how data came to be, that includes data lineage; (3) data extraction errors; (4) data processing errors, which includes errors created during data mapping and transformations; (5) traceability, the capability to identify the origins of any data variable within the final analysis datasets, is required for sound governance and nearly impossible without a formal system for managing metadata; and (6) curation, which ensures that metadata and data are available for future researchers to use for new research or to review previously published findings.

Dataflow provenance information about data sources and transformations can be used to estimate the overall quality and reliability of clinical research data (Jagadish & Olken, 2004). Elements of data quality such as integrity, trustworthiness, completeness, and identity are essential to dataflow provenance. Assessments of data quality build on data
trustworthiness, and provenance can aid in the identification of data flows that use metadata sources and transformations of unknown or incorrect origin (Bertino, Dai, & Kantarcioglu, 2009; Jagadish & Olken, 2004). A perceived lack of transparent processing in the generation of clinical decision support recommendations has limited their effectiveness. Trust is an essential element required for the success of data-driven medicine, and data flow provenance is a fundamental aspect of trustworthiness (Curcin, Fairweather, Danger, & Corrigan, 2017).

Completeness is characteristic of data quality described in (Mettler, Rohner, & Baacke, 2008). Completeness as a dimension of provenance quality is defined as "the extent to which provenance is missing or to which provenance is more than the actual amount of collectible provenance" (Cheah & Plale, 2015). Completeness in dataflow provenance assessments examine metadata gaps including orphan variables that are not traceable back to inputs, inputs that are not linked to outputs, and cases where a transformation or derivation function should be part of the trace.

Maintaining identifier integrity throughout the end-to-end data lifecycle is also essential to dataflow provenance (R. Y. Wang, Storey, & Firth, 1995). Within clinical research, each metadata element must be uniquely identified within the context of a study at a minimum. In aggregated data environments where the outputs of one process become the inputs to the next, as found with data lakes, a persistent and unique identifier policy may be a requirement (Suriarachchi & Plale, 2016). Provenance data is frequently represented in graphs, and graph nodes and edges often require unique identifiers within the context of the graph (Curcin et al., 2017).

Audit Trail. The audit trail is an essential element of data provenance (Miles, Groth, Branco, & Moreau, 2007), and 21 CFR Part 11 (FDA, 2003) establishes it as a regulatory requirement in clinical research. Data provenance within the CDISC standards often focuses on ODM’s support for a 21 CFR Part 11 compliant audit trail with electronic signature support. The ODM audit trail exists at different levels of granularity within the schema hierarchy from Study Events to individual data items. The audit trail covers the W7 model (Ram & Liu, 2009) where provenance is captured using the 7 W’s including: (1) what (an event or change of state that impacted the data), (2) where, (3) when, (4) who, (5) which
(instruments or software that were used in the event), (6) how (an action leading to the events), and (7) why. ODM’s coverage of the 7 W’s is shown in Figure 9 below.

Figure 9. CDISC ODM audit trail support (CDISC, 2008)

Replication. The Institute of Medicine defined a clinical study to be reproducible when a second study reaches the same conclusions as the first (Davis, Nolan, Woodcock, & Estabrook, 1999). There are a number of levels of replication including: (1) replication of a study by a different group following the same basic study design and methods; (2) replication of a study by the same group; or (3) replication of the analysis of the same data using different data management, programming, and statistical analysis (Zozus, Richesson, Walden, Tenenbaum, & Hammond, 2016). Replication of data derivations is essential to the validation and trustworthiness of the analysis results. Establishing data flow provenance is critical to determining the validity of scientific results by assuring they can be reproduced (Curbera et al., 2008). Dataflow provenance must include details on the derivations such that a reviewer can understand, and ideally could repeat, the process used to create the dataset under
examination (Simmhan, Plale, & Gannon, 2005b). This approach to dataflow provenance originated in the e-sciences where it was used to represent scientific workflows, and it was originally used by the scientific programmers that develop and maintain research workflows (Curcin et al., 2017). In regulatory submissions to the FDA or PMDA, the detail needed to support dataset replication is maintained in the Define-XML metadata and the documents it references.

As science has become progressively more data-driven and computationally oriented, as well as more interdisciplinary, the need for data sharing and transparency to support research reproducibility has grown correspondingly (Ludäscher, 2016). Data provenance is essential for transparency and computational reproducibility. Curcin (2016) identifies 4 levels of reproducibility that apply within the context of a learning health system: (1) auditability provides the additional details required to analyze the state of the data at each step in the research process, (2) traceability provides an unbroken chain of data from its source to the point of consumption to include any transformations and derivations that impact the data, (3) replicability provides the ability to repeat the processes using the original tooling and data, and (4) reproducibility provides the ability to independently confirm the results of the research using new data. Traceability is the most critical level for establishing trust and believability in the research processes as it provides the details establishing precisely what occurred at each step in the process (Curcin, 2016). Provenance data plays an important role in the reproducibility of the scientific processes necessary for replicating the results of published research (Missier, 2016).

Zozus et al. (2016) state that reproducibility requirements for studies using EHR data have not been sufficiently developed, but also identified traceability as an essential component of research replication. To establish traceability, research information systems must maintain when and from where EHR data was received, the state of the data received, any transformations performed to map the data into the research data model, and any algorithms used to derive research values.

Blockchain technology has been identified as a possible means to enhance the replication and reproducibility of clinical research in the future. Tracing complex data flows
from numerous diverse research participants while documenting each step in the protocol in real-time through a timestamping workflow is viewed as an important step towards establishing improved traceability as well as data consistency and inviolability (Benchoufi, Porcher, & Ravaud, 2017). Future EHR systems may provide researchers easy access to a comprehensive, immutable log of a patient's medical information via blockchain technologies (Ekblaw, Azaria, Halamka, & Lippman, 2016). Blockchain technology provides an unfalsifiable timestamp that provides the proof of existence of any piece of data through the application of public key cryptography (Benchoufi et al., 2017). A blockchain healthcare record would use this public key cryptography feature to create an append-only, immutable, verifiable, timestamped chain of content (Ekblaw et al., 2016). Equally important for empowering patients in research, blockchain technology can empower peer-to-peer exchanges where patients and researchers work together to establish consent and access to data (Benchoufi et al., 2017).

**Attribution.** As the scientific community continues to embrace data sharing, open science, and the realization that the data itself can represent a significant research contribution (Borgman, 2012); data citation becomes an extension of data provenance (Jagadish & Olken, 2004). Data citation enhances the ability for researchers to collaborate and promotes the secondary usage of existing data by making it possible to contact the people that generated a piece of data or derivation algorithm (Allen, Brand, Scott, Altman, & Hlava, 2014). Citing portions of datasets down to the individual data element level or for a specific derivation requires additional metadata as well as specific mechanisms for referencing a slice of a dataset (Hoyle et al., 2015). Technologies such as URI/URLs provide a universal locator for an electronic document, but citing content within the document can be more challenging, especially when traditional paper mechanisms like page numbers are friable in electronic documents (Buneman et al., 2000).

**Informational.** Provenance data can provide metadata descriptions of datasets that can aid in discovery activities, as well as retrieval from a data archive (Simmhan et al., 2005b). In the CDISC standards ODM supports the long-term archival of clinical research data where the integrity of the data and metadata as captured from the original systems is maintained in a system-neutral, open format. ODM represents the information necessary to achieve the
technical aspects of regulatory compliance by maintaining the full audit trail and electronic signatures for a given study. ODM utilizes an easy to understand “CRF metaphor” for organizing the data and metadata (W. Kuchinke, Aerts, Semler, & Ohmann, 2009). ODM provides a non-proprietary means to archive data that meets the federal regulations, and does not require the archival of proprietary software to support the use of the data (W. Kuchinke et al., 2009; Souza et al., 2007). This is a critical characteristic of an effective study data archival given that the lifespan of regulated clinical research can extend past 50 years. Many current electronic data capture (EDC) and study data management systems directly export to ODM facilitating its use as part of an archival solution (CDISC, 2017). ODM can be linked to EHR data and SDTM to create an end-to-end study data archival. Define-XML and Dataset-XML could be used as part of a dataset archive, and when combined with ODM provide the beginnings of an archive that covers the data lifecycle. For datasets, archived Define-XML references to external dictionaries or scoring algorithms must understand how those references will be maintained over the lifespan of the archived data (Buneman et al., 2000). If the maintainers of the external sources referenced cannot ensure their availability over the lifespan of the data archive, the external sources may need to be included in the study archive. Establishing traceability within an archive further enhances data integrity and makes it easier for new users to understand the data.

**Big Data Analytics.** Big data analytics will influence traceability within healthcare and clinical research. Y. Wang, Kung, and Byrd (2016) conducted a content analysis of 26 big data implementation cases in healthcare and identified traceability as one of the five big data analytics capabilities for healthcare. The other four capabilities were: (1) analytical capability for patterns of care, (2) unstructured data analytical capability, (3) decision support capability, and (4) predictive capability. For purposes of this study, Y. Wang et al. (2016) defined traceability as the ability to track output data from all information systems throughout the organization. The primary purpose for big data traceability is making data consistent, visible and easily accessible for analysis. Ideally systems would be designed from the outset to include the metadata necessary to support traceability, simplify maintenance, and simply the analytics work (McClatchey, Branson, & Shamdasani, 2016). Big data systems using a description-driven philosophy manage metadata describing each data item separately from the
data allowing re-use over time and traceability between data item versions as well as between the various states in the information lifecycle (McClatchey et al., 2016).

Traceability and data provenance may increase in importance in the context of big data where the same data maybe analyzed repeatedly to serve a variety of analytical purposes. As the datasets are repurposed to support new studies the separation between the original producer of the dataset and the data consumer increases, straining the original semantics. The lineage for any dataset in a big data context must be preserved to ensure the trustworthiness and believability of subsequent analytical uses of the datasets (Toivonen, 2015). Traceability and provenance within big data systems facilitates the reproducibility and verification of analysis results, as well as the validation of analyses between researchers (McClatchey et al., 2016).

Data lakes employ a schema-on-read approach where data transformations are presumed to occur within the lake leading to data items that exist in a range of stages in the data lifecycle. The availability of data in a variety of lifecycle stages increases the value of traceability and data provenance within the context of the data lake. In this case, data provenance "is the information about the activities, entities, and people involved in producing a data product" that exists in the data lake (Suriarachchi & Plale, 2016). As science becomes increasingly data-driven, as well as more collaborative and interdisciplinary, the importance of traceability to establish trust in the data and the analysis results will continue to remain a key capability (Ludäscher, 2016).

Data and Process Provenance. Data provenance, sometimes referred to as data lineage (Kifor et al., 2006) or dataflow provenance, is essential to the degree of trust placed in a data item, and plays a key role in data integration (Buneman & Tan, 2007). Data provenance requires information describing the creation, attribution, recording, processing, version history, and ownership of data items in support of its integrity (Buneman, Chapman, & Cheney, 2006). In data provenance lineage metadata is gathered about the data (Simmhan et al., 2005a). The lineage metadata is essential to preserving the scientific value of data. Data, or dataflow, provenance can describe the source for each data item as well as including the when, where, how, who, which, and why of data changes throughout the data lifecycle (Ram
& Liu, 2009). Data provenance can also be considered retrospective provenance because it maintains a log of the changes that were made for inspection after they happen (Clifford, Foster, Voeckler, Wilde, & Zhao, 2008). Technologies supporting data provenance seek to automatically capture the trace of a specific research task and its resulting data in order to facilitate reproducible research (Curcin, 2016).

Process provenance, often referred to as workflow or course-grained provenance, provides a complete description of the process used to generate a dataset to include the software programs used, external data capture instruments, and even a description of human-performed manual tasks (Buneman et al., 2006). Process provenance can also be considered prospective provenance because it captures a specification of a change or derivation before it is applied to create data changes (Clifford et al., 2008).

Moreau and Groth (2013) describe provenance as consisting of three views: (1) the data flow view, (2) the process flow view, and (3) the responsibility view. Moreau and Groth (2013) recommend starting with the data flow view. The data flow view of provenance is “concerned with the data flow within computer systems” and includes the derivations that transform the data items (Moreau & Groth, 2013). The data flow view of provenance is also referred to as traceability. In a clinical research context, the process flow view would be based on the protocol and study design, and the responsibility view would be based on the audit trail and electronic signature information.

*Regulations.* Data provenance plays an important role in addressing the regulations that guide regulated clinical research and their associated data submissions. Maintaining audit trails and traceability are established requirements stipulated in 21 CFR Part 11 (FDA, 2003), regulatory guidance on traceability (FDA, 2016a), and Good Clinical Practice (ICH, 1996a) in the US and in EudraLex Vol. 4 Annex 11: Computerized Systems in the European Union (EC, 2010). Data provenance can function as an enabling technology supporting conformance with the technical controls of 21 CFR Part 11 (Curcin, 2016). The FDA has also published guidance documents related to traceability in clinical research, including the 2003 guidance on Electronic Records; Electronic Signatures (ERES) Scope and application (FDA, 2003), and the 2007 revised Guidance for Industry Computerized Systems Used in Clinical Investigations.
(FDA, 2007b). Title 42 CFR 93, while not literally specifying traceability, indirectly points to
the need for traceability by requiring the capability to address allegations of research
misconduct for any institution funded by the US Public Health Service (Zozus et al., 2016).
Regulations with traceability requirements for non-clinical studies include 21 CFR 58.130 (e)
and 21 CFR 58.35 (b).

Granularity. The level of granularity needed to support provenance varies among
domains (Simmhan et al., 2005b). Provenance data can be collected at many different levels.
For example, the ODM audit trail exists at different levels of granularity within the schema
hierarchy from Study Events to individual data items (CDISC, 2013c). The costs associated
with capturing and maintaining provenance data can be expressed as inversely proportional to
the level of granularity (Simmhan et al., 2005a). Data provenance is considered fine-grained
provenance and describes in detail how each data item changes throughout the data lifecycle
(Ram & Liu, 2009). Data provenance is a fundamental component of data quality (Buneman
& Davidson, 2010).

Annotation and Inversion Techniques. Provenance data can either be maintained
explicitly, as it is in ODM, or deduced indirectly from the available data changes (Cui et al.,
2000). In Simmhan et al. (2005b) these approaches are categorized as annotations for the
former and inversion for the latter. With annotations derivations are recorded prospectively
(Clifford et al., 2008), or eagerly (Tan, 2004), as textual descriptions of the source data,
derivations, and processes impacting the data. These prospective provenance annotations are
pre-established and available as metadata used referenced in the provenance record. Inversion
techniques take the available data changes and invert the derivations to find the data needed to
produce the outputs (Cui et al., 2000). For example, certain queries and user-defined functions
can be inverted to identify the source data (Simmhan et al., 2005a). The inversion technique is
more efficient from a storage perspective, but the more complete and accurate annotation
approach is required for provenance in regulated clinical research.

For explicitly maintained data provenance there are three common techniques for
managing provenance data: naïve, transactional, and hierarchical (Buneman et al., 2006). The
ODM audit trail supports all 3 varieties, and the application generating and managing the
provenance data controls the implementation of a provenance technique. Naïve provenance represents the most basic technique as it stores one provenance record for each changed data item (Buneman et al., 2006). Transactional provenance is more efficient from a space and processing perspective than naïve provenance because a logical grouping of change events are captured by one audit record (Buneman et al., 2006). Hierarchical provenance can be used in combination with transactional provenance and uses a hierarchical structure to inherit provenance data from parent nodes to reduce redundancy in the audit trail information maintained (Buneman et al., 2006).

**Syntactic and Semantic Information.** In order for provenance-aware applications to be interoperable, it is critical that the process documentation they produce be structured according to a shared data model (Moreau, Groth, et al., 2008). General standards for provenance have been published by the W3C called the PROV Data Model (PROV-DM) (W3C, 2013), but many disciplines use standards specialized for their own domain. Each discipline has diverse provenance requirements (Simmhan et al., 2005a). For example, in regulated clinical research regulations such as 21 CFR Part 11 (FDA, 2003), regulatory guidance on traceability (FDA, 2016a), and data standards required for regulatory submissions (FDA, 2015a) all influence the provenance solution. Established standards for data provenance and traceability exist within the regulated clinical research domain.

A number of current provenance solutions make use of XML, including CDISC ODM, W3C PROV-XML, and OPM (Clifford et al., 2008; Miles et al., 2007). XML’s hierarchical structure naturally lends itself to provenance requirements such as audit trails and annotations. Some solutions use XML to link to ontologies to provide the needed semantics (Zhao, Goble, Stevens, & Bechhofer, 2004). Semantic web technologies, such as OWL and RDF, often play an important role in establishing the ontologies available for provenance within specific domains. The OWL/RDF model flexibility, support from provenance standards, availability of open source tools, and scalability have contributed to its broad usage in general-purpose provenance solutions (Malaverri, Mota, & Medeiros, 2013). Information models, such as Domain Analysis Models (Fridsma, Evans, Hastak, & Mead, 2008), can also support semantics within provenance. The difficulty developing and maintaining large ontologies or information models can be a barrier to establishing provenance semantics within a given
domain. In cases such as healthcare where multiple ontologies exist to serve different requirements, establishing one standard ontology to serve an entire domain may not be feasible.

**Data Provenance Standards.** Provenance models, such as the W3C PROV Data Model (PROV-DM) (W3C, 2013) and the Open Provenance Model (OPM) (Moreau, Freire, et al., 2008), provide a useful, general purpose means for tracing derivations from a specific source to an item in a particular state. According to the W3C “the goal of PROV is to enable the wide publication and interchange of provenance on the Web and other information systems” (W3C, 2013). Standards benefit data provenance since interoperability between provenance-capable applications requires a common structure or shared data model (Moreau, Groth, et al., 2008). Provenance models must support references to domain semantics, and existing domain models should be used where available (Curcin et al., 2014; Johnson et al., 2014). Within the domain of regulated clinical research, the CDISC standards provide the domain models and metadata for data element level traceability, and these benefit users since the semantics are known within the domain.

**Storage Scalability.** The number of datasets and the provenance granularity required have a significant impact on the scalability of the provenance solution, and provenance requirements vary by domain (Simmhan et al., 2005b). The number of derivations needed to create a dataset also impacts scalability. As the number of derivations increases the depth of the provenance lineage increases, along with the storage and processing requirements for the provenance solution (Simmhan et al., 2005a). The storage technique employed impacts the provenance scalability, and provenance storage solutions implement one of three previously cited approaches: naïve, transactional, and hierarchical (Buneman et al., 2006). Certain reduced provenance representations, such as Social Network Analysis, have demonstrated their ability to reduce provenance data volume while maintaining their ability to support the analysis of the provenance data (Gungoren & Aktas, 2016).

**Storage Overhead.** Process, or workflow, provenance may require manual tasks to capture and store key provenance annotations. The burden of capturing these annotations may negatively impact the completeness, accuracy, and machine-readability of the provenance
records (Zhao et al., 2004). For example, the CDISC Define-XML metadata standard is typically generated post hoc to satisfy the needs of a submission, and it often includes a number of manual tasks to record the required metadata (Hume et al., 2016). Some of the metadata referenced in Define-XML is maintained in PDF, adding to both the need to manually create metadata as well as additional storage overhead. A more progressive approach to producing Define-XML involves creating the Define-XML metadata as a specification that drives the generation of datasets and service specification documents (Lightfoot & Jansen, 2013; Maddox, 2013, 2014; Wheeldon & Burges, 2014).

Data provenance of clinical research data aids the reviewer in forming an interpretation of the integrity and authenticity of the data. Computer generated, time-stamped audit trails play a key role in establishing both integrity and authenticity by capturing details on every event that impacts the data. Care must be taken during the development of data capture software to ensure that the deployed services are capable of capturing events that impact the data as they occur such that any adverse impact to application performance is mitigated (Moreau, Groth, et al., 2008). The size of the data increases substantially with the addition of a complete audit trail, and adequate storage space must be allocated by data capture and archival solutions to maintain this information.

Exchange formats such as XML and RDF can also add to the size of the data when uncompressed (FDA, 2015b). Retrieving audit trail data for analysis requires that the design consider both the storage and retrieval tools to ensure performance is adequate for the task. Retrieval tasks are made more complex by requirements to query the audit trail for a specific data item that is related to a specified event that impacted the data while filtering out other related provenance information (Moreau, Groth, et al., 2008).

Establishing and maintaining data provenance in order to confirm the integrity and authenticity of the data requires that the provenance technology has been verified to perform the services needed to capture, store, and retrieve this information correctly. Provenance technology refers to both the hardware and software components needed to establish and maintain a complete provenance record as an end-to-end resource (Cheney, Chong, Foster, Seltzer, & Vansummeren, 2009). In regulated clinical research 21 CFR Part 11 requires the
validation of software systems that could impact the “reliability, integrity, availability, and authenticity of required records and signatures” (FDA, 2003). Computerized systems used in regulated clinical research must maintain a record of all changes, and must be re-validated to the extent demanded by the change. A software configuration management strategy is typically needed to control, audit, and report on the changes to the technology impacting electronic records in regulated clinical research environment (Burney, Saleem, Mahmood, & Jilani, 2010).

The quality of the provenance data itself can be assessed to ensure that the data quality claims based on provenance data are well-founded. Cheah and Plale (2015) uses the following dimensions to assess provenance quality: (1) correctness indicates the extent to which provenance data is correct and free of error including the evaluation of quality attributes such as provenance data accuracy, unambiguity, consistency, and homogeneity; (2) completeness indicates the amount of missing or extra provenance data; and (3) relevancy indicates the degree to which the provenance data is relevant to the needs of the reviewer.

Graph Visualization. Provenance data can typically be represented as a Directed Acyclic Graph (DAG), also referred to as causality graphs since the data flow DAG for a specific data item represents how it was created and all subsequent changes (Moreau, Groth, et al., 2008; Zhao et al., 2004). The DAG nodes represent data items and the edges represent the dependency relationships, such as versions, mapping, or derivations. Graph traversal algorithms can be used to find directed paths through a graph and test nodes for reachability. Reachability means that a directed path can be found from a source node to a specified target node within the DAG (Sedgewick & Wayne, 2011). The DAG can be used to present a visual representation of provenance data, or as a source to query against. Such visual representations can manifest patterns of real-world phenomena (Huynh, Ebden, Ramchurn, Roberts, & Moreau, 2014), such as unusually active editing patterns for a specific data point, which might initiate further examination into the “who”, “when”, and “why” provenance information. Since provenance graphs can become large, mechanisms are needed to cluster and filter the graphs in support of specific use cases (Macko, Margo, & Seltzer, 2013).
Queries and Service APIs. Datasets and databases can be searched for provenance data using standard search and query mechanisms. A significant body of literature exists in the area of provenance in database systems. Provenance data should be in a computable form and should be provided in a data store that permits querying using commonly available software.

Provenance Requirements in Healthcare Systems

"In the era of Big Data, deep learning systems such as IBM Watson, and other technologies that often rely on black-box analytical environments, it is of paramount importance to support transparency in computerized systems which actions have direct consequences on human lives" (Curcin et al., 2017).

The Standards Development Support Team for the Data Provenance Initiative under the Standards and Interoperability Framework for the Office of the National Coordinator (ONC) for Health Information Technology (IT) and the Health IT Standards Committee (HITSC) Data Provenance Task Force have been asked to provide recommendations for a standardized healthcare approach to data provenance as applied to EHRs (Gallagher, 2015). To establish a standard for capturing and exchanging EHR provenance information the Data Provenance Initiative community will: (1) define a provenance vocabulary and metadata, (2) create a technical specification for implementing data provenance, and (3) create guidance documents that help implementers understand how to implement provenance and on what level it should be applied (Gallagher, 2015). As the trend away from document formats towards atomized data continues, the need for data provenance to ensure the authenticity and integrity of the data has increased. The data provenance initiative is in the process of evaluating a number of candidate standards from the existing healthcare data standards including CDISC ODM v1.3.2, HL7 CDA Release 2, HL7 FHIR Release 1.1 Provenance Resource, W3C PROV-XML, and others.

Zozus and Bonner (2017) provide a list of the types of metadata needed to support traceability across a broad range of clinical study types, including prospective studies, retrospective studies, longitudinal studies based on EHR data, observational studies, and
studies supported by institutional data stores. Table 5 lists the sources of metadata described in (Zozus & Bonner, 2017).

<table>
<thead>
<tr>
<th>Metadata Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Audit trail</td>
<td>What, when, where, who, and why for each data change. The attribution and date/time for each data change, along with the previous value prior to the change.</td>
</tr>
<tr>
<td>Data element definition</td>
<td>ISO/IEC (2013) defines a data element as a &quot;unit of data that is considered in context to be indivisible.&quot; An MDR based on the ISO/IEC 11179 standard manages the lifecycle of a data element definition. Each data element may have a value domain associated with controlled terminology and formal semantics to establish its meaning.</td>
</tr>
<tr>
<td>Data origin</td>
<td>Identifies the source of the data from the previous step in the lifecycle or from the phenomena that was observed, questioned, or measured to produce a data value. Contextual information further specifying the origin may also be included.</td>
</tr>
<tr>
<td>Data transfer</td>
<td>The specification for data sent to or retrieved from an external information system which could include transfer log information.</td>
</tr>
<tr>
<td>Data transformation</td>
<td>The specifications for software that transforms or performs a specified operation on the data. All data transformation data changes should be captured in the audit trail.</td>
</tr>
<tr>
<td>Data quality assessment</td>
<td>The specifications for conformance rules and other constraints or logic that were applied to the data values to identify discrepancies. Any changes to the data resulting from the quality assessments should be captured in the audit trail.</td>
</tr>
<tr>
<td>Information retrieval</td>
<td>The specifications or source code for the queries executed to retrieve data values and any additional information needed to recreate the data extraction.</td>
</tr>
</tbody>
</table>

Curcin et al. (2014) provides a list of interoperable provenance implementation recommendations developed as part of the TRANSFoRm project, a European learning healthcare system initiative. This list of 21 recommendations has been reproduced in Table 6.

<table>
<thead>
<tr>
<th>#</th>
<th>Type</th>
<th>Recommendation</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Syntax</td>
<td>Use W3C PROV or the Open Provenance Model for modelling provenance data recorded in an electronic healthcare system. When choosing between the two,</td>
</tr>
<tr>
<td>#</td>
<td>Type</td>
<td>Recommendation</td>
</tr>
<tr>
<td>----</td>
<td>-------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>1</td>
<td></td>
<td>consider available tools and libraries in both systems for potential reuse or adaptation.</td>
</tr>
<tr>
<td>2</td>
<td>Vocabulary</td>
<td>Institutions involved in a distributed healthcare application should agree to a common vocabulary to use in provenance constructs where possible.</td>
</tr>
<tr>
<td>3</td>
<td>Domain models</td>
<td>Link provenance models and data with domain knowledge models and data, respectively. These may be expressed in the form of templates, similar in form to workflows or business processes, and should be shared with the community.</td>
</tr>
<tr>
<td>4</td>
<td>Detail focus</td>
<td>Always aim to model the detail of what happens, including each processing step and data item involved, rather than summary information that directly answers current provenance questions.</td>
</tr>
<tr>
<td>5</td>
<td>Granularity</td>
<td>Use the existing biomedical models and ontologies as indicators of the level of granularity of process description that users are likely to be interested in the future, and validate it against the provenance questions known at design time.</td>
</tr>
<tr>
<td>6</td>
<td>Connectivity</td>
<td>Model the provenance with the expectation that, if brought together, it would form a single graph describing the full, interconnected history of the system, as opposed to being delimited into a set of isolated records. Globally unique IDs should be used where feasible to facilitate interlinking.</td>
</tr>
<tr>
<td>7</td>
<td>Human actions</td>
<td>Include both salient human activities and automated processes in the model of provenance to be captured.</td>
</tr>
<tr>
<td>8</td>
<td>Model elements</td>
<td>In designing the provenance model, consider explicitly representing each element of the process flows and the data flows, as well as the attributions of actions to users. Layering ontologies is a useful approach that maintains logical separation between these distinct elements.</td>
</tr>
<tr>
<td>9</td>
<td>Reuse of existing data</td>
<td>Where feasible, integrate the data captured by existing mechanisms (version control, audit, logging, etc.) into the provenance record, whether by reference or translation, to provide as rich and integrated an account as possible.</td>
</tr>
<tr>
<td>10</td>
<td>Library reuse</td>
<td>If not using shared execution middleware with provenance capture support, make use of existing OPM/PROV libraries to capture provenance data within your application code.</td>
</tr>
<tr>
<td>11</td>
<td>Timely capture</td>
<td>Build the runtime provenance capture functionality into each step of the system processes, at the appropriate level of granularity, using global identifiers where possible and linking to records of preceding steps using the chosen provenance model relations.</td>
</tr>
<tr>
<td>12</td>
<td>Performance testing</td>
<td>Test the performance overhead of provenance capture within a small sample part of your application to ensure it is acceptable for your application or if the technologies used and level of detail of model need to be modified.</td>
</tr>
<tr>
<td>#</td>
<td>Type</td>
<td>Recommendation</td>
</tr>
<tr>
<td>----</td>
<td>-----------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>13</td>
<td>Database structure</td>
<td>Consider whether provenance storage needs to be provided in a relational database, if one is the standard solution in the research environment, or would an RDF store or a NoSQL graph database be an option. Privacy and data regulations will play an important role in deciding whether and how best to distribute the data in a multiple site scenario.</td>
</tr>
<tr>
<td>14</td>
<td>Data infrastructure</td>
<td>Reuse existing application infrastructure for provenance transmission if storing provenance centrally rather than at each site. Messaging systems may be used to transmit data reliably and asynchronously between application sites.</td>
</tr>
<tr>
<td>15</td>
<td>Rate of growth</td>
<td>Due to the typically large size of provenance data collected over time, it is crucial to establish early on the rate at which provenance storage requirements are expected to increase over time. The level of detail in the model may need to be adjusted accordingly and some deletion / archiving procedures introduced.</td>
</tr>
<tr>
<td>16</td>
<td>Permissions</td>
<td>When implementing provenance access control, consider the differences in access permissions between concrete data, and provenance records of that data.</td>
</tr>
<tr>
<td>17</td>
<td>Restrictions</td>
<td>When designing an access control mechanism for provenance data, decide whether restricted provenance information should be completely hidden or just have details abstracted. If the latter is the case, a mechanism is needed for answering user queries using graphs restricted to that user’s access level.</td>
</tr>
<tr>
<td>18</td>
<td>Sensitive data</td>
<td>When provenance is used to capture data about the actions of the security layers, separation of functionality needs to be introduced to avoid inadvertently exposing sensitive data. One way of doing so is by providing token identifiers which can then be used to request detail from the security mechanism.</td>
</tr>
<tr>
<td>19</td>
<td>Extensibility of queries</td>
<td>Introduce mechanisms for dynamically adding new provenance queries to the software.</td>
</tr>
<tr>
<td>20</td>
<td>Interactive querying</td>
<td>Provenance data can be used to answer individual queries or create tabular and graphical reports, but added value can be obtained by exploiting the graph representation of provenance to support open-ended, investigative querying.</td>
</tr>
<tr>
<td>21</td>
<td>Breaking abstraction</td>
<td>When creating provenance queries, whether for atomic questions, or for navigating the provenance graph space, try to find queries that can be optimized by being directly formulated in the underlying query language and investigate whether the performance gain warrants this.</td>
</tr>
</tbody>
</table>
Cognitive Load and Visualizing Traceability Information

It is assumed that traceability data can be represented by a causality graph, or DAG, that has been enriched with annotations to provide the complete information needed to trace an object back to its origin (Moreau, Freire, et al., 2008; Zhao et al., 2004). A standard graphical notation for provenance was introduced by the OPM in part to promote common tools and methods for the visual representation of provenance graphs (Moreau et al., 2011). In the e-sciences the causality graph is a common method for representing traceability, and supports accompanying methods for visualizing and browsing the graph (Simmhan et al., 2005a). Within a traceability causality graph the nodes represent data items and the edges represent the dependency relationships, such as versions, mapping, or derivations (Acar et al., 2010). The causality graph can also provide a data source to query against.

Visual representations can manifest patterns of real-world phenomena (Huynh et al., 2014) which might initiate further examination into the who, when, and why detailed provenance information. The ability to easily explore the entire traceability graph makes it easier for the users to answer many more interesting questions (Macko & Seltzer, 2011). Large causality graphs with many thousands of nodes are too large for users to navigate comfortably without mechanisms to simplify the content displayed. The traditional approach to traceability visualization shows only the relevant nodes, as specified by a filter, or as defined by a view. Graph visualizations have a number of properties that can be exploited when rendering a visualization. For example, position, size, color, edge weight, and application specific rules for how the properties influence the graph visualization (Collberg, Kobourov, Nagra, Pitts, & Wampler, 2003).

Symbolic representations of data in a tabular representation facilitate the extraction of specific data items. Tables work well for representing discrete data values. Graphs provide spatial problem representations that topologically convey information about relationships in the data. Spatially represented data facilitates viewing at a glance without requiring each data point to be addressed independently. Graphs are particularly strong at assessing a problem as a whole, and tasks that require understanding relationships or making associations between data items (Vessey, 1991). Tabular presentations work well for tasks that require an accurate
interpretation of values. Graphs work well for tasks that require assessing time dependent patterns (Dickson, DeSanctis, & McBride, 1986).

Huanga (2009) proposed a cognitive load construct within the context of graph visualizations that includes: (1) domain complexity, (2) data complexity, (3) task complexity, (4) visual complexity, (5) demographic complexity, (6) an indicator of the complexity of the data within the domain, and (7) time complexity. Huanga (2009) noted that higher levels of domain knowledge decreased the effort needed to understand the graph visualization of biology networks. When mismatches between data representation models and tasks require users to perform additional mental translations that increase the user’s cognitive load, task performance is expected to degrade. Degraded task performance can be measured by increased time to complete the task and increased probability of error. Higher performance occurs when the data representation models fit the task requirements (Goodhue, Klein, & March, 2000).
CHAPTER 3

RESEARCH METHODOLOGY

"Design science research is motivated by the desire to improve the environment by the introduction of new and innovative artifacts" (Simon, 1996).

This research utilized the DSR paradigm as described in Simon (1996) and Hevner, March, Park, and Ram (2004). This research project applied the seven guidelines established for DSR in Hevner et al. (2004). Table 7 briefly describes how this research meets the each of the Hevner et al. (2004) DSR guidelines. In DSR the objective is to create and evaluate an IT artifact that explicitly addresses an organizational problem or opportunity, as noted in Hevner et al. (2004). The implementation of the IT artifact provides “proof by demonstration” (Nunamaker Jr & Chen, 1990), and shows the feasibility of both the design process and the final designed product (Hevner et al., 2004).

Table 7. Based on Hevner et al. (2004) design science research guidelines (P. 83)

<table>
<thead>
<tr>
<th>Guideline</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design as an Artifact</td>
<td>The Trace-XML Framework for clinical research metadata traceability is the primary artifact created by this DSR project.</td>
</tr>
<tr>
<td>Problem Relevance</td>
<td>Trace-XML solves a real-world problem by extending the existing clinical research data standards in a way that can be (1) adopted as a future standard, (2) used by regulators to improve traceability within submissions, (3) used by clinical researchers to enhance data quality, and (3) used by software tool vendors seeking to enhance their product's support for the end-to-end clinical research data lifecycle. Traceability enhances the strength of clinical research analysis results by providing the source data and the processes used to determine the stated study results (Curcin et al., 2014).</td>
</tr>
<tr>
<td>Design Evaluation</td>
<td>In addition to passing a set of test cases, the Trace-XML framework was analytically evaluated using graph theory and traversal algorithms as discussed in Chapter 5. A qualitative study that used semi-structured interviews and</td>
</tr>
</tbody>
</table>
applied thematic analysis to evaluate the utility generated by Trace-XML is discussed in Chapter 6.

<table>
<thead>
<tr>
<th>Research Contributions</th>
<th>The Trace-XML Framework and its associated IT artifacts are the primary research contributions of this DSR project. More specifically, the research contributions include the Trace-XML graph model, traceability algorithms, and Define-XML extension.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Research Rigor</td>
<td>The application of graph theory and the adaptation of specific graph traversal algorithms provide the theoretical foundation for Trace-XML. This theoretical foundation enables the development of traceability proofs and represents a rigorous mechanism to validate traceability.</td>
</tr>
<tr>
<td>Design as a Search</td>
<td>Trace-XML seeks to address current regulatory requirements and extends the existing data standards. Trace-XML adapts existing graph theory and traversal algorithms to support the requirements of clinical research data traceability. The development of Trace-XML evolved during numerous build-evaluate design cycles.</td>
</tr>
<tr>
<td>Process</td>
<td></td>
</tr>
<tr>
<td>Communication of the</td>
<td>In addition to being the subject of this dissertation, this DSR project was presented in a paper at the 2017 AMIA Clinical Research Informatics conference, and an associated journal article is in progress.</td>
</tr>
<tr>
<td>Research</td>
<td></td>
</tr>
</tbody>
</table>

Build and evaluate are the two primary activities in DSR (March & Smith, 1995). Hevner (2007) describes that DSR cycles shown in Figure 10 that highlight a central design cycle. The design cycle iterates between the build-evaluate activities with the evaluation providing the feedback needed to refine the design for the next build iteration. This continues until finally a satisfactory design is realized. To more fully leverage the complementary nature of the build-evaluate cycle, this research project used a test-driven development (TDD) methodology where each iteration’s unit test scripts are developed prior to implementation as part of the build cycle. The evaluation cycle conducted a more rigorous system test that included testing the integration of all the application components. The formal evaluation will include the use of the theoretical foundations to prove that traceability has been established within the prototype implementation. It will also include a user evaluation to determine if the artifacts developed for this project improve the ability of clinical data experts understand the traceability metadata for a clinical trial. Semi-structured interviews from a panel of clinical data stakeholders will be conducted to attain the data needed to perform the user evaluation analysis.
Figure 10. Design science research cycles (Hevner, 2007)

Traceability Framework High-level Requirements

As part of the relevance cycle highlighted in Hevner (2007) and shown in Figure 10, a set of high-level requirements were identified for the clinical research data traceability environment. These requirements are listed in Table 8, and were provided as input to the build-evaluate cycle during the development of the Trace-XML artifacts. Each of the requirements listed below were covered by the final Trace-XML artifacts.

<table>
<thead>
<tr>
<th>#</th>
<th>Requirement</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>The traceability model must support references to domain semantics, and existing data and domain models where available (Curcin et al., 2014; Johnson et al., 2014).</td>
</tr>
<tr>
<td>2</td>
<td>A common vocabulary for expressing the semantics of provenance entities must be used (Curcin et al., 2014).</td>
</tr>
<tr>
<td>3</td>
<td>The metadata traceability model must be layered to represent traceability information at multiple levels of abstraction (Curcin et al., 2014; Davidson &amp; Freire, 2008).</td>
</tr>
<tr>
<td>4</td>
<td>Traceability models must be capable of forming a single graph that describes the full history of any specific data element through all stages of the clinical research data lifecycle (Curcin et al., 2014).</td>
</tr>
<tr>
<td>5</td>
<td>A query capability must exist to support the retrieval of detailed traceability information. This information may include metadata details on a specific data item, a tabular listing of items, or a trace back to the data item’s origin (Curcin et al., 2014).</td>
</tr>
<tr>
<td>6</td>
<td>The traceability framework should represent traceability constructs using a common vocabulary to support distributed healthcare applications (Curcin et al., 2014).</td>
</tr>
<tr>
<td>7</td>
<td>The traceability framework must provide the means to validate traceability as a means to assess data quality (FDA, 2016a).</td>
</tr>
</tbody>
</table>
The traceability framework must provide a mechanism for dynamically adding new traceability queries (Curcin et al., 2014).

The traceability framework must provide the information needed for a reviewer to understand the relationships between the analysis results, analysis datasets, tabulation datasets, and source data (FDA, 2016a).

The traceability framework must represent relationships between data elements in a standardized way (FDA, 2016a).

The traceability framework must be capable of identifying traceability gaps (Hume et al., 2016).

The traceability framework must attempt to resolve traceability gaps using the available metadata.

Artifact Design Table

To document the Trace-XML high-level design a DSR artifact design table was created to (1) map the research objectives to the research gaps identified during the literature review, (2) to identify the theories that will be used to address the research objectives, (3) to list the high-level features that will apply the theory to address the research objective, and finally (4) to list how the features of the artifacts produced will be rigorously evaluated. The artifact design table is shown below in Table 9.

<table>
<thead>
<tr>
<th>Research Gap</th>
<th>Research Objectives</th>
<th>Theory</th>
<th>Artifact Features</th>
<th>Evaluation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gaps exist in the computable traceability provided by the standard metadata models in the regulated clinical research domain. Computable traceability is not possible. (Davidson &amp; Freire, 2008; Dootson, 2011; Hume et al., 2016; Miles et al., 2007;</td>
<td>“Where is content missing?”</td>
<td>Graph theory and graph traversal algorithms (Sedgewick &amp; Wayne, 2011; Shankaranarayan et al., 2003)</td>
<td>Algorithm that identifies traceability gaps. The algorithm identifies missing edges or the relationships between metadata elements that represent data flows. The algorithm identifies both reachability and traceability failures.</td>
<td>Develop proofs for reachability and traceability within the Trace-XML framework using graph theory and graph traversal algorithms (Shankaranarayan et al., 2003). Conduct a qualitative assessment of the utility of the algorithm</td>
</tr>
<tr>
<td>“Where is content missing?”</td>
<td>Develop a traceability framework that identifies traceability gaps within clinical research study data.</td>
<td>Causality graphs (Acar et al., 2010; Curcin et al., 2014; Moreau et al., 2011)</td>
<td>IP-Maps (Chee et al., 2014;</td>
<td></td>
</tr>
<tr>
<td>Research Gap</td>
<td>Research Objectives</td>
<td>Theory</td>
<td>Artifact Features</td>
<td>Evaluation</td>
</tr>
<tr>
<td>--------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------------</td>
<td>-----------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------</td>
<td>--------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Gaps exist in the semantics and metadata needed to express computable traceability in the standard metadata models in the regulated clinical research domain. (Davidson &amp; Freire, 2008; Dootson, 2011; Hume et al., 2016; Miles et al., 2007; van Valkenhoef et al., 2012)</td>
<td>&quot;What content is missing?&quot; Within the traceability framework develop an algorithm that recommends the metadata content needed to resolve the traceability gaps, and extends the existing standards metadata sufficiently to express computable traceability.</td>
<td>Shankaranarayanan, 2005; Shankaranarayanan et al., 2000)</td>
<td>Algorithm that recommends resolutions to the identified traceability gaps.</td>
<td>Develop proofs for reachability and traceability, and completeness within the Trace-XML framework using graph theory and graph traversal algorithms (Shankaranarayanan et al., 2003). Conduct a qualitative assessment of the utility of the algorithm implementations to determine the utility added to clinical research data experts (Guest et al., 2011).</td>
</tr>
<tr>
<td>No capability exists to verify traceability completeness within clinical research study data (Davidson &amp; Freire, 2008; Dootson, 2011; Hume et al., 2016; Miles et al., 2007; van Valkenhoef et al., 2012).</td>
<td>&quot;Is the content correct and complete?&quot; Within the traceability framework develop algorithms that verify the correctness and completeness of the standards metadata to</td>
<td>Graph theory and graph traversal algorithms (Sedgewick &amp; Wayne, 2011; Shankaranarayanan et al., 2003)</td>
<td>Algorithm that verifies traceability completeness after validating for correctness. Constraint rules verify correctness.</td>
<td>Develop a proof for completeness within the Trace-XML framework using graph theory and graph traversal algorithms (Shankaranarayanan et al., 2003). Conduct a qualitative assessment of the utility of the</td>
</tr>
</tbody>
</table>
Trace-XML Development

Following the DSR methodology build and evaluate cycles (March & Smith, 1995), a prototype software application was developed in Java to implement Trace-XML including the creation of the traceability graph and the algorithms for querying and validating traceability. JDOM 2 was used to process the XML in the Java application. The BaseX 8.5.2 XML database engine XQuery 3.1 processor was used to implement the traceability query tool. The Define-XML extension was implemented in XML schema. The traceability graph is represented using the GraphML v1.0 schema. The final Trace-XML software artifacts rendered GraphML for two open-source graph visualization and editing tools: yEd v3.1.6 (yWorks, 2017) and Gephi v0.9.1 (Bastian, Heymann, & Jacomy, 2009; Gephi, 2017).

Trace-XML Layered Framework Implementation

The Trace-XML Framework implementation consists of 3 layers: (1) the Information Product Map (IP-Map) model: a high-level view of the manufacturing process for creating an information product (IP); (2) the CDISC standards metadata: metadata describing the IPs, data elements, and computations at a detailed level of granularity; and (3) a graph model: traceability throughout the clinical research data lifecycle that supports traceability visualization, validation, and queries. Layer 1 applies the IP-Map research to use IPs to represent computable traceability within clinical research data at a higher level of abstraction. Layer 2 represents the detailed study metadata provided by the ODM and Define-XML files. This detailed study metadata maps into the higher-level IP-Map representation found in Layer
1 of the framework. Layer 3 includes the algorithms that generate the graph, identify any traceability gaps, and validate the completed graph. Generating the graph for Layer 3 uncovered traceability gaps in the CDISC standards metadata in Layer 2. Trace-XML addresses these traceability gaps through the development of an extension to the Define-XML standard.

**Trace-XML Accessibility and License**

The Trace-XML software will be made available as open source under the Apache License, version 2.0. The system documentation and instructions on accessing and using the software are available on GitHub at https://github.com/swhume, and will also be made available at http://www.odm-review.com/ as well as http://www.cdisc.org.

**Evaluation**

The scientific evaluation of artifacts is the essence of information systems as DSR (Iivari, 2007). DSR evaluation schemes can be artificial or naturalistic in nature (Venable, 2006). Although the artifact will be evaluated in a demonstration environment, and not in a live clinical research project, it will use the metadata and data flows from the current industry standards as well as content from an actual study. As actual standards and study metadata will be used in the testing, the effectiveness will be evaluated using naturalistic test cases, while the process of creating and evaluating the traceability established within the Trace-XML framework will be performed in an artificial environment. As noted in Iivari (2007), the artifacts must be comprehensively tested in laboratory and experimental conditions prior to executing case or field studies. The evaluation should assess the artifact for utility and ensure it solves an identified problem within an organization’s technical infrastructure (Tremblay, Hevner, & Berndt, 2010).

Hevner et al. (2004) provide five categories of design evaluation methods, including: (1) observational, (2) analytical, (3) experimental, (4) testing, and (5) descriptive. In addition to testing the artifact, the quality, efficacy, and utility of the artifacts in this DSR project will be confirmed by the analytical and observational evaluation methods.
Analytical Evaluation

The analytical evaluation proves that reachability, traceability, and completeness are demonstrated within Trace-XM through the application of graph theory and specific traversal algorithms. The use of graph theory and graph traversal algorithms within the framework contributes significantly to the research rigor and the quality of the artifacts. The theoretical foundation of the artifact provides the basis for the claim that traceability has been established within the extended Trace-XM Framework, as well as the basis for the extension to the CDISC standards created as an artifact of this DSR project. Testing for traceability makes use of proofs established in Shankaranarayan et al. (2003), but applies them at the data element-level within a clinical study. The same basic algorithms work at multiple levels within the traceability framework. The analytical approach to evaluating the theoretical framework represents the most significant component of the evaluation plan. Chapter 5 describes the analytical evaluation in detail and contains the details of the algorithms and their theoretical foundations.

Observational Evaluation

An observational evaluation was necessary to determine the utility of the Trace-XM Framework from a clinical research data expert’s perspective. Causality graphs are commonly used for traceability data representation in the e-sciences (Simmhan et al., 2005a, 2005b) as well as other domains (Moreau, 2010; Moreau, Groth, et al., 2008). However, no empirical evidence exists to support either the graph-based interface or the tabular interface as a better fit for the task of assessing the traceability of regulated clinical research data. The optimal approach to applying existing knowledge about graph visualizations, or when different representations are preferred, remains an open question (Carata et al., 2014).

The observational evaluation approach consisted of demonstrating the artifact to clinical data experts and using semi-structured interviews to gather data to evaluate the utility of the artifacts for use in a field environment. Semi-structured interviews facilitated the examination of the individual expert’s understanding of traceability within the Trace-XM Framework. This provided access to the contextual issues regarding the utility of the framework for use in the field that would be otherwise difficult to access (Chee et al., 2014;
The interview method enabled personal contact with the participants, but eliminated the risk that a few participating experts dominate the interaction as can happen in focus groups (Chee et al., 2014). The observational utility and usability assessments complement the analytical evaluation and help ensure that the artifacts are solving real-world problems as experienced by the clinical research data experts. A pre-defined set of open-ended questions were used for all interviews permitting the interviewer to probe into the details as needed to gain insights into themes that could influence future versions of the artifacts. This approach permitted the respondent to share issues that might not have been considered during the development and evaluation of Trace-XML.

The interpretive qualitative study used to assess the utility created by Trace-XML used the Applied Thematic Analysis (Guest et al., 2011) method. Thematic analysis was used to analyze the interview data. The design and results of the qualitative study are detailed in Chapter 6.
CHAPTER 4

THE TRACE-XML FRAMEWORK

Design Science Research Artifacts

The primary artifact generated by this research is a clinical research data traceability framework named Trace-XML. Within this framework, artifacts contributing to the knowledge base include (1) the Trace-XML extension to the CDISC standards metadata that, when applied, provides the means to create a model that supports computable traceability; (2) the algorithms that identify the traceability gaps and validate that the resulting model supports traceability; and (3) a graph representation that provides the means to run traceability queries and visualize the traceability metadata.

This research contributes an extension to the Define-XML CDISC standard that addresses the traceability gaps uncovered by the Trace-XML algorithms. It also contributes new recommended standards implementation practices that improve traceability completeness. IP-Maps have been used to evaluate data quality, and this research extends that work to examine traceability as a dimension of data quality at the level of granularity needed to support regulated clinical research. This research contributes the algorithms for remedying, validating, and querying traceability in the Trace-XML graphs.

Traceability Framework Overview

The Trace-XML Framework provides traceability across data from all phases of the clinical research data lifecycle, from data collection through analysis datasets. Trace-XML provides new full lifecycle, end-to-end traceability features, including (1) the ability to validate end-to-end traceability, (2) the ability to run end-to-end traceability queries, and (3) the ability to visualize end-to-end traceability. These new features provide a comprehensive view of study data traceability that combines existing CDISC standards metadata across files from different phases in the clinical research data lifecycle. This replaces today's siloed
approach to traceability that provides descriptive metadata within the context of one phase of the lifecycle, as shown in Figure 11.

Each data lifecycle phase in Figure 11 has a CDISC XML file containing the metadata that describes the data for that particular state in the clinical research data lifecycle. The ADaM metadata exists in a Define-XML file, the SDTM metadata exists in a separate Define-XML file, and the CDASH metadata exists in an ODM file. Each XML metadata file exists independently of the others. Furthermore, each XML metadata file does not reference the metadata available from the other phases in the data lifecycle supporting the notion of a siloed approach to metadata traceability within the CDISC standards. Trace-XML creates new, full lifecycle traceability capabilities by enabling each of the separate XML metadata files to reference metadata in the other files, creating an integrated, full study lifecycle view of traceability.

The Trace-XML Framework shown in Figure 12 consists of 3 layers: (1) the IP-Map model represents a high-level view of the manufacturing process for creating an IP; (2) the CDISC standards metadata models represent the IPs, data elements, and computations at a detailed level of granularity; and (3) a graph model represents traceability throughout the clinical research data lifecycle that supports traceability visualization, validation, and queries. The layers in the framework work together to provide a complete view of study-level traceability at increasing levels of detail. Each layer and its associated traceability visualizations address a different traceability assessment need.
Figure 12. 3-layered Trace-XML Framework conceptual diagram

The purpose of each layer in the framework is summarized in Table 10. Each framework layer adapts existing technologies to represent the metadata describing full lifecycle traceability in the clinical research domain. Contributions of this research project include the specific technology adaptations or extensions developed to represent the metadata in the layered Trace-XML Framework, as well as the algorithms developed to work with layer 3.

<table>
<thead>
<tr>
<th>Layer</th>
<th>Model</th>
<th>Purpose</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>IP-Map models</td>
<td>This layer adapts the IP-Map's conceptual visualization of an information product manufacturing process to aid reviewers in identifying how clinical data is being captured, transformed, validated, and stored throughout a study. This layer is useful for regulatory and other data reviewers to acquire a high-level understanding of the data inputs and processes used to create a resulting information product. By design, this layer does not provide enough detail to fully satisfy the traceability requirements for regulated clinical research.</td>
<td>(Chee et al., 2011; Chee et al., 2014; Shankaranarayan et al., 2003; Shankaranarayanan, 2005; Shankaranarayanan et al., 2000)</td>
</tr>
<tr>
<td>2</td>
<td>CDISC metadata models</td>
<td>This layer adapts the Define-XML standard through the Trace-XML extension. This layer provides the detailed study metadata as described by the CDISC ODM and Define-XML models. This metadata</td>
<td>(CDISC, 2013a, 2013c; Hume et al., 2016)</td>
</tr>
<tr>
<td>Layer</td>
<td>Model</td>
<td>Purpose</td>
<td>References</td>
</tr>
<tr>
<td>-------</td>
<td>------------------------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
<td>-------------------------------------------------</td>
</tr>
<tr>
<td>3</td>
<td>Directed graph model</td>
<td>This layer implements a digraph based on the ODM and Define-XML metadata models and applies the Trace-XML extension to generate the graph's directed edges. The digraph model is represented using the GraphML XML standard. The GraphML digraph for an entire study enables traceability validation, visualization, and queries. The GraphML files can be visualized using multiple open source software applications.</td>
<td>(Sedgewick &amp; Wayne, 2011; Shankaranarayan et al., 2003)</td>
</tr>
</tbody>
</table>

**Layer 1 Overview: The IP-Map Layer.** Layer 1 applies the IP-Map research to represent computable traceability within clinical research data at a higher level of abstraction (Shankaranarayan et al., 2003). IP-Map provides a conceptual visualization of an IP’s manufacturing process that aids information consumers in identifying how data is being captured, transformed, stored, and utilized prior to becoming available to the decision maker (Chee et al., 2011; Shankaranarayanan et al., 2000). This research uses IP-Maps as an analysis tool to support traceability as a dimension of data quality. In the context of a clinical trial, and particularly a regulatory submission, IPs include the analysis datasets and analysis results that must be understood by a regulatory reviewer. For example, it is particularly important that regulatory reviewers understand the traceability for study endpoints and the efficacy datasets. The IP-Map layer provides diagrams depicting the data flow from source data to final IP. The FDA has requested diagrams to show the “electronic data flow” when electronic health records (EHR) are used as source data in clinical research (FDA, 2016b), and IP-Maps can represent this information. IP-Maps were developed as a data quality tool to improve data believability and ease of understanding (Chee et al., 2014) by displaying an information manufacturing process that includes the transformations and quality checks that precede the
creation of an IP. The addition of the data transformations and quality check processes to a basic data flow diagram should be of interest to regulatory and other reviewers with a special interest in data quality.

**Figure 13** shows a partial IP-Map for the creation of the CDISC ADaM Subject Level Analysis Dataset (ADSL) that includes EHR data sources as well as data extraction, cleaning, transformation, and validation processes. **Figure 13** also represents the flow of the data from raw sources to a final ADSL dataset IP. The IP-Map is used to establish reachability and traceability as elements of data quality within the IP manufacturing process (Shankaranarayan et al., 2003).

**Figure 13. Example layer 1 partial IP-Map for ADSL**

Layers 1 and 2 of the Trace-XML Framework are integrated in both directions, layer 1 references layer 2 and layer 2 references layer 1. Each block in the IP-Map contains metadata that includes a description of the block and specific metadata that references the layer 2 metadata XML files. The CDISC specific metadata models provided by layer 2 in the framework can explicitly reference the layer 1 IP-Map components using the *Alias* element, as shown in **Figure 14**.
Layer 2 Overview: The CDISC Standards Metadata Layer. Layer 2 represents the detailed study metadata provided by the CDISC ODM and Define-XML metadata files created as part of a study. These files contain a complete description of the study artifacts, such as CRFs and datasets, as well as the detailed data element definitions and the methods used for derivations and transformations of the data. Figure 14 shows a Define-XML v2.0 fragment that represents a partial demographics dataset. The detailed metadata maps into the higher-level IP-Map representation found in layer 1 of the framework and provides inputs into the graph model in layer 3. In this research, the CDISC standards provide the domain models and metadata for the data element level traceability, and this benefits users as these semantics are known within the clinical research domain (Curcin et al., 2014; Johnson et al., 2014).

The current versions of ODM and Define-XML provide descriptive metadata in support of traceability, but do not include the metadata needed to provide computable traceability across the data lifecycle as shown in Figure 11. For example, Define-XML for the SDTM standard provides a link to a PDF-based image of an annotated CRF, but cannot

---

**Figure 14. Using Alias to link a Define-XML domain to an IP-Map block**

```
<!-- Dataset Definition (DM) -->
<ItemGroupDef OID="SDTM.IG.DM"
    Domain="DM" Name="DM" Repeating="No" IsReferenceData="No"
    SASDatasetName="DM" Purpose="Tabulation"
    def:Structure="One record per subject" def:Class="SPECIAL PURPOSE"
    def:CommentOID="COM.DOMAIN.DM" def:ArchiveLocationID="LF.DM">
  <Description>
    <TranslatedText xml:lang="en">Demographics</TranslatedText>
  </Description>
  <ItemRef ItemID="SDTM.IT.STUDYID" OrderNumber="1"
    Mandatory="Yes" KeySequence="1"/>
  <ItemRef ItemID="SDTM.IT.DM.DOMAIN" OrderNumber="2" Mandatory="Yes"/>
  <ItemRef ItemID="SDTM.IT.USUBJID" OrderNumber="3"
    Mandatory="Yes" KeySequence="2" MethodOID="SDTM.MT.USUBJID"/>
  <ItemRef ItemID="SDTM.IT.SUBJID" OrderNumber="4" Mandatory="Yes"/>
  <ItemRef ItemID="SDTM.IT.SITEID" OrderNumber="7" Mandatory="Yes"/>
  <ItemRef ItemID="SDTM.IT.DM.COUNTRY" OrderNumber="14" Mandatory="Yes"/>
  <ItemRef ItemID="SDTM.IT.DM.RFSTDTC" OrderNumber="15"
    Mandatory="No" MethodOID="SDTM.MT.RFSTDTC"/>
  <Alias Context="ADSL-IP-Map" Name="STO1"/>
  <def:leaf ID="LF.DM" xlink:href="dm.xpt">
    <def:title>dm.xpt</def:title>
  </def:leaf>
</ItemGroupDef>
```
display the full lifecycle of a variable. The Trace-XML Define-XML extension created as part of this research project extends the existing standards to include explicit references to the source data and methods such that a directed graph can be created in layer 3 of the framework. The extension augments the metadata definition of each variable to include references to its immediate predecessor variables defined elsewhere within the study metadata. This source reference metadata is typically located in another Define-XML or ODM file that represents the previous stage in the clinical research data lifecycle. The layer 2 metadata is displayed within the framework using stylesheets to create a data dictionary style view, the same view provided as part of a regulatory submission. Layer 2 of the Trace-XML Framework integrates views of traceability generated using the layer 3 graph that presents the full data lifecycle trace for each variable. Integrating layer 3 views into the style sheet used for regulatory submissions of Define-XML promotes the alignment of Trace-XML artifacts with the existing standards.

**Layer 3 Overview: The Graph Layer.** Layer 3 includes the algorithms that generate the directed graph, identify the traceability gaps, and validate the completed graph. The forms, datasets, variables, and computational methods metadata provided by the CDISC models in layer 2 become the nodes in the layer 3 graph. Each graph node includes the CDISC metadata details drawn from layer 2. The flow of the directed graph matches the flow provided in the IP-Maps in layer 1. Generating the graph using the Define-XML and ODM metadata uncovered gaps in the layer 2 metadata needed to generate the complete set of graph edges. These gaps were addressed through the Trace-XML extension to the Define-XML standards metadata that added the explicit references to source variables. These source variable references were used to generate the directed edges in the full lifecycle graph. Once the directed graph has been generated the data flow for a study can be visualized as shown in Figure 15.
Validation of CDISC standards metadata beyond mere XML schema validation has become a critical step in the regulatory submission process. Using the directed graph and depth-first search (DFS) based algorithms provided by Trace-XML, metadata validation can be extended to cover traceability across the full clinical research data lifecycle to improve data integrity. When Trace-XML identifies an unexpected gap in reachability or traceability a report can be generated to show the unexpected gaps. The Unreachable Nodes report fragment shown in Figure 16 shows two nodes that are unexpectedly unreachable in a Trace-XML graph. When valid traceability exists across the study lifecycle, no unexpected reachability gaps should exist.

### Unreachable Nodes

<table>
<thead>
<tr>
<th>#</th>
<th>Element OID</th>
<th>Origin</th>
<th>Expected?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SDTM.IT.DM.ETHNIC</td>
<td>CRF</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>ADAM.IT.ADSL.ETHNIC</td>
<td>Predecessor</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>SDTM.IT.DM.COUNTRY</td>
<td>Assigned</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>SDTM.IT.VS.VSORRES</td>
<td>Assigned</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>SDTM.IT.VS.VSORRESSU</td>
<td>Assigned</td>
<td>yes</td>
</tr>
</tbody>
</table>

Once the software confirms that the graph is valid, a Trace-XML query traces a variable back to its original source to include any transformations or derivations that impact
the data. The results of a Trace-XML query show the full lifecycle for a variable and include metadata retrieved from the layer 2 XML files. For example, the Trace-Query report fragment in Figure 17 shows each step of the trace from analysis variable back to the original data collection source, including the detailed metadata describing the content in each step. This Trace-Query output uses the Trace-XML graph and includes metadata details drawn from three separate XML files: (1) the ADaM Define-XML file, (2) the SDTM Define-XML file, and (3) the CDASH ODM file.

<table>
<thead>
<tr>
<th>Trace for ItemDef Element Named SEX (ANALYSIS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>#</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>6</td>
</tr>
</tbody>
</table>

Figure 17. Example Trace-Query output for the ADSL SEX variable

**Trace-XML Application Overview.** In addition to the Trace-XML Define-XML extension, the Trace-XML application consists of the three primary software applications listed in Table 11. These applications were developed explicitly for Trace-XML in Java and represent a significant part of the overall Trace-XML artifacts developed as part of this research project. See Table 15 for a list identifying each software component developed as part of the applications summarized in Table 11.
Table 11. Main Trace-XML Framework software applications

<table>
<thead>
<tr>
<th>Software Application</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TraceXML</td>
<td>The Trace-XML application uses the metadata files provided in layer 2 with the Trace-XML Define-XML extension to generate the full lifecycle graph in the GraphML standard. This graph provides computable traceability. Trace-XML validates the graph for reachability and traceability and reports any gaps found. The GraphML graph can be visualized or analyzed using several open source software tools that support the graph standard.</td>
</tr>
<tr>
<td>TraceQuery</td>
<td>The Trace-Query application generates trace data or a report for any given node in the graph generated by Trace-XML. The query returns the full trace for a node that includes every step in the study data lifecycle up to that point. The graph nodes can be variables, methods, forms, sub-forms, or datasets. The report can be generated for any 1 variable, or a set of variables.</td>
</tr>
<tr>
<td>TraceKB</td>
<td>The Trace-KB application creates a knowledge base of edge metadata that can be used to generate the source reference metadata included in the Trace-XML Define-XML extension. This application uses CDISC standards content pulled from the SHARE MDR to populate the knowledge base. Not all the required edge metadata is available from the CDISC standards, and sponsors interested in generating the Trace-XML extension content will need to supplement the knowledge base with metadata from their own mapping applications or organizational standards.</td>
</tr>
</tbody>
</table>

Trace-XML: An Integrated Framework

The 3 layers of the Trace-XML Framework are integrated to provide a comprehensive view of traceability at increasing levels of detail as shown in Figure 12 and described in Table 10. The data flow within the clinical research lifecycle, from data collection through analysis datasets, is represented by directed edges within the graph. Tracing a variable’s lineage requires following these edges backwards from analysis content back to the source of the collected data. The Trace-XML Framework saves the graph as XML using the standard GraphML format. GraphML is supported by several open source software tools for viewing, filtering, and analyzing the resulting graph. Figure 18 shows the traceability for the ADaM ABLFL (vital signs analysis baseline flag) variable in a predecessor graph fragment created using the yEd (yWorks, 2017) software. Figure 18 shows an ADAM.IT.VS.ABLFL variable that can be traced back through the method that derives it, ADAM.METHOD.VABLFL,
through to the original data collection variable, ODM.ITEM VISIT, on the vital signs CRF, ODM.FORM.VS.

Layering enables the framework to represent traceability at multiple levels of abstraction. The hierarchical nature of the framework provides data reviewers with a high-level, abstract view of the entire information manufacturing process in layer 1 that is integrated with increasingly detailed views of traceability in the subsequent layers. Trace-XML’s hierarchical traceability framework fosters an improved understanding for each IP by permitting reviewers to seek additional traceability details as needed to answer specific questions about individual data elements (Chee et al., 2011). Trace-XML provides a more comprehensive assessment of traceability by integrating the conceptual layer, the clinical study artifact metadata layer, and a data-flow or graph layer (Chee et al., 2014).

Creating framework layers that integrate metadata generated in each phase of the data lifecycle creates a new, integrated view of metadata. Today’s standards provide views of one phase of the lifecycle at a time and at one level of detail. By integrating the 3 layers of the framework and covering the entire lifecycle for a study, Trace-XML provides additional traceability metadata along two different dimensions. Figure 19 shows both dimensions of traceability created by the Trace-XML Framework. On the vertical dimension, the Trace-XML Framework provides increasing levels of detail from the IP-Maps at layer 1 to the
study-level digraph at level 3. On the horizontal dimension, the Trace-XML Framework connects each phase of the clinical research data lifecycle into a single view. Figure 19 shows the horizontal dimension representing 3 phases of the clinical research data lifecycle as represented by their CDISC content standards. Integrating these dimensions creates a more complete and diverse representation of study metadata traceability.

![Figure 19. Trace-XML Framework creates new dimensions of traceability](image)

In Trace-XML the CDISC standards provide the domain models and metadata for the data element level traceability, and these benefit users as the semantics are known within the regulated clinical research domain. However, computable traceability across the clinical research lifecycle is not possible using the current CDISC standards because the traceability metadata provided in the *Origin* element provides only descriptive metadata used to identify the prior step in the process. Therefore, a Trace-XML extension to Define-XML was developed to include specific references to source variables found in a study’s Define-XML and ODM files. The new *Trace* and *TraceItem* elements shown in Figure 20 contain the source variable references and use the *trc* namespace prefix that has been designated to identify Trace-XML extension content. The *leafID* element provides a reference to the ODM or Define-XML file containing the reference and the *ItemOID* element contains the reference to the source variable. Optional identifying information can also be provided in *TraceItem*, including *ItemGroupOID* and *FormOID* attributes used to further constrain the *ItemOID*
source reference.

```xml
<ItemDef OID="SDTM.IT.USUBJID" Name="USUBJID" DataType="text" Length="30" SAS>
  <Description>
    <TranslatedText xml:lang="en">Unique Subject Identifier</TranslatedText>
  </Description>
  <def:Origin Type="Derived">
    <trc:Trace>
      <trc:TraceItem leafID="LF.ODM" ItemOID="ODM.IT.Common.StudyID"/>
      <trc:TraceItem leafID="LF.ODM" ItemOID="ODM.IT.Common.SubjectID"/>
    </trc:Trace>
  </def:Origin>
</ItemDef>
```

Figure 20. Example Trace-XML extension shown with the trc namespace prefix

For example, although an ItemOID identifier may reference a unique ItemDef element, that same ItemDef may appear in many ItemGroupDefs or FormDefs leading the trace to include multiple elements containing the source ItemDef. If the source of an ItemDef was a single instance of that variable, then referencing only the containing ItemGroupDef and FormDef elements provides a more accurate and simple trace.

The ability to explicitly reference source variables enables the Trace-XML software to generate the edges that connect the variables, computational methods, datasets, sub-forms, and forms into a graph representation. These references integrate the study metadata across the various stages in the clinical research data lifecycle. The ODM and Define-XML content provide the variables, computational methods, datasets, sub-forms, and forms that become the nodes in the graph. The source references for each variable provided by the Trace-XML extension are added to the Origin element to support the instantiation of a directed edge between the source and the target nodes. The data flow within the clinical research lifecycle, from data collection through analysis datasets, is represented by directed edges within the graph. Any derivation or transformation that impacts the variable is also represented in the graph. Tracing a variable’s lineage requires following these edges from analysis content back to the data collection metadata.

Figure 21 shows a hierarchical visualization of a Trace-XML graph fragment for a study lifecycle that includes data collection, standardized tabulations, and analysis datasets. This example fragment highlights the demographic domain and shows a relatively small
portion of a complete study graph that might typically include over 20 domains. Visualization tools, such as the yEd software used to render Figure 21, support the graph navigation and partitioning needed to analyze large graphs. This graph fragment shows the directed edges that connect nodes within one phase of the lifecycle, say connecting the SEX and RACE variables to the demographic sub-form (ODM.IG.DM), as well as connecting nodes across different phases of the lifecycle. Figure 21 shows the SEX variable with directed edges connecting the CDASH SEX variable to the SDTM SEX variable and the SDTM SEX variable to the ADaM SEX variable. Additional directed edges show that the SDTM SEX variable exists in the SDTM demographics dataset (SDTM.IG.DM), and the ADaM SEX variable exists within the subject-level analysis dataset (ADAM.ADSL).

Figure 21. Full lifecycle Trace-XML graph fragment in a hierarchical layout

Each node on the graph can be opened to reveal the detailed metadata pulled from the ODM or Define-XML content, such as a description of an algorithm used to transform the variable. For example, Figure 22 shows another predecessor graph generated by the yEd software with the metadata details for the selected node displayed on the right in the properties window. These details are taken directly from the Define-XML, and the details available in the graph are limited by the content provided in the Define-XML file.
A subset of the metadata needed to create the edges required to generate an integrated digraph was retrieved using the Trace-KB component of the Trace-XML Framework. Trace-KB generates a knowledge base of the relationships needed to connect the XML metadata across the phases of the clinical research data lifecycle. Trace-KB retrieves these metadata relationships from the SHARE MDR Application Programming Interface (API) (CDISC, 2015). When the CDISC CDASH standards are used the SHARE metadata can be applied by the Trace-KB software to automatically generate the extended source reference metadata required by the Define-XML extension. Figure 23 shows a report of the edges added to a Define-XML file using the Trace-XML extension to represent missing edges detected by Trace-KB. The edge content was retrieved from SHARE using the API and inserted into the Define-XML file the Trace-KB application. The report in Figure 23 lists the status as "Unconfirmed" meaning that the automated content generation by Trace-KB should be reviewed before published as final.

**Edges added from SHARE to the Define-XML file: d:/src/odm-prov/ip-map-xml/sdtm-define-dm-v03-updated.xml**

<table>
<thead>
<tr>
<th>#</th>
<th>Original Item OID</th>
<th>Original Item Name</th>
<th>Source Item OID</th>
<th>Source Item Group OID</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SDTM.ITDM.ETHNIC</td>
<td>ETHNIC</td>
<td>ODM.ITDM.ETHNIC</td>
<td>ODM.IG.DM</td>
<td>Unconfirmed</td>
</tr>
<tr>
<td>2</td>
<td>SDTM.ITDM.SEX</td>
<td>SEX</td>
<td>ODM.ITDM.SEX</td>
<td>ODM.IG.DM</td>
<td>Unconfirmed</td>
</tr>
</tbody>
</table>

Figure 23. Trace-XML edges added to Define-XML using SHARE content
Establishing Traceability Using Trace-XML

Reachability must be established to prove traceability exists within the graph generated from the ODM and Define-XML files. Given the directed graph $G_a$, any node $m$ is reachable from node $n$ in $G_a$ if there exists a directed path from $n$ to $m$. A DFS algorithm for digraphs will identify all and only those nodes reachable from a given node $n$ in the digraph $G_a$ (Sedgewick & Wayne, 2011; Shankaranarayan et al., 2003). Nodes that cannot be reached, but are expected to be reachable, are flagged as potential validation issues. Nodes with an Origin Type of “CRF”, “Derived”, and “Predecessor” must be reachable to be valid. The reachability test proceeds end-to-end across the clinical research lifecycle. The examples and tests developed for this research project demonstrate reachability that starts with the data collection CRF content in an ODM file, connects to nodes in a standardized tabulation Define-XML file, which in turn connects to nodes in an analysis Define-XML file. Once reachability has been established, it can be shown that if node $m$ is reachable from node $n$, then node $n$ is traceable from node $m$. Thus, achieving reachability for the nodes in $G_a$ asserts that traceability also exists (Shankaranarayan et al., 2003).

Once traceability has been confirmed, the full trace of any individual variable or node can be shown in a report that returns the basic metadata for each node. The Trace-XML query report generated by the Trace-Query application takes as input the unique identifier of the variable, or node, of interest and returns every connected node that precedes it. The Trace-XML digraph with confirmed traceability makes this feasible. Trace-Query uses XQuery to return the metadata for each preceding node in the trace. The metadata shown in Table 12 lists a subset of the information returned from a traceability query of the Pooled Site Group 1 analysis variable. The actual results include more details, such as a description of the computational method listed in row #3, than are shown in Table 12.
Table 12. Example Trace-XML query results for the Pooled Site Group 1 variable

<table>
<thead>
<tr>
<th>#</th>
<th>OID</th>
<th>Phase</th>
<th>Element</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ADAM.IT.ADSL.SITEGR1</td>
<td>Analysis</td>
<td>ItemDef</td>
<td>Variable</td>
<td>Pooled site group 1</td>
</tr>
<tr>
<td>2</td>
<td>ADAM.IG.ADSL</td>
<td>Analysis</td>
<td>ItemGroupDef</td>
<td>Dataset</td>
<td>Subject level analysis dataset</td>
</tr>
<tr>
<td>3</td>
<td>ADAM.MT.ADSL.SITEGR1</td>
<td>Analysis</td>
<td>MethodDef</td>
<td>Derivation</td>
<td>Computation method</td>
</tr>
<tr>
<td>4</td>
<td>SDTM.IT.SITEID</td>
<td>Tabulation</td>
<td>ItemDef</td>
<td>Variable</td>
<td>Study site identifier</td>
</tr>
<tr>
<td>5</td>
<td>SDTM.IG.DM</td>
<td>Tabulation</td>
<td>ItemGroupDef</td>
<td>Dataset</td>
<td>Demographics dataset</td>
</tr>
<tr>
<td>6</td>
<td>ODM.IT.COMMON.SITEID</td>
<td>Data Collection</td>
<td>ItemDef</td>
<td>Variable</td>
<td>Study site identifier</td>
</tr>
<tr>
<td>7</td>
<td>ODM.IG.COMMON</td>
<td>Data Collection</td>
<td>ItemGroupDef</td>
<td>Sub-form</td>
<td>Common variables</td>
</tr>
<tr>
<td>8</td>
<td>ODM.F.DM</td>
<td>Data Collection</td>
<td>FormDef</td>
<td>CRF</td>
<td>Demographics form</td>
</tr>
</tbody>
</table>

A hyperlink to an HTML rendering of each variable’s Trace-XML query can be included in the output generated by the Define-XML stylesheet to make reviewing traceability easier for reviewers and decision makers. **Figure 24** shows a partial view of an ADaM Define-XML that lists the vital signs analysis dataset ADVS with links to the individual variable traceability queries shown in the Source/Derivation/Comment column. These links provide data reviewers access to the detailed traceability information returned by a Trace-Query query that reaches back to the original source variable.
Using the graph and DFS-based algorithms provided by Trace-XML, metadata validation can be extended beyond individual Define-XML documents to cover the full clinical research data lifecycle as a means to improve data integrity. Using Trace-XML, the ODM and multiple Define-XML documents may be validated as one study to ensure end-to-end validity across the clinical research data lifecycle. Unreachable or untraceable nodes may be reported as validation errors so that the Define-XML or ODM files can be corrected to more accurately reflect the complete data flow through the lifecycle. Figure 25 shows a partial list of unreachable nodes generated by Trace-XML. The SDTM.IT.DM.ETHNIC and ADAM.IT.ADSL.ETHNIC variables are flagged as unexpectedly unreachable nodes. With Origin values of "CRF" and "Predecessor", respectively, these nodes are expected to be reachable within the Trace-XML graph, and these variables represent a validation issue for this study. In this case, the Trace-XML extension content for the SDTM.IT.DM.ETHNIC variable was removed from the SDTM Define-XML causing the validation issue.
Unreachable Nodes

<table>
<thead>
<tr>
<th>#</th>
<th>Element OID</th>
<th>Origin</th>
<th>Expected?</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SDTM.IT.DM.ETHNIC</td>
<td>CRF</td>
<td>no</td>
</tr>
<tr>
<td>2</td>
<td>ADAM.IT.ADSL.ETHNIC</td>
<td>Predecessor</td>
<td>no</td>
</tr>
<tr>
<td>3</td>
<td>SDTM.IT.DM.COUNTRY</td>
<td>Assigned</td>
<td>yes</td>
</tr>
<tr>
<td>4</td>
<td>SDTM.IT.VS.VSORRES</td>
<td>Assigned</td>
<td>yes</td>
</tr>
<tr>
<td>5</td>
<td>SDTM.IT.VS.VSORRESU</td>
<td>Assigned</td>
<td>yes</td>
</tr>
</tbody>
</table>

Figure 25. Report showing unexpectedly unreachable nodes in a Trace-XML graph

The GraphML standard used by Trace-XML can be rendered or analyzed using a number of open-source software tools, and Trace-XML can be configured to include GraphML extensions used by specific software packages. Open-source software tools such as yEd and Gephi provide alternative ways of conducting exploratory metadata analysis using visual analytics to quickly access how all the variables used within a study are related to one another. These tools provide another mechanism for assessing traceability within a study. They generate a wide variety of visualization layouts based on the same study graph to suit specific exploratory analysis preferences. These tools also generate graph metrics useful for analyzing and comparing study graphs. The graph shown in Figure 26 was created using the yEd software using the directed tree layout. Large, full lifecycle graphs are useful for exploring high-level data flows and permit a reviewer to zoom in on a graph fragment for a more detailed analysis.

Figure 26. Full lifecycle Trace-XML graph in a directed tree layout
Visual Assessment of Traceability using Graph Visualizations

After the gaps in the IP-Graph were identified and resolved, and the metadata standards were enhanced with the additional metadata, the IP-Map was re-instantiated and evaluated for reachability, traceability, and completeness. The evaluation process used the same algorithms as previously described with the expectation that the previously identified gaps had been resolved. After the algorithms provided the evidence of traceability and completeness, the software prototype generated a graph representation from the metadata to provide the traceability query capability and the graph visualization. The graph representation of the CDISC standards metadata is a contribution of this research.

The graph traceability queries can be used to test individual variables for traceability. The query returns every node in the trace back to its source, and missing or additional nodes in the trace can be identified when compared to the expected results. Rendering traceability graphs visually provides the means to explore traceability and is another means of identifying traceability issues. For example, the graphs in Figure 27 show an organic layout rendered by the yEd software for a test study. This layout shows that certain methods, instantiated as blue nodes, had outputs without inputs. Although this scenario is possible, in this case it highlighted errors in the test study metadata. The graph on the right in Figure 27 shows the graph rendered after the corrections to the test study have been applied.
Traceability can also be visually assessed using the generated graph by selecting a node from the full graph and generating the predecessor sub-graph as was shown in Figure 22. This view provides a complete view of traceability for the selected variable, in this case the vital signs analysis baseline flag. The ODM and Define-XML node identifiers are listed to the left of the predecessor graph and the metadata details for the selected node are shown on the right.

**Summary of the Trace-XML Traceability Rules**

This chapter has described the Trace-XML Framework and how it achieves traceability using CDISC standard metadata. It also described a number of Trace-XML specific traceability rules. These rules are summarized in Table 13. These traceability rules describe specific detailed requirements implemented in the Trace-XML prototype to generate the trace graph.
<table>
<thead>
<tr>
<th>#</th>
<th>Rule</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td><em>OIDs must be unique within a study</em></td>
<td>The OID, or identifier, for an element referenced by a <em>TraceItem</em> must be unique within the context of the study.</td>
</tr>
<tr>
<td>2</td>
<td><em>Zero or more TraceItem elements may be used to identify the source variables</em></td>
<td>An <em>ItemDef</em> variable may provide 0, 1, or n <em>TraceItem</em> elements as sources. Not all <em>Origin Type</em>'s support traceability, as is noted in subsequent rules, and non-traceable origin type variables will have 0 <em>TraceItem</em> elements. Many variables take their source from one variable, oftentimes as a carry-over from the previous phase in the clinical research data lifecycle, and these variables will include one <em>TraceItem</em> element. In cases where a variable references a <em>MethodDef</em>, that method may take multiple source variables as its input. Those input variables become <em>TraceItem</em> elements included in the <em>ItemDef</em> that receives the result of the <em>MethodDef</em> element's transformation or derivation.</td>
</tr>
<tr>
<td>3</td>
<td>Variables with multiple <em>TraceItem</em> sources must reference a <em>MethodDef</em></td>
<td>When an <em>ItemDef</em> has multiple <em>TraceItem</em> elements, an <em>ItemRef</em> to that <em>ItemDef</em> must reference a <em>MethodDef</em>. The <em>TraceItem</em> elements become the inputs to the <em>MethodDef</em>. The method either transforms the inputs or uses the inputs in a derivation such that the output of the method provides the source for the <em>ItemDef</em>.</td>
</tr>
<tr>
<td>4</td>
<td><em>Origin Type</em>=&quot;CRF&quot; variables require a <em>TraceItem</em></td>
<td>A variable with <em>Origin Type</em>=&quot;CRF&quot; must include a <em>TraceItem</em> element that references the source variable. This variable must be reachable from its source.</td>
</tr>
<tr>
<td>5</td>
<td><em>Origin Type</em>=&quot;Derived&quot; variables require a <em>TraceItem</em></td>
<td>A variable with <em>Origin Type</em>=&quot;Derived&quot; must have an associated <em>MethodDef</em>, and must include the <em>TraceItem</em> elements that reference the source variables. These source variables become inputs to the <em>MethodDef</em>, and the output of the <em>MethodDef</em> becomes the input to the &quot;Derived&quot; variable. Alternatively, a <em>MethodDef</em> may take no source variable parameters and in this case <em>Origin</em> includes the attribute trc:NoTraceItems=&quot;Yes&quot;.</td>
</tr>
<tr>
<td>6</td>
<td><em>Origin Type</em>=&quot;Predecessor&quot; variables require a <em>TraceItem</em></td>
<td>A variable with <em>Origin Type</em>=&quot;Predecessor&quot; must include a <em>TraceItem</em> element that references the source variable. This variable must be reachable from its source.</td>
</tr>
<tr>
<td>#</td>
<td>Rule</td>
<td>Definition</td>
</tr>
<tr>
<td>----</td>
<td>----------------------------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>7</td>
<td>Origin Types in [&quot;Protocol&quot;, &quot;Assigned&quot;, &quot;eDT&quot;] must not include a TraceItem</td>
<td>No traceability is expected for these variables based on the Origin Type. They are tested for traceability only since reachability will not exist, and these variables are expected to terminate the trace.</td>
</tr>
<tr>
<td>8</td>
<td>Derived variables referencing a MethodDef without input parameters must specify NoTraceItems</td>
<td>When a MethodDef referenced by a &quot;Derived&quot; variable does not define input parameters, then the Origin must set the attribute NoTraceItems=&quot;Yes&quot; to indicate that there will be no arguments referenced as source variable inputs to the method. This variable is tested for traceability only since reachability will not exist beyond the MethodDef due to the lack of source argument inputs.</td>
</tr>
<tr>
<td>9</td>
<td>Origin Type=&quot;Assigned&quot; variables require a comment</td>
<td>A variable with an &quot;Assigned&quot; Origin Type must have an associated CommentDef. This is checked by the conformance rules and is not part of the current version of the traceability graph.</td>
</tr>
<tr>
<td>10</td>
<td>MethodDef elements may not be reachable</td>
<td>MethodDef elements may not be reachable. MethodDef elements that are referenced by at least one variable are traceable.</td>
</tr>
<tr>
<td>11</td>
<td>FormDef and ItemGroupDef elements are not reachable</td>
<td>ItemGroupDef and FormDef elements are not reachable. ItemGroupDef and FormDef elements are traceable if they reference at least one variable.</td>
</tr>
<tr>
<td>12</td>
<td>FormDef, ItemGroupDef, and ValueListDef elements reference the variable nodes included in the graph</td>
<td>The trace graph is implemented by tracing variables defined as part of ItemGroupDef and ValueListDef elements. ItemDef elements not referenced by an ItemGroupDef or ValueListDef are not included in the trace graph. Within ODM only, ItemGroupDef elements must be referenced by a FormDef element to be included in the trace graph.</td>
</tr>
<tr>
<td>13</td>
<td>ValueListDef elements are not graph nodes</td>
<td>ValueListDef elements are processed when referenced by a variable, but are not added as nodes on the trace graph.</td>
</tr>
<tr>
<td>14</td>
<td>ItemDef elements referenced by a ValueListDef element are reachable and traceable</td>
<td>The ItemDef elements referenced by a ValueListDef should include the Origin element, as well as TraceItem elements where appropriate based on the value of the Origin Type attribute. In Define-XML, it is appropriate to include a traceable Origin Type on the ItemDef that references the ValueListDef. However, in Trace-XML it is more accurate to include the traceable Origin Type and TraceItem elements on the value level ItemDef.</td>
</tr>
<tr>
<td>#</td>
<td>Rule</td>
<td>Definition</td>
</tr>
<tr>
<td>----</td>
<td>---------------------------------------------------------------------</td>
<td>------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>15</td>
<td>Variables with <em>ValueLists</em> must not include a <em>TraceItem</em></td>
<td><em>ItemDef</em> elements that reference a <em>ValueListDef</em> should not themselves include a traceable <em>Origin Type</em> as the traceability should be defined by the value list <em>ItemDef</em> elements. <em>ItemDef</em> elements that include a <em>ValueListDef</em> may use an <em>Origin Type</em>=&quot;Assigned&quot;, and the assignment will be provided by the value list.</td>
</tr>
<tr>
<td>16</td>
<td>Variables that reference a <em>ValueListDef</em> must not reference a <em>MethodDef</em></td>
<td>Items may include a reference to a <em>MethodDef</em> or to a <em>ValueListDef</em>, but not both. <em>MethodDef</em> elements can be referenced from the <em>ItemRef</em> elements that are defined within a <em>ValueListDef</em>, and in this way methods can be referenced from a variable that has a value list as part of its definition.</td>
</tr>
<tr>
<td>17</td>
<td>A variable may reference source variables from the same lifecycle phase</td>
<td>Variables in one clinical research lifecycle phase may reference other variables in the same phase, as well as variables in previous lifecycle phases. For example, an ADaM <em>ItemDef</em> may include a <em>TraceItem</em> that references another ADaM <em>ItemDef</em>. The Trace-XML algorithms back track to find references to source variables that may not have been added to the graph when the original <em>ItemDef</em> is processed.</td>
</tr>
</tbody>
</table>

Trace-XML graphs are variable-centric meaning that traceability exists between variables, or *ItemDef* elements, and therefore many graph edges connect one variable to another. In the case of value level metadata, the value level *ItemDef* functions as a virtual variable within the graph. Trace-XML graph nodes are created using *ItemDef*, *MethodDef*, *ItemGroupDef*, and *FormDef* elements. *MethodDef*, *FormDef*, and *ItemGroupDef* elements may not be reachable in the traceability graph due to the *ItemDef*-centric nature of the graph. However, each of these node types is traceable since they have relationships to variables, such as functioning as a collection element to organize variables into a CRF or dataset.

Trace-Query also adheres to requirements that dictate how the trace query is displayed in a tabular report format. Trace-Query executes a recursive DFS search on the Trace-XML graph, and this search returns the nodes in an order that matches the graph traversal, but may not seem correct when the nodes are listed in a sequential report. The
Trace-Query display rules, listed in Table 14, adjust the nodes returned from the query to better suit the needs of a traceability report.

<table>
<thead>
<tr>
<th>#</th>
<th>Rule</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Redundant nodes are removed</td>
<td>Redundant nodes are removed so that each node is displayed only once. For example, two different variables may be a part of the same ItemGroup or two ItemGroup elements may be a part of the same form, but in each case the element is displayed only once per report.</td>
</tr>
<tr>
<td>2</td>
<td>Nodes are ordered by lifecycle phase</td>
<td>The nodes in the report are grouped by clinical research lifecycle phase. In cases where a variable has multiple inputs from different lifecycle phases, content from the different phases may be interleaved in the sequential query results. Lifecycle phase grouping ensures that analysis content comes before tabulation content which comes before data collection content in the query report. However, it is important to note that within each lifecycle phase the nodes remain in graph traversal order.</td>
</tr>
<tr>
<td>3</td>
<td>Fully qualified TraceItem elements filter unwanted ItemGroup or FormDef elements</td>
<td>A fully qualified TraceItem element includes the ItemGroupOID and, if relevant, the FormOID in addition to the ItemOID. For the tabulation and data collection phases, these additional references work as a filter and eliminate ItemGroup and FormDef elements that are not referenced by a TraceItem. For example, this filter eliminates the case where a common ItemGroupDef containing key variables, such as SiteID and SubjectID, is referenced in many forms, but the trace should only display one specific form. This filter is optional.</td>
</tr>
</tbody>
</table>

**Technologies Used in the Prototype Implementation**

The prototype software application was developed in Java to implement Trace-XML including the creation of the traceability graph and the algorithms for querying and validating traceability while identifying any gaps. JDOM 2 was used to process the XML in the Java
application. The BaseX 8.5.2 XML database engine XQuery 3.1 processor was used to implement the traceability query tool. The Define-XML extension was implemented in XML schema. The CDASH and SDTM standards were retrieved from the CDISC SHARE MDR, as were the CDASH to SDTM mapping relationships. The traceability graph is represented using the GraphML v1.0 schema. The Trace-XML prototype discussed in this paper rendered GraphML for two open-source graph visualization and editing tools: yEd v3.1.6 and Gephi v0.9.1. A detailed listing of the source code files created as part of the Trace-XML prototype are listed in Table 15.

<table>
<thead>
<tr>
<th>Source File (Application)</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tracexml.java (Trace-XML)</td>
<td>Java (main)</td>
<td>Taking as inputs the CDISC ODM and Define-XML files representing the metadata for a full study, Trace-XML uses a Define-XML extension to generate a full life-cycle study graph that provides computable traceability. In addition to validating traceability, the Trace-XML graph can be used to query traces for a variable or to visualize traceability with a visual rendering of the graph.</td>
</tr>
<tr>
<td>DFS.java (Trace-XML)</td>
<td>Java</td>
<td>Depth-First Search (DFS) class - searches the trace graph to establish node reachability.</td>
</tr>
<tr>
<td>CdiscGraphML.java (Trace-XML)</td>
<td>Java</td>
<td>CdiscGraphML generates a GraphML version of the internal graph built from the ODM and Define-XML files.</td>
</tr>
<tr>
<td>DefineGraph.java (Trace-XML)</td>
<td>Java</td>
<td>Generates a graph from the Define-XML metadata. Assumes that the XML files are loaded in life-cycle phase order. Assumes that OIDs are unique within a study.</td>
</tr>
<tr>
<td>Digraph.java (Trace-XML)</td>
<td>Java</td>
<td>The Digraph class represents the directed graph built from the ODM and Define-XML files.</td>
</tr>
<tr>
<td>Display.java (Trace-XML)</td>
<td>Java</td>
<td>The Display class contains static methods used to generate output listings, reports, and console messages.</td>
</tr>
<tr>
<td>ODMGraph.java (Trace-XML)</td>
<td>Java</td>
<td>The ODMGraph class generates the directed graph from the ODM metadata.</td>
</tr>
<tr>
<td>Origin.java (Trace-XML)</td>
<td>Java</td>
<td>The Origin class contains constants for the origin types and the tests determining whether or not a specific origin is traceable.</td>
</tr>
<tr>
<td>Source File (Application)</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>-----------</td>
<td>--------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Phase.java (Trace-XML)</td>
<td>Java</td>
<td>The Phase class contains the constants for the life-cycle phase text.</td>
</tr>
<tr>
<td>TraceNode.java (Trace-XML)</td>
<td>Java</td>
<td>TraceNode shares code with the TraceQuery application to trace a graph fragment to its conclusion.</td>
</tr>
<tr>
<td>ValidateXml.java (Trace-XML)</td>
<td>Java</td>
<td>The ValidateXml class runs schema validation on the XML files.</td>
</tr>
<tr>
<td>Vertex.java (Trace-XML)</td>
<td>Java</td>
<td>The Vertex class represents the node objects that comprise the graph generated from the ODM and Define-XML metadata files.</td>
</tr>
<tr>
<td>XsltTrace.java (Trace-XML)</td>
<td>Java</td>
<td>The XsltTrace class transforms the XML outputs of Trace-XML.</td>
</tr>
<tr>
<td>TraceQuery.java (Trace-Query)</td>
<td>Java (main)</td>
<td>TraceQuery queries the Trace-XML graph using XQuery to produce a full life-cycle trace for any given variable OID.</td>
</tr>
<tr>
<td>XsltTrace.java (Trace-Query)</td>
<td>Java</td>
<td>XsltTrace runs style sheets to transform XML output into html and other formats.</td>
</tr>
<tr>
<td>ConfigReader.java (Trace-Query)</td>
<td>Java</td>
<td>ConfigReader loads the contents of the Trace-XML configuration file and provides accessors to retrieve the information.</td>
</tr>
<tr>
<td>TraceKB.java (Trace-KB)</td>
<td>Java (main)</td>
<td>TraceKB generates a CDISC standards metadata knowledge base that includes the mappings between standards. These TraceKB uses these metadata mappings to add the Trace-XML extension content to Define-XML XML files.</td>
</tr>
<tr>
<td>ShareAPIClient.java (Trace-KB)</td>
<td>Java</td>
<td>ShareAPIClient uses the SHARE API to build the Trace Knowledge Base of links between CDISC standards elements.</td>
</tr>
<tr>
<td>XsltTrace.java (Trace-KB)</td>
<td>Java</td>
<td>XsltTrace runs style sheets to transform XML output into html and other formats.</td>
</tr>
<tr>
<td>ConfigReader.java (Trace-Query)</td>
<td>Java</td>
<td>ConfigReader loads the contents of the Trace-XML configuration file and provides accessors to retrieve the information.</td>
</tr>
<tr>
<td>trace-1-0-0.xsd (Define-XML extension)</td>
<td>XML Schema</td>
<td>Primary schema for the Trace-XML Define-XML extension.</td>
</tr>
<tr>
<td>trace-extension.xsd (Define-XML extension)</td>
<td>XML Schema</td>
<td>The Trace-XML extension to Define-XML imports the define-extension schema and the trace-ns schema, and redefines the define-ns schema.</td>
</tr>
<tr>
<td>trace-ns.xsd (Define-XML extension)</td>
<td>XML Schema</td>
<td>The Trace-XML extension v1.0.0 namespace schema. Trace-XML extends Define-XML to explicitly reference source items in</td>
</tr>
<tr>
<td>Source File (Application)</td>
<td>Type</td>
<td>Description</td>
</tr>
<tr>
<td>--------------------------</td>
<td>------</td>
<td>-------------</td>
</tr>
<tr>
<td>Origin in order to generate a directed graph from the ODM and Define-XML metadata for a study.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>trace-get-target.xql (Trace-KB)</td>
<td>XQuery</td>
<td>Find the mapping target for generating the Trace-XML extension content from the SHARE knowledge base.</td>
</tr>
<tr>
<td>trace-get-source.xql (Trace-KB)</td>
<td>XQuery</td>
<td>Find the mapping source for generating the Trace-XML extension content from the SHARE knowledge base.</td>
</tr>
<tr>
<td>trace-get-edges.xql (Trace-KB)</td>
<td>XQuery</td>
<td>Generates a report of the missing edges added to the Define-XML file using the Trace-XML extension.</td>
</tr>
<tr>
<td>trace-node.xql (Trace-XML, Trace-Query)</td>
<td>XQuery</td>
<td>Recursive node traversal query that runs on the GraphML file and return the nodes in the trace for a given variable.</td>
</tr>
<tr>
<td>trace-node-unique.xql (Trace-XML, Trace-Query)</td>
<td>XQuery</td>
<td>Returns unique set of nodes returned from a trace.</td>
</tr>
<tr>
<td>trace-node-origin.xql (Trace-XML)</td>
<td>XQuery</td>
<td>Returns the Origin value for each node in the trace.</td>
</tr>
<tr>
<td>trace-node-oid.xql (Trace-XML, Trace-Query)</td>
<td>XQuery</td>
<td>Uses the results of the GraphML trace look up the nodes in the appropriate XML file including the file name and path.</td>
</tr>
<tr>
<td>trace-node-filters.xql (Trace-Query)</td>
<td>XQuery</td>
<td>Returns a listing of nodes that removes ItemGroupDef and FormDef nodes that are not referenced in a TraceItem element for the tabulation and data collection lifecycle phases.</td>
</tr>
<tr>
<td>trace-kb.xql (Trace-KB)</td>
<td>XQuery</td>
<td>Builds initial trace knowledge base using metadata from the SHARE API.</td>
</tr>
<tr>
<td>trace-kb-edge.xsl (Trace-KB)</td>
<td>XSLT</td>
<td>Transforms the Trace-XML extension edges added to the Define-XML files into and HTML report of the edges added.</td>
</tr>
<tr>
<td>unreachable.xsl (Trace-XML)</td>
<td>XSLT</td>
<td>Transforms the list of unreachable nodes in the graph into HTML.</td>
</tr>
<tr>
<td>unreachable-txt.xsl (Trace-XML)</td>
<td>XSLT</td>
<td>Transforms the list of unreachable nodes in the graph into a tab delimited text file.</td>
</tr>
<tr>
<td>trace-node.xsl (Trace-Query)</td>
<td>XSLT</td>
<td>Transform the list of nodes returned from a trace query and their associated detailed metadata into an HTML report that displays the full trace for a variable.</td>
</tr>
</tbody>
</table>
| text-trace-node.xsl (Trace-Query) | XSLT | Transform the list of nodes returned from a trace query and their associated detailed metadata into a tab delimited text file report that
<table>
<thead>
<tr>
<th>Source File (Application)</th>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>can be opened using spreadsheet software, such as Microsoft Excel, to display the full trace for a variable.</td>
</tr>
</tbody>
</table>
CHAPTER 5

THE ANALYTICAL EVALUATION OF TRACE-XML

This chapter provides an analytical evaluation of the Trace-XML Framework DSR artifact. The analytical evaluation provides the primary means of establishing the artifact's efficacy. It uses the Trace-XML graph model and the DFS-based algorithms to demonstrate the validity of Trace-XML's claims of establishing traceability within clinical research metadata. The analytical approach is a static analysis (Hevner et al., 2004) used to ensure that reachability, traceability, and completeness can be accessed using the Trace-XML prototype. The analytical evaluation of Trace-XML is complemented by a qualitative assessment of utility described in Chapter 6, and a set of test cases created for use during the design-cycle.

Creating a Simple IP-Map to Evaluate Trace-XML

To test the Trace-XML Framework several test studies were created with increasing complexity. Each test study consisted of three CDISC standard XML files: (1) a CDASH ODM v1.3.2 file, (2) an SDTM Define-XML v2.0 file, and (3) an ADaM Define-XML v2.0 file. A one domain test study was implemented as the simplest possible study that enabled full lifecycle testing of the Trace-XML Framework algorithms as implemented in the Trace-XML prototype software. The detailed analysis of the Trace-XML algorithms that follows uses just the CDASH ODM and the SDTM Define-XML metadata phases of this simple one domain test. This small test is sufficient as it fully covers the Trace-XML algorithms. Extending the test to additional phases in the lifecycle simply repeats the same algorithms. The CDISC CDASH standard, often represented using ODM, describes the basic data collection fields for the data domains used in the study. Both the CDASH and ODM standards are widely supported by the EDC systems used to collect patient data during a study. Define-XML provides the metadata describing the content of the CDISC SDTM and ADaM tabular datasets.
The simple IP-Map in **Figure 28** represents the first two phases of the one domain test study used to analyze the Trace-XML functionality. **Figure 28** shows the transformation of one CDISC domain from the CDASH data collection standard to the SDTM standardized tabulation standard. This represents two of the essential phases of the overall clinical research data lifecycle using the CDISC standards. The detailed CDISC standard metadata for this test study was retrieved from the SHARE MDR formatted as ODM and Define-XML. This test stops after creating submission ready SDTM datasets. Additional tests that match the regulatory submission requirements, and go beyond the simplified test described here, include the ADaM, or analysis, datasets that are created using the SDTM data as an input. ADaM metadata is also described using Define-XML.

![Figure 28. Simplified IP-Map for creating a submission-ready dataset](image)

The simplified standards-based IP-Map for the CDISC demographics (DM) domain is shown in **Figure 28**. The two data sources are DS₁ and DS₂. DS₁ provides the metadata describing the CRFs used for data collection, as well as the data collected from patients at the research site. DS₁ contains CDISC CDASH demographics CRF data and metadata represented using the ODM standard and represents the source metadata and data for the transformation described in P₁. DS₂ contains the metadata describing CDISC SDTM dataset content represented using the Define-XML standard and represents the target format for the subsequent transformation described by P₁.

DS₁ and DS₂ provide the raw data inputs RD₁ and RD₂, respectively, to P₁. P₁ performs the transformations converting the subject demographics CRF data in DS₁ into a standardized SDTM dataset described by DS₂ metadata. Component Data outputs CD₁ and CD₂ flow from P₁ into data stores STO₁ for the Define-XML file that contains the updated SDTM metadata, and STO₂ for the Dataset-XML file that contains the subject demographic
dataset. These two files provide the Component Data inputs CD3 and CD4 that flow into process P2. Process P2 converts them into the submission-ready SDTM demographics dataset information product, IP1, for use by the statistical programmers. This simplified map provides a sufficient sample to test the traceability available in the current standards model, as well as demonstrating IP-Map’s usefulness as a tool for establishing traceability as a measure of data quality.

**Testing Traceability in Trace-XML**

Testing begins at layer 1 within the Trace-XML Framework where traceability must be confirmed in the IP-Map. The reachability and traceability algorithms used in this research are based on those described in Sedgewick and Wayne (2011) and Shankaranarayan et al. (2003). In order to prove that traceability exists within the IP-Map, IP-Map blocks must be mapped onto a corresponding IP-Graph. Any IP-Map x can be represented as a unique IP-Graph G(N, L) where each node \( n \in N \) represents a block in \( x \) and each flow \( e \in L \) represents an ordered pair \((n, m)\) where \( n, m \in N \). By definition, each block in the IP-Map has a unique, non-null identifier. Thus, each node in the IP-Graph is also unique. Each IP-Map flow is represented by a directed edge linking two adjacent nodes in the IP-Graph making G(N, L) a directed graph, or digraph. In this case, the digraph is a directed acyclic graph (DAG) because none of the directed paths have the same first and last node. Because each node \( n \) is unique and G(N, L) is a DAG, then each ordered pair \((n, m)\) is also unique. To establish traceability within the IP-Map from CB1 back to DS1 in the IP-Map, the IP-Graph must be used to demonstrate that the node \( n \) representing the CB1 block is reachable from node \( m \) representing the data source block DS1. Block CB1 is reachable from block DS1 if in G(N, L) there exists a path from node \( m \) to node \( n \) indicating there exists a sequence of stages within the IP-Map that constitute a path from block DS1 to block CB1.

The techniques for establishing reachability within a DAG will be discussed in the next section, but for the simplified graph in Figure 28 reachability from data sources DS1 and DS2 through to the information product CB1 can easily be established by inspection. Since the IP-Map diagram was constructed at the conceptual level to serve as an analysis tool, traceability is expected to exist. However, regulatory requirements for clinical research
demand a more granular accounting of traceability to include each data element in a submission rather than the higher level conceptual flow of data. In order to establish traceability as an attribute of data quality for the IP-Map, a more detailed analysis must be conducted.

IP-Maps are developed top-down using blocks and flows and then the detailed metadata is added for each block. However, IP-Map traceability must be validated from the bottom-up. In order to validate that the flows linking the blocks in the IP-Maps accurately represent the data flows, a detailed analysis of the data flows within the attribute-level content must be performed. Validating traceability within the attributes follows the same process as validating the higher-level IP-Map content, except in graph G(N, L) the nodes are attribute-level data elements and the edges are the data flows between the data elements. To prove traceability exists within the attributes reachability, traceability and completeness must be established in the attribute-level IP-Grapha.

Creating the Attribute-level Graph

To create the attribute-level graph and test for it reachability the algorithms used at the IP-Map level must be applied (Sedgewick & Wayne, 2011; Shankaranarayan et al., 2003) to the attributes found within the blocks in the IP-Map. To prove that reachability exists within the set of attributes that comprise the IP-Map blocks we first mapped the attributes in the attribute-level IP-Mapa onto the corresponding attribute graph IP-Grapha. Any attribute level IP-Mapa can be represented as a unique IP-Grapha Ga(Na, La) where each node n ∈ Na represents an attribute a within block b in the IP-Map, and each link e ∈ La is defined by the ordered pair (n, m) where n, m ∈ Na. Each block and each attribute within the block has a unique, non-null identifier. Each flow in the IP-Map is a directed edge that represents the flow of the data elements in Ga. Thus, Ga is a DAG that reflects the direction of the higher-level IP-Map flows. The process of mapping each attribute of a block in the IP-Map to a node in Ga generates mapping set M. Each member of M is an ordered pair <a, n> where a ∈ B and n ∈ Na. Because every block and attribute within the IP-Map has a unique, non-null identity each corresponding node in graph Ga is unique, as is the mapping set M.
Testing for Reachability in the Attribute-level Graph

After creating the IP-Graph\textsubscript{a} from the IP-Map\textsubscript{a}, establishing reachability is the next step in verifying traceability. Given the attribute-level digraph G\textsubscript{a} generated previously, node m is reachable from node n in G\textsubscript{a} if there exists a directed path from n to m. A depth-first search (DFS) algorithm for graphs will identify all and only those nodes reachable from a given node n in the digraph G\textsubscript{a} (Sedgewick & Wayne, 2011; Shankaranarayan et al., 2003). Given a start node, DFS marks every node it encounters as reachable. When no more nodes are reachable down a particular path, DFS backtracks and proceeds down the next link in the DAG. The DFS terminates when no more outbound links are accessible from the start node. The DFS marks all reachable nodes from the starting node, and unmarked nodes are considered unreachable.

If G\textsubscript{a} node F\textsubscript{i} in block DS\textsubscript{1} is the start node, Figure 29 shows the reachable nodes identified using DFS. Figure 29 shows that the DFS did not find edges reaching outside of DS\textsubscript{1}, and therefore only DS\textsubscript{1} nodes were marked. To improve readability, Figure 29 represents the graph for just the first 3 blocks of the IP-Map shown in Figure 28 (DS\textsubscript{1}, DS\textsubscript{2}, and P\textsubscript{1}), and yet immediately it is clear that CB\textsubscript{1} is not reachable from DS\textsubscript{1} since P\textsubscript{1} is not reachable from DS\textsubscript{1}. In other words, no path exists connecting the DS\textsubscript{1} attributes to P\textsubscript{1}. This has two important implications: (1) the RD\textsubscript{1} flow indicated from DS\textsubscript{1} to P\textsubscript{1} in the IP-Map is not supported by the attribute-level graph, and reachability and traceability do not exist in the attribute metadata; and (2) that the standards metadata used to represent the IP-Map attributes are incomplete and do not support the reachability or traceability drawn in the IP-Map. The DFS algorithm demonstrates that more metadata is needed to represent the complete flow of the data elements and establish reachability from DS\textsubscript{1} to P\textsubscript{1} and beyond. Essentially, this reachability test demonstrates that the current metadata standards operate within one phase of the clinical research data lifecycle and are missing the metadata needed to establish full lifecycle traceability that functions across lifecycle phase boundaries. This test demonstrates the siloed nature of the existing metadata standards.
Identifying the Reachability Gaps

The outbound flow $RD_1$ from $DS_1$ to $P_1$ in the IP-Map shown in Figure 28 indicates the expectation of reachability from $DS_1$ to $P_1$. Executing the DFS algorithm on the graph $G_a$ demonstrates that reachability does not exist from $DS_1$ to $P_1$. In order to establish reachability at the attribute level, additional metadata is needed. The siloed metadata representing each phase of the clinical research data lifecycle cannot be explicitly linked to metadata in the next phase of the lifecycle using the current standards. The existing CDISC standards specifications do not provide the metadata needed to establish the links between the attributes in $DS_1$ and $P_1$. However, during the development of the CDISC SHARE MDR (CDISC, 2016) these attribute-level links were instantiated as $maps$-$to$ relationships in the SHARE MDR model. As an input to the creation of the additional metadata needed for reachability, these relationships were extracted from SHARE in the form of a mapping table. Where the relationships were not available in SHARE, they were manually created and added to the mapping table. A graph $G_s$ was created from this mapping metadata. The two graphs $G_a$ and $G_s$ were then compared to identify the reachability gaps in the existing metadata standards as expressed in $G_a$. The algorithm shown in Figure 30 was used to find the missing edges that prevented reachability from being established.
Equipped with the hash table generated using the SHARE MDR relationships and the algorithm shown in Figure 30, the missing edges must next be inserted into the graph \( G_a \). The SHARE relationships do not link the source data in \( DS_1 \) directly to \( P_1 \), but instead link \( DS_1 \) into the \( DS_2 \) target metadata. That is, the mapping table shows which \( DS_1 \) data elements map to specific \( DS_2 \) data elements without representing the process needed to perform the transformation or mapping. The IP-Map indicates that every \( DS_1 \) attribute should flow to \( P_1 \). The \( P_1 \) transformation processes include data transformations, derivations, and direct mapping to the output variables described by \( DS_2 \). As \( P_1 \) transforms \( DS_1 \) into outputs described by \( DS_2 \) they flow as component data \( CD_1 \) and \( CD_2 \) to create output files \( STO_1 \) and \( STO_2 \). In order to support the process depicted in the IP-Map, the relationships provided by SHARE must be interpreted to accurately represent the processes in \( P_1 \). The algorithm performing this interpretation is shown in Figure 31.

The algorithm in Figure 31 took the missing metadata identified using the algorithm in Figure 30, and used it to create the missing edges and nodes needed to update the graph \( G_a \). As a result of this process, an additional gap in the standards metadata was discovered. In a separate test case that included the CDISC adverse events (AE) domain, an attribute was discovered with no relationship to the outputs described in \( DS_2 \). This AE attribute also has no relationship to the metadata in \( P_1 \) and therefore appears to be a terminal attribute. In this case, data management used this data element as a flag to confirm that no adverse events exist. Otherwise, data managers cannot know if a subject experienced no adverse events, or adverse events exist but the study site had not yet entered the data for them. In order to more completely represent traceability, this data management data element prompted the implementation of a new rule that requires every attribute to reach a data sink or be identified as an exception. For this case, a new data management IP would be created, and additional blocks would be added to the IP-Map. Once the IP-Map was updated, new graphs would

---

**Figure 30. Identifying the metadata missing from the attribute graph**

Given the IP-Map attribute graph \( G_a(N, L) \)
Given SHARE mapping attribute graph \( G_s(N, L) \)
For every edge \( e \in L_s \) represented by a directed pair of nodes \( (n, m) \) in \( G_s \)
   if \( (\pi \in N_a \) and \( \pi \in N_a \) and \( e \in L_a \) then
     insert \( e \) into a hash table as the directed pair of nodes \( (n, m) \)
need to be generated to include the necessary nodes and edges required to support reachability to that IP. Alternatively, certain attributes will be flagged as not traceable because they represent known exceptions. For example, attributes with certain Origin values, such as “Assigned”, will be identified as traceability exceptions since data is assigned to the variable rather than flowing from a data source in the previous phase of the data lifecycle. In cases where the source variable does not flow to the next phase an additional attribute will be added to flag the variable as a known traceability exception. Additionally, variables with an Origin Type of "Assigned" are expected to have a CommentDef element associated with them that further explains the source of the variable content.

![Algorithm to create the missing metadata in the attribute graph](image)

**Figure 31. Algorithm to create the missing metadata in the attribute graph**

**Resolving the Traceability Gaps in the CDISC Standards Metadata**

The prototype Trace-XML algorithms revealed metadata missing from the CDISC standards needed to establish computable traceability in full lifecycle regulatory submission datasets. The current standards address traceability using descriptive metadata. For example, an SDTM variable with an Origin Type of "CRF" would include a link to an annotated blank CRF in PDF format, and an ADaM variable with an Origin Type of "Predecessor" would provide text with the domain and variable name of the source variable. This descriptive metadata provides limited traceability information. Computable traceability that supports a query capability would provide a solution that more completely addresses the regulatory requirements. The gaps in the standards metadata were filled in the attribute-level graph using a combination of the model relationships available in the SHARE MDR, a graph difference
algorithm that identified the missing edges, and an algorithm that transcribed the relationship metadata to determine the nodes and edges needed to support the process shown in the IP-Map (Archambault, 2009).

Traceability has been a Define-XML requirement from the beginning and extending Define-XML with additional traceability metadata represents a natural enhancement to the existing specification and schema. ODM and Define-XML were designed to accommodate extensions, and Define-XML is a standardized extension of the ODM standard. Providing a Define-XML extension named Trace-XML makes the enhancements available for immediate use by the clinical research community. Figure 32 shows a small example of the extension in use. The new Trace and TraceItem elements containing the source variable references are identified using the trc namespace prefix used to classify Trace-XML extension content. The TraceItem element includes only the ItemOID attribute, the minimum required content to establish a reference to the source variable. This new Trace-XML content complements the existing descriptive metadata that shows the immediate source variable as part of the PDF annotated CRF document.

The ability to explicitly reference source variables enables the Trace-XML software to generate the edges that connect the variables, methods, datasets, sub-forms, and forms into a graph representation. For example, a variable with an Origin of Type “CRF” now includes the ItemOID to explicitly reference that variable’s metadata in addition to maintaining a link to the descriptive blank-crf.pdf file, as shown in Figure 32. In another case, a variable with an Origin Type of “Derived” must include references to the variables that operate as inputs to the derivation or transformation provided by the MethodDef associated with the variable, as shown in Figure 20. Using the TraceItem references, the Trace-XML software applies a recursive algorithm to link to the sources of all previous steps in the data flow. In addition to the Define-XML extension, ODM can use the existing Alias element to add the metadata needed to map CRF content into the SDTM datasets.
Resolving Reachability Gaps using Traceability

Certain variables will be unreachable because their source variables have an Origin Type that indicates that the variable is not traceable, such as "Assigned" or "Protocol". When testing for reachability, every variable that uses an "Assigned" variable as a source will be unreachable. "Assigned" origins represent known reachability exceptions, but variables that reference a source variable that has an Origin Type of "Assigned" are expected to be traceable. For example, if an SDTM variable is "Assigned", an ADaM variable should be reachable from the SDTM variable. In this way, we expect to trace from the ADaM variable back to the "Assigned" SDTM variable, but no further.

To resolve this dilemma, the Trace-XML algorithm re-tests unreachable nodes using traceability. Traceability for unreachable nodes is performed by reversing the graph direction, and testing for reachability in the opposite direction. Starting at the last phase of the lifecycle and working backward, a variable traces its lineage back to a source. In cases where a source variable has an Origin Type of "Assigned", the assigned variable is considered the origin and the trace is marked as valid. In this case, the "Assigned" variable represents the original source. If tracing the variable in the reverse direction through the lifecycle yields a node that should itself have a source, such as when the Origin Type of an SDTM variable is "CRF", then the trace should continue until if finds a terminal node or a gap in the expected traceability. Thus, Trace-XML tests the graph for reachability in both directions to create an accurate traceability assessment.
Re-testing for Reachability and Traceability in the Attribute-level Graph

After the gaps in the IP-Graph were identified and resolved, and the metadata standards were augmented with the additional metadata, the IP-Map was re-instantiated and the IP-Graph was again evaluated for reachability, traceability, and completeness. The evaluation process used the same algorithms as previously described with the expectation that the previously identified gaps had been resolved.

Using the demographics domain example, the missing edges and nodes were added to the graph G, and the DFS was executed to re-test for reachability. As shown in Figure 33, reachability was achieved for the DS1 to P blocks. Expanding the test to include DS2 yields all the nodes shown in Figure 34 as reachable. Expanding this process to the entire IP-Graph shows that reachability was established for DS1 and DS2 through to CB.

Figure 33. Re-testing for reachability from DS1 to P using the updated graph

Now that reachability has been achieved, we show that if node n in Na is reachable from node m, then node m is traceable from node n. In a DAG traceability is not achievable since all the edges are directed. The first step in proving traceability for an attribute-level IP-graph that demonstrates reachability is to create the transpose of Ga that reverses the direction of all links in the graph. We create G_a'(N, L_a') as the transpose of G_a(N, L_a). Each node in G_a
is also a node in the transpose graph $G'_a$. Each link in $G_a$ has the same endpoints in $G'_a$, but the direction of the link is reversed. Once the transpose graph $G'_a$ has been created, DFS is again used to show reachability. Since the nodes in $G_a$ and $G'_a$ are identical, and the links are identical except the direction is reversed, if a node is reachable in $G'_a$ it is traceable in $G_a$ (Shankaranarayan et al., 2003). Thus, by achieving reachability for the attributes in $DS_1$ to the attributes $CB_1$ we can show that the $CB_1$ attributes can be traced back to the $DS_1$ attributes.

Achieving reachability in IP-Graph$_a$ means we have reachability in IP-Map$_a$ which in turn confirms the flows drawn between the blocks in the IP-Map.

**Testing the Attribute-level Graph for Completeness**

This research project has defined completeness for the IP-Graph$_a$ $G_a(N, L)$ as the state where every node $n \in N$ is reachable from a data source block data element and traceable from a data sink data element, or it has been flagged as an exception. Other prerequisites for establishing completeness include: (1) the IP-Graph$_a$ has been created by a complete mapping of all the IP-Map composite attributes; (2) correctness in the XML metadata is confirmed; (3) the IP-Graph$_a$ is directed, and every pair of distinct nodes is connected by a unique directed edge that is reversible; and (4) coverage exists. Correctness in
the XML metadata is confirmed when the XML metadata passes both Trace-XML schema validation and the CDISC conformance rules created for compliance with the standards. These rules confirm that the resulting graph conforms to the typing, ordering, and impossibility constraints, among others. The Apache Xerces 2.11 XML parser was used to run schema validation against the trace-1-0-0.xsd XML schema, and the XML4Pharma Define.xml checker (aerts, 2017) was used to run the CDISC Define-XML v2.0 conformance rules. Coverage exists where all data sources are traceable from at least one IP output, and all IP outputs are reachable from at least one data source except for the known traceability exceptions that are flagged as such. As noted previously, additional metadata must be added to the existing industry standards to achieve completeness. Relevance was another dimension of provenance quality asserted by Cheah and Plale (2015) to complement correctness and completeness. In Trace-XML, the traceability metadata added to provide computable traceability is highly relevant as it extends the existing concept of Origin within Define-XML and significantly extends the existing traceability capabilities in clinical research to better meet the existing regulations.
CHAPTER 6
THE QUALITATIVE EVALUATION OF TRACE-XML

The qualitative evaluation of Trace-XML evaluated the utility of Trace-XML as a DSR artifact that addresses research objectives related to end-to-end traceability within the domain of clinical research. This study collected data by interviewing clinical research data experts that represent a range of stakeholder categories. The interviews followed a semi-structured format, and the data was analyzed using thematic analysis. This same approach was used as the primary means to evaluate a traceability framework for the business intelligence domain in a recent journal article (Chee et al., 2014). The Trace-XML project seeks to use this approach as a secondary means of evaluation that complements the primary analytical evaluation method. The following sections of this chapter cover the essential research design elements for the qualitative evaluation project (Myers, 2013), as well as the study results. A discussion of the implications of the study results is covered in Chapter 7.

Research Questions

In addition to gaining feedback to feed into future design cycles, this applied thematic analysis qualitative research project sought to explore the utility of Trace-XML to clinical research data experts. To better understand Trace-XML utility as it relates to different aspects of traceability, this research sought to address three exploratory research questions.

1. How does Trace-XML impact the task of assessing end-to-end traceability for variables within a clinical research study?
2. How does Trace-XML impact the task of assessing the validity of end-to-end traceability within a clinical research study?
3. What features of an end-to-end traceability solution are most important for the data reviewer tasks of assessing and validating traceability for variables within a clinical study?
Research Design

Research method

This qualitative study used Applied Thematic Analysis (Guest et al., 2011) as the research method for the evaluation of Trace-XML utility. This research method was applied using an interpretive philosophical approach, and applied inductive reasoning to explore the data generated by the interviews. The themes established by the analysis were driven by the data, and focused on the primary research questions. As a DSR evaluation tool, applied thematic analysis of the semi-structured interview data provided the method to assess the utility of the artifacts. This qualitative study represents applied research that seeks to understand and improve the solution to a practical problem, traceability within clinical research. This evaluation method is secondary to a static, analytical evaluation method used to prove the artifacts meet the project research objectives.

Data collection technique

Semi-structured interviews with clinical research data experts with a stake in clinical research study traceability were used to gather data for this study. These clinical research data experts represented 5 different categories of organizations. Each interview began with a detailed demonstration of Trace-XML. Eight questions were created in advance of the study to direct the interview and ensure the data collected would support an evaluation of the primary research questions. The questions were designed to assess the usefulness of Trace-XML to support traceability from a number of perspectives. The demonstration took approximately 15 minutes followed by a 30-40 minute interview. Most of the 8 questions included optional probing questions to further explore the primary questions. As a semi-structure interview with subjects from 5 different categories of organizations, each interviewee was given latitude to pursue relevant topics related to their areas of interest and expertise. This ensured a richer and more varied set of data to analyze, and provided unanticipated new features to feed into the next Trace-XML development design-cycle.
Data analysis approach

The qualitative data gathered from the interview process was analyzed using thematic analysis. Thematic analysis seeks to identify the explicit and implicit ideas and patterns, or themes, within the data. An interpretive philosophy was used during analysis to identify the underlying themes. In general, thematic analysis relies more heavily on interpretation than it does strict analytical techniques such as word counting and other algorithms for text analysis. Thematic analysis excels at capturing the "complexities of meaning" within the data (Guest et al., 2011) and builds themes using inductive reasoning. Thematic analysis is the most commonly used analysis method in qualitative research (Guest et al., 2011). The thematic analysis was performed using the QDA Miner Lite v2.0 qualitative analysis software (Provalis-Research, 2017).

Written record

The qualitative research methods described in (Myers, 2013) include a step for describing the written record. As part of the evaluation of the larger DSR project, this research has been documented in this dissertation. A journal article covering the full project including the evaluation steps will be authored, as well as a conference article that focuses on the qualitative evaluation of the DSR artifacts.

Sampling and Sample Size

The subjects for this research were selected from the categories of clinical data experts listed in Table 16 below. Each category represents a different type of organization giving the associated subjects different perspectives on traceability within the CDISC standards. These diverse viewpoints helped evaluate Trace-XML utility from a number of different perspectives and provided well-rounded input for new Trace-XML features. The sample size was 10 clinical data experts distributed evenly across the different categories. Each participant is a well acknowledged expert with at least 15 years of clinical research data experience. Randomized selection or assignment will not be used in this study. The subjects that participated in the study were recruited using email.
<table>
<thead>
<tr>
<th>Category</th>
<th>Description and Examples</th>
<th># of Subjects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Regulatory</td>
<td>Works for a government regulator</td>
<td>2</td>
</tr>
<tr>
<td>Standards developers</td>
<td>Members of CDISC teams that develop the standards.</td>
<td>2</td>
</tr>
<tr>
<td>Biopharmaceutical / Contract Research Organizations</td>
<td>Experts that work for a biopharmaceutical company or a contract research organization that implement the standards.</td>
<td>2</td>
</tr>
<tr>
<td>Technology vendors</td>
<td>Technology experts that work for an organization that develops and markets a technology that supports one or more parts of the clinical research data lifecycle.</td>
<td>2</td>
</tr>
<tr>
<td>Academic</td>
<td>Academics with experience applying the CDISC standards in academic clinical research.</td>
<td>2</td>
</tr>
</tbody>
</table>

**IRB Approval and Consent**

Institutional Review Board (IRB) approval was granted by the Dakota State University (DSU) IRB to conduct this research and work with human subjects. The project was granted expedited project approval as described in 45 CFR 46.110 on January 24th, 2017 with approval # 2016-2016-114 for a term of 1 year. The research activity was deemed to be no greater than minimal risk as defined in 63 FR 60364-60367. The DSU IRB approval letter is available for review in Appendix F.

A consent form was created for subjects participating in the study, and each subject received the form prior to the interview. The highlights of the consent form were covered with subjects at the outset of the interview, and each subject was given the opportunity to ask questions prior to beginning the demonstration. Each subject retained the right to withdraw from the study at any time. Subjects were not required to sign the consent form, and their agreement to participate in the study implied consent. Upon receiving IRB approval, a pilot interview was conducted to ensure the questions worked in the context of an interview and would provide the data needed to answer the research questions. The informed consent form is available for review in Appendix E.
The Interview Demonstration and Questions

After reviewing the informed consent, each subject was provided with an overview of the Trace-XML Framework followed by a demonstration of the Trace-XML software. This detailed information on Trace-XML was intended to educate the subject as well as to provide the necessary understanding of the Trace-XML Framework artifacts to answer detailed interview questions regarding the utility of Trace-XML to provide a solution to the real-world problem of traceability within clinical research. The investigator will also demonstrate the current state-of-the-art method for assessing traceability to facilitate a discussion on the relative utility of Trace-XML. The state-of-the-art traceability demonstration will be based on the current regulatory requirements for CDISC traceability in submissions, and will be confirmed by an industry expert. All subjects that participated in the study were familiar with the current state-of-the-art in traceability as well as the relevant regulations.

Traceability Definitions and Background

The following concepts were defined for each subject during the course of the demonstration:

- **Metadata traceability**: Trace-XML addresses the need for metadata traceability. Metadata traceability facilitates the understanding of the relationship between a variable to its source dataset(s) and variable(s), including a description of the algorithm used to derive or populate a variable from its predecessor.

- **Regulatory submission context**: This project focuses primarily on traceability in the context of a CDISC study as would be found in a typical FDA or PMDA submission.

- **End-to-end traceability**: End-to-end describes traceability from analysis datasets (or even analysis results) back to the original source data and covers the different phases of the clinical research data lifecycle.

Trace-XML Demonstration

Prior to the demonstration, a brief review of the current standard for assessing metadata traceability within a regulatory submission was provided, followed by an overview of Trace-XML. The Trace-XML Framework demonstration focused on the main attributes
and did not attempt to cover every feature. The features covered in the Trace-XML orientation and demonstration included:

1. Use the Define-XML v2.0 SDTM example to demonstrate how traceability is established in the context of a regulatory submission using the current state-of-the-art.
2. Present brief 7-slide PowerPoint introduction to the problem space and the Trace-XML Framework
3. Show the IP-Map overview of traceability
   a. Demonstrate possibility of including EHR as source content
   b. Demonstrate example used for the Trace-XML demonstration
4. Demonstrate Trace-XML Validation
   a. Show validation and the unreachable report
   b. Remove an item and re-generate the report
5. Demonstrate Trace-XML Query to show an end-to-end trace for one variable
   a. Generate a query trace for an analysis variable
   b. Show the trace query report added to a Define-XML document
6. Demonstrate Trace-XML graph visualizations
   a. Show the full study graph as an exploratory tool
   b. Show the predecessor graph

Questions on Trace-XML as an End-to-end Traceability Solution

The semi-structured interviews began by the investigator asking the prepared questions. However, the subject could direct much of the discussion towards relevant topics that most interested them. Not every interview spent the same amount of time on the 8 questions prepared for the interview, depending on how much the subject directed the discussion. Probe questions were also prepared as an aid to the investigator, but may or may not have been used depending on the specific interview.

1. Assuming Trace-XML has been implemented as a study review tool, what impact would Trace-XML have on assessing end-to-end traceability?
   a. [Probe] Do you think there are end-to-end traceability benefits that Trace-XML provides that are not available using the current methods?
b. [Probe] Assuming no implementation barriers, what impact would Trace-XML have?

2. Given what you know of regulatory requirements for traceability, what impact would Trace-XML have on the ability to meet those requirements?

3. What impact does the ability to visualize traceability have on assessing end-to-end traceability?
   a. [Probe] Do you think there are benefits that Trace-XML provides in support of end-to-end visualization that are not available using the current methods?
   b. [Probe] How does the visualizing study metadata using the full graph representation or predecessor view impact your understanding of study traceability?

4. What impact of does end-to-end traceability validation have on assessing study traceability?
   a. [Probe] Do you think there are benefits that Trace-XML provides in support end-to-end validation that are not available using the current validation methods?
   b. [Probe] How could Trace-XML impact how the industry thinks about study metadata validation?

5. What impact does an end-to-end traceability query feature have on a reviewer’s ability to assess traceability of a variable?
   a. [Probe] Do you think there are benefits that Trace-XML provides in support of traceability queries that are not available using the current methods?
   b. [Probe] Does including the query results for each variable in the Define-XML output improve the ease of reviewing end-to-end traceability for a variable?

6. What impact does the IP-Map have on understanding traceability within a study?
   a. [Probe] How does the higher-level view of traceability provided by the IP-Map diagram impact your understanding of the detailed traceability outputs such as the graph visualization or traceability queries?
   b. [Probe] Would the inclusion of an IP-Map diagram with Define-XML impact a data reviewer’s understanding of traceability within the study?

7. What features of a traceability solution are the most important?
a. [Probe] What makes the priority features more important than others?

b. [Probe] If implementation feasibility, money and time were not issues, describe the most important features of an ideal traceability solution.

8. Given what you have seen during the demonstration, what recommendations do you have for improving Trace-XML?

Analysis

Methods

Applied Thematic Analysis (Guest et al., 2011) was used within the context of the Trace-XML DSR project as the method for evaluating the utility of the artifacts. This qualitative study included 10 subjects with expertise in CDISC standards metadata taken from organizations in the 5 different categories described in Table 16. The scripted questions were asked during the semi-structured interview, but the subject was given the latitude to respond and comment in a manner that allowed them to cover the topics they felt were most relevant. The questions were designed to assess the usefulness of Trace-XML to support traceability from several perspectives. The questions themselves impacted the themes derived from the coded text, but did not alter the main substance of the responses.

The interviewer took detailed notes during the interview and refined the content just after the conclusion of the interview to remove ambiguous pronouns and any proper nouns that might identify the subject or the subject's organization. A response template was created to guide and structure the investigator's note taking during and immediately after the interview session. The detailed interview notes were coded using the QDA Miner Lite v2.0 software. The code table was developed, refined, and condensed over 4 iterations through the 10 interview cases loaded into the project. The initial and final coding iteration counts are listed in Table 17.

<table>
<thead>
<tr>
<th>Initial and Final Coding Phase</th>
<th>Categories</th>
<th>Codes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Code Book Development</td>
<td>9</td>
<td>49</td>
</tr>
<tr>
<td>Final Code Book</td>
<td>5</td>
<td>24</td>
</tr>
</tbody>
</table>

Table 17. Number of Categories and Codes
The text and codes related to each research question were isolated to identify themes and quotes specific to each of the 3 research questions, as shown in Table 18 in the Results section.

**Results**

This qualitative assessment of Trace-XML was completed to assess the usefulness of the Trace-XML framework to support full study lifecycle traceability. The variety of roles represented by the subject categories ensured a wide range of perspectives, while their shared background in clinical research data and the CDISC standards ensured common interests in improving traceability within clinical research.

The key themes and code categories identified during this evaluation were: *adds utility*, *adds information*, *simplifies implementation*, *verifies data quality*, and *promotes understanding*. The *adds utility* theme indicates that Trace-XML was viewed as adding utility to the demonstrated traceability tasks by the clinical research data experts. The *adds information* theme indicates that Trace-XML makes available and utilizes additional information not previously available to those assessing traceability. The *simplifies implementation* theme encapsulates those codes that mean that Trace-XML makes implementing traceability simpler through such individual codes as *building on existing technologies* such as the ODM and Define-XML standards or instilling a *tool focus* by providing a software tool that automates the traceability tasks under review. The *verifies data quality* theme indicates that Trace-XML provides data quality verification capabilities such as validation, a completeness assessment, and produces traceability reports and visualizations. The *verifies data quality* theme produced 12% of the total codes, accounting for the fewest codes of the major themes. Finally, the *promotes understanding* theme indicates that Trace-XML improves a user's understanding of their data and produced 27.9% of the total codes, the most of any major theme.

After the final coding iteration, the code frequency analysis shown in Table 18 was generated. The table shows the codes for each category, the number of times the code appears, the percentage of all codes applied represented by the code, the number of cases or interviews
where the code was applied, and the percentage of interviews where the code was applied. For example, the codes for *new information* (9.8%) created using Trace-XML and showing *how things fit together* (9.8%) each totaled nearly 10% of the total codes. Other codes, such as *mapping metadata* (0.8%) and information *re-use* (0.8%) represented less than 1% of the codes.

<table>
<thead>
<tr>
<th>Code Category</th>
<th>Code</th>
<th>Count</th>
<th>% Codes</th>
<th>Cases</th>
<th>% Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>adds utility</td>
<td>useful</td>
<td>21</td>
<td>7.9%</td>
<td>7</td>
<td>70%</td>
</tr>
<tr>
<td></td>
<td>role independent</td>
<td>6</td>
<td>2.3%</td>
<td>5</td>
<td>50%</td>
</tr>
<tr>
<td></td>
<td>regulatory impact</td>
<td>24</td>
<td>9.1%</td>
<td>10</td>
<td>100%</td>
</tr>
<tr>
<td></td>
<td>new capability</td>
<td>8</td>
<td>3.0%</td>
<td>6</td>
<td>60%</td>
</tr>
<tr>
<td></td>
<td>too advanced</td>
<td>3</td>
<td>1.1%</td>
<td>3</td>
<td>30%</td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td>62</td>
<td>23.4%</td>
<td>6.2</td>
<td>avg</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code Category</th>
<th>Code</th>
<th>Count</th>
<th>% Codes</th>
<th>Cases</th>
<th>% Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>adds information</td>
<td>new information</td>
<td>26</td>
<td>9.8%</td>
<td>9</td>
<td>90%</td>
</tr>
<tr>
<td></td>
<td>missing information</td>
<td>6</td>
<td>2.3%</td>
<td>6</td>
<td>60%</td>
</tr>
<tr>
<td></td>
<td>mapping metadata</td>
<td>2</td>
<td>0.8%</td>
<td>2</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>all dataset types</td>
<td>5</td>
<td>1.9%</td>
<td>1</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td>re-use</td>
<td>2</td>
<td>0.8%</td>
<td>2</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>complex</td>
<td>5</td>
<td>1.9%</td>
<td>3</td>
<td>30%</td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td>46</td>
<td>17.5%</td>
<td>3.8</td>
<td>avg</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code Category</th>
<th>Code</th>
<th>Count</th>
<th>% Codes</th>
<th>Cases</th>
<th>% Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>simplifies implementation</td>
<td>tool focus</td>
<td>6</td>
<td>2.3%</td>
<td>1</td>
<td>10%</td>
</tr>
<tr>
<td></td>
<td>simplify</td>
<td>13</td>
<td>4.9%</td>
<td>7</td>
<td>70%</td>
</tr>
<tr>
<td></td>
<td>embedded</td>
<td>4</td>
<td>1.5%</td>
<td>2</td>
<td>20%</td>
</tr>
<tr>
<td></td>
<td>builds on existing technology</td>
<td>12</td>
<td>4.5%</td>
<td>6</td>
<td>60%</td>
</tr>
<tr>
<td></td>
<td>generate</td>
<td>20</td>
<td>7.5%</td>
<td>8</td>
<td>80%</td>
</tr>
<tr>
<td>Totals</td>
<td></td>
<td>55</td>
<td>20.7%</td>
<td>4.8</td>
<td>avg</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Code Category</th>
<th>Code</th>
<th>Count</th>
<th>% Codes</th>
<th>Cases</th>
<th>% Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>verifies data quality</td>
<td>validation</td>
<td>17</td>
<td>6.3%</td>
<td>7</td>
<td>70%</td>
</tr>
<tr>
<td></td>
<td>completeness</td>
<td>5</td>
<td>1.9%</td>
<td>4</td>
<td>40%</td>
</tr>
</tbody>
</table>
### Table 34

<table>
<thead>
<tr>
<th>Code Category</th>
<th>Code</th>
<th>Count</th>
<th>% Codes</th>
<th>Cases</th>
<th>% Cases</th>
</tr>
</thead>
<tbody>
<tr>
<td>promotes understanding</td>
<td>comprehension</td>
<td>18</td>
<td>6.8%</td>
<td>8</td>
<td>80%</td>
</tr>
<tr>
<td></td>
<td>visual</td>
<td>19</td>
<td>7.1%</td>
<td>7</td>
<td>70%</td>
</tr>
<tr>
<td></td>
<td>identify differences</td>
<td>5</td>
<td>1.9%</td>
<td>3</td>
<td>30%</td>
</tr>
<tr>
<td></td>
<td>big picture</td>
<td>6</td>
<td>2.3%</td>
<td>5</td>
<td>50%</td>
</tr>
<tr>
<td></td>
<td>how things fit together</td>
<td>26</td>
<td>9.8%</td>
<td>8</td>
<td>80%</td>
</tr>
<tr>
<td></td>
<td><strong>Totals</strong></td>
<td><strong>74</strong></td>
<td><strong>27.9%</strong></td>
<td><strong>6.2\text{avg}</strong></td>
<td><strong>62%\text{avg}</strong></td>
</tr>
</tbody>
</table>

All the major themes identified can be related to the central theme of *adding utility*, or the concept that Trace-XML provides additional utility not currently available to address full lifecycle study traceability. Most of the codes deal directly or indirectly with adding utility. Utility may be added by improving comprehension or easing traceability implementation issues by building on existing technology. Thus, *adding utility* represents the primary theme in this study with the remaining themes of *adds information*, *simplifies implementation*, *verifies data quality*, and *promotes understanding* supporting the main theme. This hierarchical relationship is depicted in **Figure 35**.

![Figure 35. The Trace-XML thematic hierarchy](image)

The *adds utility* central theme directly supports the primary research objective to evaluate if clinical research data experts found that Trace-XML was a useful innovation to support the real-world problem of assessing traceability within the context of a clinical study. In addition to this overarching objective, this qualitative evaluation sought to explore the three research questions listed in the Research Questions section of this chapter. Each research
question is listed in Table 19 along with the salient themes and supporting example quotes from the interviews.

Table 19. Salient themes for each research question

<table>
<thead>
<tr>
<th>Research Question</th>
<th>Salient Themes</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>How does Trace-XML impact the task of assessing end-to-end traceability for variables within a clinical research study?</td>
<td>Presents new information</td>
<td>&quot;The Define-xml embedded piece is extremely useful. This is the most obvious first thing to get a quick win from Trace-XML because it's in the same format they're already using but now they get more information with 1-click.&quot;</td>
</tr>
<tr>
<td></td>
<td>Shows how things fit together</td>
<td>&quot;The Trace-XML query report provides a much clearer description than what is available today. The reviewers would definitely want this. Putting content into the Define-XML style sheet makes it simple to use and understand.&quot;</td>
</tr>
<tr>
<td></td>
<td>Aids traceability comprehension</td>
<td>&quot;Certainly useful information for regulators. It provides a seamless way for FDA reviewers to identify a sponsor's complete process. It shows the regulators the full picture for any variable.&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Today it's hard to pull traceability information together and you need the mapping files. Trace-XML is a lot easier. You can load the study and generate the content needed to understand traceability.&quot;</td>
</tr>
<tr>
<td>How does Trace-XML impact the task of assessing the validity of end-to-end traceability within a clinical research study?</td>
<td>Provides traceability validation</td>
<td>&quot;Taking variables and making sure that they're traceable. It adds a more sophisticated level of validation. Doesn't validate a silo, but an entire study.&quot;</td>
</tr>
<tr>
<td></td>
<td>Simplifies generating traceability content</td>
<td></td>
</tr>
</tbody>
</table>
Benefits regulators

"Useful to understand if the traceability is complete. Think it would be very useful for regulators receiving a submission."

"I think users should be interested in end-to-end quality. Validation is a good use of this technology. I would certainly want this. Makes it easy to trace everything. Does it in a simple way."

"This highlights dead points in a study in which data points are based on assumptions that are not documented – not linked into the overall study data flow."

<table>
<thead>
<tr>
<th>Research Question</th>
<th>Salient Themes</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>What features of an end-to-end traceability solution are most important for the data reviewer tasks of assessing and validating traceability for variables within a clinical study?</td>
<td>Shows <em>how things fit together</em></td>
<td>&quot;Knowing where the inputs to data come from, understanding the source.&quot;</td>
</tr>
<tr>
<td></td>
<td>Traceability validation</td>
<td>&quot;Establishing the sources for each variable.&quot;</td>
</tr>
<tr>
<td></td>
<td>Visualize traceability</td>
<td>&quot;Understanding the data flow through all their data.&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;The transparency provided by the end-to-end flow of information&quot;</td>
</tr>
<tr>
<td></td>
<td></td>
<td>&quot;Validation. Finding the gaps in a study trace.&quot;</td>
</tr>
</tbody>
</table>

The most codes were applied (27.9%) under the *promotes understanding* category which covers how Trace-XML improved a subject's ability to understand the flow of data through the clinical research data lifecycle, from data collection through analysis. The most frequent code applied in this category was *how things fit together* (9.8%) followed by *visual* (7.1%) and *comprehension* (6.8%). Trace-XML's potential use to *promote understanding* was a major theme that emerged from the interviews. A summary of the code category frequency totals is shown in Figure 36.
As noted earlier, the major themes identified from the interviews all support the overarching theme that Trace-XML *adds utility* to assessing traceability and understanding how study data fits together. Within the category of *adds utility*, the code *regulatory impact* was applied the most frequently (9.1%) and had the highest coverage as it was applied to 100% of the interviews. This indicates that *regulatory impact*, by adding utility for regulatory reviewers, was a key theme that emerged from the interviews. Every interview subject mentioned that Trace-XML should interest the regulators.

There were a wide range of responses to the question regarding the impact of the IP-Map layer within the Trace-XML framework on understanding traceability, but the most common code was *big picture* under the *promotes understanding* category. Other frequent codes were *visual* and *how things fit together* which also indicate that the IP-Map helps promote understanding. One subject summarized the impact of an IP-Map for traceability as:

*It would be useful to give someone the high-level view and to understand what CRFs contributed to the ADSL dataset. You may only have 1 or 2 key datasets for your primary or secondary endpoints and the IP-Map could help find the key analysis datasets and variables and how they were created and what the inputs were. This could be used to direct a search to find more detailed information.*
Another summed up the value of IP-Map this way:

_This could be a useful way to present traceability to business users. This information is known to the standards folks, but not explicitly documented for those that are less familiar with the information used to generate submission content._

The question regarding the impact of the graph visualization on understanding traceability was similarly varied. The *visual* code was most frequently applied, but *how things fit together* was also frequently applied and indicates that the visualization can help to improve one's understanding of the data. In this instance, there were some cautionary codes as well, including *complex* indicating that the graph was complex for the uninitiated to understand and *too advanced* for immediate use by less knowledgeable users.

_The visualization makes it easy to see where things came from. What derivations are done. Reviewers spend a lot of time figuring out what is this stuff how do these fit together._

Another subject identified the benefits as well as the impact of the complexity of the graph visualization:

*Useful information for those with a curious mind. Less immediately important than the tabular HTML visualization since it takes more work and knowledge to get something out of it.*

Subject's also had the opportunity to discuss what ideas or new features might improve Trace-XML. The two most common codes were *generate* and *new information*. For *generate*, subjects recommended that the additional information needed to support Trace-XML be automatically generated, as it can be done from SHARE. For example, one subject noted

*Trace-XML feels like something that's of interest to folks doing mapping since the Trace-XML metadata can be generated. Don't want to add the Trace-XML metadata manually. Target those doing mappings.*
For new information, subjects recommended extending Trace-XML to include different types of metadata to augment what can currently be communicated. For example, one subject recommended

*Visualizing the standards is useful. Generating the visualizations from the standards metadata, would be very useful. Annotate the graphs with hyperlinks that go to the SDTM web page or a specification.*

The subjects were not informed that Trace-XML includes the Trace-KB tool to generate Trace-XML content using the relationships available in the SHARE MDR as it was not demonstrated. Thus, none of the subjects commented on this capability.
CHAPTER 7

DISCUSSION AND CONCLUSION

Primary Benefits of Trace-XML

Trace-XML contributes two features that immediately benefit data reviewers: the ability to validate traceability across the full study lifecycle, and the ability to query the complete trace for a variable. Traceability improves a reviewer’s ability to understand a study, and has been identified as essential for a regulatory reviewer’s ability to assess a submission. This same task of assessing study data is important to non-regulatory data reviewers as well, including cases where (1) a research organization seeks conduct a meta-analysis using data from different studies, (2) a research organization seeks to aggregate clinical research data from a variety of sources, (3) a researcher seeks to perform a data quality assessment on an existing study, (4) a researcher attempts to replicate research findings, or (5) a research organization in-licenses a compound and its associated study data. In general, advancing traceability improves the quality of the study metadata and increases the efficiency with which data reviewers begin to understand the data. Making the data easier to understand encourages the secondary use of the data.

Today, computable traceability does not exist within the clinical research metadata. Currently, the standards include a limited set of descriptive metadata that provides information on the previous step in the process. Identifying the full trace for a variable in a CDISC study today is a manual, labor intensive process. Even when manually assembled the data collection metadata is not provided in submissions, and instead a PDF rendering of an annotated CRF provides the descriptive metadata. Manually assembling the traceability metadata for key analysis variables is inefficient and error-prone. It is effectively infeasible to manually perform this task at the scale of an entire clinical research study. Thus, Trace-XML provides new traceability features that better enable the assessment of traceability within a clinical study, and thus, better address the existing regulatory requirements.
The ability to conduct an exploratory analysis of traceability for a study, or to compare the end-to-end data lifecycle for similar submissions has not been a common practice. It is infeasible to visualize a full study or subsets of a study using the current state-of-the-art. Trace-XML's ability to generate full lifecycle study graphs makes this analysis possible. Combining Trace-XML graphs with graph visualization software enables new analysis capabilities not available in the current standards and supporting tools.

**Implementation Feasibility**

"Good design science research often begins by identifying and representing opportunities and problems in an actual application environment" (Hevner, 2007).

The Hevner (2007) relevance cycle shown in Figure 10 establishes the environment from which the solution requirements are drawn as well as implementation constraints and acceptance criteria. The environment for the solution domain consists of the existing processes, people, information systems, and organizational systems that establish the context within which the solution must operate. To achieve success, the design science artifacts must be implementable in the existing environment. The relevance cycle provides iterative input into the design cycle to constrain the design and to provide the feedback needed to adjust the artifacts to meet the contextual requirements. Clinical research represents a complex, global environment with significant levels of regulatory oversight. Operating within the constraints applied by this environment and maximizing implementation feasibility were important objectives for the Trace-XML artifacts.

Trace-XML provides a repeatable method for creating full lifecycle study traceability using the CDISC metadata and supporting the requirements of the regulated clinical research environment. The Trace-XML extension to the Define-XML standard is small and builds on existing Define-XML concepts. The Trace-XML Framework provides the Trace-KB tool that generates some of the extended Define-XML content using metadata available in the SHARE MDR eliminating the need to add this content manually. Subjects participating in the qualitative assessment study noted that their Extract-Transform-Load style software could use the CDISC mapping metadata they maintain to generate Trace-XML content for the Define-
XML files they produce. The CDISC leadership has agreed to promote Trace-XML as a Define-XML extension to its membership, and the Trace-XML extension has been proposed for inclusion in the next version of the Define-XML standard. The Trace-XML software will be made available as open source accessible from the CDISC web site. The combination of the Trace-XML extension and software makes possible a repeatable process for generating the Trace-XML graph and subsequently enabling full lifecycle study traceability.

The feasibility for implementing Trace-XML in practice has been enhanced by building on the existing industry data standards. Easing implementation barriers and working within the existing data and information system infrastructure is critically important to adoption of a traceability solution at the industry level. Domains such as clinical research and routine healthcare make use of a diverse range of existing data and technologies within their IT infrastructure that cannot be replaced without incurring significant expense and disruption (Curcin, 2016). A traceability solution must minimize the impact to the existing infrastructure while maximizing the benefits provided to reviewers to increase the feasibility of adoption in the near term. As a regulatory requirement for data submissions to the FDA and PMDA, the use of Define-XML by Trace-XML reduces the barriers to implementation for submissions. Leveraging the existing standards to the maximum extent possible, and requiring only a small extension to the Define-XML standard, allows potential implementers to take full advantage of their existing standards investments. In a regulatory context, any additional information required for a submission can be viewed as an additional regulatory burden, despite obvious benefits. The expansion of existing tools that generate Define-XML files to include the Trace-XML content would minimize the impact of requiring the Trace-XML extension while providing the additional traceability benefits.

As an extension to Define-XML, Trace-XML metadata does not require additional information technology infrastructure to function which reduces barriers to implementation and acceptance. Trace-XML generates a graph in the GraphML standard XML format. Because the regulatory authorities, as well as organizations generating data for submissions, already work with XML documents such as Define-XML, the Trace-XML graph file does not represent a new technology that requires additional investment or training. The Trace-XML software is implemented in Java, and the regulatory authorities and CDISC membership have
demonstrated the ability to run JVM-based applications, such as the Pinnacle-21 conformance tool. If users of Trace-XML become interested in graph visualization, Java-based open source software is available to render and analyze the Trace-XML graph, including Gephi (Bastian et al., 2009).

Trace-XML uses ODM to provide the metadata for the data capture phase of the clinical research data lifecycle. ODM v1.3.2 is a mature standard and "ODM metadata has become the language of choice for describing CRFs" (Hume et al., 2016). Most of the major EDC systems support ODM and the ability to export ODM CRF metadata (CDISC, 2017). Using EDC systems to export ODM-based CRF metadata eliminates the need for new technology implementations to support the use of ODM by Trace-XML and better leverages the existing features of widely used data capture platforms. Style sheets exist to generate annotated CRF visualizations of ODM-based CRFs. As Define-XML is an ODM extension, using the Trace-XML extension to link the metadata across the clinical research data lifecycle creates the ability for the CDISC XML metadata standards to represent the metadata for an entire study, and breaks down the existing silos caused by each metadata file working in isolation to represent only one phase of the lifecycle.

This research sought to better address the stated requirements for full lifecycle study-level traceability within the context of a regulatory submission. Trace-XML query results can be embedded into the existing Define-XML stylesheet outputs, as shown in Figure 24, to eliminate the need for additional training to use the Trace-XML output. Providing simple means to access Trace-XML content embedded within an existing standards implementation minimizes barriers to implementation and improves the feasibility of Trace-XML as a traceability solution that could be used by the regulators.

**Trace-XML as a Validation Tool**

Validation of the Define-XML documents beyond mere XML schema validation has become a critical step in the regulatory submission process necessitating the development of validation rules and the software to apply them (Hume et al., 2016). The regulatory authorities publish additional conformance rules for the CDISC standards used in submissions, and apply
these rules to each submission using software tools. The study metadata provided in Define-XML is among the content checked for standards conformance as part of a regulatory submission. Full lifecycle, or end-to-end, study metadata traceability validation is an immediate benefit provided by Trace-XML to improve the quality of study metadata. The Trace-XML graph generation and validation ensures metadata traceability exists within the context of a study, and thus would augment the current conformance rules. Today's validation rule checks exist in silos and address one phase of the lifecycle at a time. Trace-XML adds a full study lifecycle quality and conformance check that is made possible by the Trace-XML extension to Define-XML, as well as the Trace-XML algorithms. Prior to Trace-XML this has not been feasible.

To effectively generate and validate traceability graphs for clinical research, new traceability rules must be created to establish end-to-end traceability requirements. Trace-XML has implemented several of these rules. For example, a variable that has multiple source variables should reference a method that describes the derivation or transformation used to create one result from multiple sources. This may be as simple as a concatenation to create a full date field or a calculation used to derive a result. This research project also added a rule to ensure that OIDs are unique within a Define-XML or ODM file, and ideally OIDs would be unique across the entire study. New traceability rules should be considered as additions to the existing CDISC standards and applied as conformance rules that verify traceability quality within a study. Table 13 lists a summary of the Trace-XML specific traceability rules.

As Trace-XML verifies traceability within the graph, it reports those variables that are not traceable based on its rules and algorithms. Trace-XML uses the variable's Origin Type to assess the degree to which traceability should be expected for a variable and whether or not it should be traceable back to the data collection phase of the lifecycle. By reporting the traceability gaps, those implementing the study metadata can remedy the issues and retest the updated files for complete traceability. When traceability gaps are addressed prior to study deployment this can ensure that the data needed to support analysis are being collected, and that unnecessary variables are not collected.
Trace-XML as a Tool to Enhance Data Quality

Creating Define-XML for use as a specification has been recommended as a best practice for improving study data quality (Hume et al., 2016). This practice can be improved by including Trace-XML metadata into the Define-XML file. As standards that provide study metadata, both ODM and Define-XML can be used to support process automation and aid in study setup (Hume et al., 2016). There are three fundamental approaches for creating Define-XML files: (1) create it post-hoc as a means to document the metadata used to create the final study datasets; (2) synchronize the generation of the metadata content with the ongoing transformations used to create the study datasets; and (3) create the Define-XML metadata before the datasets are created, and use it as a specification that drives the creation of the study datasets. This last approach has been extended by Lightfoot and Jansen (Lightfoot & Jansen, 2013) to describe how a Define-XML file created by the sponsor can be used as a template for study setup, and then applied to create a Define-XML file that describes a study to be included in a regulatory submission. Wheeldon and Burgess (Wheeldon & Burgess, 2014) describe how creating Define-XML during study setup promotes reuse and governance of study designs, reuse of mapping specifications, and automated dataset validation. When the Trace-XML extension is created as part of the study specification, traceability gaps can be identified and resolved prior to study start improving data quality and reducing the need for re-work or mid-study amendments to resolve data issues.

IP-Maps were developed as a tool to promote data quality and Trace-XML extends that research to examine traceability as a dimension of data quality at the level of granularity needed to support regulated clinical research. An IP-Map's ability to communicate traceability at a high-level of abstraction is the primary benefit of layer 1 of the framework. The IP-Map also highlights the data quality steps applied during the production of the information product, including data cleaning, data transformations and imputations, and data standards conformance. This helps data reviewers to better understand the steps taken to ensure data quality within the study. The IP-Maps are particularly useful to those reviewers interested in attaining a high-level understanding of the study data and traceability. They provide another way to review the study data flows and the processes that could impact data quality. A high-level conceptual model has become an increasingly important data quality communication
tool as new information sources and new validation mechanisms have been introduced into
the clinical research lifecycle. For example, FDA draft guidance on the use of Electronic
Health Record (EHR) data in Clinical Investigations recommends that sponsors include a
diagram of the data flow between the EHR and the clinical research systems (FDA, 2016b). In
cases where the reviewer does not need the detailed understanding of variable lineage
provided by layers 2 and 3 of the framework, the IP-Maps in layer 1 may provide all the
information needed understand the study data flows across the lifecycle. The links between
layers 1 and 2 of the framework are not essential if the IP-Maps are not used to help automate
the generation of the Trace-XML graph. In these cases, layer 1 can stand alone as an
informative addition to the detailed traceability features provided by Trace-XML.

**Trace-XML as a Visualization Tool**

Trace-XML provides a more comprehensive understanding of the clinical data for a
study by integrating the conceptual view, the clinical study artifact and data element view,
and the graph view of the study metadata (Chee et al., 2014). The integrated, hierarchical
representation of traceability provided by Trace-XML improves the efficiency with which
data reviewers come to understand the data and traceability within a study. Reviewers are able
to drill into more detail as needed to answer specific questions about the data (Chee et al.,
2011). Layer 1 in the framework provides a conceptual visualization of the data flow and
information manufacturing processes for an information product at a high-level of abstraction.
However, the primary traceability visualization capability is provided by a rendering of the
Trace-XML graph produced by layer 3 of the framework.

Rendering the Trace-XML graph to visualize the data flow for a full study provides a
unique mechanism for exploratory analysis. Patterns in the graphs can be visually identified
and reviewed. Graph visualizations can aid in the detection of traceability anomalies and can
improve a reviewer's understanding of how the data flows throughout the study lifecycle. To
aid the reviewer, nodes within a visualization can be color coded to convey different
meanings, the size of the nodes can vary according to an established metric, or the distance
between nodes can be adjusted to highlight certain relationship metrics. For example, node
size can be increased to reflect the number of links a node has to other nodes making it easy to identify those parts of a study that reference the most data elements.

A graph covering the full lifecycle for an entire study can grow too large to easily analyze visually. To improve the ability to work with large graphs, filtering or collapsing parts of the graph can be a useful way to reduce the number of nodes presented at any one time. For example, using the yEd graph editor portions of a graph can be grouped together and consolidated. A group can be opened to show each of the member nodes, or closed to represent the group as a single node within the graph. Grouped nodes can also be viewed independently of the full graph permitting simplified interrogation of the individual nodes contained in the group. Such graph organization tactics can be used to improve targeted visualization tasks.

The graph visualization of metadata traceability represents a feature that does not exist in today's standards or tools. It also represents an exploratory capability that could provide useful insights into the data and help data reviewers better understand a study. Outside of this ability to explore data traceability and better understand the data, the tasks to which graph visualizations will be applied are not clear. The applications of graph visualization as a traceability tool will evolve as reviewers become more comfortable with the tools and better understand the questions that can be answered using them. Trace-XML generates the graph model, but the visualization features vary with the software used to render and analyze the graph. This research project has tested two open-source tools that work with GraphML graphs, but others tools are available. Additionally, the GraphML graph can be converted to other formats such as JSON where other visualization tools such as D3.js can be used to produce different visualizations to support specific analytic needs.

**Trace-XML Framework Utility**

As part of the artifact evaluation, Trace-XML was demonstrated to a panel of clinical data experts, including regulators, to establish whether or not this work represents an advancement in traceability utility that provides new and useful traceability capabilities. The feedback received during the interviews with the clinical research data experts both confirmed
the utility of Trace-XML as well as provided instructive feedback to be used in future build cycles. The overarching theme resulting from the analysis was *adds utility*, supported by themes that highlight the type of value added including *adds information*, *verifies data quality*, *promotes understanding*, and *simplifies implementation*. The most prominent supporting theme was that Trace-XML *promotes understanding* of traceability and *how things fit together*. Within the *adds utility* category, the code *useful* was the most prominent at 7.9% of the total codes produced during the analysis.

The interview subjects were unanimous in their confirmation that Trace-XML added new, meaningful traceability features. Every interviewee noted that Trace-XML improves a data reviewer's understanding of their data by significantly simplifying the work needed to trace the inputs to any variable in a study. This conclusion is supported by the fact that the *promotes understanding* theme was the most prominent with the code *comprehension* appearing in 80% of the interviews and the code *how things fit together* also appearing in 80% of the interviews. In response to the question about assessing end-to-end traceability validation, two of the salient themes were the top two codes applied in terms of number of instances: *new information* and *how things fit together*. These codes each accounted for nearly 10% of the total codes and support the claim that Trace-XML presents new information that aids data reviewers in understanding a study.

Another theme in the findings indicates that Trace-XML should add utility to the regulatory data review process where traceability is a requirement. Within the *adds utility* theme which accounted for 23.4% of the total codes, the *regulatory impact* code was the most frequently applied at 9.1%. The *regulatory impact* code was applied in 100% of the interviews and had the broadest coverage of any code in the study. Several non-regulatory study participants explicitly requested that Trace-XML be demonstrated to the FDA with the understanding that the Trace-XML benefits would apply directly to regulatory data reviewers, and that regulatory agency interest would help accelerate industry acceptance. One subject commented "the FDA is quite interested in a capability like this; they really want a tool that allows them to trace through things."
Regarding the theme to *simplify implementation*, many subjects identified the need to *generate* the content used as inputs to Trace-XML, along with a general need to have a *tool focus* and hide the details of the XML. As the use of Define-XML v2.0 has increased, implementers have increasingly realized the need to automate the process of generating the metadata document to the extent possible. This theme was pronounced when interviewing those currently involved with generating Define-XML files. Two subjects referenced the possibility of using existing *mapping metadata* to generate the content needed to fulfill the Trace-XML Define-XML extension. Define-XML implementers were also particularly excited that Trace-XML *builds on existing technology*. Using just a small extension to Define-XML to implement Trace-XML, the subjects observed that this reduces barriers to implementation while benefiting from the significant industry knowledge base for Define-XML. One subject expressed this idea in the following quote:

*This builds on ODM which is a big positive. I like that we’re just adding a few things to existing standards – not starting from scratch.*

The *tool focus* code complements the *generate* code by noting that users benefit more from tools that make implementation easy and automatic, as one subject noted:

*To maximize its impact users of Trace-XML should be using it with their tools.*

*I would not ask users to add information manually as part of a Define-XML generation, but would generate it using tools. Not sure how interested end-user implementers would be in implementing this, but they would like to take advantage of the features through a tool.*

The Trace-KB feature that generates the Trace-XML extension metadata from content retrieved using the SHARE API was not demonstrated prior to the interviews so subjects were not aware of this feature.

The *verifies data quality* theme was the least applied with only 12% of the overall codes. Codes for this theme included *validation, completeness*, and *traceability*. This outcome may have been influenced by the focus of the questions as well as the role of the study participants. It also may reflect that case that many organizations look to the regulatory
authorities to establish the types of data quality assessments that are needed for a study, and are not likely to volunteer to apply more rigor to the assessments without external pressure to do so. Most subjects felt that the FDA would be interested in both the improved level of traceability as well as the additional level of validation offered by Trace-XML. As one subject noted, "conformance rules that include traceability validation will be important."

The primary cautionary feedback received during some interviews are highlighted by the codes complex and too advanced. These codes made rare appearances with complex at 1.9% and too advanced at 1.1% of the overall codes, but should not be overlooked. Complexity became a topic to describe the full graph views of a study. These graphs are large and contain a significant amount of detail. Furthermore, this is a new way to visualize study metadata that provides new information not available today, so it does represent an entirely new way of analyzing study metadata when compared to today's processes. Although all interviewees saw the graph visualization as useful and a step forward, some noted that it might not meet an obvious need for some roles. As one subject noted "not every user will understand what problem the graph visualization solves for them."

The too advanced code indicates that Trace-XML does not merely represent an incremental improvement to existing processes, but offers new capabilities not yet available to data reviewers. Again, this code was more likely to be applied to the graph visualization than other aspects of Trace-XML, but it was also applied to indicate that this represents an advancement to the industry state-of-the-art that will take time to adopt. One subject noted that "Trace-XML is 3-5 years ahead of the game now."

A number of recommendations for improving or expanding Trace-XML came from the interviews. As previously noted, a number of subjects recommended automatically generating Trace-XML content to the extent possible, and focusing on building tools that deliver users benefits while isolating them from exposure to the CDISC XML standards. To this end, one subject noted that changing the report labels and naming of graph nodes to remove content that reflects the CDISC XML standards would improve usability. Another recommendation was to make the colors configurable to show different types of categorization.
A subject also recommended generating the IP-Map content from the Trace-XML metadata changing the process from a top-down approach to a middle-out approach. A middle-out approach would generate the lower-level Trace-XML graph, but would use this same information to generate an analysis level depiction of the data flow similar to an IP-Map. Essentially, this recommendation calls for the Trace-XML Framework level 2 metadata to be used to generate both the level 3 graph, as well as the level 1 IP-Map. In general, minimizing manual steps in the process of using Trace-XML was seen as desirable to ensure broad usage. Another recommendation was to add the ability to show an annotated CRF generated by Trace-XML using a style sheet. This has been demonstrated to be feasible and would add another benefit to encourage the inclusion of ODM XML with the Define-XML files as part of a study regulatory submission.

One subject recommended creating traces or reports that focused on the key study endpoints and the most relevant study datasets. Highlighting this information would help ensure regulatory reviewers had easy access to the information that was of most interest to them, without overwhelming them with information of lesser importance. Another subject recommended using Trace-XML to show how a standard evolves over time with additions and changes being added to the graph to visually depict changes. This would provide a visualization of the standards that would help implementers better understand how they are organized and the impact of changes.

The key themes that emerged from the thematic analysis of the interview data affirm that Trace-XML adds utility to the task of creating and assessing end-to-end clinical research study traceability across several different user roles and perspectives. The interviewees were unanimous and unambiguous in their conclusions regarding the utility added via new traceability capabilities demonstrated in Trace-XML. The diversity of perspectives provided a number of varied alternatives for new features to add value to future Trace-XML builds. Subjects also reported that the cautionary notes on complexity and the advanced nature of the tool can be addressed by targeting more precise use cases with automated tooling.
Impact on Regulatory Submissions

Trace-XML advances traceability using CDISC metadata and provides capabilities not available using the current standards. To fully realize the Trace-XML advances, regulatory authorities should request that ODM be included in regulatory submissions to represent CRF metadata instead of the current blank CRF in PDF format. ODM generates annotated blank CRF visualizations in HTML using style sheets. These visualizations are simpler to create than their PDF counterparts. The PDF annotated CRFs are often created manually. Style sheets exist today capable of generating annotated CRF visualizations using ODM metadata. As noted previously, ODM CRF metadata exports are supported by most commonly used EDC systems making it simpler to generate ODM metadata that it is to create Define-XML metadata which draws metadata content from a wider variety of sources. Since the ODM metadata is machine readable, this enables Trace-XML to function fully and could also enable other new features, including traceability links to EHR electronic source data.

Define-XML v1.0 was published in 2005 to replace define.pdf. While the eventual replacement did not happen immediately, machine-readable Define-XML documents are now an important part of a regulatory submission (FDA, 2017). Although, as previously noted, generating ODM CRF metadata is simpler than creating a Define-XML, and it would be simpler to generate an ODM-based annotated CRF, ODM is not currently accepted as part of a regulatory submission. Replacing PDF documents for representing CRF metadata not only provides the machine-readable metadata used by Trace-XML, but would also make the metadata submissions more consistent. The use of ODM-based standards as a consistent means of representing study metadata would simplify submissions while providing opportunities to leverage more complete machine-readable metadata for each study included in a submission.

ODM could also be used to optionally submit the raw data collected at the site. Currently, limited CRF data is submitted in PDF form, but this information is incomplete, lacks machine-readability, and is difficult to generate. ODM provides the machine-readable metadata and data for CRFs. It also provides audit trail information which would enable regulators to implement standard tools for detecting fraud. The FDA has stated that PDF is an inadequate format for submitting EDC CRFs and in 2007 proposed to pilot ODM for
submitting CRFs. In the pilot notice the FDA noted that they were interested in ODM as a standard that could "reliably provide all three components of the CRF in an electronic format: data, metadata, and audit trail" (FDA, 2007a). Combining Trace-XML with ODM metadata and data would significantly enhance the provenance information included with each submitted study.

**Future Work**

The study data lifecycle used for this research covers from data collection to final analysis datasets. Future research will extend this lifecycle to include electronic source data from EHR systems as well as analysis results metadata. Pre-populating CRF content with EHR data is an area of active research, and including EHR sources in Trace-XML would improve a reviewer's understanding of study traceability (Erturkmen, Bain, & Sinaci, 2014). The FDA's recent final guidance on the use of eSource within clinical investigations (FDA, 2016b) notes that eSource has the potential to improve the integrity, traceability, provenance, quality, and reliability of data included in electronic submissions (Curcin, 2016). Extending ODM to capture references to the eSource content used to pre-populated CRFs would provide the additional metadata needed to add EHR content to Trace-XML study traceability. Without such an ODM extension to support this addition to Trace-XML, no existing infrastructure includes the eSource information as part of the traceability metadata provided in a regulatory submission.

Analysis results metadata represents the study findings produced using the analysis datasets. Currently, the Analysis Results Metadata (ARM) extension to Define-XML provides the additional metadata needed to represent the study analysis results, and extending ARM to support Trace-XML would enable the analysis results to be included in the full lifecycle study traceability graph. Regulatory reviewers may be particularly interested in tracing key study endpoints and efficacy variables. Although ARM is not yet a regulatory requirement, it is strongly recommended for submissions to the PMDA. Extending Trace-XML to include both eSource and analysis results fits into the Trace-XML Framework model and significantly enhances the value of traceability metadata within the context of a regulatory submission.
Graph representation of clinical research studies can be used to compare studies and identify differences. A future investigation will generate graphs using standard metadata for a CDISC therapeutic area standard and compare it to actual study metadata. The comparison of the study graph against the standard graph using graph difference algorithms will highlight where novel additions have been added to the study or where the study deviates from the standards. Highlighting such differences will greatly aid the review process by focusing a reviewer's attention on novel aspects of the study or highlighting deviations from the standard. As the standards are better known to reviewers than the novel aspects of a study, such a graph difference, would limit the review space significantly.

This initial version of Trace-XML does not implement the W3C PROV or Open Provenance Model because these standards are not currently used in the CDISC standards or by regulators, but future versions will support these standards (Curcin et al., 2014). The use of PROV would add complexity to Trace-XML with correspondingly limited benefits in the near term. However, as more healthcare systems provide data for use in research a more general provenance capability may be useful. As more EHR applications and other healthcare systems implement standards like PROV, there will be increased utility in supporting these standards in Trace-XML.

Trace-XML generates a GraphML XML representation of the traceability graph. This aligns well with the existing ODM and Define-XML standards, and minimizes the new tools and training needed to work with the graph. Alternative JSON and RDF representations of the Trace-XML may interest users working with alternative data formats and should be supported in the future. This has the benefit of supporting a broader set of software tools for working with the Trace-XML graph.

Future work will include expansion of the current example studies to include more domains and data to enable larger graphs and a broader range of traceability scenarios to evaluate. Larger Trace-XML graphs will require additional work to establish the optimal means to visualize traceability content.

The metadata gaps discovered in this research and referenced in Chapter 5 represent the current siloed nature of the standards metadata and subsequently an inability to explicitly
reference metadata in adjacent phases of the lifecycle. As an outcome of this research project, the metadata needed to establish the missing graph edges has been implemented as the Trace-XML extension to the Define-XML standard. To expand traceability to include EHR electronic source data, ODM might also benefit from the Trace-XML extension in a future version. The Trace-XML extension has been submitted to the CDISC XML Technologies Team for consideration. Future work could include incorporating the Trace-XML extension into an upcoming version of the ODM or Define-XML standards.

As Trace-XML becomes more broadly used additional traceability rules should be added to the CDISC standards to improve traceability completeness. Certain variables, such as those included in CRFs for data management purposes, might not be used in later phases of the lifecycle, but these nodes should link to a terminal node in the graph. This improves the ability to verify traceability accuracy and helps reviewers identify sets of variables where reachability into the next phase of the lifecycle is not expected. Another example of a new traceability rule involves methods that can exist at the variable and value-level. Trace-XML assumes that an Item does not reference a MethodDef both at the variable level and at the value level. Methods should be placed on the value level metadata ItemRef in this instance. This improves the accuracy of the Define-XML as well as traceability and should be considered as a future rule for Define-XML. This rule has been implemented in the current version of the Trace-XML prototype. The development of new rules that address the full clinical research data lifecycle should improve both traceability as well as overall data quality. A summary of the traceability rules implemented to support Trace-XML can be found in Table 13.

As the Trace-XML graph visualization usage increases, additional flexibility in configuring the trace graphs should be considered. For example, setting node sizes, colors, and shapes in the configuration file would provide a useful way to tailor the graph visualizations produced to address specific analysis interests. Additional features may also be added to generate graphs that implement the features of specific graphing packages beyond the yEd software.
A new version of Define-XML, version 2.1, is currently in the final stages of development and future work includes enhancing Trace-XML to support this new version of Define-XML. Define-XML v2.1 updates the Origin element and will require updated traceability rules.

Research Contributions

This research contributes the Trace-XML Framework for creating computable traceability using the CDISC metadata standards within the domain of clinical research. This framework contributes a new Trace-XML extension to the Define-XML standard that addresses traceability gaps uncovered during the development of the Trace-XML software. The Trace-XML software contributes the algorithms needed to remedy and validate traceability in the Trace-XML graphs, in addition to adding support for traceability queries. These DFS-based algorithms make use of contextual rules that adapt the algorithms to the CDISC standards and make it possible to effectively add full lifecycle traceability to CDISC standard metadata. These rules, summarized in Table 13, adapt the DFS-based algorithms to support traceability within the Trace-XML graphs and are a contribution of this research. Trace-XML also contributes the graph model generated using the extended CDISC metadata to represent full study lifecycle traceability. In addition to the layered traceability perspective, the primary benefits of the Trace-XML Framework are (1) to identify and resolve traceability gaps in clinical study metadata, (2) validate metadata traceability in a clinical study, and (3) query and visualize traceability metadata. The Trace-XML Framework maximizes the use of the existing standards models and technology to minimize barriers to implementation.

Conclusion

The artifacts presented in this research provide the means to advance traceability within clinical research. The artifacts extend the existing standards and provide a bridge between the previous traceability solutions and a new more advanced traceability solution. These artifacts also make implementation in a real-world environment feasible. The evaluation of the artifacts supports the claim that Trace-XML provides new and useful functionality that improves a reviewer's ability to assess traceability and understand the data.
The Trace-XML extension to Define-XML and the software implementation of the framework will be published as open source. The Trace-XML software and documentation are available on GitHub at https://github.com/swhume in the trace-xml and trace-query repositories. Trace-XML content will also be made available on the CDISC web site (http://www.cdisc.org), as well as on the ODM Review web site (http://www.odm-review.com).
APPENDICIES

APPENDIX A: GLOSSARY OF TERMS

Table 20. Glossary of Terms / Acronyms

<table>
<thead>
<tr>
<th>Term / Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADaM</td>
<td>Analysis Dataset Model</td>
</tr>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>ARM</td>
<td>Analysis Results Metadata</td>
</tr>
<tr>
<td>BI</td>
<td>Business Intelligence</td>
</tr>
<tr>
<td>CCD</td>
<td>Continuity of Care Document</td>
</tr>
<tr>
<td>CDA</td>
<td>Clinical Document Architecture</td>
</tr>
<tr>
<td>CDASH</td>
<td>Clinical Data Acquisition Standards Harmonization</td>
</tr>
<tr>
<td>CDE</td>
<td>Common Data Element</td>
</tr>
<tr>
<td>CDISC</td>
<td>Clinical Data Interchange Standards Consortium – a standards development organization for clinical research data standards.</td>
</tr>
<tr>
<td>CFR</td>
<td>Code of Federal Regulations</td>
</tr>
<tr>
<td>CRF</td>
<td>Case Report Form</td>
</tr>
<tr>
<td>CRO</td>
<td>Contract Research Organization</td>
</tr>
<tr>
<td>eCRF</td>
<td>Electronic Case Report Form</td>
</tr>
<tr>
<td>Define-XML</td>
<td>CDISC standard for specifying the tabulation data definitions in XML format. A required standard for CDISC standard dataset metadata (e.g. SDTM, ADaM, SEND) submissions to regulatory authorities such as the FDA and PMDA.</td>
</tr>
<tr>
<td>DSR</td>
<td>Design Sciences Research</td>
</tr>
<tr>
<td>DAG</td>
<td>Directed Acyclic Graph</td>
</tr>
<tr>
<td>DSU</td>
<td>Dakota State University</td>
</tr>
<tr>
<td>EDC</td>
<td>Electronic Data Capture</td>
</tr>
<tr>
<td>EHR</td>
<td>Electronic Health Record</td>
</tr>
<tr>
<td>EMA</td>
<td>European Medicines Agency</td>
</tr>
<tr>
<td>EVS</td>
<td>Enterprise Vocabulary Services (NCI EVS)</td>
</tr>
<tr>
<td>ePRO</td>
<td>Electronic Patient Reported Outcomes.</td>
</tr>
<tr>
<td>FDA</td>
<td>US Food and Drug Administration</td>
</tr>
<tr>
<td>FHIM</td>
<td>Federal Health Information Model</td>
</tr>
<tr>
<td>Acronym</td>
<td>Description</td>
</tr>
<tr>
<td>---------</td>
<td>-------------</td>
</tr>
<tr>
<td>FHIR</td>
<td>Fast Healthcare Interoperability Resources – an HL7 data exchange standard</td>
</tr>
<tr>
<td>GCP</td>
<td>Good Clinical Practice</td>
</tr>
<tr>
<td>GraphML</td>
<td>Graph Markup Language – an XML standard for representing graphs</td>
</tr>
<tr>
<td>HITSP</td>
<td>Health Information Technology Standards Panel</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language – used to represent web pages and forms for rendering in browsers.</td>
</tr>
<tr>
<td>ICH</td>
<td>International Conference on Harmonization of Technical Requirements for Registration of Pharmaceuticals for Human Use</td>
</tr>
<tr>
<td>IP-Map</td>
<td>Information Product Map</td>
</tr>
<tr>
<td>IRB</td>
<td>Institutional Review Board</td>
</tr>
<tr>
<td>IS</td>
<td>Information Systems</td>
</tr>
<tr>
<td>ISO</td>
<td>International Standards Organization</td>
</tr>
<tr>
<td>IT</td>
<td>Information Technology</td>
</tr>
<tr>
<td>IVRS</td>
<td>Interactive Voice Response System</td>
</tr>
<tr>
<td>JSON</td>
<td>Java Script Object Notation</td>
</tr>
<tr>
<td>LOINC</td>
<td>Logical Observation Identifiers Names and Codes</td>
</tr>
<tr>
<td>MDR</td>
<td>Metadata Repository</td>
</tr>
<tr>
<td>NCI</td>
<td>National Cancer Institute (US)</td>
</tr>
<tr>
<td>NIH</td>
<td>National Institutes of Health (US)</td>
</tr>
<tr>
<td>ODM</td>
<td>Operational Data Model</td>
</tr>
<tr>
<td>ONC</td>
<td>Office of the National Coordinator for health information technology (US)</td>
</tr>
<tr>
<td>PDF</td>
<td>Portable Document Format</td>
</tr>
<tr>
<td>PMDA</td>
<td>Japan Pharmaceuticals and Medical Devices Agency</td>
</tr>
<tr>
<td>PRM</td>
<td>Protocol Representation Model (CDISC)</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Format</td>
</tr>
<tr>
<td>RIM</td>
<td>Reference Information Model (HL 7)</td>
</tr>
<tr>
<td>SDO</td>
<td>Standards Development Organization</td>
</tr>
<tr>
<td>SDTM</td>
<td>Study Data Tabulation Model</td>
</tr>
<tr>
<td>SEND</td>
<td>Standard for Exchange of Nonclinical Data. SEND is a CDISC standard based on the SDTM model.</td>
</tr>
<tr>
<td>SHARE</td>
<td>Shared Health and Research Electronic library. The metadata repository for creating, maintaining, and publishing the CDISC standards.</td>
</tr>
<tr>
<td>TCG</td>
<td>Technical Conformance Guide</td>
</tr>
<tr>
<td>TDD</td>
<td>Test-Driven Development</td>
</tr>
<tr>
<td>ToC</td>
<td>Transitions of Care initiative</td>
</tr>
<tr>
<td>W3C</td>
<td>World Wide Web Consortium – community developing Web standards</td>
</tr>
<tr>
<td>Term</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>XMI</td>
<td>XML Metadata Interchange</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language – developed by the W3C</td>
</tr>
<tr>
<td>XQuery</td>
<td>XQuery is a query language that provides the means to extract information from an XML file or XML database.</td>
</tr>
<tr>
<td>XSL</td>
<td>eXtensible Stylesheet Language – a language used for expressing stylesheets used to display XML content</td>
</tr>
</tbody>
</table>
APPENDIX B: TRACE-XML USER DOCUMENTATION

Trace-XML

Trace-XML generates a GraphML graph model that implements full lifecycle study traceability. This version of Trace-XML requires an ADaM Define-XML file, an SDTM Define-XML file and an ODM file. The Trace-XML extension to Define-XML must also be implemented to represent the relationships that reference the source variables for each of the Define-XML files. Please reference the Trace-XML extension documentation for details.

Running a Java Program

Make sure that the Java Runtime Environment (JRE) is listed in the PATH environment variable or set the PATH environment variable to include the JRE. For example, on Windows you could run:

> set path=%path%;C:\Program Files\Java\jdk1.8.0_65\bin

Trace-XML was developed using Java version 1.8.0_65.

Installing Trace-XML

Copy and unzip the Tracexml.zip file in the directory from which you would like to run the application.

Configuration File

Prior to running Trace-XML the configuration file must be created. An example configuration file is listed below:

*trace-node-unique=trace-nodes-unique.xml*
*trace-node=trace-nodes.xml*
*define-xsd-file=/Users/shume/Documents/Temp/schema/trace1-0-0/trace-1-0-0.xsd*
*trace-node-details=trace-node-details.xml*
trace-xsl=trace-node.xsl
unreachable-html=unreachable-nodes.htm
unreachable-xml=unreachable-nodes.xml
unreachable-text=unreachable-nodes.txt
unreachable-xsl=unreachable.xsl
xml-path=/Users/shume/Documents/Temp/
L1-graph=L1-safety-ip-graph.graphml
trace-node-oid=trace-node-oid.xml
unreachable-text-xsl=unreachable-txt.xsl
xquery-path=/Users/shume/Documents/xml/trace-xml/
data-analysis-file=/Users/shume/Documents/Temp/adam-define-test.xml
trace-html=trace-node-detail.htm
odm-xsd-file=/Users/shume/Documents/Temp/schema/odm1-3-2/ODM1-3-2.xsd
L3-graph=Trace-test-graph.graphml
data-tabulation-file=/Users/shume/Documents/Temp/sdtm-define-test.xml
fix-invalid-byte-1=No

The table below provides a brief explanation of each item in the configuration file. Those rows with the "Change?" column set to "Yes" are the settings that most typically need to be set to run Trace-XML and Trace-Query.

<table>
<thead>
<tr>
<th>Config. Name</th>
<th>Configuration Value</th>
<th>Change?</th>
</tr>
</thead>
<tbody>
<tr>
<td>data-collection-file</td>
<td>Path and file name of the ODM file used to represent the metadata in the data collection phase of the clinical research data life cycle.</td>
<td>Yes</td>
</tr>
<tr>
<td>data-tabulation-file</td>
<td>Path and file name of the Define-XML file used to represent the SDTM metadata in the data tabulations phase of the clinical research data lifecycle.</td>
<td>Yes</td>
</tr>
<tr>
<td>data-analysis-file</td>
<td>Path and file name of the Define-XML file used to represent the ADaM metadata in the data analysis phase of the clinical research data lifecycle.</td>
<td>Yes</td>
</tr>
<tr>
<td>Config. Name</td>
<td>Configuration Value</td>
<td>Change?</td>
</tr>
<tr>
<td>------------------------</td>
<td>-------------------------------------------------------------------------------------</td>
<td>---------</td>
</tr>
<tr>
<td>xml-path</td>
<td>File path to the XML files used and created by Trace-XML.</td>
<td>Yes</td>
</tr>
<tr>
<td>xquery-path</td>
<td>File path to the XQuery files used by Trace-XML</td>
<td>Yes</td>
</tr>
<tr>
<td>L3-graph</td>
<td>File name of the GraphML graph produced by Trace-XML.</td>
<td>Yes</td>
</tr>
<tr>
<td>define-xsd-file</td>
<td>The Define-XML schema with the Trace-XML extension - trace-1-0-0.xsd.</td>
<td>Yes</td>
</tr>
<tr>
<td>edm-xsd-file</td>
<td>File and path to the ODM v1.3.2 schema file - ODM1-3-2.xsd.</td>
<td>Yes</td>
</tr>
<tr>
<td>trace-node-unique</td>
<td>List of unique nodes returned from a query.</td>
<td></td>
</tr>
<tr>
<td>trace-node</td>
<td>List of nodes returned from a query.</td>
<td></td>
</tr>
<tr>
<td>trace-node-details</td>
<td>Metadata details for each node returned from a trace query.</td>
<td></td>
</tr>
<tr>
<td>trace-xsl</td>
<td>Creates HTML output from the results of a query.</td>
<td></td>
</tr>
<tr>
<td>unreachable-html</td>
<td>Output file for the HTML rendering of the results of the test for unreachable nodes in the graph.</td>
<td></td>
</tr>
<tr>
<td>unreachable-xml</td>
<td>List of unreachable nodes produced by the Trace-XML algorithms and queries.</td>
<td></td>
</tr>
<tr>
<td>unreachable-text</td>
<td>Tab delimited text file listing the unreachable nodes.</td>
<td></td>
</tr>
<tr>
<td>unreachable-xsl</td>
<td>Style sheet that generates the HTML report for unreachable nodes.</td>
<td></td>
</tr>
<tr>
<td>trace-node-oid</td>
<td>XML file name containing nodes returned from the GraphML graph with reference information from the appropriate ODM or Define-XML file such as the node id, OID, lifecycle phase, and XML file name and path.</td>
<td></td>
</tr>
<tr>
<td>unreachable-text-xsl</td>
<td>Style sheet used to generate the tab delimited text content for the list of unreachable nodes.</td>
<td></td>
</tr>
<tr>
<td>trace-html</td>
<td>Trace-Query HTML output report containing the results of a query.</td>
<td></td>
</tr>
<tr>
<td>fix-invalid-byte-1</td>
<td>A value of “Yes” forces Trace-XML to eliminate an UTF-8 generation bug - Invalid byte 1 of 1-byte UTF-8 sequence.</td>
<td></td>
</tr>
</tbody>
</table>

**Running Trace-XML**

Trace-XML was developed using Java version 1.8.0_65. It can be built to package the JRE with the application to ease deployment to those environments running older versions of Java. The following example shows a command-line that runs the Trace-XML jar file:
> java -jar Tracexml.jar cfg=d:/temp/config/trace-xml.cfg yed

Specifying the location of the configuration file is required unless a configuration named trace-xml.cfg is located in the same directory as the Tracexml.jar file. The yed command-line argument is required to use the GraphML output in the yEd editor software. The Trace-XML command-line arguments are listed in the following table:

<table>
<thead>
<tr>
<th>Cmd-line Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cfg=</td>
<td>Path and file name of the Trace-XML configuration file. Trace-XML will attempt to find a trace-xml.cfg file in the jar directory if this command-line argument is not set. The configuration file is required to run Trace-XML.</td>
</tr>
<tr>
<td>yed</td>
<td>Include the minimum extensions required to generate a GraphML file that can be loaded into the yEd graph editor software.</td>
</tr>
<tr>
<td>reachable</td>
<td>Print the reachable nodes to the console.</td>
</tr>
<tr>
<td>unreachable</td>
<td>Print the unreachable nodes to the console.</td>
</tr>
<tr>
<td>verbose</td>
<td>Turn on verbose mode to get more feedback to the console on Trace-XML execution.</td>
</tr>
<tr>
<td>display</td>
<td>Attempt to automatically load the HTML Unreachable report and Query results in the default browser.</td>
</tr>
<tr>
<td>validate</td>
<td>Run Trace-XML, Define-XML, and ODM schema validation on the XML files prior to generating the graph.</td>
</tr>
<tr>
<td>help</td>
<td>Print the Trace-XML usage information to the console.</td>
</tr>
</tbody>
</table>

**Trace-Query**

Trace-Query generates a trace report using the GraphML study graph given a variable OID. You must run Trace-XML prior to Trace-Query to ensure the study graph has been generated. Trace-Query uses the Trace-XML configuration file.
Running Trace-Query

Running Trace-Query is similar to running Trace-XML, but with different command-line arguments. The following example shows a command-line that runs the Trace-Query jar file:

> java -jar TraceQuery.jar cfg=d:/temp/config/trace.xml.cfg oid=ADAM.IT.VS.ADY

The oid argument is required, and the configuration file must be accessible either by specifying it on the command-line or by placing trace-xml.cfg in the same location as the Trace-Query jar file.

<table>
<thead>
<tr>
<th>Cmd-line Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>cfg=</td>
<td>Path and file name of the Trace-XML configuration file. Trace-XML will attempt to find a trace-xml.cfg file in the jar directory if this command-line argument is not set. The configuration file is required to run Trace-XML.</td>
</tr>
<tr>
<td>oid=</td>
<td>The oid of the variable for which the trace will be executed. The oid must exactly match an oid in the graph.</td>
</tr>
<tr>
<td>help</td>
<td>Print the Trace-XML usage information to the console.</td>
</tr>
</tbody>
</table>
APPENDIX C: TRACE-XML DEFINE-XML EXTENSION

The Trace-XML schema documentation has been automatically generated using the Oxygen XML editor version 18.0 and the Trace-XML extension schema files. The complete documentation can be found on-line at:

https://github.com/swhume/trace-xml

Trace-XML is a small Define-XML extension adding the minimal elements and attributes needed to support traceability. The extension schema files include: trace-1-0-0.xsd, trace-extension.xsd, and trace-ns.xsd. The following schema code blocks display the Trace-XML extension content without the core Define-XML schemas.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.cdisc.org/ns/def/v2.0"
    xmlns="http://www.cdisc.org/ns/def/v2.0"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:trc="http://www.cdisc.org/ns/trace/v1.0"
    elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:include schemaLocation="trace-extension.xsd"/>
  <!-- Trace-XML 1.0.0 trace-1-0-0 schema -->
</xs:schema>
```

trace-1-0-0.xsd
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.cdisc.org/ns/def/v2.0"
    xmlns="http://www.cdisc.org/ns/def/v2.0"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:odm="http://www.cdisc.org/ns/odm/v1.3"
    xmlns:def="http://www.cdisc.org/ns/def/v2.0"
    xmlns:trc="http://www.cdisc.org/ns/trace/v1.0"
    elementFormDefault="qualified"
    attributeFormDefault="unqualified">
    <!-- Trace-XML 1.0.0 adds the trace-ns to Define-XML 2.0.0 -->
    <xs:import
        namespace="http://www.cdisc.org/ns/odm/v1.3"
        schemaLocation="../define2-0-0/define-extension.xsd"/>
    <xs:import
        namespace="http://www.cdisc.org/ns/trace/v1.0"
        schemaLocation="trace-ns.xsd"/>
    <xs:redefine
        schemaLocation="../define2-0-0/define-ns.xsd">
        <xs:attributeGroup name="OriginAttributeExtension">
            <xs:attributeGroup ref="def:OriginAttributeExtension"/>
            <xs:attribute ref="trc:NoTraceItems" use="optional"/>
        </xs:attributeGroup>
        <xs:attributeGroup name="OriginElementPostExtension">
            <xs:sequence>
                <xs:group ref="def:OriginElementPostExtension"/>
                <xs:element ref="trc:Trace" minOccurs="0" maxOccurs="1"/>
            </xs:sequence>
        </xs:attributeGroup>
    </xs:redefine>
</xs:schema>
<?xml version="1.0" encoding="UTF-8"?>
<xs:schema targetNamespace="http://www.cdisc.org/ns/trace/v1.0"
    xmlns:xs="http://www.w3.org/2001/XMLSchema"
    xmlns:odm="http://www.cdisc.org/ns/odm/v1.3"
    xmlns:def="http://www.cdisc.org/ns/def/v2.0"
    xmlns:trc="http://www.cdisc.org/ns/trace/v1.0"
    xmlns:xlink="http://www.w3.org/1999/xlink"
    elementFormDefault="qualified" attributeFormDefault="unqualified">
    <!-- Trace-XML 1.0.0 share-ns schema -->
    <xs:annotation>
        <xs:documentation>Trace-XML 1.0.0 trace-ns schema</xs:documentation>
    </xs:annotation>
    <xs:import namespace="http://www.cdisc.org/ns/odm/v1.3"
        schemaLocation="../odm1-3-2/ODM1-3-2-foundation.xsd"/>
    <xs:attribute name="NoTraceItems" type="odm:YesOnly">
        <xs:annotation>
            NoSourceItems indicates that there are no source items for an ItemDef Origin that would normally have source items, but in this instance does not. For example, an Origin type of Derived where the derivation takes no source Items as inputs.
        </xs:annotation>
    </xs:attribute>
    <xs:element name="Trace" type="trc:Trace"/>
    <xs:complexType name="Trace">
        <xs:annotation>
            The Trace element is intended to define the TraceItems that reference the sources for an Item. The TraceItems are child elements of the Trace parent element.
        </xs:annotation>
        <xs:sequence>
            <xs:element ref="trc:TraceItem" minOccurs="1" maxOccurs="unbounded"/>
        </xs:sequence>
    </xs:complexType>
</xs:schema>
Continued on the next page...
The TraceItem elements are intended to reference the source ItemDefs that function as the explicit Origin of a given ItemDef. If the Origin type is Derived, the TraceItem elements reference the ItemDefs that represent the inputs to the MethodDef performing the derivation.

- **leafID** references the document containing the TraceItem ItemDef.
- **ItemOID** references the OID of the ItemDef that represents the source item. ItemOIDs should be unique per study in Trace-XML.
- **FormOID** references the OID of the FormDef containing the source item (via an ItemGroup). This is primarily used when a source item is referenced in multiple forms, but only one instance of this item in one form provides the input for this ItemDef.
- **ItemGroupOID** references the OID of the ItemGroupDef containing the source item. This is primarily used when a source item is referenced in multiple ItemGroups, but only one instance of this item in one ItemGroup provides the input for this ItemDef.
APPENDIX D: TRACE-XML SOURCE CODE DOCUMENTATION

The Trace-XML applications include three distinct programs: Tracexml, TraceQuery, and TraceKb. The source code for these applications include comments that comply with the Javadoc format for generating standardized documentation. Due to the volume and format of this documentation, it has not been embedded in this dissertation. Instead, the Trace-XML source code and documentation have been released under the Apache 2.0 open source license and are accessible via GitHub:

- [https://github.com/swhume/trace-xml](https://github.com/swhume/trace-xml)
- [https://github.com/swhume/trace-query](https://github.com/swhume/trace-query)

A trace-kb repository will be created at a later date, once access to the SHARE API becomes more widely available.

Links to the Trace-XML documentation will also be made available on the CDISC web site ([http://www.cdisc.org](http://www.cdisc.org)) as well as on the ODM Review web site ([http://www.odm-review.com](http://www.odm-review.com)).
APPENDIX E: TRACE-XML QUALITATIVE ASSESSMENT INFORMED CONSENT

DSU Institutional Review Board

Date: 2/7/2017

Dear Subject:

I am conducting a research project entitled "Enhancing Traceability in Clinical Research Data Through an Information Product Framework" as part of a dissertation at Dakota State University.

The purpose of the study is to evaluate the utility of Trace-XML, a framework for end-to-end metadata traceability within the domain of clinical research.

You as a clinical research metadata expert are invited to participate in the study by participating in a traceability demonstration and interview. We realize that your time is valuable and have attempted to keep the requested information as brief and concise as possible. It will take you approximately 30 minutes of your time. Your participation in this project is voluntary. You may withdraw from the study at any time without consequence.

There are no known risks to you for participating in this study. There are no direct benefits to you from participation in this study.

Your responses are strictly confidential. When the data and analysis are presented, you will not be linked to the data by your name, title or any other identifying item.
Your consent is implied by agreeing to participate in the demonstration and interview. Please keep this letter for your information. If you have any questions, now or later, you may contact me at the number below. Thank you very much for your time and assistance.

If you have any questions about the research study or content, please feel free to contact me at swhume@gmail.com or 484-354-0873. If you have any questions regarding your rights as a research participant in this study, you may contact the DSU Office of Sponsored Programs at 605-256-5100 or at irb@dsu.edu.

Sincerely,

**Sam Hume**

swhume@gmail.com

This project has been approved by the DSU Institutional Review Board, Approval No.: #2016-2017-114
Dear Sam Hume,

The IRB approved your project using expedited procedures as described in 45 CFR 46.110. The activity was deemed to be no greater than minimal risk, and the following expedited category from 63 FR 60364-60367 was found to be applicable to your activity:

(7) Research on individual or group characteristics or behavior (including, but not limited to, research on perception, cognition, motivation, identity, language, communication, cultural beliefs or practices, and social behavior) or research employing survey, interview, oral history, focus group, program evaluation, human factors evaluation, or quality assurance methodologies.

One-year approval of your project will be dated starting January 24, 2017. If you require additional time to complete your project or wish to extend the activity, please submit a request for extension before December 24, 2017. The request can be submitted by email to IRB@dsu.edu. If there are any unanticipated problems involving risks to subjects or others, or if there are changes in the procedures during the study, please contact the Sponsored...
Programs Office at IRB@dsu.edu. Any protocol changes must be approved by the IRB prior to implementation. At the end of the project please inform the committee that your project is complete.

As a university operating within the United States, our faculty, staff and students are required to follow U.S. law. As a researcher affiliated with Dakota State University, you are required to comply with all regulations as outlined in the U.S. Office of Human Research Protections (OHRP)/ Health and Human Services (HHS) Code of Federal Regulations, Title 45, Part 46 (45 CFR 46): Protection of Human Subjects.

Protections for international human subjects should be equal to those in 45 CFR 46. It is your responsibility, as the Principal Investigator of research protocol #2016-2017-114, to ensure that human subjects research conducted internationally adheres to all applicable U.S. law and university policy. If you have questions or concerns about these rules, please work with the Director of Sponsored Programs to resolve them before beginning any phase of your research that requires contact of any kind with human subjects.

If I can be of further assistance, please don’t hesitate to let me know.

Yours truly,

Jack H. Walters, Ph.D.
Chair, DSU Institutional Review Board
REFERENCES


Buneman, P., & Davidson, S. B. (2010). Data provenance--the foundation of data quality.) (Eds.): ‘Book Data provenance--the foundation of data quality ’(2013, edn.).


CDISC. (2008). Operational Data Model (ODM) Course: CDISC.


CDISC. (2011b). Representing Controlled Terminology in ODM: CDISC.


De Lusignan, S., Liaw, S., Krause, P., Curcin, V., Vicente, M., Michalakidis, G., . . . Mendis, K. (2011). Key concepts to assess the readiness of data for International research: Data
quality, lineage and provenance, extraction and processing errors, traceability, and curation. *Yearb Med Inform*, 6(1), 112-120.


FDA. (2010). *Study Data Specifications v1.5.1*. FDA: FDA.


FDA. (2017). *Study data technical conformance guide*. FDA.


Tremblay, M. C., Hevner, A. R., & Berndt, D. J. (2010). Focus groups for artifact refinement and evaluation in design research. *Communications of the Association for Information Systems*, 26, 1.


