2004

Geodatabase use in fire sciences research| The development lifecycle

Lee T. Macholz
The University of Montana

GEODATABASE USE IN FIRE SCIENCES RESEARCH:

THE DEVELOPMENT LIFECYCLE

by

Lee T. Macholz

B.A. University of Montana, 1994

Presented in partial fulfillment of the requirements

for the degree of

Master of Arts

The University of Montana

July 2004

Approved by:

Chairperson

Dean, Graduate School

8/4/04

Date

Macholz, Lee T., M.A., June 2004 Geography

Geodatabase Use in Fire Sciences Research: The Development Lifecycle

Chair: Paul B. Wilson, Ph.D. PW

The Joint Fire Sciences Program (JFSP) is currently sponsoring rapid response research concerning wildland fire. As a component of this research, the JFSP requested that a common database architecture be investigated to facilitate data sharing between multiple research projects. It is the intent of staff at the National Center for Landscape Fire Analysis (NCLFA) to design and develop a functional ArcSDE geodatabase that will integrate the rapid response data collected at the site of the Cooney Ridge fire, which burned in August of 2003 southeast of Missoula, MT. The resulting geodatabase will allow researchers to share their data and build a common data source without duplication of effort or data. The ArcSDE geodatabase is intended to provide multiple-users with access, editing, and analysis capabilities through multiple ESRI GIS applications. This thesis will document the lifecycle of the rapid response geodatabase development process.

The development lifecycle for the rapid response geodatabase will capture all of the stages of the development process including the conceptualization, pre-design, design, development and implementation of the geodatabase. A protocol for geodatabase development is prepared through the combination of software process theory, the principals of database design, spatial database theory, and the rapid response geodatabase lifecycle. This protocol is presented to serve as a guide for future applications of the technology in the Federal fire science research arena.

ACKNOWLEGEMENTS

Thank you to Paul Wilson, Ph.D., Sarah Halvorson, Ph.D., Lloyd Queen, Ph.D., Colin Hardy, Don Helmbrecht, and all the members of the rapid response geodatabase data committee. Thank you also to Anne Rys-Sikora, all my friends and family, and especially my husband, Craig.

TABLE OF CONTENTS

Appendix

LIST OF TABLES

Table | Page

LIST OF ILLUSTRATIONS

Figure | Page

LIST OF ABBREVIATIONS

AES	Autonomous Environmental Sensor
ANSI	American National Standards Institute
ATIR	Airborne Thermal Infrared
BIA	Bureau of Indian Affairs (U.S. Department of the Interior)
BLM	Bureau of Land Management (U.S. Department of the Interior)
CASE	Computer-aided Software Engineering
DBMS	Database Management System
ER	Entity-relationship model
ESRI	Environmental Systems Research Institute, Inc.
FBP	Fire Behavior Package
FGDC	Federal Geographic Data Committee
FTP	File Transfer Protocol
FWS	Fish and Wildlife Service (U.S. Department of the Interior)
GAO	United States General Accounting Office
GIRM	Geospatial Interoperability Reference Model
GIS	Geographic Information System
GPS	Geographic Positioning System
IRMWT	NWCG's Information Resource Management Working Team

ISO	International Standards Organization
JFSP	Joint Fire Science Program
NCLFA	National Center for Landscape Fire Analysis
NIFC	National Interagency Fire Center
NIST	National Institute of Standards and Technology
NPS	National Park Service (U.S. Department of the Interior)
NSDI	National Spatial Data Infrastructure
NWCG	National Wildfire Coordinating Group
OGC	Open GIS Consortium
OMB	Office of Management and Budget
SDE	Spatial Data Engine
SDTS	Spatial Data Transfer Standard
UML	Unified Modeling Language
USDA	U.S. Department of Agriculture
USDI	U.S. Department of the Interior
USGS	U.S. Geological Survey

CHAPTER 1

INTRODUCTION

This thesis explores the creation of a protocol for implementing geospatial technologies in support of wildland fire research. Geospatial technologies are those digital technologies that capture data describing the spatial and non-spatial properties of geographic features on the earth. Advances in the technologies driving remote sensing, geographic positioning systems (GPS), and geographic information systems (GIS) allow geography and its processes to be displayed visually with increasing accuracy. One of the many fields to use these new geospatial technologies is the fire sciences.

Geospatial technologies have been used for wildland fire research and management for many years. However, the standard techniques currently being used are becoming outdated. Fire researchers often utilize GIS to store, display, and analyze geospatial data. Different researchers use different GIS software, and as is the case in many industries, geospatial data is not often shared between researchers. A lack of data sharing leads to isolated islands of data and often to the existence of multiple data sets representing the same phenomenon or location. New GIS technologies provide the ability for multiple geospatial data sets to be combined and shared among users at multiple locations, thus linking islands of data and eliminating redundant data sets. However, this technology has not been widely implemented in the field of fire sciences.

The intent of this thesis is to provide a protocol for the development and implementation of state-of-science GIS technology for data sharing in Federal fire research.

This chapter will introduce the demonstration project that this thesis documents by describing the rapid response research projects and the Cooney Ridge Fire. The purpose of documenting this project lifecycle will then be addressed. This chapter will state the purpose, objectives, and scope of this thesis and the parties involved in the project will be identified.

The Rapid Response Project

Fire is a naturally occurring phenomenon that is often seen by our society as a necessary evil. It is necessary because many natural plant communities depend on fire for the removal of dead materials, nutrient cycling, and regeneration. It is "evil" because it can cause extensive damage to human property and threatens human life. Because fire is a hazard to humans, our society has attempted to exert control over it. This is relatively easily done in our cities and towns but has proven much more difficult on wild lands. Unfortunately, decades of fire exclusion policies have resulted in the accumulation of uncharacteristic amounts of fuel (dead forest materials) in the wildlands of the United States. In the presence of such large fuel sources, wildfires can be larger, harder to control, and more devastating to human property. Land managers have recently recognized the importance of fuels mitigation and the lack of knowledge concerning this topic.

In 1998, Congress directed the U.S. Department of the Interior (USDI), including the Bureau of Land Management (BLM), Bureau of Indian Affairs (BIA), National Park

Service (NPS), Fish and Wildlife Service (FWS), the U.S. Geological Survey (USGS), and the U.S. Department of Agriculture's (USDA) Forest Service to create the Joint Fire Sciences Program (JFSP). The purpose of the JFSP was to fill the knowledge gap regarding wildland fire fuels by addressing four critical issues: fuels inventory and mapping; evaluation of fuels treatments; scheduling of fuels treatments; and monitoring and evaluating fuels treatments.[1]

The JFSP is overseen by a governing board consisting of five representatives from the USDA Forest Service and five representatives from the USDI. This board of governors makes the final decisions on which research projects are funded by the JFSP.[2]

In October of 2002, the JFSP announced a call for proposals that would specifically "obtain, document, and evaluate critical, time-sensitive information or data during or following wildland fire incidents or post-fire land treatments."[3] The resulting proposed projects have been broadly labeled as "rapid response" projects and they will be referred to as such in this document.

There were many rapid response proposals submitted to the JFSP, and the board of governors recognized that there were close ties between several of the proposed projects. In one instance, the board of governors offered a counter-proposal requesting that the proposals by Philip J. Riggan and others and Colin C. Hardy and others be combined into one proposal.[4] The board of governors also directed that the combined

---

[1] Joint Fire Science Program, Joint Fire Science Plan (Available at: *http://jfsp.nifc.gov/JointFire.html*).

[2] Ibid.

[3] Joint Fire Science Program, Announcement for Rapid Response Proposals. (Available at: *http://jfsp.nifc.gov/2003-2_AFP.htm*).

[4] Philip J. Riggan et al., *Wildfire Remote Sensing and Modeling in Support of Operational Fire Management*. (Initial proposal to the Joint Fire Science Program submitted under JFSP RFP-2003-2 Task 1, 2003); Colin C. Hardy et al., *Advancing the Capabilities for Rapid Response Fire Monitoring and Intelligence*. (Initial proposal to the Joint Fire Science Program submitted under JFSP RFP-2003-2 Task 1,

proposal include the development of a database that would be common to several rapid response projects, thereby exploiting the linkages between the projects and allowing researchers to share data. The database would be designed to allow data sharing between the rapid response projects being conducted by Hardy, Riggan, et al.; Finney et al.; and Morgan et al.[5]

Hardy approached the National Center for Landscape Fire Analysis (NCLFA) at the University of Montana College of Forestry and Conservation to lead the effort of designing and developing the common database. Hardy requested that a common database be designed to hold both spatial and tabular data that supports the rapid research projects identified by the JFSP Board of Governors. The NCLFA viewed this project as an opportunity to demonstrate the recent advancements in geospatial database design and GIS technology. NCLFA staff worked with the rapid response project investigators, their staff, and consultants from Environmental Systems Research Institute, Inc. (ESRI) to design, develop, and implement an ArcSDE geodatabase using ESRI technology.[6] The ArcSDE platform allows the viewing, editing, and dissemination of data through several ESRI software applications.

2003); Colin C. Hardy, Philip J. Riggan, et al., *Demonstration and Integration of Systems for Fire Remote Sensing, Ground-Based Measurement, and Fire Modeling*. (Combined proposal to the Joint Fire Science Program submitted under JFSP RFP-2003-2 Task 1, 2003).

[5] Colin C. Hardy, *Demonstration and Integration of Systems*; Mark Finney et al., *Modeling Surface Winds in Complex Terrain for Wildland Fire Incident Support*. (Proposal to the Joint Fire Science Program submitted under JFSP RFP-2003-2 Task 1, 2003); Penelope Morgan et al. *Assessing the Causes, Consequences and Spatial Variability of Burn Severity: A Rapid Response Approach*. (Proposal to the Joint Fire Science Program submitted under JFSP RFP-2003-2 Task 1, 2003).

[6] See Glossary for definitions of "SDE" and "geodatabase."

The Cooney Ridge Fire Database Prototype

A geodatabase was developed for, and populated by, a subset of data collected by the rapid response projects during a burnout operation at the Cooney Ridge fire.[7] The Cooney Ridge fire, located south of Missoula, MT and Interstate 90, west of Rock Creek, and east of the Bitterroot Valley foothills (Figure 1), burned approximately 24,000 acres in August of 2003.

The Cooney Ridge fire was chosen for the rapid response studies because the researchers identified that they needed to study a wildfire within one day's travel of Missoula, MT where a sample site could be efficiently and safely accessed. The study requirements also necessitated: 1) pre-burn data exist for the area; 2) the fire be of mixed-severity; 3)the fire be in mixed-conifer forest; 4)the site be of moderate terrain; and 5) a vantage point from across a drainage or valley provide an oblique view of the sample site.[8] The location and characteristics of one flank of the Cooney Ridge fire satisfied all of these requirements.

Data collected by the principal investigators at the site of the Cooney Ridge burnout were numerous and diverse. Pre-burn data were collected by Hardy's team from the Missoula Fire Sciences Laboratory (U.S. Forest Service, Rocky Mountain Research Station) and included a site characterization for fuels composition and canopy density. Hardy's and Riggan's research teams also collected data during the fire, including: radiometric data (radiant and total heat flux) both on-site and off-site; weather data such as temperature, wind speed, and wind direction; airborne thermal remote sensing; and

---

[7] Burning out is when a fire is lit within a control line for the purposes of consuming fuel between the control line and the fire. National Wildfire Coordinating Group. *Firefighters Guide.* (Boise, 1986) NFES 1571 PMS 414-1. 70-2.

[8] Colin C. Hardy, Personal Communication, November 24, 2003.

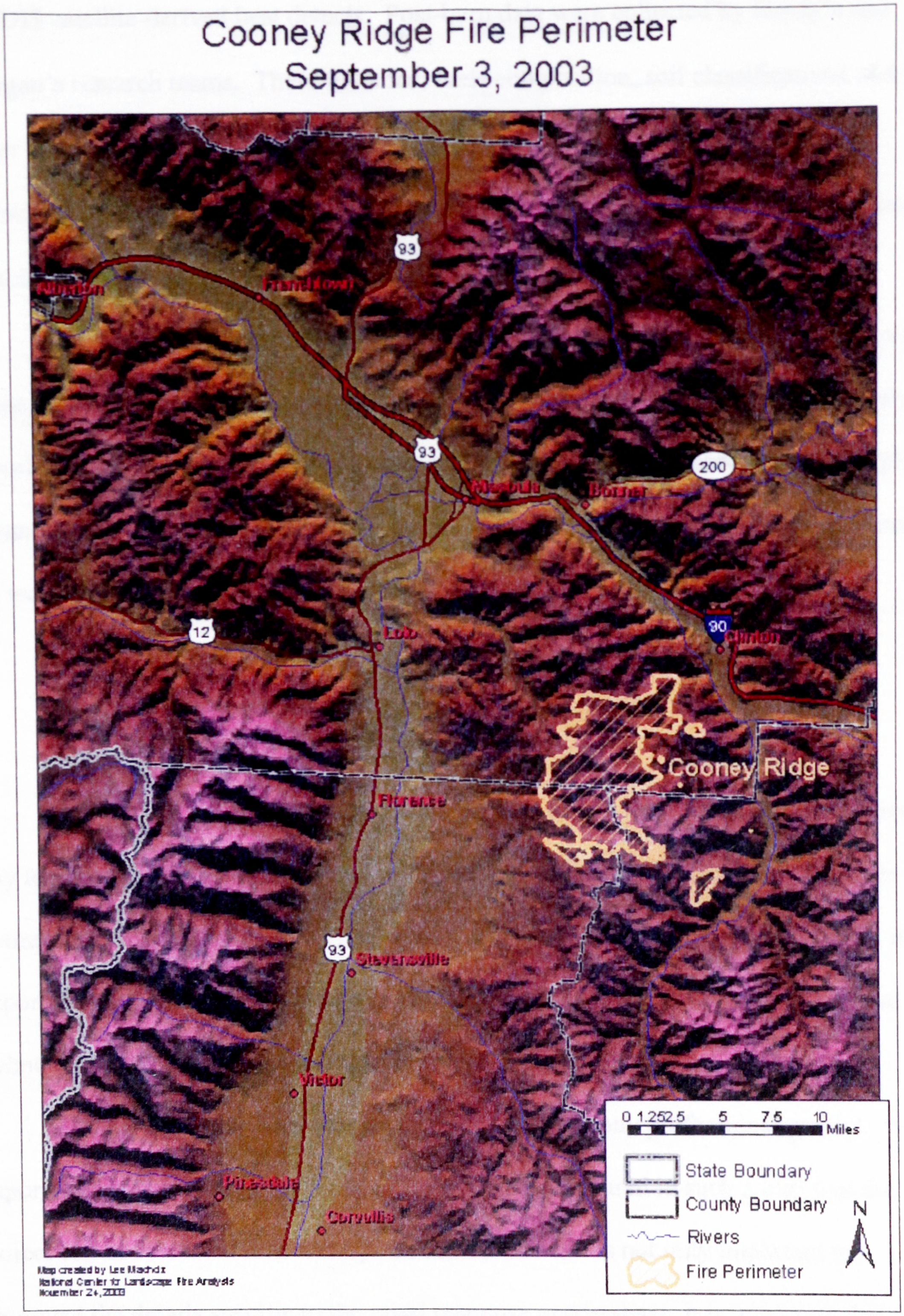

Figure 1—Map showing Cooney Ridge fire.

MODIS satellite-derived heat detects. Post-burn data were collected by Hardy's and Morgan's research teams. These included, fuels composition, soil classifications and soil water repellency, canopy density in combination with slope and aspect data, and re-vegetation measurements. The rapid response geodatabase incorporates all of the above data and their spatial components.

The geodatabase developed by the NCLFA will be presented to the JFSP as a part of the rapid response project conducted by Hardy, Riggan, and others. It will not only be a working database for the rapid response projects, but it will also serve as an example of advanced geospatial database and GIS technology for those who may wish to implement the technology in the future.

Documenting the Lifecycle

Providing the JFSP with an example of SDE geodatabase technology is a good way to showcase the advantages and utility of an advanced GIS. However, this does not guarantee that the technology will be widely accepted or even understood. Thus, it is important that the development process be captured and presented in such a way that the technology is accessible and useful to everyone.

The documentation process is common to almost any professional project. It is important that decisions, processes, and variables be captured in such a way that the project can be recreated if necessary. In this instance, it is not only important to document the details specific to the rapid response geodatabase, it is important that the lifecycle of the project be captured in such a way that it can be used to guide a new geodatabase project. This thesis will synthesize the rapid response geodatabase lifecycle into a protocol for ArcSDE geodatabase development.

The federal government dominates the research and management of wildland fire-related issues. Because the NCLFA is not a federal entity, it is not limited by the standards and business practices enforced by federal agencies. However, the rapid response geodatabase was created to support a series of federal research projects and is a product for the JFSP, which is a federal entity. Any implementation of this technology by the JFSP or anyone else in the future will most likely be associated with a federal agency. Therefore, it is important that federal standards and business practices be acknowledged and included in a protocol for developing an ArcSDE geodatabase system.

Statement of Purpose

An ArcSDE geodatabase is being created by the National Center for Landscape Fire Analysis to support research sponsored by the Joint Fire Sciences Program. The rapid response geodatabase will contain data collected by Hardy, Riggan, and others; Finney and others; and Morgan and others at the site of the 2003 Cooney Ridge Fire. The geodatabase will serve as a data repository that the researchers and their cooperators can access to view, analyze, edit, or retrieve data. The geodatabase will also serve as an example for future applications of the technology.

The purpose of this thesis is to document the lifecycle of the rapid response geodatabase from conception through implementation. This thesis will also identify each step in the geodatabase development process where federal standards or business practices would impact

the lifecycle. The result will be a protocol for geodatabase development for Federal fire sciences research.

Objectives

The following objectives will be addressed in this thesis:

- Identify the steps involved in the design, development, and implementation of an SDE geodatabase.
- Describe the general constraints on the lifecycle of development and implementation.
- Determine the processes and constraints specific to a federal agency.
- Identify existing standards that would apply to this lifecycle.

Scope

The scope of the research will be confined to the development of an ArcSDE geodatabase for the rapid response research being conducted on the Cooney Ridge fire of 2003. This thesis is intended to capture the lifecycle of the geodatabase development process to serve as a guide for future applications of the technology in similar circumstances.

The Players

There are several individuals and groups involved in the rapid response geodatabase development project. First, there are the principal investigators (PIs) and their staff who are conducting the rapid response research. These researchers are the clients, and the geodatabase is being created to allow them to share data and to meet their research goals. Second, there is the NCLFA—the service provider. The NCLFA GIS Program Manager (Don Helmbrecht) and the author are leading the development of the geodatabase. Third is ESRI. The geodatabase will be based on ESRI technology and the NCLFA will contract with ESRI to provide training and support throughout the development process. It is important to note that no single player will bear sole responsibility for the rapid response geodatabase as a whole. Rather, each group will be responsible for various aspects of the project, with the author also being responsible for the documentation of the lifecycle (this thesis).

Chapter Summary

This chapter described the emergence of geospatial database technologies within the field of the fire sciences. The rapid response geodatabase project was introduced, as was the data source for the prototype geodatabase. The purpose of this thesis was identified as the documentation of the development lifecycle of the rapid response geodatabase and the subsequent creation of a protocol for geodatabase development. The objectives and scope of this thesis were stated and the project participants were identified. The following chapter will address the theoretical background and methodology for this thesis.

CHAPTER 2

THEORY AND METHODOLOGY

This chapter will discuss the theoretical basis and methodology for this thesis. The theoretical basis of the thesis will be drawn from several areas of academic literature addressed here in six sections. First, the author will examine the state of spatial database use in federal fire management. Second, an overview of software process theory will be provided. Third, the principals of database design will be discussed. Fourth, the theory behind spatial database design will be addressed. Fifth, the importance of interoperability in the design and use of shared GIS applications will be reviewed. Lastly, the existing and proposed federal standards for geospatial information specific to the fire sciences will be investigated.

Spatial Database Use in Federal Fire Management

The federal government has identified that the interagency management of geospatial information and technology is not adequate for the successful use of these technologies for federal fire management. The U.S. General Accounting Office (GAO) recently issued the report, *Geospatial Information: Technologies Hold Promise for Wildland Fire Management, but Challenges Remain.*[9] This report identifies the technologies being used in support of wildland fire management, the challenges to using those technologies effectively, and opportunities to improve the effective use of geospatial technologies nationally.

The challenges identified by the GAO include: geospatial data are not always readily available; neither data nor systems are interoperable; there is no inventory of the systems that are in use; there is often limited access to equipment, software, communications, or the internet at remote fire sites; the training of GIS technicians is inconsistent; and there has been a failure to use state-of-science technology.[10] These problems will not be easily rectified.

The National Wildfire Coordinating Group (NWCG) initiated the development of a strategic plan for the use of geospatial technologies at the interagency level. The NWCG has also initiated the development of an Information Resource Management strategy and an enterprise architecture.[11] Though these initiatives are steps in the right direction, they have not received the support or funding necessary to be successful.

Software Process Theory

The field of software engineering can be linked to almost any field of study, including the fire sciences. Technology is advancing so rapidly that software applications are being developed for almost any field of study that can be named. The rapid response geodatabase being discussed here is not a custom software application, but

---

[9] U.S. Department of the Interior. General Accounting Office. *Geospatial Information: Technologies Hold Promise for Wildland Fire Management, But Challenges Remain.* GAO-03-1047 (Washington, D.C., September 2003).

[10] Ibid, 2-3.

[11] Ibid, 3.

the same software process can be applied to the development of this geodatabase as would be applied to the development of any custom application.

A software process is a sequence of activities that result in the production of a software product.[12] In his book, *Software Project Management*, Joel Henry explains that the software process "forms the basis of all the work your team will do—how team members know when to do what, and why; what lies ahead and what just passed; what tasks are performed and how they fit together."[13] Just like a house should not be built without a set of blueprints, a software product should not be built without a documented process.

When beginning a software development project, it is necessary to identify the software process that will be used. There are many process models existing in the literature that can be tailored to fit the proposed project. Two common models are the staged (called "waterfall") model, and the spiral model. The waterfall model represents the software process as a series of stages. In theory, each stage is completed before the next stage is begun, and the process for each stage is dependent on the results from previous stages.[14] In reality, the stages overlap and feed each other incrementally. The spiral model represents the software process as a series of loops, where each loop represents a phase of the project and where each loop circles back through a risk analysis phase.[15] There is less backtracking in the spiral model, and there is more focus on risk throughout the process than in the waterfall model. The rapid response geodatabase project will use a software process based on the waterfall model.

---

[12] Joel Henry. *Software Project Management.* (Boston: Addison-Wesley, 2003), 25. Ian Sommerville. *Software Engineering.* (England: Pearson Education, Ltd., 2001), 8.

[13] Henry, 25.

[14] Sommerville, 45.

[15] Ibid, 53.

Sommerville identifies the four basic tasks in a software process as **software specification, development, validation**, and **evolution.**[16] **Software specification** is intended to determine the requirements and constraints for the software product. A feasibility study must be conducted to identify existing hardware and software and determine additional technology needs.[17] System requirements are then elicited from the users, analyzed, documented, and validated. The documentation takes the form of specifications that describe the system requirements in abstract terms for the users and describe the system functionality in detail for the developers. Validation ensures that the requirements are realistic, consistent, and complete. The result of the software specification task is a detailed requirements document that will be used to guide the development task.

Sommerville defines **software development** as, "the process of converting a system specification into an executable system."[18] In other words, the development process incorporates the design, creation, and implementation of the software. The development process may employ structured design methods, which provide guidelines, tools, and standardized notation for the software design.[19] The notation used in a structured design method is actually provided by an underlying modeling language. The Unified Modeling Language (UML) is a standardized modeling language that can be used to provide the graphical notation to any given design method.[20]

---

[16] Ibid, 55.
[17] Ibid, 56.
[18] Ibid, 56.
[19] Ibid, 58.
[20] Martin Fowler and Kendall Scott. *UML Distilled: Applying the Standard Object Modeling Language*. (Reading: Addison Wesley Longman, Inc., 1997), 1.

Structured methods typically involve producing models of the system in graphical format. There are many types of models that can be used, one of which is the entity-relationship model. The entity-relationship model is the most common model for describing databases and will be described in more detail in the Principals of Database Design section. The result of the software development task is a fully functional software product.

The **software validation** task is intended to take the fully functional software product and test it to ensure that it follows all of the specifications and meets all of the requirements. Depending on the software process being used, the validation task can occur either incrementally throughout the development process or wholly at the end. Either way, testing is approached in stages. Individual software components are tested independently first, then in increasingly large collections through the sub-system and system levels.[21] Validation is a cyclical process of testing and modification, which will result in a final software product.

Software evolution is an ongoing task throughout the life of the software product. In this time of rapid technological growth, it is not uncommon that as soon as a software product is released, it is antiquated. In order to be viable, the software product must be flexible enough that it can change and evolve as the users' needs and requirements change over time.[22]

Software processes can be enhanced through the use of Computer-aided Software Engineering (CASE) tools. CASE tools are used to develop and maintain software by

---

[21] Sommerville, 61.
[22] Ibid, 63.

automating certain tasks.[23] There are innumerable CASE tools available ranging from simple macros to more complex software applications. It is important to carefully consider the use of CASE tools because improper use can hinder the software process.[24]

According to Henry, the final issue to address in the discussion of software process is process assessment.[25] One of the advantages to implementing a software process is that a well-documented process is re-usable. In order to improve upon the process next time, it is important to perform a process assessment. Too often this task is set-aside until the end of the project. When this happens, details are forgotten, people move on to other tasks, and the same undocumented inefficiencies plague the next project. Henry suggests that process assessment tasks be incorporated with development tasks and concurrent documentation. Thus, the post-project assessment activities are confined to analyzing existing data and implementing improvements into the software process.

Principals of Database Design

The purpose of a database is to store, organize, catalog, and retrieve a collection of information. The goal of database design is to "ensure efficient data processing through the elimination of redundant information and the minimization of update and deletion problems."[26] There are three models for database design: relational, distributed, and object-oriented. Relational database design is based on the theory of a mathematical

---

[23] Ibid, 64.
[24] Henry, 5.
[25] Ibid, 40.
[26] R. Norbeto Fernandez, Marek Rusinkiewicz, Lucia Morais da Silva, and Chris J. Johannsen, "Design and Implementation of a Soil Geographic Database for Rural Planning and Management," *Journal of Soil and Water Conservation*. 48 (1993): 141.

relation—ordering values into tables.[27] The tables are then linked together through columns of common values. Distributed database design allows for multiple databases to be integrated into a distributed system by creating linkages between the independent databases.[28] Object-oriented database design is a newly evolving design theory for storing and processing object-oriented programming data structures.[29] Object-oriented databases are compatible with a number of object-oriented programming languages. They allow the designer to more completely incorporate databases into software applications by providing for storage of complex objects and their associated operations.[30] The rapid response geodatabase will be developed on the relational database model.

Elmarsi and Navathe describe the database design process as starting with the collection of user requirements.[31] Subsequent analysis breaks the list of requirements into **functional requirements** and **data requirements. Functional requirements** consist of the operations that the users apply to the database. The result of the functionality analysis is a document specifically detailing the users' requirements of the database.[32] **Data requirements** are used to create a conceptual schema, visually describing the users' needs. The conceptual schema details the database structure, which consists of the data types, relationships, and constraints that will maintain the integrity of the data.[33] The schema allows designers to ensure that all data requirements are met

---

[27] Ramez Elmarsi and Shankant B. Navathe, *Fundamentals of Database Systems*, 4th ed. (Boston: Pearson Education, Inc., 2004), 125.

[28] David M. Kroenke, *Database Processing Fundamentals, Design, and Implementation*, 7th ed. (Upper Saddle River: Prentice Hall, 2000), 20.

[29] Ibid, 20.

[30] Ibid, 640.

[31] Elmarsi and Navathe, 50.

[32] Ibid, 52.

[33] Ibid, 26, 52.

without conflicts. The conceptual schema does not include implementation details, thus it provides a less-technical diagram of the database and can be used as a communication tool.

After the functional analysis and conceptual schema tasks are complete, the next step is the logical design of the database. The logical design involves the implementation of the conceptual schema in a commercial database management system (DBMS).[34] Once the data model has been implemented, the database design process moves into the physical design stage. During this stage, the internal storage structure of the database and the applications that will be used to interact with the database are developed.[35] The result is the final implementation of the database and its associated applications.

The relational database structure can be portrayed using the Entity-Relationship (ER) model. The ER model uses UML to express a series of ER diagrams. These ER diagrams describe the data as entities and attributes and further describe the relationships that exist among these entities and their attributes. Entities are objects and attributes are properties that describe each entity. Entities can be classified into types. Entity types are collections of entities that share attributes, where each entity maintains its own attribute values. An entity type in an ER diagram is equivalent to a table, where entities are stored in table rows and attributes are stored in table columns. Entities within an entity type are constrained by the fundamental rule that each entity must be unique. The attribute, or attributes, that uniquely identify each entity are known as key attributes.[36]

The interactions between entity types within the ER model are described as relationships. When two entity types (tables) contain different attributes for the same

---

[34] Ibid, 52.
[35] Ibid.
[36] Ibid, 53-57.

entities, the entity types can participate in a relationship. However, the two entity types must contain a common attribute field containing the same attribute values in order to establish a relationship.[37] The relationships present within a database can be described by their cardinality. Cardinality refers to the number of times a particular entity occurs within an entity type. The relationships between entity types are often expressed with a cardinality ratio; or the maximum number of times each entity can participate in the relationship. There are three cardinality ratios expressed in a database: one-to-one (1:1), one-to-many (1:M), and many-to-many (M:M).[38]

A one-to-one cardinality ratio specifies that each entity in one entity type matches only one entity in the other entity type. A one-to-many cardinality ratio specifies that each entity in one entity type match multiple entities in the other entity type. A many-to-many cardinality ratio specifies that multiple entities in one entity type match up with multiple entities in the other entity type.[39] Entities are "matched up" when the attribute values within the attributes common to each entity type are equal.

The process of logically grouping entities and their attributes and creating relationships between entity types through the use of the ER model is formalized through a process called normalization. Normalization minimizes data redundancy and update anomalies.[40] Fundamentally, this process checks for the desirability and correctness of relationships within a database.[41] Ideally, relationships should provide a table structure in which the minimum number of duplicated attribute fields occurs within the database.[42]

---

[37] Ibid, 62.
[38] Ibid, 65.
[39] Ibid.
[40] Ibid, 313.
[41] Kroenke, 113.
[42] Elmarsi and Navathe, 298.

This is important because data duplication reduces storage space and efficiency within the database.

The removal of update anomalies is also important to the integrity of the database. Update anomalies can be classified into three specific error anomalies: insertion, deletion, and modification. Insertion anomalies occur when data entered into attributes that participate in a relationship are entered inconsistently or when a parent attribute that is required to establish a relationship is not present within the database before dependent attributes are entered. Deletion anomalies occur when the deletion of an entity results in the loss of required attributes of another entity. Modification anomalies occur when changing one attribute value requires that all related attribute values also be changed.[43] The normalization process is intended to minimize the occurrence of update anomalies.

There are five levels of normalization: first normal form, second normal form, third normal form, Boyce-Codd normal form, and fourth normal form.[44] A relationship is in first normal form if it meets the definition of a relation.[45] Kroenke identifies the following five rules as defining a relation: cells may contain only one single value; columns may contain only values of the same kind; columns within a table must be uniquely named; each row in a table must be unique; column and row order do not matter.[46] A relationship is in second normal form if all attributes within the tables are functionally dependent.[47] A functional dependency means that if the user has one attribute value, that can be used to find other attributes of the same object.[48] In other

---

[43] Ibid, 300.
[44] Elmarsi and Navathe, 315-326; Kroenke, 120-125.
[45] Elmarsi and Navathe, 315; Kroenke, 120.
[46] Kroenke, 114
[47] Elmarsi and Navathe, 318; Kroenke, 121.
[48] Kroenke, 114

words, in second normal form, if one attribute is removed, the integrity of the table fails. A relationship is in third normal form if it meets the requirements of second normal form and there are no transitive dependencies.[49] An example of a transitive dependency would be when three attributes are dependent upon one another where attribute A determines attribute B, and attribute B determines attribute C, but attribute C does not determine attribute A.[50] A relationship is in Boyce-Codd form if every attribute within the relation must be used to define the primary key.[51] That is, every attribute field must be used to identify each entity as unique. Finally, a relationship is in fourth normal form if it meets the requirements for Boyce-Codd normal form and there are no multi-value dependencies.[52] Multi-value dependencies occur when each entity can occur within a table multiple times, each time with different attribute values.

Relational database design should incorporate the use of both the ER model and the normalization process. The ER model allows the designers to group entities and attributes logically and the normalization process provides a way to check the resulting database structure for sources of redundancy and error.

Geospatial Database Design

There is a wide body of literature regarding the design of spatial databases. A majority of sources discuss this topic in respect to the creation of project-specific databases. There were no articles found that simply presented a "how-to" or protocol for

[49] Elmarsi and Navathe, 320; Kroenke, 122.
[50] Kroenke, 585.
[51] Elmarsi and Navathe, 324; Kroenke 123.
[52] Kroenke, 124.

spatial database design. The theory of spatial database design culminates into three essential phases: **conceptual modeling, logical design**, and **physical design.**[53]

The purpose of the **conceptual modeling** phase is to model the user's view of the data by describing the geographic objects of interest and the relationships between those objects.[54] Zeiler recommends that these objects and their relationships be identified and drawn using simple UML class diagrams.[55] Once all of the objects of interest have been identified, their attribute values should be identified, and cataloged.[56] The result of the conceptual modeling phase is a conceptual schema—a graphical representation of the spatial database structure including all of the data objects, their attributes and their relationships to each other.[57]

During the conceptual modeling phase it is important to pay attention to the semantics of objects and their attributes. This is the stage where it is necessary to create naming conventions for object and attribute field names. A spatial database must use clear, concise, standardized terminology for its naming convention. It is necessary to consider the fact that multiple user groups within and between disciplines may use different terminology to refer to the same phenomenon; or they may use the same terminology to refer to a different phenomenon.[58] If standardized terminology exists

---

[53] Fernandez et al., 141; Philippe Rigaux, Michel Scholl, and Agnes Voisard. *Spatial Databases: With Application to GIS* (San Francisco: Morgan Kaufmann, 2002), 5.

[54] Rigaux et al., 6.

[55] Michael Zeiler. *Modeling Our World: The ESRI Guide to Geodatabase Design*. (Redlands: Environmental Systems Research Institute, Inc., 1999), 16.

[56] David Arctur, David Hair, George Timson, E. Paul Martin, and Robin Fegeas. "Issues and Prospects for the Next Generation of the Spatial Data Transfer Standard (SDTS)." *International Journal of Geographical Information Science* 12 no. 4 (1998): 403-425.

[57] Rigaux et al., 7.

[58] Yaser Bishr, "Overcoming the Semantic and Other Barriers to GIS Interoperability," *International Journal of Geographical Information Science* 12 no. 4 (1998): 299-314.

within a discipline, it is important that it be used to allow the system to be shared between user groups.

Naming conventions must adhere to the limitations of the DBMS being used. Each system will have specific limitations on the number and type of characters in an object's name. For example, when utilizing ESRI's ArcSDE geodatabase technology, the total number of characters in the owner and feature class name should not exceed 25 characters.[59] If this limit is exceeded, the object will not be editable.[60] Special characters such as spaces, asterisks, quotation marks, etc., are often not allowed in object or field names. Each system will also have a list of reserved words that may not be used as object or field names within a database. Reserved words are often words such as: date, time, timestamp, year, and zone.

The second phase of the spatial database design process is **logical design**. The first task during the logical design phase is to identify the geographical representation for each object in the conceptual schema: point, line, polygon, or raster image.[61] The relationships between geographic objects—or their topology—need to be described in more detail during the logical design phase. The relationships between tables also need to be described in more detail. The issue of cardinality must be addressed and table structures re-organized, if necessary, to support the required functionality of the spatial database.

---

[59] See glossary for a definition of "feature class."

[60] Environmental Systems Research Institute, Inc. Web site: *http://support.esri.com/knowledgeBase/documentation/FAQs/sde_/WebHelp/faq.htm.*

[61] Zeiler, 190.

In theory, the logical design phase includes an assessment of the processing requirements for the database, and a choice of which DBMS will be used.[62] In reality, the software already used by, or available to, the user typically dictates this choice. However, it is important to evaluate the processing requirements for the database and ensure the chosen DBMS will meet those requirements. The choice of a DBMS is particularly important if the spatial database is being created independent of an existing GIS application. A purely relational database is not adequate for handling and processing spatial data.[63]

There are two approaches to the problem that a pure relational DBMS will not handle spatial data. The first is a loosely-coupled DBMS approach where descriptive data is stored in a DBMS and the spatial data is managed in a separate structure outside the DBMS.[64] An example of this approach is the shapefile. The ultimate goal, however, is to store all of the data in one structure. The second approach is to extend the functionality and query language of a DBMS, to be able to handle spatial data.[65] A DBMS can be extended either by building a subsystem to interact with the database and performing the spatial querying operations outside of the DBMS or by modifying the query language and creating new algorithms that will perform the processing requirements within the DBMS.[66] ESRI's ArcSDE software is an example of an application that runs parallel to a DBMS, controlling the server-client connections and interpreting spatial queries between the GIS application and the DBMS.

---

[62] Rigaux et al., 7.

[63] David J. Able, Beng Chin Ooi, Kian-Lee Tan, and Soon Huat Tan. "Towards Integrated Geographical Information Processing." *International Journal of Geographical Information Science* 12, no. 4 (1998): 353-371; Rigaux et al., 22.

[64] Rigaux et al., 24.

[65] Ibid, 25.

[66] Able et al., 353-371; Rigaux et al., 25.

The result of the logical design stage is a modified conceptual schema that includes all of the detail necessary to create the physical structure of the spatial database.

The third phase of the spatial database design process is the **physical design** phase. The physical design phase is when the conceptual schema is converted into the data structure of the chosen DBMS, the DBMS is populated with data, and the user interface is built and implemented.[67] The physical design phase is analogous to the software validation phase in software process theory where the system cycles through testing and modifications until it meets the functionality requirements of a spatial database.

Once the physical structure of the DBMS has been created from the conceptual schema, the database can be populated with data. Before using the populated database, it is necessary to create the external structure of the DBMS. The external structure is the view of the database through an application.[68] For the rapid response geodatabase, the external structure involves the use of ArcSDE to allow various ESRI applications and multiple users to access the geodatabase.

The theory behind spatial database design is somewhat vague because there are innumerable ways—and reasons—to implement spatial databases. The theoretical basis of spatial database design will be complemented by the theory behind both the chosen software environment and the field of study. The rapid response geodatabase will be built upon basic spatial database theory in conjunction with the specific theory behind the ESRI geodatabase data model.

---

[67] Rigaux et al., 8.
[68] Ibid, 5.

ESRI's Geodatabase

ESRI has developed a spatial database structure commonly known as the geodatabase. Until recently, spatial data was most commonly stored in file-based formats where the coordinate information and attribute information were stored in separate files. Shapefiles, coverages, and CAD files are all examples of file-based spatial data models. The geodatabase differs as it allows both the coordinate information and the attribute information to be stored within one database.[69] Collections of similar geometry types (points, lines, or polygons) are stored as feature classes.[70] Feature classes that share the same spatial reference can be grouped into feature datasets. Feature datasets can include spatial features and relationships.[71]

The geodatabase data model allows spatial data (objects and their attributes) to be stored together in a way that supports advanced rules and relationships between the data. The geodatabase supports several methods of attribute and spatial validation. Attribute validation can occur through the use of subtypes, domains, and relationship classes. Subtypes allow spatial features within a feature class to be grouped into subsets based on attribute values.[72] Domains limit the values that can be entered into an attribute field by declaring explicit acceptable values (coded domain) or a range of acceptable values (range domain).[73] Relationship classes create a permanent link between two feature classes, between a feature class and a table, or between two tables within the geodatabase. Spatial validation can occur through the use of topology rules or geometric

---

[69] Makram Murad-al-shaikh, Krista Page, Mark Stewart, and Marnel Taggart. *Introduction to ArcGIS I.* (Redlands: Environmental Systems Research Institute, Inc. 2003). 5-10.

[70] Zeiler, 64.

[71] Zeiler, 8.

[72] Makram Murad-al-shaikh, Krista Page, Mark Stewart, and Marnel Taggart. *Introduction to ArcGIS II.* (Redlands: Environmental Systems Research Institute, Inc. 2003). 9-3.

[73] Ibid, 9-3

networks. Topology rules allow the user to define and enforce the spatial relationships between features within one or more feature classes. Topology rules can define the adjacency, coincidence, or connectivity between features.[74] Geometric networks allow the user to define the connectivity and direction of flow between features. A geometric network provides the ability to perform direction-of-flow and other path-based analyses.[75]

A geodatabase can be developed at two functional levels: the personal geodatabase, or the enterprise (multi-user) geodatabase. Both levels use a DBMS engine in combination with GIS software to provide spatially-based functionality. The enterprise geodatabase uses a more robust DBMS (SQL Server, Oracle, Informix, and DB2) than a personal geodatabase, which uses Microsoft Access. The enterprise geodatabase also requires a more robust method for connecting multiple users to the DBMS. These connections are handled by ESRI's ArcSDE platform.

ArcSDE provides the infrastructure that links ESRI GIS software to a geodatabase and controls access, querying, editing, and versioning of the geodatabase. For most users, ArcSDE is an invisible string that connects their desktop GIS with the data they require. For more advanced users, ArcSDE allows them to view different versions of the same geodatabase, edit the same feature that another user is editing and choose which edit to save, and check-out portions of the geodatabase to view and edit on their desktop or a mobile unit and check those edits back in to the geodatabase.[76] ArcSDE can be used

---

[74] Ibid, 9-14

[75] Ibid, 7-7.

[76] Environmental Systems Research Institute, Inc. *Introduction to ArcSDE: Lectures*, (Redlands: Environmental Systems Research Institute, Inc. April 2003).

to establish a connection between a geodatabase and a host of ESRI GIS software including ArcGIS (desktop GIS), ArcIMS (Web-enabled GIS), and ArcPad (mobile GIS).

Whether implementing a personal geodatabase or an enterprise geodatabase, the proper spatial reference information must be determined. The spatial reference of a geodatabase refers to the spatial domain for the entire geodatabase as well as the coordinate system, spatial domain, and precision for each feature class. The spatial domain is the allowable coordinate range for x,y coordinates. Essentially, the spatial domain defines the maximum spatial extent to which the data can grow. The precision describes the number of decimal places that will be stored for each spatial coordinate. In other words, the precision value allows the user to tell the geodatabase to store a specific number of decimal places in order to maintain the desired coordinate precision.

A geodatabase stores coordinate values as positive integers. The maximum integer value that can be stored is about 2.14 billion map units.[77] However, spatial coordinates rarely come as positive integers. Thus, the geodatabase must be able to shift the original coordinate values into positive coordinate space, be able to retain the decimal values, and result in a value that is less than 2.14 billion.[78]

In order to position the data within the maximum spatial domain allowed by the geodatabase, an x,y shift is applied to the spatial data (Figure 2). The geodatabase stores the modified values, shifting them back to their original values for display.

[77] Map units are determined by the projection. For example, the map units for a coordinate value projected in UTM will be meters, and the map units for an unprojected coordinate value will be decimal degrees.

[78] Environmental Systems Research Institute, Inc. *Introduction to ArcSDE: Lectures*, 5-3 – 5-7.

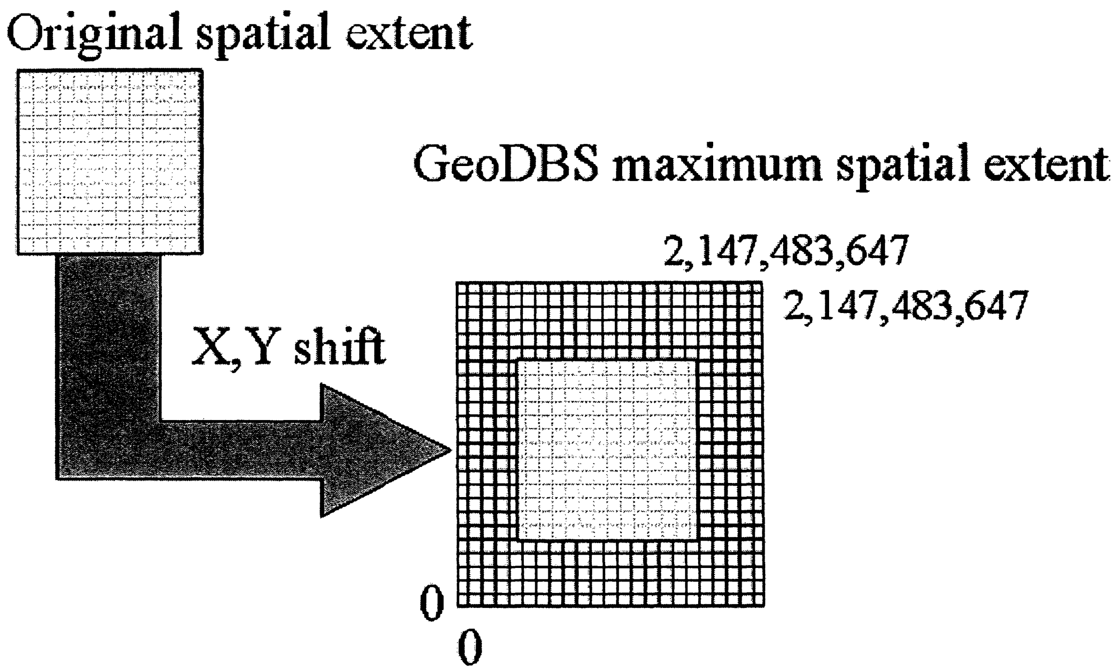

Figure 2—An x,y shift is applied to center the data within the maximum spatial extent of the geodatabase.

In order to retain the decimal values of the original coordinate value, the original value is multiplied by a given precision value. Thus, the precision describes the number of decimal places that will be stored for each coordinate value. Figure 3 shows an example of the effect of the chosen precision value. It illustrates that a low precision value will cause the coordinate value to be generalized, causing the feature to loose resolution. However, a precision value that is set too high will cause the coordinate to fall beyond the maximum spatial extent of the geodatabase. In Figure 3, when the precision is set to 1,000,000,000 the stored coordinate value exceeds the maximum integer value that a geodatabase will store.

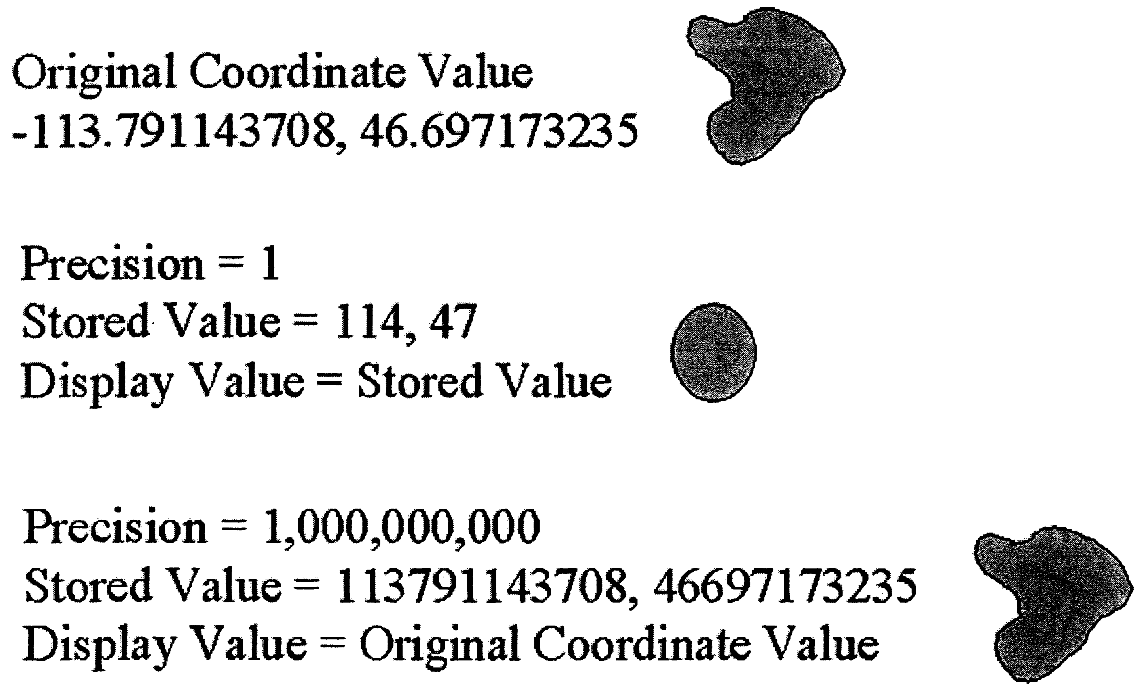

Figure 3—An example of the effect of precision.

The spatial domain and spatial precision values are inversely proportional. Increasing the precision increases the stored coordinate value, which decreases the spatial domain. The spatial reference must be assigned such that the spatial domain and precision are balanced and an acceptable coordinate accuracy is maintained, while allowing for growth within the geodatabase.

Interoperability

There are many definitions of interoperability in the literature. Bishr defines interoperability as the "ability of a system, or components of a system, to provide information portability and inter-application cooperative process control."[79] ESRI focuses its definition of interoperability from a general system to a GIS:

---

[79] Bishr, 299.

> "...[A]n open GIS system allows for the sharing of geographic data, integration among different GIS technologies, and integration with other non-GIS applications. It is capable of operating on different platforms and databases and can scale to support a wide range of implementation scenarios."[80]

The definition that will be used for the rapid response SDE geodatabase project is: the ability to share disparate data sets among multiple platforms, databases, development languages, and applications.

Bishr identifies six levels of interoperability: network protocols, hardware and operating systems, spatial data files, database management systems (DBMS), data models, and application semantics.[81] Because you can have some level of interoperability at any of these six levels, the question becomes: "When is an information system considered interoperable?"

According to Bishr, there is "no known GIS that provides interoperability at the data model and application semantics levels."[82] For the purpose of this research, the focus will be placed on interoperability at what Bishr calls the DBMS level. In order for an information system to be interoperable at the DBMS level, the users need to be able to establish a connection between systems and query the remote system with their own query language to display and analyze the remote data.[83] In order to do this, the users will need to have prior knowledge of the data model and semantics being used.

Devogele and others identify several solutions that will lead to interoperability.[84] One solution is to create a global catalog of information sources and their metadata, all of

---

[80] Environmental Systems Research Institute, Inc. *Spatial Data Standards and GIS Interoperability*, (Redlands: Environmental Systems Research Institute, Inc. January 2003).

[81] Bishr, 300.

[82] Ibid, 312.

[83] Ibid, 310.

[84] Thomas Devogele, Christine Parent, and Stefano Spaccapietra. "On Spatial Database Integration." *International Journal of Geographical Information Science* 12 no. 4 (1998): 335-352.

which can be browsed online. Unfortunately, nobody has assumed responsibility for this huge task. The next solution is standardization. Standardized data models, naming conventions, and metadata allow data exchange between heterogeneous sources and systems. However, the problem of converting existing data sets to meet new data standards remains. Another solution that Devogele and others explore is interoperability through software connectivity, where software packages can connect to different databases and allow data exchange. In this solution, proprietary applications and data structures are problematic. Of course, all of these options are easier said than done, and the key to interoperability lies in a combination of the above approaches.

Federal Standards

The federal government recognized early on that duplication and redundancy of spatial data would be a costly issue in time, money, and quality control. Thus, the federal government has been trying to coordinate mapping efforts since 1953. Their efforts have had varying degrees of success and the issues were only compounded with the development of digital geospatial data.

In 1953, the U.S. Office of Management and Budget (OMB) issued Circular A-16.[85] The purpose of this document was to coordinate surveying and mapping activities within the federal agencies so these activities were carried out efficiently and without duplication. Circular A-16 was revised in 1967 in order to define the responsibilities of the Departments of the Interior, Commerce, and State in the coordination of surveying and mapping activities. In 1990, the document was again revised to include digital

[85] U.S. Office of Management and Budget. *Coordination of Geographic Information and Related Spatial Data Activities*. Circular No. A-16 Revised. (Washington, D.C. 19 August 2002.)

geospatial data activities and additional reporting requirements. The 1990 revision also established the Federal Geographic Data Committee (FGDC) as an interagency committee intended to coordinate the use of geospatial data on a national level. Circular A-16 underwent revision most recently in 2002 to reflect advancements in digital geospatial technologies and data management.

Between the 1990 and 2002 revisions of Circular A-16, the federal government took several steps to further coordinate geospatial data activities. In 1994, President Clinton issued Executive Order 12906, which directs the FGDC to "develop, in cooperation with State, local, and tribal governments, and the private sector, a coordinated National Spatial Data Infrastructure to support public and private sector applications of geospatial data."[86] Executive Order 12906 also directs the FGDC to develop standards for the National Spatial Data Infrastructure (NSDI) and to establish a National Geospatial Data Clearinghouse. Federal agencies are directed to adhere to FGDC standards, to document all spatial data holdings in accordance to FGDC standards, and to make the resultant metadata available through the Clearinghouse. Additionally, Executive Order 12906 directs the FGDC to develop a plan for the implementation and maintenance of a national digital geospatial data framework and to submit this plan to the OMB.

In 1998, the OMB issued a revision of Circular A-119 that directs agencies to "use voluntary consensus standards in lieu of government-unique standards except where

[86] U.S. President. 1994. Executive Order 12906. "Coordinating Geographic Data Acquisition and Access: The National Spatial Data Infrastructure." *Federal Register* 59, no. 71 (13 April 1994): 17672, 17671-17674.

inconsistent with law or otherwise impractical."[87] This applies specifically to the Executive Order 12906 requirement that the FGDC develop standards for the NSDI. Thus, in their process of developing geospatial data standards, the FGDC must implement existing voluntary consensus standards where appropriate rather than develop new standards.

Circular A-16 defines standards as "documented agreements containing technical specifications or other precise criteria to be used consistently as rules, guidelines, or definitions of characteristics to ensure that materials, products, processes, or services are fit for their purposes."[88] There are several different types of standards that are available for spatial data. Government-unique standards are those that the government develops for internal use. Industry standards are those developed in the private sector without the benefit of the consensus process. Voluntary consensus standards are those developed and/or accepted by voluntary consensus standards bodies.[89] These standards go through a rigorous process of development, evaluation, and acceptance by private, academic, and governmental representatives. As stated above, Circular A-119 requires that voluntary consensus standards be used if they exist. In the case of geospatial data standards, the use of voluntary consensus standards promotes interoperability. The FGDC participates with the Open GIS Consortium (OGC) and the American National Standards Institute (ANSI), the International Organization for Standardization (ISO), and the National Institute of Standards and Technology (NIST) to develop geospatial-specific voluntary consensus standards.

---

[87] U.S. Office of Management and Budget. *Federal Participation in the Use of Voluntary Consensus Standards and in Conformity Assessment Activities*. Circular No. A-119 Revised (Washington, D.C., 10 February 1998): 1.

[88] OMB, Circular A-16.

[89] OMB, Circular A-119.

Geospatial standards deal with several aspects of geospatial data such as development, maintenance, and processes. Regardless of the purpose of the standard, there are several common characteristics of geospatial standards. Maitra and Andersen describe a number of common characteristics, as follows.[90] Geospatial standards relate to geospatial data, they standardize data and data sharing, and they minimize duplication. They are future-focused in that they are intended to advance the sharing of data, linking government entities and private industry. Geospatial standards are structured because they provide minimal guidelines for development of data while enhancing the understandability and usability of geospatial data. They are technology independent because they do not limit the development of technology or vendor systems or their use. Geospatial standards are integrated with each other and with related standards. Thus, definitions and procedures do not overlap between standards. Geospatial standards are backward compatible and evolve with changes in technology. They are intended to be complete and consistent in form and format. Geospatial standards are publicly available. There is public notice of their availability; they are available electronically; and geospatial standards are not copyrighted.

The FGDC has written a Standards Reference Model that is intended as a guide for developers and users of FGDC standards.[91] The reference model identifies four main types of standards for geospatial data—data, process, organizational, and technology—and their subtypes. The FGDC is involved in the development of data and process

---

[90] Julie Binder Maitra, and Norman Andersen. *Geospatial Standards (Article 1 of 4)*. Federal Geographic Data Committee. (Available at: *http://www.fgdc.gov/publications/documents/standards/geospatial_standards_part1.html*).

[91] U.S. Federal Geographic Data Committee. *FGDC Standards Reference Model* (Washington, D.C., March 1996. Available at: *http://www.fgdc.gov/standards/refmod97.pdf*).

standards but are not addressing the development of organizational and technology standards.

The FGDC's Standard Reference Model describes several subtypes of data standards, including: classification, content, symbology, transfer, and usability standards.[92] Data classification standards provide rules for grouping data into categories. Soil and land cover classifications are examples of data classification standards. Data content standards provide definitions for sets of objects. Data symbology standards define graphic symbols and the language used to describe those symbols. Data transfer standards provide specifications for moving data between systems—independent from technology or applications. The Spatial Data Transfer Standard (SDTS) is the FGDC-endorsed data transfer standard. Data usability standards provide a structure for documenting metadata—data quality, accuracy, and contents. The FGDC Content Standard for Geospatial Metadata is the standard that federal agencies must use to document metadata.

The FGDC's Standard Reference Model describes process standards as those standards that provide descriptions of how geospatial information and technology are used to complete tasks.[93] Process standards provide a comprehensive set of procedures to guide the user through a given geospatial process. The FGDC's Standard Reference Model identifies the following types of process standards: general data transfer procedures, existing data access procedures, classification methodologies, data collection, storage procedures, presentation standards, data analyzing procedures, data integration

---

[92] Ibid, 7.
[93] Ibid, 8.

procedures, and quality control and quality assurance processes.[94] The fundamental difference between data and process standards is that process standards describe how to perform a technique and data standards describe how to apply a technique.

Standards that are more specific to resource applications may also exist. The NWCG's Information Resource Management Working Team (IRMWT) established a Geospatial Task Group in 1999. Among other things, the Geospatial Task Group is responsible for supporting interagency wildland fire management by recommending and developing strategies for managing and storing geospatial data, coordinating the development of geospatial applications, and recommending geospatial data standards.[95] The Geospatial Task Group is recommending that a standard be developed for creating fire perimeter data and that geospatial technology use for incident support be standardized.[96] So far, no geospatial data standards have been implemented by the NWCG.

After a review of the available standards from the FGDC, ANSI, ISO, and the OGC, the author has determined that there are no data standards for fire-specific geospatial data. The standards that will apply to various aspects of the rapid response geodatabase will be primarily related to the interoperability of the system.

The FGDC has written the *Geospatial Interoperability Reference Model* (GIRM) to help managers and decision makers understand interoperability and choose standards

---

[94] Ibid, 8-9.

[95] National Wildfire Coordinating Group, Information Resource Management Working Team, *Geospatial Task Group Charter* (Boise, 1999. Available at: *http://www.nwcg.goc/teams/irmwt/gtg/Charter.pdf*).

[96] NWCG IRMWT Geospatial Task Group Issues Web site: *http://www.nwcg.gov/teams/irmwt/gtg/gtg_issues.htm*

that will allow a certain activity or technology to achieve interoperability.[97] When determining which standard to use for a given situation, it is important to take several things into consideration, and GIRM identifies five criteria on which to evaluate a standard:[98]

1. How open is the standard? Is it a voluntary consensus standard?
2. What level of interoperability does the standard support? Does the standard allow geospatial systems to work together?
3. Is the standard documented clearly, accessibly, and is it consistent with other standards?
4. Has the standard been successfully implemented by others?
5. How mature is the standard? Has it been adopted by a recognized standards body?

See Appendix A for a list of currently existing geospatial data standards.

Once a standard is chosen, it is very important that it be implemented properly. Circular A-119 notes, "the use of standards, if improperly conducted can suppress free and fair competition; impede innovation and technical progress; exclude safer or less expensive products; or otherwise adversely affect trade, commerce, health or safety."[99]

Methodology

There are two aspects to the rapid response geodatabase project that are discussed in this thesis. The first is the process of designing, developing, and implementing the geodatabase itself. The second is the documentation of this process. The purpose of this

---

[97] U.S. Federal Geographic Data Committee. John D. Evans, ed. *A Geospatial Interoperability Reference Model, v1.0* (Washington, D.C, May 2003. Available at: *http://gai.fgdc.gov/girm/*).

[98] Ibid.

[99] OMB, Circular A-119.

thesis is to document the development lifecycle of the rapid response geodatabase, not perform the development. It is important to clarify that the methods described in this section are those that the author used in the documentation process. The methods that were implemented to develop the geodatabase are discussed in the next chapter.

The methodology the author followed for this thesis was to participate in, observe, and document every step that the NCLFA took to design, develop, and implement an ArcSDE geodatabase for the rapid response research projects. In so doing, first-hand knowledge of the paths taken throughout the process and the barriers encountered was obtained in order to write an accurate protocol for development. The documentation took the form of a daily journal intended to capture the thought processes and daily tasks throughout the project's lifecycle. The journal then became the primary resource for this thesis.

Several meetings were held over the course of the rapid response geodatabase project. The author attended, participated in, and documented these meetings. A formal, moderated, workshop was held on March 9-10, 2004. The workshop was hosted by the NCLFA at the USDA Forest Service Fire Sciences Laboratory in Missoula, MT and was moderated by Colin Hardy. The author participated in the workshop in the role of database designer. The workshop proceedings were recorded both manually (by a note-taker), and electronically (through the use of a SONY portable minidisk recorder). The author reviewed and took notes from the sound recordings of the meeting and from the document provided by the note-taker.

The daily journal includes all of the steps taken throughout the lifecycle of the rapid response geodatabase from the pre-design stage through the development stage and resulting in its implementation. The author also collected the design documents, educational documents, and work plans produced throughout the project lifecycle. These data were synthesized with elements of software process theory, database design theory, and spatial database design theory to produce a step-by-step protocol for the development of a geodatabase for federal fire research.

The protocol for geodatabase development is a theoretical step-by-step guide for the development process. The author analyzed the process and results of each task in the rapid response geodatabase development lifecycle and determined the effectiveness of each task and its timing. These tasks were then ordered as they should theoretically occur when following a waterfall process model. In order to identify the theoretically ideal tasks in the geodatabase design process, the author analyzed the processes inherent to software process theory, database design theory, and spatial database design theory. The two lists of tasks, the observed and the theoretical, were synthesized into one list, still following the waterfall process model. Using the experience gained by having completed the rapid response geodatabase development process, the author added, removed, and rearranged tasks. Thus, ineffective tasks were modified or eliminated and effective and important tasks were emphasized. Finally, detailed descriptions of each task were written, the result being a complete protocol for geodatabase development.

Chapter Summary

This chapter discussed several theories that could contribute to the geodatabase development lifecycle. The topics of discussion included the state of spatial database technology in federal fire management, the applicability of software process theory, the principals of database design, the theory behind geospatial database design, the specifics of ESRI's geodatabase data model, the importance of interoperability, and the state of federal geospatial standards. Elements from each of these theoretical foundations are important to the rapid response geodatabase development project and to the subsequent protocol for geodatabase development developed in this thesis. This chapter also addressed the methodologies followed by the author throughout this thesis.

CHAPTER 3

THE LIFECYCLE

In theory, the lifecycle for the rapid response geodatabase would have conformed to a strict software process following the waterfall model for design and would have looked somewhat like Figure 4.

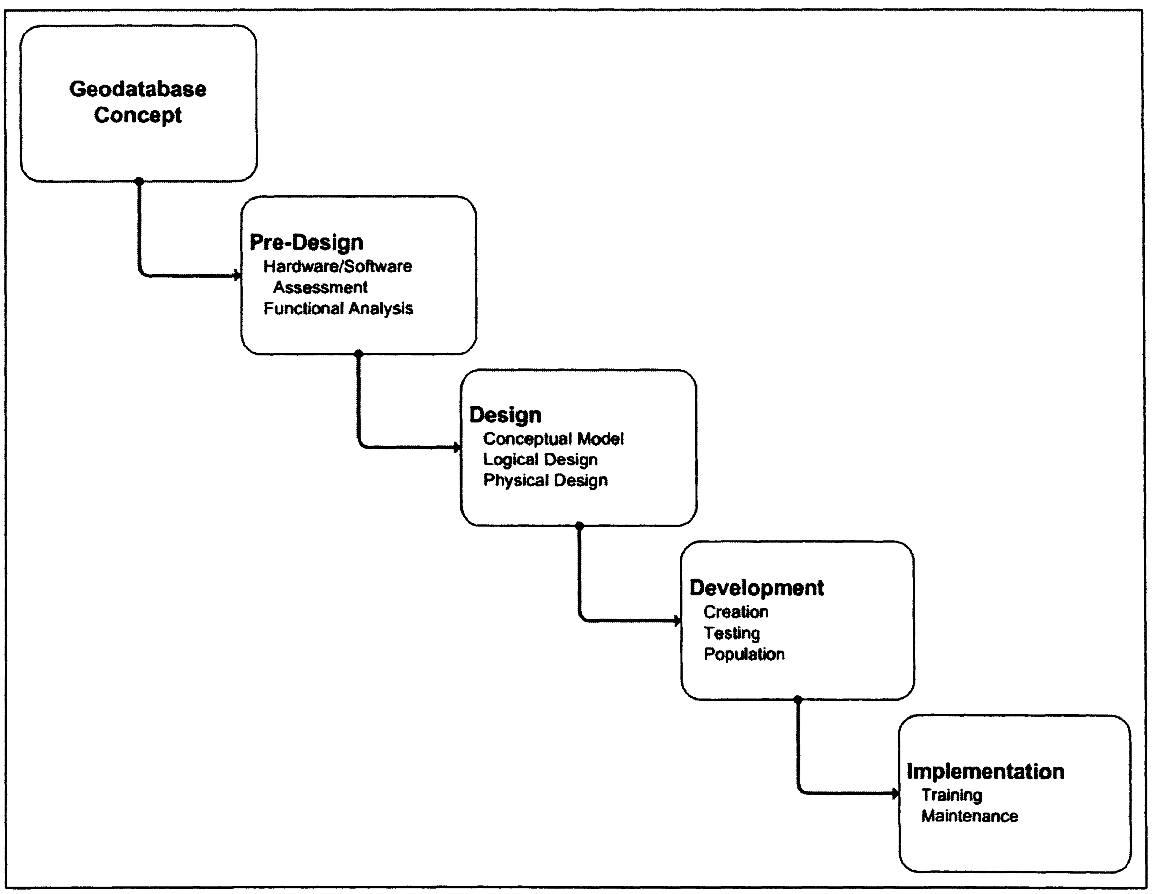

Figure 4—Waterfall model representing the rapid response geodatabase development process (source—author).

However, reality rarely conforms to theory. The lifecycle captured here has followed the theoretical process only very loosely. The goal of this chapter is to portray the development lifecycle by presenting the actual processes followed by the geodatabase developers. The process begins with the conceptualization of the project, and continues through the hardware and software assessment, the pre-design, design, development, and implementation stages, and ends with the process assessment.

Conceptualization

The JFSP board of governors initially identified the need for a rapid response geodatabase. The JFSP did not use the term "geodatabase" or conceptualize the project as such, but rather, they realized the utility of a centralized database through which several researchers could share their data. Subsequently, they instructed Hardy and Riggan to "investigate" a common database for several rapid response research projects. The use of the word "investigate" rather than one such as "create" or "develop" indicates that, though the JFSP recognized the applicability of a database, they did not have a clear idea of what it would consist of.

There are four main alternatives for the architecture of a common database. The first alternative is to create a custom-built database management system (DBMS) resulting in a unique relational database architecture that would require custom interfaces and applications to allow users to retrieve the stored data. The second alternative is to use a proprietary DBMS such as Oracle, SQL Server, etc., to store the data. The use of a proprietary system may or may not require custom interfaces and applications to allow access to the data, depending on the DBMS used and the requirements for data access.

Because it is necessary to extend the functionality and query language of a DBMS to be able to handle spatial data, the use of a stand-alone DBMS, custom or proprietary, would only allow researchers to store and retrieve non-spatial data. This type of system would not support the storing, viewing, or editing of spatial data. The third alternative is to create a custom geographic information system (GIS) that would provide the functionality to store and query spatial and non-spatial data within a database (custom or proprietary). The GIS would allow single users to view and edit spatial data stored in the database. In this alternative, it would be necessary to distribute the custom GIS and database to each researcher individually. The fourth alternative is more complex.

If the intent of the common database is to act as a repository of the original data from which the researchers can retrieve a copy of the data for use in their own file storage system, any of the first three alternatives will work. However, the technology exists that allows a spatial data set to be stored in one location and accessed by multiple users simultaneously. The data need not by copied by every researcher into their own file management system in order to view, edit, and analyze it. Therefore, the fourth alternative for database architecture is a solution based on ESRI software and a proprietary DBMS that can be designed to allow multi-user viewing and editing. The spatial and non-spatial data are organized into a relational database model called a "geodatabase" and loaded into the DBMS using an intermediary application called ArcSDE. ArcSDE is a software platform that essentially runs on top of the DBMS and manages connections between the geodatabase and multiple ESRI applications that access the data stored within the geodatabase. Users can access the data using ArcGIS desktop GIS applications (for example, ArcCatalog and ArcMap) and the Web-based

ArcIMS. Data may also be exported from the geodatabase in a variety of interoperable data formats, which would allow data to be used in non-ESRI applications.

The NCLFA decided to pursue the fourth alternative to design and develop a common database for the rapid response researchers. There are several reasons for this decision. A stand-alone DBMS would not satisfy all requirements of the data due to the spatial element to the rapid response data. Also, the ArcSDE geodatabase model provides the highest level of interoperability. It allows for data access through multiple ESRI as well as non-ESRI applications. The use of ArcSDE has the potential to allow multiple users to view, edit, and analyze the data stored in a single, centralized location. The decision to use the ESRI software and geodatabase data model was also based on the assessment that ESRI software is a common factor between all of the rapid response researchers. ESRI GIS software is one of the most widely used GIS packages in the private sector and the USDA Forest Service maintains a contract with ESRI and provides ESRI desktop GIS applications to all personnel.

Hardware and Software Assessment

The common database for the rapid response researchers will be built using ESRI software and the geodatabase data model. The geodatabase will be created as a multi-user system utilizing ESRI's ArcSDE technology. Thus, the geodatabase will not be localized to the researchers' personal computers; it will be located on a server that will be hosted by the NCLFA. The NCLFA owns a licensed copy of ArcSDE and the required hardware. Also, ArcSDE is a complicated application that requires some experience and skill to administer, as does the underlying DBMS. At the time of this project, the

NCLFA did not employ a database or ArcSDE administrator. However, the NCLFA had access to the resources necessary to effectively host the ArcSDE geodatabase.

The software assessment continued with the decision to use SQL Server as the DBMS in which the geodatabase would ultimately be stored. Again, this decision was made on the premise that the NCLFA owned a licensed copy of SQL Server. Additional software applications that were deemed necessary for the geodatabase development process included: Microsoft Office, Microsoft Visio, and ESRI's ArcGIS (with an ArcInfo-level license).

Pre-Design

As in many large projects, the course of the project lifecycle was affected by politics. In this particular situation, the problem began with the fact that the decision to investigate a common database came from the top down, rather than from the bottom up. The JFSP acted as the "management" and requested this addition to the list of rapid response research projects. Then a relatively unknown third party, the NCLFA, was brought in as the "service provider" to create and implement an ArcSDE geodatabase. At this point, the researchers had no real plans to use a common database to complete their research or share data, let alone a geodatabase. Many were left asking, "What is a geodatabase?" Or "Why can't we use this other system that we are developing over here?" The result of these underlying politics was that the users of the proposed system were neither dependent on, nor responsible for, the success of its development.

The first step taken to mitigate these politics was to create a Web site that the project participants could use to keep informed of the project's progress.[100] The NCLFA also drafted and distributed a white paper describing the center's intent to develop a geodatabase for the rapid response research.[101] The paper included an overview of the geodatabase model and several Internet links to ESRI Web pages giving more detailed information about this model and the ArcSDE platform. At this time, the NCLFA requested that the researchers submit comprehensive data dictionaries listing all data fields and spatial objects that would be included in the geodatabase. The request for data dictionaries was made in September of 2003 with a deadline for submission of November 25, 2003.

Through the course of several meetings between NCLFA staff and Colin Hardy, the key issues of the geodatabase design process were identified. The first issue was the question of who was going to design the geodatabase and how. The NCLFA had been charged with developing the geodatabase in cooperation with the rapid response researchers. The initial intent was to designate a database design team that would work together to design the geodatabase. The team members would be expected to understand the geodatabase model and be able to help educate the rest of their respective research teams. The team members would be expected to understand their data and be able to identify how they would be using other researchers' data. The team included staff from the NCLFA (including the author), selected members of each contributing rapid response

---

[100] *http://firecenter.forestry.umt.edu/rapid_response/*

[101] Don Helmbrecht, "Demonstration and integration of systems for fire remote sensing, ground-based fire measurement, and fire modeling project: Database development," (Missoula: National Center for Landscape Fire Analysis, 2003).

research teams, the designer of FIREMON, and a consultant from ESRI.[102] The plan was to have ESRI facilitate a database design workshop at which the database schema would be designed and the geodatabase created. The NCLFA would then populate the geodatabase and address the functionality of the system. Things did not necessarily go as planned.

Though the need for a data committee was recognized, cooperation was not guaranteed. The prerequisite for holding the database design workshop was that data dictionaries from all contributing research projects be submitted to the NCLFA and combined into a master data list. The initial deadline for the principal investigators to submit their data dictionaries was November 25, 2003. On that date, only two of the six expected dictionaries had been received. By December 15, 2003 only four of the six had been received. The final two were not received until March 9, 2004 when the data committee workshop was finally held.

Despite the lack of participation by the researchers, the design task was begun with the information that was available. The first task was to complete a functional analysis of the geodatabase. This process started with a series of questions: What are the data? How will they be used? What is spatial about the data? What are the relationships within the data and between the projects? How are the data entered into the system? How will the users access the data? What are the derived products? Will those products be stored within the geodatabase? What is the timeline for the geodatabase development process?

---

[102] FIREMON is a protocol for fire effects monitoring and inventory. It provides a set of standards for data collection and a database and set of analysis tools for monitoring the effects of wildland fire. Information can be found on the Web at: *http://fire.org/firemon/default.htm*.

In the process of the functional analysis, the NCLFA was unable to satisfactorily answer the question of how the geodatabase would be used. It was known that not all of the researchers used ESRI software to process and analyze their data. However, neither the analysis tasks nor the applications used to perform analysis were known. The NCLFA determined that it was important to be able to demonstrate common spatial analysis tasks with data stored within the geodatabase. The NCLFA also determined that because the researchers would likely be using non-ESRI applications in conjunction with data stored within the geodatabase, the interoperability of the system would be important. In order to facilitate the interoperability of the system, a custom application needed to be developed to allow the researchers to export the data in non-ESRI data formats.

At this time, a plan of work was developed for the rapid response geodatabase project.[103] The plan had three goal areas. The objectives for each goal area were detailed including the tasks, methods, and outcomes for each objective. The objectives for **goal area one** included the completion of the functional analysis, the geodatabase development, and the design of the data-sharing framework. **Goal area two** included the demonstration of the ability to interpolate a raster surface by using point data from the geodatabase and if necessary, build a custom tool to provide the interpolation functionality. **Goal area three** was intended to demonstrate the interoperability of the geodatabase by integrating FARSITE modeling functionality through the use of a custom tool. The work plan included a data committee workshop during goal area one for the

[103] Don Helmbrecht and Lee Macholz. "Plan of Work for the Rapid Response Project Within the National Center for Landscape Fire Analysis." (Missoula: National Center for Landscape Fire Analysis, 2003).

purpose of developing the database schema after the functional analysis had been completed. The timeline for the geodatabase is shown in Table 1 below.[104]

Table 1—Timeline for the rapid response geodatabase as given in the NCLFA work plan.

	2003		2004				
	Nov.	Dec.	Jan.	Feb.	Mar.	Apr.	May
Goal Area 1:							
Functional Analysis	X						
Develop Geodatabase		X	X				
Dissemination Framework			X	X	X		
Goal Area 2:							
Test Interpolation Functionality	X						
Build Interpolation Functionality		X	X	X			
Goal Area 3:							
FARSITE Integration					X	X	X

Continuing the functional analysis, Don Helmbrecht and the author worked with Colin Hardy to further answer the question of how the data would be used. The three data dictionaries that had been received to date were compiled into an outward-looking matrix. The intent was to identify the relationships between the data elements. Thus, by determining the derived products, the question of how the system would be used could be answered. Unfortunately, this small group of people could not answer these questions for all of the research projects. The data matrices were only useful for depicting the non-spatial relationships between data levels within one research project.

Questions concerning what is the data and how will it be used persisted. The author began using the data dictionaries that had been received to start creating the geodatabase schema. Several things became obvious: the schema could not be completed without all of the data dictionaries; more direct user input was necessary to

[104] Ibid.

determine the spatial elements and relationships within the data; and table structure within the geodatabase would differ depending on the requirements of the users. It was evident not only that a data committee workshop needed to be held, but also the goals of that workshop were different than initially planned.

The goals for the data committee workshop became four-fold. The first goal was to review the master data dictionary in order to assure accuracy and completeness. The second goal was to define the relationships among individual data elements and those existing between projects. The third goal was to complete the functional analysis of the database. And the fourth goal was to finalize the project timeline.

To prepare for the data committee workshop, a variety of documents were created. The first document was a thorough explanation of why the NCLFA chose to use the geodatabase data model, what a geodatabase was, and the role of ArcSDE. This document also requested help from the researchers in the form of responses to an accompanying survey, submission of their data dictionaries and data, and attendance at the data committee workshop.

The second document created in preparation for the data committee workshop was a survey (Appendix A). The intent of the survey was to prepare both the researchers and the NCLFA for the functionality analysis to be completed at the workshop. The survey's intent was to prepare the researchers to think about the questions that might be asked of them at the data committee workshop. It served to prepare the NCLFA by further clarifying the researchers' expectations and requirements for the geodatabase. The survey addressed the following issues: the researchers' expectations of the geodatabase system; the researchers' functionality requirements; how each researcher intended on using other researchers' data; what derived products would the researchers be creating; and what software packages would be used for analysis.

Nine surveys were sent to the primary investigators and selected members of their research teams. Six were returned to the NCLFA. The results of the survey were very thought-provoking. While several of the researchers' expectations matched the purpose of the geodatabase, many of the expectations for the rapid response geodatabase were unrealistic. Where the intent of the geodatabase was to provide a common data repository for the rapid response field data that would be accessible through a GIS interface, the expectations included the ability to view historical research themes and a Web-based service for software upgrades. The responses regarding functionality requirements were also varied. It seemed that unrealistic expectations led the researchers to desire functionality above and beyond what the NCLFA could provide with a demonstration project. However, the responses can be summarized by a common theme: the need to query, view, and retrieve data. The fact that there were unrealistic expectations for the project meant that the educational efforts made by the NCLFA had not reached all of the researchers. This would need to be addressed at the data committee meeting.

The focus of the data committee meeting had shifted from the original intent of creating the geodatabase schema to the goal of validating an existing geodatabase schema and completing the functional analysis. Therefore, the creation of the geodatabase schema was continued using the data dictionaries that had been submitted to date. The schema was arranged into two UML diagrams—an inheritance diagram and a relationship diagram. On the inheritance diagram, objects were organized by type.

Objects that contained only tabular data were grouped by project and listed as tables. Objects that contained spatial information as well as tabular attribute data were listed as feature classes. On the relationship diagram, table and feature class objects were grouped by project. Within each project, the relationships between objects and the cardinality of those relationships were shown. The table structure was created such that attribute information was divided into appropriate tables and related back to the feature classes. In the initial conceptual schema, all relationships had a cardinality of either one-to-one or one-to-many.

In addition to the conceptual schema, the author prepared a comprehensive data dictionary listing all of the fields that had been submitted. The master data dictionary reflected the table structure in the conceptual schema. Included in the data dictionary were the original field name, the field name used in the geodatabase schema, the data type, and comments.

The data committee workshop was scheduled for March 9th and 10th, 2004 at the Missoula Fire Sciences Laboratory. Colin Hardy was asked to moderate the workshop. Fifteen individuals were invited to attend. Those invited included the principal investigators from each of the contributing rapid response projects and one or two members of their research teams. Also invited were several individuals not associated with a rapid response project but who could provide additional insight and technical information during the workshop. The NCLFA provided a note-taker and a digital recording device to capture the proceedings of the workshop. Eighteen individuals attended the data committee meeting (Table 2) representing all but one of the contributing rapid response research projects.

Table 2—Data committee meeting attendees.

Name	Organization	Project
Lloyd Queen	Director, NCLFA	Geodatabase Development
Don Helmbrecht	NCLFA	Geodatabase Development
Lee Macholz	NCLFA	Geodatabase Development
Colin Hardy	RMRS Missoula Fire Lab	Fuel Loading and Ground Thermal Infrared*
Sharon Hood	RMRS Missoula Fire Lab	Fuel Loading Characterization
Helen Smith	RMRS Missoula Fire Lab	Fuel Loading Characterization
Bryce Nordgren	RMRS Missoula Fire Lab	Ground Thermal Infrared
Bret Butler	RMRS Missoula Fire Lab	Fire Behavior Package*
Kyle Shannon	RMRS Missoula Fire Lab	Fire Behavior Package
Dan Jimenez	RMRS Missoula Fire Lab	Fire Behavior Package
Phil Riggan	RMRS Riverside Fire Lab	Airborne Thermal Infrared*
Penny Morgan	University of Idaho	Burn Severity*
Carter Stone	University of Idaho	Burn Severity
Andrew Hudak	Moscow RMRS	Burn Severity
John Caratti	SEM	Observer†
Brig Bowles	ESRI	Observer†
Mike Sweet	University of Montana	Observer†
Patricia Williams	University of Montana	Note-taker†

*Principal Investigator
†Not associated with a rapid response project

The data committee members worked as teams representing each rapid response project to complete the tasks they were given. Team sizes ranged from one to three persons. Staff from the NCLFA and a consultant from ESRI circulated the room answering questions and giving assistance as needed. The first task of the data committee workshop was to review the data dictionary. The teams were instructed to verify that every data field they would be providing to the geodatabase was present in the data dictionary. They were also instructed to modify the field names as necessary to accurately reflect the contents of the data field. Finally, the teams were asked to identify the spatial object that each data element was related to. Each research team received a UML diagram of the geodatabase conceptual schema (both the inheritance and relationship diagrams) on which they were asked to mark their changes and additions. Each team was then asked to stand in front of the group and explain their rapid response project, including data collection methodologies and the data elements to be contributed to the geodatabase.

During the first task, teams were instructed to work only with the data fields that they would be contributing to the geodatabase. When each team stood and explained their project and their data, they were not only answering the NCLFA's question of what is the data, but many of their fellow researchers were hearing about the data for the first time. Even though they had been instructed by the JFSP to exploit the linkages between projects, and they had made initial plans to do so, none of the researchers had actually contacted one another. This exercise helped overcome the underlying political barrier and opened new lines of communication between the researchers.

The second task of the data committee workshop was to define the relationships present between data elements. The teams were asked to highlight all of the data fields they expected to use in their analysis, both their own and other researchers'. Again the teams were asked to stand and explain how they would use each other's data in their projects. This discussion started slowly, but as the research teams started talking, they got more and more animated. It became obvious that through different combinations of different researchers' data, new questions could be posed, and answers could be reached that were previously thought to be unattainable. The attitude toward the rapid response geodatabase project suddenly changed and the researchers became very interested in the idea of a shared spatial database.

The third task of the data committee workshop was to complete the functional analysis of the geodatabase. In order to do this, the data committee members were asked to consider the following questions: Do you need to edit your original data? How will you use the data in the geodatabase? What products are you going to derive from the geodatabase? Will those derived products need to be stored in the geodatabase? The teams were then asked to draw a flow chart showing their research process starting with the primitive data and including all anticipated derived products. Team members were asked to specifically address what programs would be used to perform data analysis, whether or not each derived product would be posted back to the geodatabase, and what file formats those products would be in. The teams were then asked to present their flow charts to the group.

The third task helped the NCLFA answer the question of how the researchers would use the geodatabase. There seemed to be two primary themes to the data analysis being done. First, tabular data were being processed in spreadsheets and statistical software. Second, raster data were being processed in a variety of environments from custom applications to mainstream off-the-shelf spatial applications such as ERDAS, ENVI, IDL, and ArcGIS.

ESRI's ArcGIS, specifically ArcCatalog, contains the functionality to transform data from a geodatabase to any of the data formats required by the non-ESRI applications being used by the researchers. Thus, custom programming for the import and export of data to and from the geodatabase would not have to be written. Federal agencies however, are required to implement geospatial standards as identified by the FGDC. The SDTS has been identified as the desired standard for the transfer of spatial data.[105] The purpose of the SDTS is to allow digital spatial data to be transferred between different spatial applications. The SDTS is implemented through the use of profiles.

---

[105] Information regarding the SDTS can be found at the following Web site: *http://mcmcweb.er.usgs.gov/sdts/*.

Unfortunately, the requirements for SDTS profiles are very complicated and are not supported by ESRI applications at this time. The NCLFA determined that the creation of a custom import/export tool for SDTS profiles would not be pursued during the rapid response geodatabase demonstration project.

During the functional analysis discussion, the researchers also addressed data input. For the purpose of the demonstration project, the researchers would submit their data to the author at the NCLFA who would then populate the geodatabase. However, if the rapid response geodatabase were to be implemented beyond the demonstration project, the NCLFA would not be responsible for administering the system. The researchers illuminated the fact that if the entry of primitive field data into the system was not easy to do, the geodatabase would not be used. If the geodatabase project were to be implemented beyond the demonstration phase, custom data entry screens would need to be created for each of the contributing research teams.

The next functionality requirement that the researchers identified was the ability to do calculations on the data within the geodatabase. Again, this functionality is already provided within ArcGIS, specifically ArcMap.

When the NCLFA posed the question, what type of spatial analyses will be done within the geodatabase, the answers were fairly simple. For the most part, the researchers would be performing geostatistics and overlay analyses. These requirements helped dictate the table structure within the geodatabase during the design phase.

The data committee workshop concluded with the researchers committing to submit all of their data collected at the site of the Cooney Ridge Fire to the NCLFA. The workshop was successful in meeting three of the four main goals that had been established:

1. The data dictionary was corrected;
2. The relationships within the data and between the projects were apparent;
3. The functionality analysis was complete.

The real success of the workshop was that new lines of communication had been established between the research teams and the NCLFA. The underlying political barrier had been overcome and the researchers were finally willing participants in the rapid response geodatabase demonstration project.

Immediately following the data committee workshop, Lloyd Queen, Don Helmbrecht, and the author, all members of NCLFA, met with Colin Hardy, Brig Bowles of ESRI, and Mike Sweet of the University of Montana to debrief the proceedings of the workshop. The goals of this meeting were to discuss the results of the functionality analysis, determine the next steps for the geodatabase project, establish the timeline for the completion of the geodatabase, and determine the deliverables for the project. This meeting was important because it allowed time for the ERSI consultant to give the NCLFA some detailed feedback on the development requirements of the geodatabase.

It was determined that the functionality requirements of the users could be fully satisfied by the existing functionality within ESRI's ArcGIS software and that no custom applications needed to be created for the demonstration project. This determination was significant because it is different than what the NCLFA had previously determined would be required by the project. The significance is that the initial determination was based upon functionality that the NCLFA predicted the users would require, not upon direct input from all of the users. The lesson to be learned here is that the design team cannot attempt to determine the user requirements without consulting the users.

Design

Once the users' functionality requirements had been determined and all of the data dictionaries had been received at the data committee workshop, the author could continue the process of designing the geodatabase. In theory, this would be the beginning of the design process; in reality the initial conceptual schema was created to assist in the functional analysis. During the course of the data committee workshop, the researchers had used copies of the conceptual schema to make changes and additions to the geodatabase design. The first task of the design phase was to incorporate all of these changes into the conceptual schema.

A better understanding of the data fields and their relationships lead to a re-organization of the table structure and relationships. All tables participating in one-to-one relationships were normalized to eliminate unnecessary tables. At this point, all relationships within the geodatabase were believed to have one-to-many cardinality.

The data committee workshop also allowed the geographic representation of each object to be identified. This answered the question of what is spatial about the data set, justifying the use of a geodatabase rather than a non-spatial relational DBMS. The author went through the schema and identified the geographic representation for each object within the geodatabase. The geographic representation categories were: none (tabular data), point, line, polygon, raster image, or TIN. The researchers had collected latitude and longitude coordinate locations representing their instrument locations and plot centers; thus, all of the feature classes within the geodatabase contained point features. Two projects had collected thermal infrared (ground and air) imagery that would be stored in the form of raster imagery within the ArcSDE geodatabase.

Naming conventions were then developed to standardize the terminology used within the geodatabase. The naming convention for the geodatabase was established in accordance with the naming practices of the researchers. It was important to be able to distinguish which project each object belonged to, so a three- or four-letter prefix was used as an identifier. Abbreviations commonly used by the researchers were used in object and field names. Abbreviations were important because ArcSDE requires that feature class and table names be 25 characters or less. This 25-character limit includes both the object name and the name of the user that owns the table in SDE.[106] There are also approximately 500 reserved words and keywords that cannot be used as field names within SQL Server.[107] Moreover, special characters were avoided in the rapid response geodatabase naming convention; only the underscore (_) was used. Finally, numbers were not used at the beginning of field names, though they were used in the middle or at the end of some fields.

The next task in the geodatabase design process was to determine the data type for every field within the geodatabase. Data types define the way each data field is stored in memory. Fields containing text, a combination of text and numbers, or numbers that will not be used to perform calculations are stored as a "string" data type. Fields containing numbers are stored as "integer" or "double" data types depending on whether or not they

---

[106] The user that creates any given feature class or table in an SDE geodatabase is designated as the "owner" of that object and their username is appended to the beginning of the object's name. Environmental Systems Research Institute, Inc Web site: *http://support.esri.com/knowledgeBase/documentation/FAQs/sde_/WebHelp/faq.htm.*

[107] *http://www.bairdgroup.com/reservedwords.cfm.*

contain decimals. String data types require a value for length, which determines the maximum number of characters that can be entered into that field. Double data types require values for both precision and scale, which determine the total number of numerical characters that can be entered into that field and the number of decimal places it contains, respectively. Integer data types require only a value for precision. When using Visio in conjunction with CASE Tools to create a geodatabase, the data types must be set in the UML diagram in Visio, and the length, precision, and scale values may be set either in the UML diagram or in the Schema Wizard in ArcCatalog.

The NCLFA had requested that the researchers submit their rapid response data for the Cooney Ridge fire before the data committee workshop was held. That request was again made at the data committee workshop. Within the month following the workshop, five of the six rapid response projects involved with the geodatabase had submitted their data. For the most part, the data was contained within Microsoft Excel worksheets or comma-delimited text files. The default format for data cells within MS Excel is "general" and, as such, numeric data types are not embedded within the worksheet unless they are specified. Likewise, text files inherently contain only string data. None of the data submitted in these types of files had assigned data types. One project submitted its data in the form of a Microsoft Access database within which string and numeric data types had been distinguished, but neither the precision nor the scale were set within numeric fields. Thus, the data types, field lengths, precision or scale could not be determined without help from the researchers.

Since the data were being used for statistical analyses, it was very important that numeric fields retain the appropriate precision and scale. The data dictionary was updated and revised, adding blank fields for data type, precision/length, scale, description, and units (for metadata). The revised data dictionary was sent to the researchers with a request that they fill in the blanks and return the completed dictionary. Upon the return of the completed data dictionaries, the author again updated the geodatabase schema, setting all of the necessary values within the UML diagram of the schema.

Next, the issue of value domains within the geodatabase was addressed. Value domains are sets of valid attributes that are associated with a data field.[108] Domains are used to limit the contents of a data field, thereby enforcing consistency in repeated values and reducing input error. When a coded value domain is assigned to a field, only values that exist within that domain can be entered into the associated field. When a range domain is assigned to a field, only values that fall within that specified range can be entered into the associated field. Value domains are assigned to a data field by setting the data type of the data field to be the corresponding value domain.

Several coded value domains were created for the rapid response geodatabase, primarily to support the FIREMON protocol for data collection implemented by Hardy's research team.[109] These domains were copied from the FIREMON database in which the Fuel Loading data had been submitted to the NCLFA. During the data committee workshop, the researchers indicated that a coded value domain for vegetation species would be useful within the geodatabase. The list of species used within the FIREMON database was obtained from John Caratti. This list turned out to be in tabular format, containing 11 columns of attributes and 82,120 rows of values. The geodatabase does not

---

[108] Zeiler, 78.

[109] The FIREMON data collection protocol and database can be downloaded from the following Web site: *http://fire.org/firemon/default.htm.*

support the use of a free-form table as the source of values for a coded value domain. The values must be entered into the proprietary domain structure in ArcCatalog. It was not reasonable for the NCLFA to manually enter the code and species name for 82,120 species unless this task could be automated. It was determined that the task could be automated programmatically through ArcObjects. However, the data that would populate the geodatabase for the purpose of the demonstration project only contained references to five species classes. Thus it was decided that a coded valued domain would be created to contain those species referenced within the data for Cooney Ridge and an automated solution would be investigated at a later date.

The last task for the design stage was to determine the metadata that would accompany the geodatabase. Metadata in data dictionaries formed the foundation on which the geodatabase design was built. The maintenance of this metadata will also serve as a guide for the system's users. ESRI's ArcCatalog will capture metadata pertaining to spatial properties of the feature classes within the geodatabase as the geodatabase is populated. However, some of these properties—specifically the spatial reference and projection—were documented during the design stage. This information was used to determine the spatial reference and projection that would be used within the geodatabase. Each research team collected spatial data primarily through the use of GPS units. All research teams collected these points using the WGS84 datum. However, some researchers transformed the coordinates into the NAD83 datum and then projected the features into a local projection (UTM, Zone 11). Due to the potential for growth within the geodatabase as new study sites are added, the decision was made to store the spatial data as unprojected geographic coordinates. The geodetic datums would be standardized to NAD83, which is recommended datum for federal geospatial data.

It was determined that metadata documenting the content of the geodatabase should comply with the FGDC's Metadata Content Standard. Thus, the data for specified FGDC metadata fields were requested from the researchers at the same time they were requested to supply the data types and field lengths in the revised data dictionary. The following metadata fields were requested: contact information, field descriptions, units, and data collection methods. Not all metadata information was resubmitted. It was determined that the author would populate the metadata fields to the extent possible during the implementation of the geodatabase. The NCLFA would subsequently train the researchers how to enter additional metadata through the use of the FGDC metadata style sheet available within ArcCatalog.

Development

The conceptual schema was completed in early April 2004. The schema contained a complete representation of the geodatabase including all feature classes, tables, relationships, and value domains. All feature classes had been assigned their corresponding geographical representation (point, line, or polygon), and all data fields had been assigned data types, field lengths, and values for precision and scale. The schema did not contain placeholders for raster data sets; raster data would be added after the ArcSDE environment was established. The schema was converted from a UML diagram in Visio to an XML document using ESRI's CASE Tools within Visio. A personal geodatabase was created within ArcCatalog and the schema wizard was used to create the structure of the geodatabase from the XML document.

Personal geodatabases were used as the testing environment during the development phase. This was recommended by ESRI because all of the functionality of an enterprise geodatabase required to test the geodatabase design is present within a personal geodatabase, and it is a more forgiving environment to work in when making changes.

Populating the initial trial geodatabase was a time-consuming process. The tabular data were compiled into an Access database (not a geodatabase) before populating the geodatabase. This was done for several reasons. First, because more than one trial geodatabase would likely be created in addition to the final geodatabase, having the data in one location and in an easily loadable format was important. Second, to minimize error, it is beneficial to ensure that data types are embedded within the source data file. Third, it is helpful to organize the data under headings that will match the destination field names. The Access database was then used to load data to the existing tables within the geodatabase.

One of the first things discovered was related to the schema wizard. Even though the values for field length, precision, and scale were set in the UML diagram, they were not retained through the transformation into the actual geodatabase. This loss of field precision was noted, but the issue was set aside to return to later.

The next problem encountered involved the loading of the feature classes into the geodatabase. Spatial data were submitted in two formats, coverages and coordinates within text files. The coverages were converted into shapefiles while text files were formatted, displayed as event themes, and converted into shapefiles.[110] These shapefiles were then loaded into pre-defined, empty feature classes within the geodatabase using the simple data loader within ArcCatalog. When the new feature classes were subsequently displayed with the original shapefiles, the points were not coincident. The author went through an extensive process to examine this problem by testing 47 separate scenarios of loading data into the geodatabase. Brig Bowles of ESRI was contacted for assistance with this problem. It was determined that the source of the problem was determined to be that the geodatabase does not automatically maintain the same level of precision in the spatial coordinates as the source files. It was recommended that the spatial reference properties be accessed within the schema wizard and the precision should be set at 100,000, but the spatial reference itself should not be set within the schema wizard. Instead, the spatial reference should be imported from the source shapefiles after the empty feature classes are created within the geodatabase and before they are populated with data. Finally, the simple data loader should be used to import the data into the feature class from a shapefile. This solution appeared to correct the problem.

During the testing phase, it was recognized that even though the structure of the relationships within the geodatabase supported the initial data, several of the relationships were flawed. Relationships for two of the projects (Burn Severity and Fuel Loading) were keyed on PlotID fields with a cardinality of one-to-many. However, the one-to-many cardinality did not support all of the original data. Thus, due to the realities of the data, several many-to-many relationships emerged within the geodatabase structure. The decision was made not to normalize these tables because splitting the data into several

---

[110] The geodatabase will accept spatial data from both coverages and event themes. In this instance the data were converted to shapefiles because coverages cannot be edited within ArcGIS 8.3.

tables would not comply with the user requirements for data processing outside the geodatabase.

The conceptual schema was modified within Visio to reflect the change in relationships. Many-to-many relationships in the geodatabase require an intermediary table. This intermediary table is used to store the key fields from each table in order to allow the database to match up the corresponding fields. The intermediary table was specified in the conceptual schema by assigning the table a stereotype of relationship class.

Meanwhile, staff at the NCLFA were working on installing and configuring the SQL Server database and ArcSDE. Two scenarios for the configuration of ArcSDE user accounts and permissions were explored; a centralized system and a distributed system. In a centralized system, a database administrator maintains the ArcSDE geodatabase. The administrator is the owner of the data and controls the users' permissions to the data. In a distributed system, the ArcSDE geodatabase is maintained by the users and there may or may not be a database administrator. The users own their own data within the geodatabase and they control other users' permissions to the data they own.

For the purposes of the rapid response geodatabase demonstration project, the geodatabase was set up as a centralized system. The NCLFA would act as the database and ArcSDE administrator. An administrative account would be created through which the geodatabase structure would be created and all of the data would be loaded. User accounts would be created for each of the research projects through which the teams would access the geodatabase. The user accounts would be given only *CreateTable* permissions within ArcSDE. This would limit the users' ability to change the structure of the geodatabase itself, allowing them only to append data to existing feature classes and tables. All of the users would receive read privileges to all of the feature classes and tables within the geodatabase. Each research team would then receive update and delete permissions for the data objects containing only their own project data.

Because the ArcSDE geodatabase data model is so new to the researchers, the decision was made that it would be best for the NCLFA to retain control over the geodatabase structure while educating the researchers in its use. If the rapid response geodatabase were to be implemented beyond the purposes of the demonstration project, the NCLFA would have to decide if it would be willing to fill the role of the ArcSDE geodatabase administrator. Whether or not the NCLFA accepts that role in the future, the users' permissions will have to be extended to allow them to add new feature classes, tables, and raster data to the geodatabase and control the privileges to those objects.

Implementation

The implementation stage of the rapid response geodatabase development lifecycle involved the creation of the final ArcSDE environment and the SQL Server database, followed by the creation, population, and testing of the geodatabase structure. The SQL Server database was created during the development stage; and this same database was used for the implementation. The geodatabase structure and user accounts used to test the ArcSDE and SQL Server environments during the development stage were deleted from the server. The final administration and research team user accounts were then created in SQL Server and their permissions set in ArcSDE. The final account structure and permissions matched the structure outlined during the development stage.

The structure of the geodatabase itself was created next. A database connection was established from the author's desktop computer to the ArcSDE server through ArcCatalog. The first task in creating the geodatabase structure was to create the value domains so that they would be available as their dependent fields were created. During the testing phase, it was discovered that the settings for field precision and scale were not retained through the use of the Schema Wizard or the *Import* command. In order to set values for field precision and scale, the table and feature class structures were created manually. All of the tables were created in ArcCatalog using the *New Table* dialog. The *New Table* dialog allowed fields to be imported from existing tables in the trial personal geodatabase and allowed the field properties to be altered. However, there were still problems with setting the field properties of data type, precision, and scale. Where the data type was set to long integer and the precision was less than or equal to 10, the software automatically changed the field precision to 10 upon final creation of the table. If the precision was more than 10, the precision value was retained, but the data type was automatically changed to double. Where the data type was set to double and the precision value was less than 10, the software automatically changed the data type to float, but retained the proper precision and scale values. These automatic changes were not encountered when testing within a personal geodatabase. It was necessary to address the fact that the field precisions and scales were not set as required. In order to enforce the required field precision, value domains were created within the geodatabase. For example, fields that contained percentages were assigned a range domain of 0 to 100. It was not possible to enforce specific field scales in this manner, but the required field precisions were maintained.

After the tables were created, the feature classes were created using the *New Feature Class* dialog. This dialog was used in the same manner as the *New Table* dialog with the exception of the *SHAPE* field. The geometry type property for the *SHAPE* field was set, the default grid value was accepted, and the spatial reference was not defined. After a feature class was created, its properties were accessed and the spatial reference was imported from the original shapefile containing the corresponding data. This ensured that the spatial extents and precision of the feature class matched those of the shapefile.

Finally, the relationship classes were created through the *New Relationship Class* dialog. The relationship classes were structured as they were defined within the conceptual schema.

Once the skeletal structure of the geodatabase was fully created, the tables and feature classes were loaded using the simple data loader. During the development stage, the tabular data was organized into an Access database where data types were assigned to each field. The simple data loader recognizes where field names and data types match and allows assignment of source fields to destination fields where data types are compatible. Unfortunately, those destination fields whose data types had been automatically changed to float were not considered compatible with source fields whose data types were defined as double. In order to be able to load these fields and retain the decimal values, the data types of the source fields had to be redefined as string fields within the Access database. Data integrity was ultimately maintained through this process and the population of the geodatabase tables was successful. The feature classes were not subject to this problem as they were loaded from the original shapefiles rather than from the Access database.

The feature classes were loaded into the ArcSDE geodatabase following the same process that was established during the testing stage. Unfortunately, the loaded feature classes again were not coincident with the source shapefiles. The problem was determined to be that the spatial reference settings were still not correct. The following solution was implemented to correct this problem. A shapefile was created that contained state boundaries for the western U.S. including Alaska, Washington, Oregon, California, Idaho, Montana, Utah, Nevada, Arizona, North Dakota, South Dakota, Wyoming, and New Mexico. This shapefile used the NAD83 datum but was not projected. The author used an ESRI executable (CalcLoadingParams.exe) to determine the proper spatial domain for the ArcSDE geodatabase based on the shapefile of the western U.S. The existing feature classes were deleted from the geodatabase and new empty feature classes were created in their places. A new feature class was added to the geodatabase to accommodate the state boundaries for the western U.S. and this feature class was loaded first in order to establish the spatial domain for the whole geodatabase. The rest of the feature classes were then loaded using the same spatial reference information as the western states feature class. This process resulted in the feature classes being fully coincident with the source shapefiles.

Performing a series of tests through each research team user account tested the functionality of the final geodatabase. The tests ensured that the users could access and view all of the data within the geodatabase, that users could edit their own data, and that the relationships between feature classes and tables were properly defined.

Metadata collected within the master data dictionary was added to the geodatabase through ArcCatalog in two ways. Metadata that had been entered into the personal geodatabase during the development stage was exported from the personal geodatabase in XML format and then imported into the ArcSDE geodatabase. Metadata that had not previously been entered was entered through the FGDC style sheet provided by ArcCatalog.

A training session with the research teams was planned, completing the implementation process. The training session would address user accounts and their permissions, how to connect to the ArcSDE geodatabase, how to view, edit, and analyze the data within the geodatabase, and how to import and export data to and from the geodatabase. The training session has not yet been held at the time of completion of this thesis.

Process Assessment

Finally, a process assessment was completed for the rapid response geodatabase development lifecycle. The purpose of the process assessment was to address the effectiveness of the development process. The author conducted the process assessment by analyzing the documentation of the project lifecycle. The process assessment addressed the following questions: What tasks were effective? What tasks were not effective? Were all of the requirements met? What areas can be improved upon for the future? The following section is an overview of the results of the process assessment.

The majority of the tasks undertaken throughout the development process were effective. However, even though these tasks were effective, many were not completed in accordance to the timelines established throughout the lifecycle. Thus, it was often the inefficiency of task performance and unrealistic timelines that caused problems during development, rather than the fundamental effectiveness of a task.

There were several tasks that were performed during the course of the project that were ineffective and should be eliminated or improved upon for future implementations of similar projects. The Web site created to disseminate information regarding the project was largely ineffective. Web sites can be very effective communication tools if the resources are available to keep them accurate, up-to-date, and ensure that the relevant parties access them. The rapid response Web site was updated only once throughout the course of the project. Ultimately, it provided only preliminary project information and contact information for the project participants.

The initial request for data dictionaries made by the NCLFA to the researchers was relatively ineffective. This request simply asked for a listing of data relevant to the principal investigator's rapid response research. The information contained within the data dictionaries that were submitted was insufficient for the full design and development of the geodatabase. The initial request should have included a more comprehensive list of fields to submit and detailed instructions on the information required by the developers.

The task of creating a database design team consisting of members from each research team was not effective in its original purpose. The original intent was for this team to collaborate in the actual design of the geodatabase, in practice, the database design team acted to verify the design created by the NCLFA. Thus, a distinction should be made between a design team and a data committee. Both of these entities must be informed of their duties and possess the knowledge base required to complete the assigned tasks.

The attempt by the NCLFA to perform the functional analysis with the input of only one primary investigator was ineffective and it had a negative effect on the development lifecycle. This group did not possess the full list of data and products that would be contributed to the geodatabase and thus could not answer all of the necessary questions when performing a functional analysis. Because of this, they came to the wrong conclusions of how the geodatabase would function. For future implementations of a project of this type, it is recommended that a data committee representing all relevant parties perform the functionality analysis very early in the project lifecycle.

The requirements for the rapid response geodatabase were never fully documented, so it is difficult to tell if they had been met or not. Ultimately, the rapid response geodatabase exceeded the requirements of the JFSP. The JFSP had asked that a common database architecture be *investigated*. Their request was answered with the full development of a common database architecture in the form of a geodatabase. The rapid response geodatabase is a functional, interoperable GIS that allows multiple users to view, edit, analyze, and export spatial data relating to several JFSP-funded rapid response research projects.

All of the successes and failures identified in the process assessment and throughout the course of the development lifecycle provide the developers with opportunities to learn from their mistakes and improve upon the process the next time around.

Chapter Summary

This chapter described the geodatabase development lifecycle from conceptualization to implementation. The author addressed all of the tasks undertaken throughout the hardware and software assessment, and the pre-design, design, development, and implementation stages, concluding with the process assessment. The next chapter will describe in detail the final structure of the rapid response geodatabase.

CHAPTER 4

THE RAPID RESPONSE GEODATABASE STRUCTURE

This purpose of this chapter is to describe the final structure of the rapid response geodatabase. The six research areas included in the geodatabase and the data included by each area will be described. The organization of the data into feature classes and tables will be discussed as well as the relationship classes and value domains that were implemented in the geodatabase.

The rapid response geodatabase was designed and developed around data collected by multiple researchers at the site of the Cooney Ridge Fire. However, it was the intent of the designers to ensure that the geodatabase was scalable such that data collected at other rapid response research sites could be entered into the same geodatabase structure. The feature classes and tables within the geodatabase were designed based on the idea that the data collected at new sites would be appended to the existing structure rather than creating new feature classes and tables for each site. Each feature class and table within the geodatabase contains the *FireID* field to distinguish data collected at any given rapid response research site.

The rapid response geodatabase was designed to incorporate data from six rapid response research areas. These research areas are as follows: the Autonomous Environmental Sensor (AES), the Fire Behavior Package (FBP), Fuel Loading (Fuel), Burn Severity (Sev), Airborne Thermal Infrared (ATIR), and Ground Thermal Infrared (GTIR). The data associated with each of these research areas were submitted by different research teams. The goal of the rapid response geodatabase was to incorporate all of these data sources into a multi-user geodatabase through which each researcher could perform analysis on their own data and the data provided by their fellow researchers.

The research teams were asked to identify all of the data fields that would be created throughout the course of their research beginning with the original data collected in the field and ending with the highest level derived data field that could be identified. The researchers objected, stating that the original field-level data would not be used in analysis, rather the data used in analysis would be the summary statistics derived from the original data. The database designers pointed out that the inclusion of the lowest level of data would enable the geodatabase to become the first point of data entry and thus eliminate the use of spreadsheets for data storage and manipulation. Subsequently, functionality could be built into the geodatabase to automate the calculation of the necessary summary statistics.

Throughout the pre-design process, the NCLFA worked with the researchers to identify all of the data fields that would be included in the geodatabase. The data fields were first organized by research area. The data fields from each area were then organized into feature classes and tables. The organization process employed the use of UML diagrams created in Microsoft Visio to provide a graphical representation of the geodatabase.

The following sections address each research area in turn, discussing the data organization and the relevant relationship classes and attribute domains. A UML diagram of the structure of each research area is shown. Detailed descriptions of the data fields and field properties of the feature classes and tables for each research area can be found in Appendix C. Detailed descriptions of the value domains present within the rapid response geodatabase can be found in Appendix D. Detailed descriptions of the relationship classes established within the rapid response geodatabase can be found in Appendix E.

Autonomous Environmental Sensor Research Area

The AES research area involved the placement of AES instruments, each of which consisted of multiple sensor units and a data recording instrument, in the path of a fire. At the Cooney Ridge fire, the sensors on the AES instruments measured air temperature, wind speed and direction, relative humidity, and thermal flux. The data recorders captured time series data from all of the sensor units simultaneously. The locations of the AES instruments were recorded with a GPS unit at the time that the instruments were set up.

The AES data were organized into one feature class and one table. The feature class, *AES_InstPT*, contains the identification and location information for each instrument placed in the field. The table, *AES_Data*, contains the time series data collected from each of the instruments.

Each record in the *AES_Data* table contains the instrument identification field upon which a one-to-many relationship class was built. Figure 5 shows the final geodatabase structure for the AES research area.

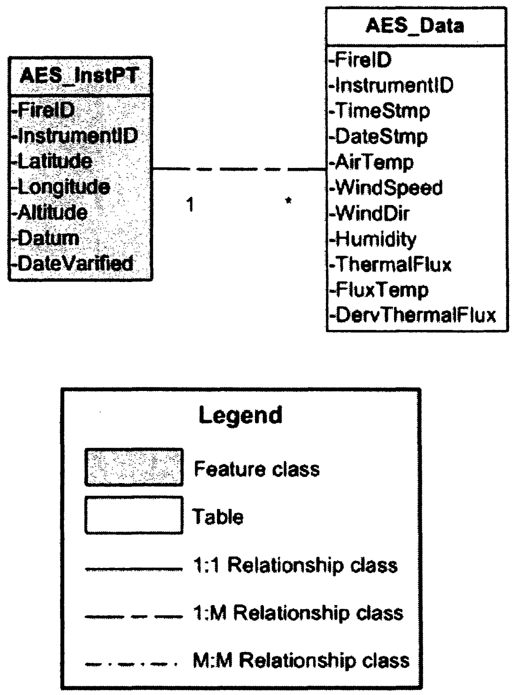

Figure 5—UML diagram of the AES research area data structure.

The AES research area required no coded value domains for attribute validation. Several range value domains were implemented for the AES data to enforce required field precisions. The assignment of these domains is given in Appendix C.

Fire Behavior Package Research Area

The FBP research area involved the placement of FBP instruments, each of which consisted of multiple sensor units in conjunction with a data recording instrument and a digital camera. The sensors on the FBP instruments measured air temperature, total heat flux, radiant heat flux, vertical and horizontal air velocities, and flame emissivity. Just like in the AES research area, the FBP data recorders captured time series data from all of the sensor units simultaneously. The camera recorded images that could be linked to the time series data. Also like the AES research area, the FBP instrument locations were recorded using a GPS unit at the time the instruments were set up.

The FBP data were organized into two feature classes and three tables. The *FBP_InstPT* feature class contains the point location and instrument information (instrument identification and sensor calibration information) for the FBP instruments. The *FBP_CameraPT* feature class contains the point location and instrument information for the FBP digital camera. The original data collected by the FBP instruments were raw, uncalibrated data and were stored in the *FBP_Data_Raw* table. Calibrated data were derived from the original raw data and stored in the *FBP_Data_Calibrated* table. Fire behavior data were derived from the calibrated FBP data and stored in the *FBP_FireBehav_Derv* table.

The *FBP_InstPT* feature class participates in one-to-many relationships with both the *FBP_Data_Raw* and *FBP_Data_Calibrated* tables based on the *PackageID* field. Neither the *FBP_CameraPT* feature class, nor the *FBP_FireBehav_Derv* table participate in any relationships. A UML diagram of the FBP research area structure is shown in Figure 6.

The FBP research area required no coded value domains for attribute validation. Several range value domains were implemented for the FBP data to enforce required field precisions. The assignment of these domains is given in Appendix C.

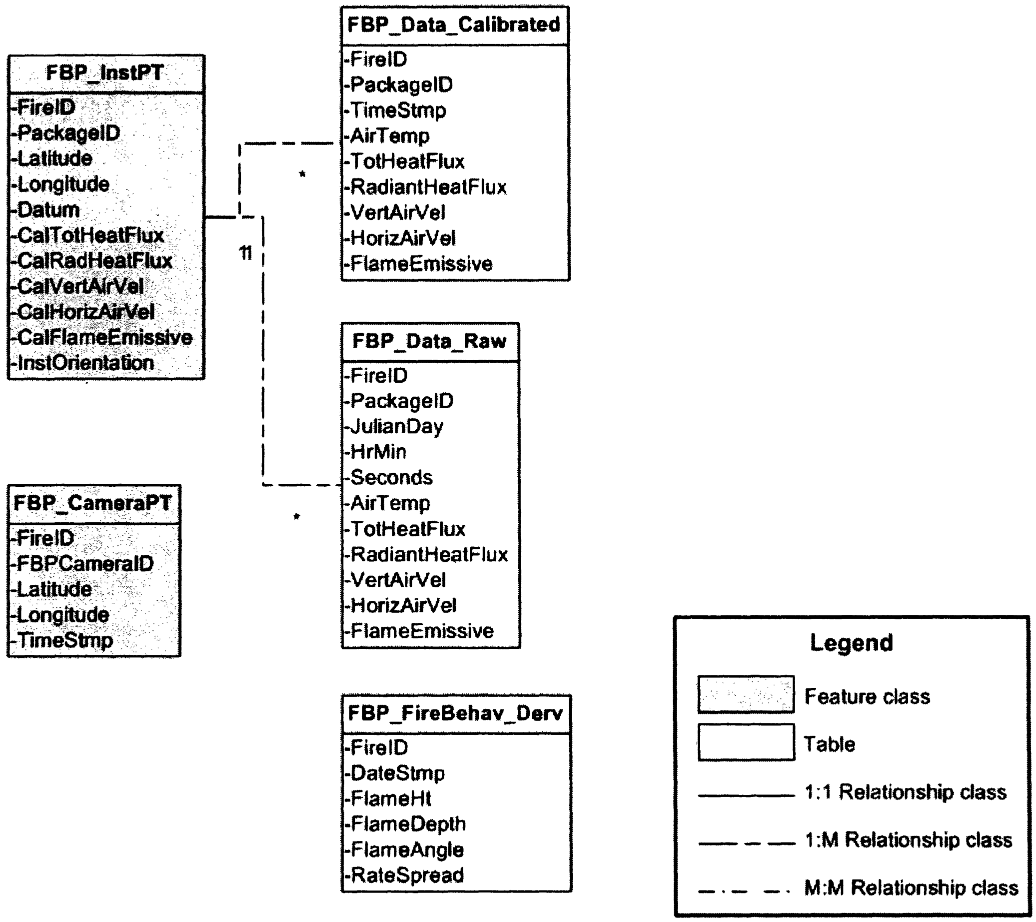

Figure 6—UML diagram of the FBP research area data structure.

Fuel Loading Research Area

The Fuel Loading research area involved extensive field observations collected by research personnel. Field observations were made within a plot that was established within an unburned area near the Cooney Ridge fire that was expected to burn during the course of the fire. Observations were made both before and after the plot burned over. The center of the Fuel Loading plot was recorded using a GPS unit.

The Fuel Loading data were organized into one feature class and eleven tables. The *Fuel_PlotPT* feature class contains the point location information for the center of the field plot. The *Fuel_PlotPT* feature class also contains general descriptive information collected only once at any given Fuel Loading plot site. The Fuel Loading data that were organized into tables were separated first by category. The categories included plot information, fuel moisture observations, live fuel observations, and dead fuel observations. Within each category, data fields were organized by the number of observations per field plot, per sample date, and the data level (observed or derived). For example, the researchers made observations of each of the mature trees within the Fuel Loading plot both before and after the plot burned over. Summary statistics were then derived from these observations to describe the number and general condition of mature trees within the Fuel Loading plot both before and after the plot burned over. Thus, the fields that contained multiple field-level observations for each sample date were organized into tables and subsequently related to the tables containing the derived data. These relationship classes were defined as many-to-many relationships because the relationship was based on the *PlotID* field and each *PlotID* value occurred multiple times for each sample date. The tables that contained the derived data were then related to the *Fuel_PlotPT* feature class with one-to-many relationship classes. A UML diagram of the structure of the Fuel Loading feature class and tables is shown in Figure 7.

Ten coded value domains were created to provide attribute validation for the Fuel Loading data. These value domains provide selection values for the units of measurement, decay class, vegetation size class, fuel category, tree (health) status, fire type, percent live crown ranges, mortality codes, and species codes. The assignments of the coded value domains are given in Appendix C. The contents of the coded value domains within the rapid response geodatabase are listed in Appendix D.

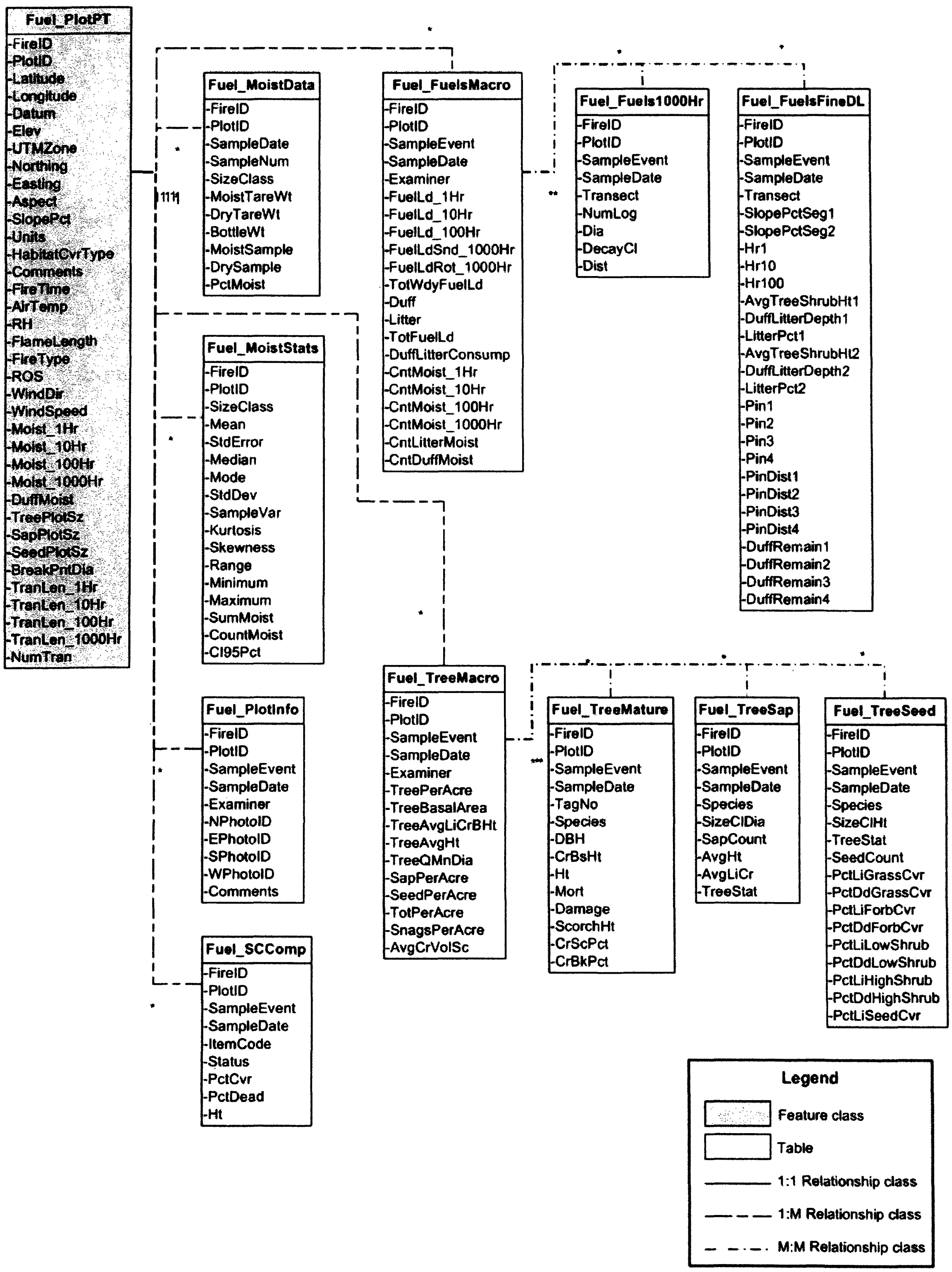

Figure 7—UML diagram of the Fuel Loading research area data structure.

Burn Severity Research Area

The Burn Severity research area involved extensive field observations of the burned site both immediately after the fire and one year later. Field observations were made within a series of plots and subplots established at the site. At each Burn Severity study site, the area as a whole is defined as the site. The site encompasses one or more set of plots. Each set of plots consists of nine plots unevenly spaced along intersecting 60-meter transects. Each individual plot contains 15 subplots spaced 1-meter apart. In addition, the plot located at the intersection of the two transects contains three vegetation plots of 1/50-, 1/100-, and 1/750-hectare. The center of each plot is recorded using a GPS unit. Figure 8 shows the Cooney Ridge site layout for the Burn Severity research area.

The Burn Severity data were organized into two feature classes and ten tables. The *Sev_PlotPT* and *Sev_SubPlotPT* feature classes contain point location and identification information for the severity plot and subplot centers. The severity data fields were organized by site-level data, plot-level data, and subplot-level data. Within each level, the data were organized by category. The categories included soil data, water infiltration data, and vegetation data from the 1/50-, 1/100-, and 1/750-hectare plots. Soil and water data were collected at the subplot level, thus the *Sev_SoilSubPlo*t and *Sev_WaterSubPlot* tables were created (see Figure 9). One-to-many relationship classes were created to relate these two tables to the *Sev_SubPlotPT* feature class based upon the *SubPlotID* field. The soil and water data collected at the subplot level were summarized to the plot level and these summary statistics were included in the *Sev_SoilPlot* and *Sev_WaterPlot* tables respectively (see Figure 10). These tables were related to the

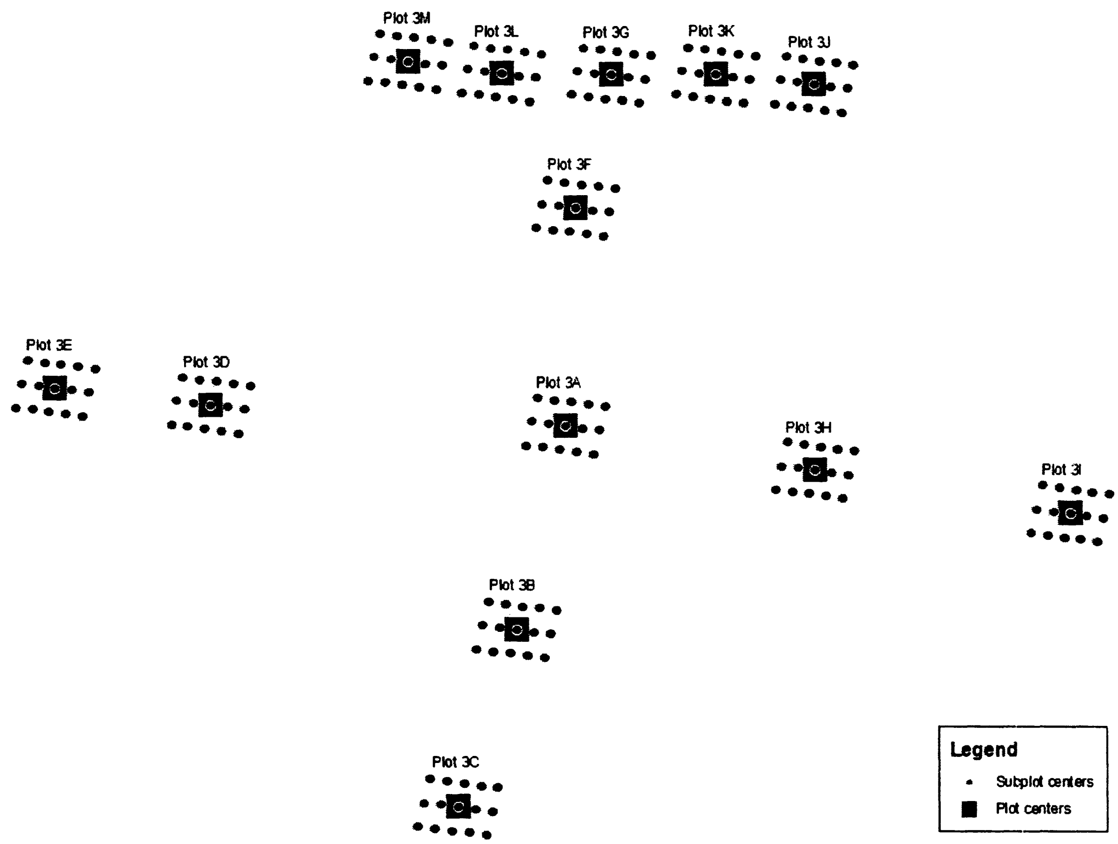

Figure 8—Burn Severity plot layout at the Cooney Ridge fire.

Sev_PlotPT feature class with one-to-many relationship classes based on the *PlotID* field. Observed and derived vegetation data from each of the three vegetation plots were organized into four different tables, *Sev_Tree50Data*, *Sev_Tree50*, *Sev_Veg100*, and *Sev_Veg750* (see Figure 10). The data fields collected at the 1/50-hectare vegetation plot were split into two tables because the observed data consisted of multiple observations per plot per sample date. Thus, the observed data were placed in the *Sev_Tree50Data* table and the derived summary data were placed in the *Sev_Tree50* table. A many-to-many relationship class was created to link these two tables through their *PlotID* fields. Next, the *Sev_Tree50*, *Sev_Veg100*, and *Sev_Veg750* tables were related to the *Sev_PlotPT* feature class with one-to-many relationship classes based on their *PlotID*

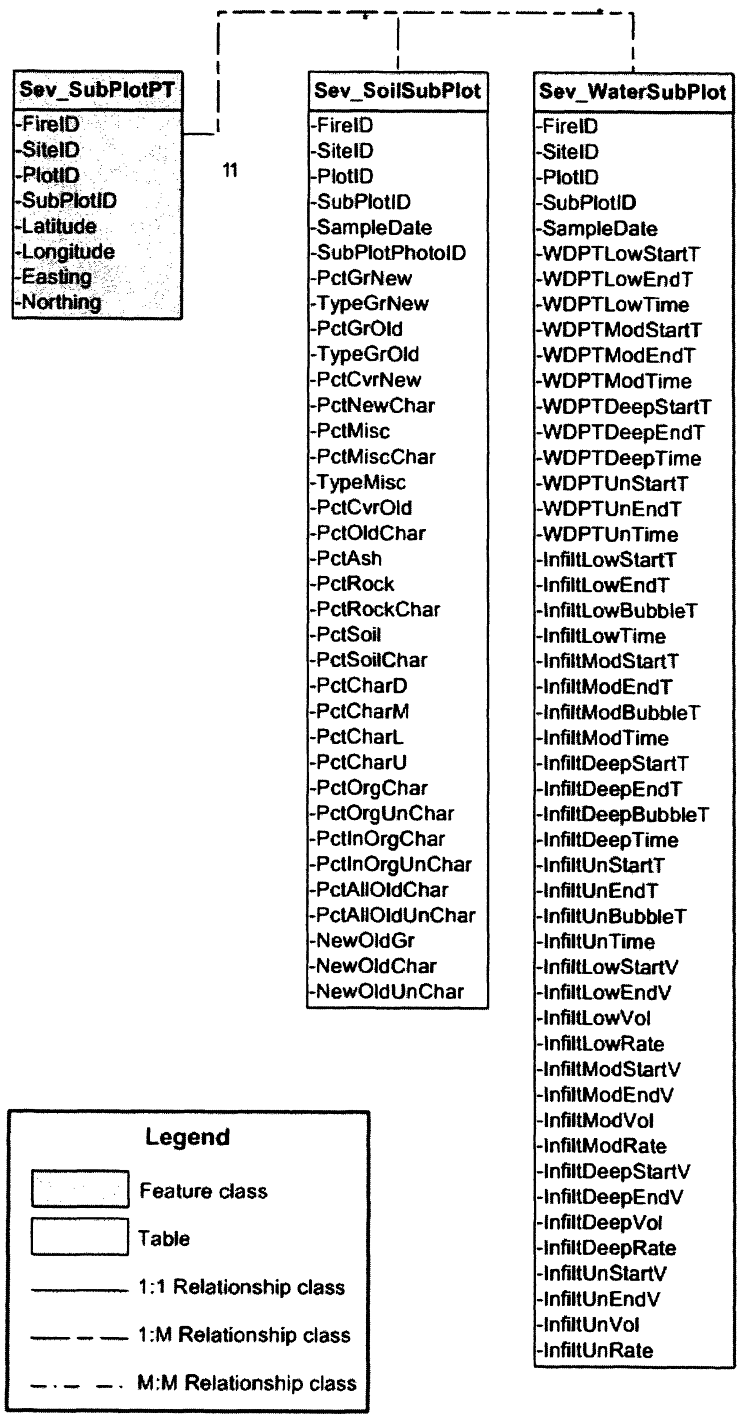

Figure 9—UML diagram of the Burn Severity research area subplot data.

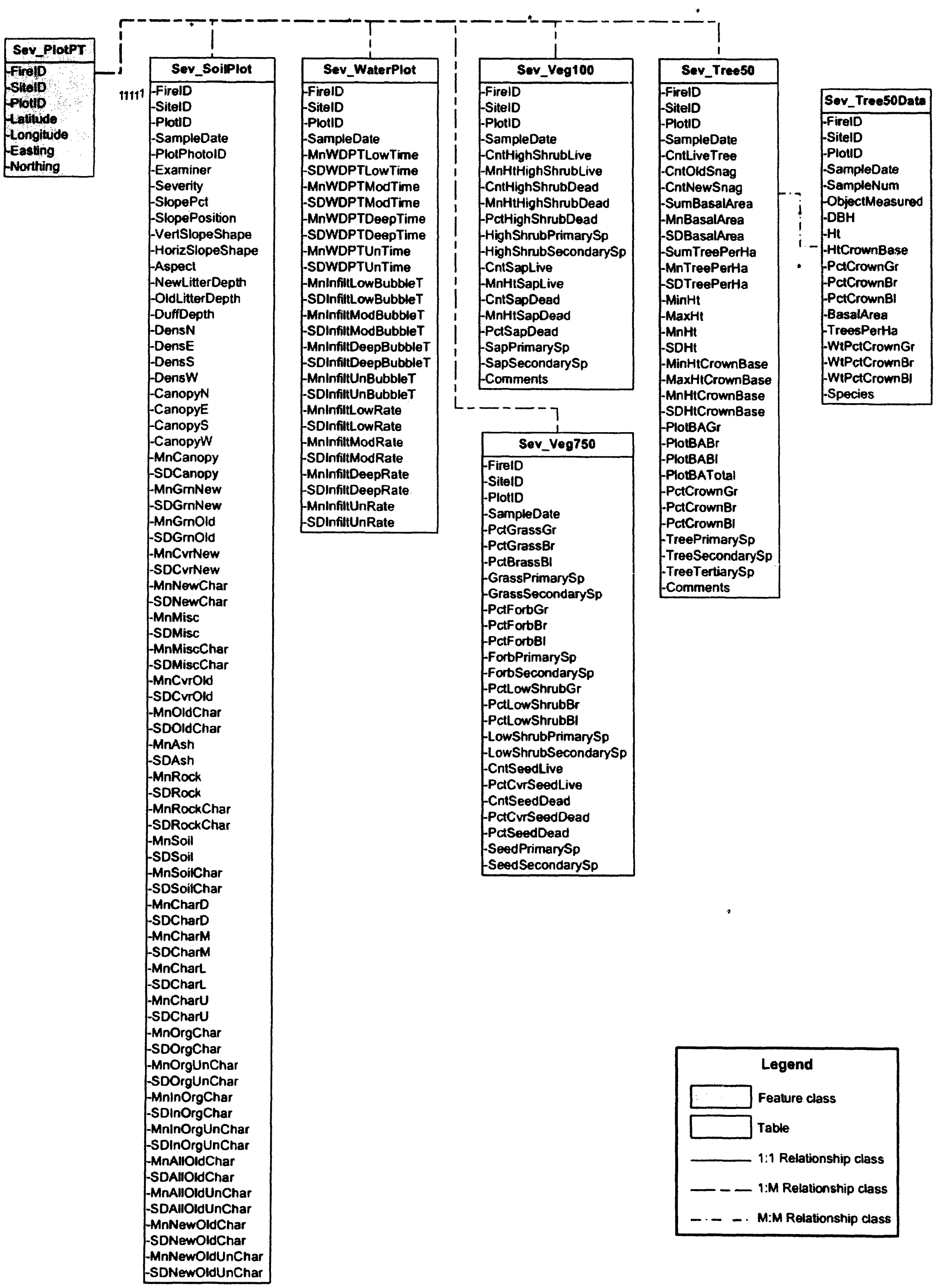

Figure 10—UML diagram of the Burn Severity research area plot data.

fields. Finally, the soil and water data were summarized to the site level. These data are contained within the Sev_SoilSite and Sev_WaterSite tables (see Figure 11). There were no site-level features to relate these tables to.

Range domains were assigned to enforce required field precisions for numeric data. This was especially important for validating that values entered into fields specified as percentages fall between 0 and 100. There were no coded value domains assigned to severity fields.

Airborne Thermal Infrared Research Area

The ATIR research area involved remotely sensed thermal imagery being collected from an aircraft flying over the fire as it burned over the identified research site. The ATIR data included in the rapid response geodatabase is primarily raster data. A table was created to catalog the name, date, time, and location of each raster image (see Figure 12). At the time this thesis was completed, the final raster imagery had not yet been received by the NCLFA.

Ground Thermal Infrared Research Area

The GTIR research area involved the collection of thermal video imagery of the fire from a vantage point on the ground across the valley from the identified research site. Still images were extracted from the video imagery and projected onto a raster image (DEM) of the research site.

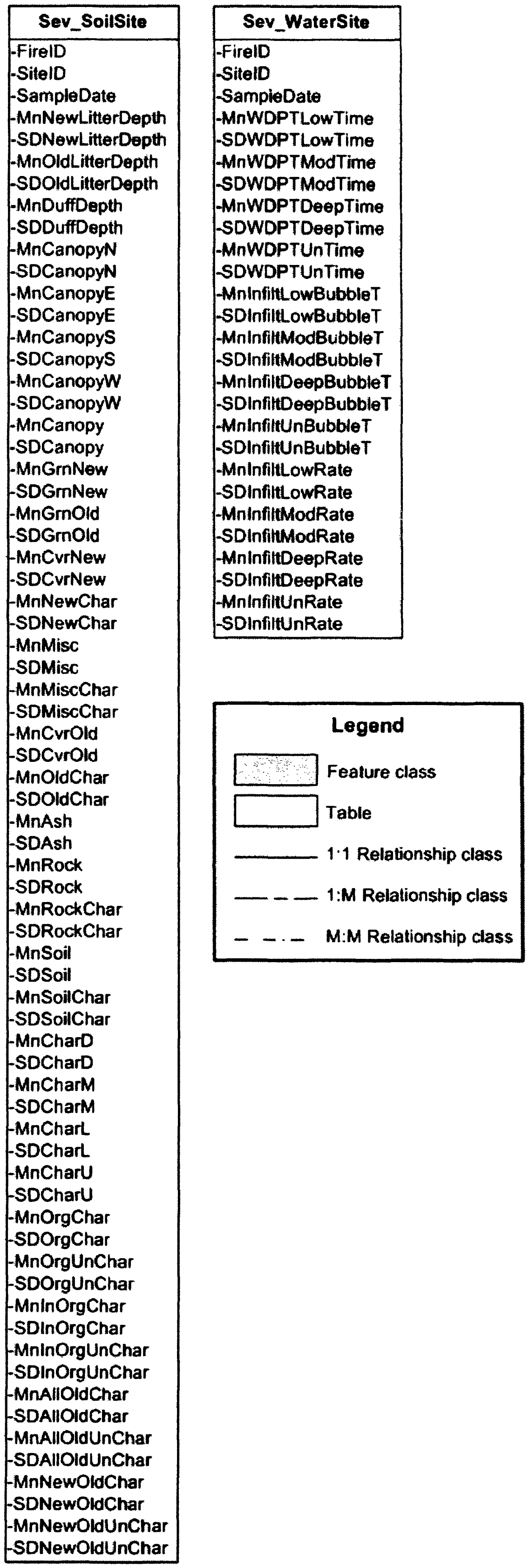

Figure 11—UML diagram of the Burn Severity research area site data.

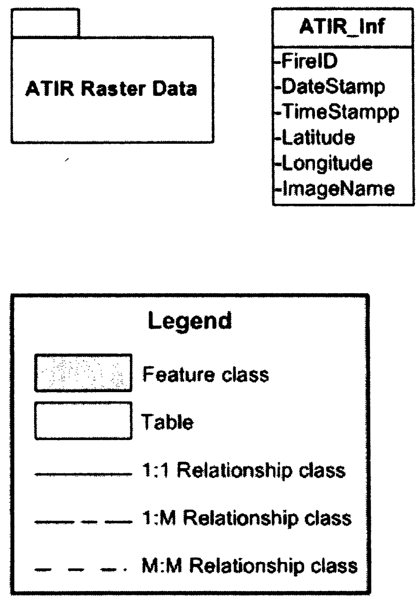

Figure 12—UML diagram of the ATIR research area data.

The GTIR data were organized into two feature classes and three tables. The *GTIR_CameraPT* feature class contains point location and identification information for the thermal video camera. The *GTIR_LandscpPOLY* feature class contains polygon features depicting the boundaries of the projected thermal imagery. A table, *GTIR_Video*, was created containing the camera identification information, date, time, and file name and location for each of the video files captured by the GTIR research. The *GTIR_StillData* table was created to catalog the identification information for each still image captured from the GTIR video files. These two tables were related to the *GTIR_CameraPT* feature class with one-to-many relationship classes based on their *CameraID* fields. The *GTIR_StillData* table was also related to the *GTIR_LandscpPOLY* feature classes with a one-to-one relationship class based on their *StillID* fields. Additional data was derived regarding the pixel centers for each pixel within each still image cataloged within the *GTIR_StillData* table. These data were subsequently cataloged in the *GTIR_StillPixelCenters* table. The *GTIR_StillData* table and the *GTIR_StillPixelCenters* table were related with a one-to-many relationship class based on their *StillID* fields. Figure 13 shows the UML diagram of the GTIR research area data.

The derived raster images depicting the GTIR data projected onto a DEM were to be included in the geodatabase. At the time this thesis was completed, the NCLFA had not yet received the raster data.

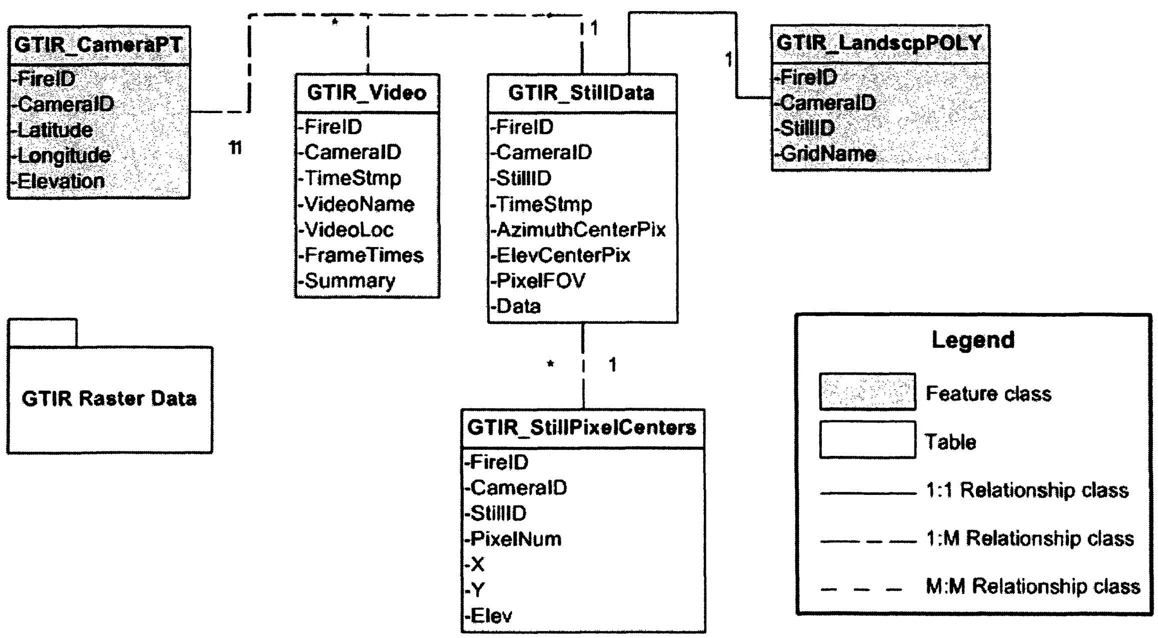

Figure 13—UML diagram of the GTIR research area data.

Range domains were assigned to enforce the required field precisions for the numeric data. There were no coded value domains assigned to GTIR fields.

The final ArcSDE geodatabase structure, as it appears in ArcCatalog, is shown in Figure14.

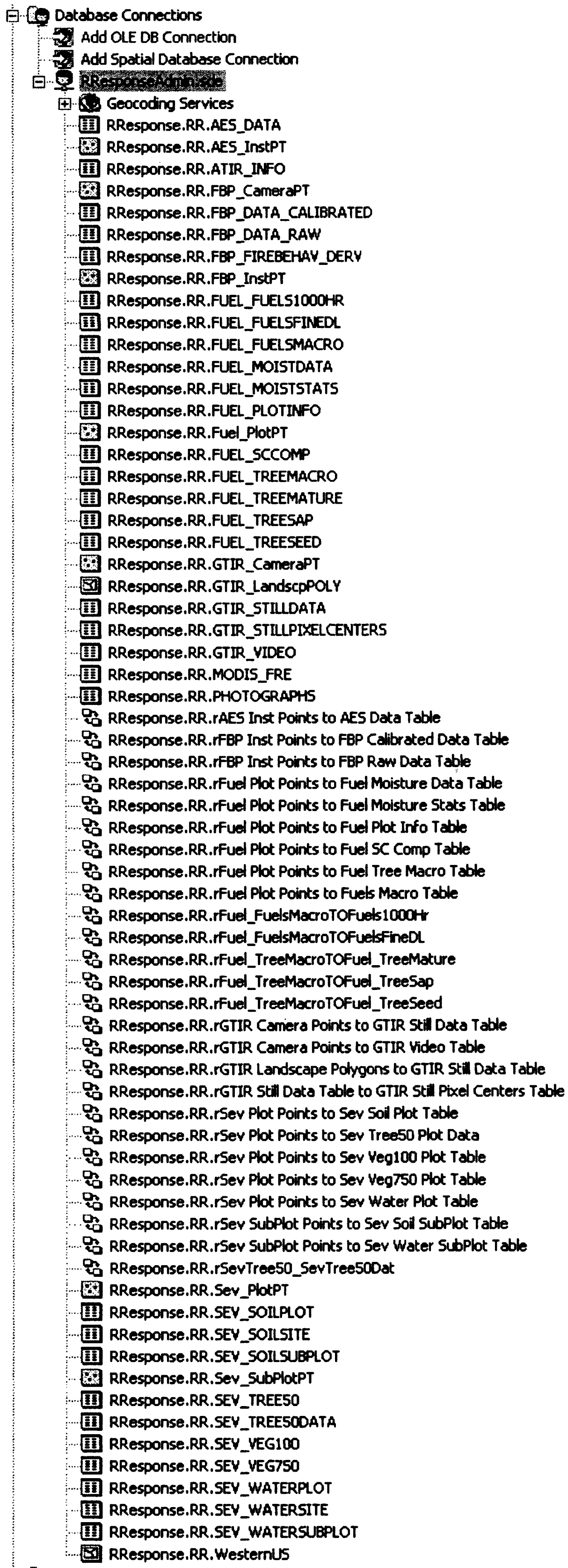

Figure 14—The final structure of the rapid response geodatabase as seen in ArcCatalog.

Chapter Summary

This chapter addressed the structure of the rapid response geodatabase. The creation of feature classes and tables for each of the six research areas were described in detail. UML diagrams of the structure of each research areas were provided in this chapter and a detailed listing of the fields and field properties of the feature classes and tables are provided in Appendix C. The contents of the coded value domains contained within the rapid response geodatabase are provided in Appendix D. The structure of each of the relationship classes contained within the rapid response geodatabase is detailed in Appendix E.

CHAPTER 5

PROTOCOL FOR GEODATABASE DEVELOPMENT

The purpose of this chapter is to provide a protocol for developing a common geodatabase through which multiple research projects can perform spatial analyses and share research data. The protocol is written specifically for developing an enterprise geodatabase using ESRI's GIS technology (ArcGIS and ArcSDE) and SQL Server. It is not intended to teach the reader how to use the software or methodologies discussed, but rather describe how they should be implemented in the development process.

Geodatabase design and development involves the creation of a database structure that accommodates both tabular and spatial data within one single database. Traditional GIS design and development has focused mainly on the creation of structures to accommodate spatial data and then allow the user to link tabular data stored in binary files. Traditional database design and development has not addressed the storage of spatial data within a database, database design has focused entirely on the storage of tabular data. Thus, in order to accomplish true geodatabase design, the concepts of GIS design must be combined with those of database design.

When undertaking a large-scale design and development project, it also is important that the project follow a well-defined process. The field of software engineering specializes in the development of software processes. General process models are available that can be adapted to fit a given development project.

This protocol for geodatabase development was created based on the waterfall process model.[111] The protocol is divided into six stages: conceptualization, pre-design, design, development, implementation, evolution, and process assessment. These stages and the tasks associated with them were derived from a combination of software process theory, the principals of database design, and spatial database design theory. In addition, the author analyzed the efficiencies and inefficiencies encountered throughout the rapid response geodatabase development lifecycle; thus, the theoretical phases and tasks were modified and supplemented by the observed phases and tasks undertaken throughout the lifecycle of the rapid response geodatabase development project. Each section of this protocol contains a footnote referencing a relevant discussion of the task at hand in chapters 2 (theory) or 3 (observed development lifecycle) of this thesis.

This protocol for geodatabase development was written with the federal fire research community in mind. However, its use can certainly extend beyond that community. There were several assumptions to recognize when considering the use of this protocol. The first assumption being made is that the organization using this protocol is a federal agency. Second, it is assumed that the protocol is being used to develop a geodatabase for multiple user groups wishing to share data between themselves. The third assumption is that the contents of the geodatabase being developed are research-related and that the data have a recognizable spatial component. Fourth, it is assumed that the waterfall model will be applied as the underlying process model for the project. Finally, it is assumed that the decision has been made to use ESRI software products. It

[111] The waterfall process model is described in detail on page 13.

is assumed that the designers have ArcGIS 8.3 with an ArcInfo-level license and that the users have ArcGIS 8.3 with an ArcEditor- or ArcInfo-level license. It is also assumed that there is a designated party to host ArcSDE and that all of the users can access this host.

The following protocol should not be viewed as a rigid structure that should be followed to the letter, but rather as recommendations to help structure the geodatabase development process.

1. **Conceptualization**[112]—The decision to create an enterprise geodatabase must be an informed one. First, it is necessary to ask: Who will be contributing data to and using this data repository? It is important to identify the participants and make certain that all of the researchers agree to participate. Some educational efforts may be required at this point to ensure that all of the primary investigators understand the purpose of the project, the basis of the technology that will be used, and how they will benefit from it. Second, it is necessary to ask: What is spatial about this data? If there is no spatial component to the data, a geodatabase is not the appropriate storage format. If there is a spatial component to the data, a well-designed geodatabase can provide a superior environment for data storage and analysis. Third, it is necessary to ask: Are the resources available to complete this project? The task of developing an enterprise geodatabase is a complicated one. It is important that the personnel involved in the development process have an understanding of geospatial data processes and formats, specifically ESRI's geodatabase data model. It is also important to consider whether the resources are available for the maintenance of the

[112] See pages 43-45.

system and where they are located. For example, an enterprise geodatabase will benefit from the expertise of a dedicated ArcSDE/database administrator. In a project that spans several research teams, it is necessary to identify which group will take responsibility for serving the geodatabase to the rest of the users. If the decision is made to develop an enterprise geodatabase, a risk analysis should be performed for the geodatabase project.

1.1. **Risk Analysis**[113]—The purpose of a risk analysis is to identify factors that could cause the project to fail. Once these factors are identified, they can be mitigated to reduce the risk of failure. It is important to broaden the frame of reference when performing a risk analysis in order to identify all foreseeable sources of risk. It is necessary to pay particular attention to the underlying politics of the project, as hidden agendas, group dynamics, and other issues will take a toll on the development process. It is also important to address the level of understanding of the technology within each contributing research team. Lack of understanding can lead to unreasonable expectations of the geodatabase and/or unwillingness or reluctance to participate in the project. The result of the risk analysis should be a written document listing sources of risk and proposing mitigation techniques for each area of risk identified.

2. **Pre-design**[114]—The pre-design stage of a geodatabase development project will form the foundation for the geodatabase. During the pre-design stage, a hardware and software assessment and a functional analysis of the system should be conducted. The results of the pre-design stage will include three documents. The first document

---

[113] See page 13.
[114] See pages 14 and 46-58.

will be a listing of the hardware and software requirements for the project including details on what hardware and software will need to be acquired. This document will also address who is responsible for acquiring these items, and who is responsible for hosting the enterprise geodatabase. The second document will be a comprehensive requirements document detailing the functionality required in the final system. The third will be a process document detailing the steps and tools that will be used throughout the development process.

2.1. **Hardware and Software Assessment**[115]—The first task in the hardware/software assessment is to compile a list of all the software applications that will be used throughout the project. This list should include (but is not limited to): word processing and spreadsheet software; database software including both Microsoft Access and SQL Server (SQL Server can be replaced here with Oracle, IBM's DB2, or Informix, depending what may be available); UML modeling software (Microsoft Visio or Rational Rose); ESRI's ArcGIS software with either an ArcEditor- or ArcInfo-level license; ESRI's ArcSDE; ESRI's ArcIMS (if a Web interface is to be created for the geodatabase); and any other geospatial software programs that will be used during the course of the individual research projects. The second task during this assessment is to expand the list of software by including the hardware requirements for each software application to be used. Hardware requirements will depend greatly upon the software being used. Some issues to consider regarding hardware requirements include: sufficient storage space to accommodate the estimated future size of the geodatabase; sufficient network capabilities to serve an enterprise geodatabase;

[115] See pages 14 and 45-46.

and availability of desktop and/or mobile computers for users. The third task during the hardware/software assessment is to identify those software applications and hardware items that need to be purchased, and who will be responsible for purchasing those items. Finally, it is necessary to decide who will be responsible for the various aspects of hosting the geodatabase. The result of the hardware and software assessment should be a written document addressing what hardware and software systems are already in place, what needs to be acquired, and putting each group's responsibilities in writing.

2.1.1. **ESRI Software**[116]—The use of ESRI technology is advantageous for many reasons. In the federal arena, ESRI software is common among researchers because the USDA Forest Service and the USDI both have contracts with ESRI to provide ArcGIS software to all employees that have a need for it. ESRI is the leader in geodatabase technology and provides extensive resources for users through the Internet and other publications.

2.1.1.1.**ArcGIS**—Find information about the ArcGIS suite of products at: *http://www.esri.com/software/arcgis/index.html.*

2.1.1.2.**ArcSDE**[117]—Find system requirements at: *http://www.esri.com/software/arcgis/arcsde/about/sys-reqs.html.*

2.1.2. **DBMS Software**[118]—ESRI's geodatabase data model can be created at two levels: the personal geodatabase and the enterprise geodatabase. The personal geodatabase provides functionality to the single user through a desktop-only environment. At this level, the data is stored in a Microsoft

---

[116] See page 26.
[117] See page 27.
[118] See pages 24 and 27.

Access database. The enterprise geodatabase provides extended functionality to multiple users distributed over a network through a desktop, mobile, or Internet environment. At the enterprise level, the data is stored in a Microsoft SQL Server, Oracle, IBM DB2, or Informix database management system. The decision of which DBMS to use can be based primarily on what is available to the group that will host the geodatabase. Further discussion of the advantages and disadvantages of each DBMS can be found on ESRI's on-line support center at: *http://support.esri.com/.*

2.2. **Functional Analysis**[119]—The functional analysis of a geodatabase is one of the most important steps in the development process. If the functionality required of the geodatabase is not known, how can the geodatabase development succeed? It is very important to involve the end users in analyzing the functionality of the geodatabase. If the geodatabase is developed without user input, it will most likely not meet the needs of the users and they will resist its implementation. For example, if the designers develop the geodatabase such that data entry is done through Excel spreadsheets, but the users want to be able to add data through a mobile device using ArcPad, the geodatabase does not meet the users' needs.

The following questions (among others) can be used to guide the functional analysis: How will the geodatabase be used? How will the users access the geodatabase? What is spatial about the data? What are the analysis techniques that the geodatabase will need to accommodate? How will data be input into the geodatabase? How does each user group currently manage their data? What built-in functionality would help automate each group's data processing? Will data

[119] See pages 17, 48-49, 55, 57, and 74.

need to be exported from the geodatabase into a non-ESRI application? If so, what format is required and will a derived product be imported back into the geodatabase? Will data need to be edited within the geodatabase? What metadata needs to be captured within the geodatabase?

A data committee should be formed to assist with the functional analysis. The functional analysis should address user requirements, user constraints, and metadata. The result of the functional analysis will be two documents—a requirements document and a process document—both of which will be discussed in more detail below.

2.2.1. **Data Committee**[120]—The functional analysis should begin with the formation of a data committee. Select two to three members from each research team (including the primary investigator) to participate on this committee. These individuals should have a thorough understanding of their data and collection methods, the processing and analysis techniques to be employed, and an idea of the derived products that may be created. Schedule a meeting of the data committee as early as possible. When preparing for the data committee meeting, it may be necessary to prepare some educational resources. If the data committee members are educated about the geodatabase data model they will be better able to distinguish between unreasonable expectations and attainable functionality requirements. If given free reign, the users may underestimate their needs because they do not know the range of functionality that could be provided, or they may over-inflate their expectations of the system and ask for more

[120] See pages 47-48, 50-51, and 53-58

than can be provided with the available resources and time frame. The designers should come away from the data committee meeting with the ability to prepare a detailed description of the user requirements for the geodatabase.

2.2.2. **User Requirements**[121]—The functionality analysis should address user requirements from two points of view. First, how do users currently process data for management and analysis? Second, how can those processes be improved through the use of a geodatabase?

2.2.2.1. **Interface**[122]—The functional analysis should address the interface through which users will access the geodatabase. ESRI's ArcIMS provides a customizable Web-based interface to access a geodatabase. ESRI's ArcCatalog and ArcMap each provide standard interfaces to a geodatabase. If the users require additional (or simplified) functionality in their interface, custom tools can be created in ArcIMS, ArcCatalog, and ArcMap.

A completely custom-built interface can also be created to suit the users' needs. The creation of custom tools should be carefully considered. In a research environment, custom tools meant to automate analysis can be timesaving, but have the potential to introduce systematic errors. If these types of custom tools are to be created, it is important to ensure that the researcher has sufficient

[121] See pages 14, 17, 49, and 56.
[122] See pages 49, and 55-57.

control over the process and understands it to the extent that they can recreate it manually.

By automating analysis processes for the researcher new to GIS, some of the researcher's ability to utilize the technology to its fullest extent is taken away. This is acceptable for a researcher that may not need to become GIS literate, but may be unacceptable for one that needs to learn the underlying processes. Thus, creating a user-friendly, easy-to-use interface that provides a full range of comprehensive analysis tools is a balancing act. If custom interfaces or tools are to be created, their design and development should be detailed fully during the functional analysis.

2.2.2.2. **Data Entry**[123]—Users must have the capability to enter data accurately, efficiently and easily. The functionality analysis should address who will be entering data, what format the data will be in and what type of interface will best facilitate data entry. The functional analysis should also address the entry of both spatial and tabular data.

When designing a geodatabase for research activities, it is also necessary to address multiple levels of data entry, including the primitive data and multiple levels of derived data. In a field research situation, primitive observations can be recorded on three mediums: on paper-based forms, on a laptop, or on a hand-held mobile computing device. If paper-based forms are used, the subsequent data entry should be done directly into the geodatabase through a series of

[123] See page 57.

custom forms. If a laptop is used, the observations should be entered directly into the geodatabase, again through a series of custom forms. The technology is available to enter field observations directly into an ArcSDE geodatabase through a hand-held mobile computing device with the use of ArcPad software. This last scenario also allows the researcher to connect a GPS unit to the hand-held unit and collect coordinate locations that are then stored directly in the geodatabase.

Whichever data entry scenario is used to enter primitive data, the functional analysis should address the specific fields that will be collected and any other user requirements for them. The functional analysis should also address the tools required for populating fields with derived data. Identifying which tools are already available through the chosen interface and what custom tools will need to be created should do this.

2.2.2.3. **Editing**[124]—The functional analysis should address the question of whether or not the users will need to edit their data once it has been entered into the geodatabase. If editing capabilities are required, identify and document which users need to edit which data sets. In the realm of research, it is generally safe to assume that each researcher should be able to edit their own data but no one else's. This information will help determine the settings for user permissions when configuring ArcSDE.

[124] See pages 51-52, 57.

2.2.2.4. **Analysis**[125]—When a geodatabase is being accessed through ESRI's suite of GIS software, there is a comprehensive suite of spatial analysis tools available through the software. In this case, the functionality analysis should address two issues.

First, what analysis tools will be used, and what underlying table structure do they require? This question is important because improper table structure can limit the spatial analyses that can be performed. For example, if a user needs to interpolate a raster based on a field in a related table, he first needs to perform a join on the two tables in question. A join can only be performed on tables with a one-to-one relationship. Thus, if the table structure is such that there is a one-to-many relationship between the two tables in question, the user will not be able to perform the required task without first changing the table structure. It is important to identify these types of requirements so the geodatabase table structure can be designed to accommodate the users' needs.

Second, what types of custom analysis tools will be required? This issue is again one of customization. Custom tools can be created or processes automated according to the users' analysis needs. This is the time to identify those requirements and document them in detail.

2.2.2.5. **Derived Products**[126]—Derived products can range from statistical summaries to derived raster images to model results. Start by asking

[125] See pages 51-52, 55-57.
[126] See pages 55-56.

what types of derived products will be produced. Follow up by asking what types of data formats will those products be in and how will they be stored in the geodatabase. It is certainly not possible to predict all of the derived products that may be created, and it is not necessary to do so. What is important is that the geodatabase be scalable in that derived products can be added to the geodatabase without changing its integrity.

2.2.3. **User Constraints**[127]—The functionality analysis should address any user constraints that will limit the functionality of the geodatabase. One form of user constraints is hardware and software availability. Another form is standards that the geodatabase must meet.

2.2.3.1. **Standards**[128]—Standards can be viewed both as structural support and system constraint. The purpose of standards is to enforce a certain level of uniformity and interoperability on and between projects. Standards can dictate the structure, terminology, and access to a geodatabase. In the world of geospatial data, OMB's Circular A-16 requires all federal agencies to adhere to Federal Geographic Data Committee (FGDC) data and metadata standards. Executive Order 12906 requires that all federal agencies post all metadata to the National Spatial Data Clearinghouse and that all data be accompanied by appropriate metadata.

---

[127] See page 14.

[128] See pages 32-38, 56, 63, and 146-149.

Federal agencies should utilize the FGDC's Geospatial Interoperability Reference Model (GIRM) to help determine which standards should be used throughout the course of a geodatabase project. This document can be found on the Internet at: *http://gai.fgdc.gov/girm/.*

There are many standards that can be applied to digital geospatial data; however, only three will be specifically addressed here. First, all digital geospatial data must be accompanied by metadata that describes the content of the data, its spatial extents, and by whom it was created. This metadata should conform to the FGDC's Content Standard for Digital Geospatial Metadata (FGDC-STD-001-1998). Second, the transfer of digital geospatial data must adhere to the Spatial Data Transfer Standard (SDTS, FGDC-STD-002, ANSI NCITS 320-1998). Thus, a geodatabase should be able to import and export files in SDTS profiles. Third, the data collection procedures used by researchers often follow industry standards.

These standards will vary depending on data collection methods. It is important that the functional analysis include these standards as they may dictate terminology and structure within the geodatabase. In the world of federal fire research in particular, the Geospatial Task Group of the National Wildfire Coordinating Group's Information Resource Management Working Team is responsible for recommending geospatial data standards. At this time there are no geospatial standards to guide geodatabase development for the federal fire sciences other than the FGDC's metadata content standard.

2.2.4. **Metadata**[129]—Metadata is essentially data about data. It is a source of information describing contents, sources, quality, and format of the data that exists within your geodatabase. Metadata should serve as a guide for designers when creating the geodatabase and as a guide for subsequent users as they utilize the geodatabase.

There are two components to metadata: documentation and properties. Documentation is information about every object within the geodatabase that is entered by the user. This information describes the contents of the geodatabase including field descriptions, sources, quality, code definitions, and contact information. Properties are information that is automatically captured about feature classes by ESRI's ArcCatalog as they are entered into the geodatabase. This information includes the spatial reference and projection of the spatial data, the number of features within a feature class, and the spatial extents.

Determine the fields that will be included in the documentation metadata. Tell the researchers what metadata will be expected to accompany the data when it is submitted. Federal agencies must provide metadata according to the FGDC Metadata Content Standard. ArcCatalog provides a template for metadata entry that meets this standard. Metadata entry should occur during or after the population of the geodatabase. Once

[129] See pages 30, and 61-64.

the metadata has been entered into ArcCatalog, it is stored in an XML file that can be viewed either in ArcCatalog or in a Web browser.

2.3. **Requirements Document**[130]—The results of the functional analysis should be captured within a requirements document. This document should provide an outline of the identified functionality and give specific details of system requirements. For example, the ability for users to enter data through a Web interface may be a requirement. Details such as which variables need to be entered, what the interface should look like, and where the Web site will be hosted should be included in the requirements document. The requirements document will be used to guide the development of the geodatabase, so it should be as detailed as possible.

2.4. **Validation**[131]—Validation during the pre-design phase is the process of ensuring that the functionality requirements identified are realistic, consistent, and complete. Each requirement in the requirements document should be reviewed to determine if it could be achieved with the resources available and within the project timeline. The final requirements document should reflect changes made during validation.

2.5. **Process Document**[132]—A process document is a detailed, step-by-step outline used to guide the development team. A process document should include the model that will be used to guide development, the teams that will participate in the process and their roles and assignments throughout the process, and the timeline for the project. The process document should be task-oriented in that

[130] See page 14.
[131] See pages 14-16.
[132] See pages 14-16.

the development process should be broken down into an ordered series of steps that lead to the completion of the project.

2.5.1. **Models**[133]—There are several theoretical models on which a geodatabase development project can be based. This protocol for development follows the waterfall model of software design. The waterfall model is divided into distinct stages of development where each stage is completed in full before the next stage begins, with results from each stage flowing into the next. The process document should give instructions on the completion of each stage in the project. There are other models that are included in the geodatabase development process. These models include, but are not limited to, the overall process model, the entity-relationship model, models used for custom application design and development, and geodatabase-specific models such as those used to create a geometric network.

Each model that will be used during the geodatabase development process should be described in the process document, as it will be applied in the project. It is also necessary to include the modeling language that will be used to express each model, where appropriate. For example, this protocol describes the use of Unified Modeling Language (UML) to express an entity-relationship model describing the geodatabase schema.

2.5.2. **Teams**[134]—Depending on the extent of the geodatabase development project, there may be one or more teams working on the project, with each of these teams consisting of one or more individuals. Each team should have a

---

[133] See pages 13, 14, 18, and 22.
[134] See pages 46-48.

leader with recognized authority to make decisions, enforce deadlines, and the ability to motivate team members.

A large project requiring many custom tools may have separate teams responsible for the schema, the custom tools, the interfaces, and the database management system (DBMS) and ArcSDE as well. A smaller project may have only one or two individuals responsible for the entire geodatabase development process.

No matter how many people are working on a project, if they do not have specific assignments and strong leadership, it is likely that productivity will decline and the process will get off track. In the process document, the purpose of each team should be well described, and their assignments and associated timelines should be clear.

2.5.3. **Timeline**[135]—Within the process document, assign each task a reasonable time for completion. This timeline should give personnel adequate time to complete each task without compromising the overall time restrictions of the project. A firm deadline for known deliverables will help keep the development process on track.

3. **Design**[136]—Once the pre-design stage has been completed and the requirements and process documents are available, the geodatabase development process moves into the design stage. During the design phase the geodatabase will be modeled, developed, and tested. The result of the design phase will be a finalized conceptual

---

[135] See pages 49-50, and 58.
[136] See pages 18, 22-25, and 59-64.

schema, a functional DBMS/ArcSDE environment, and a plan for implementing user accounts and permissions in ArcSDE.

3.1. **Conceptual Modeling**[137]—Conceptual modeling is the process of conceptualizing the geodatabase structure and putting that structure into writing. In this case, the "writing" is a UML diagram drawn in a software program (e. g., Microsoft Visio). This process involves creating a list of all the data fields, combining them into functional groupings to create tables and feature classes, and developing a naming convention to accurately and consistently name each data object and field.

3.1.1. **Data Dictionaries**[138]—Data dictionaries should be collected from each of the research teams during or after the functional analysis. Initially, these data dictionaries need to contain a comprehensive listing of the primitive data fields, known derived data fields (for example, fields for statistical means and standard deviations), a description of all observed spatial entities, and data collection methodologies. The methodologies with which data are collected in a research situation is important because they will help identify the functional groupings and relationships among the data.

The data dictionaries need to include complete metadata for every entity. Ultimately, the data dictionary will also need to include the following information for each data field: the name of the data field, the table in which it is located within the geodatabase, the data type, length, precision, scale, units, and description. This information needs to come

[137] See pages 22-23, 50, 52-53, and 59-60.
[138] See pages 47, 53-54, and 59-62.

from the research teams and can be collected either when the data dictionaries are initially submitted or after the conceptual schema has been created.

Once all of the research teams have submitted data dictionaries, the database design team will combine them into one master data dictionary. This list should be organized into functional groupings, field names changed to conform to a naming convention, and then used to create the conceptual schema.

3.1.2. **Functional Groupings**[139]—Functional groupings serve as the foundations of feature classes and tables in a geodatabase. For each research project to be included in the geodatabase, first identify the spatial objects within the data set. Next, identify what data are associated with each of those spatial objects. Finally, for each spatial object, identify the major themes (or methodologies) around which data are collected. For example, if data regarding trees, soils, and water are collected at a field site, group the data by the headings of *trees*, *soil*, and *water*.

3.1.3. **Define Relationships**[140]—Once the data have been organized into functional groupings, the relational table structure can be identified. This is perhaps the most important step in the geodatabase design process.

If the feature classes and tables within your geodatabase are not structured properly, the system will loose efficiency and functionality. First, organize the data within the functional groupings. Do this by initially

[139] See pages 26, 52-53, and 76-93.
[140] See pages 18-21, 52-53, 55, 59, and 76-93.

identifying the spatial features. Spatial features representing the same types of data objects will form the foundation for feature classes. For example, data representing plot points and subplot points should be organized into two feature classes. Next, identify the attribute information that is associated with each feature class.

Attribute information stored within a feature class must be data that occurs only once per feature. For example, each feature has only one attribute value for things such as identification number, name, and area. The remaining data fields should be organized into tables that will be related to the feature classes. Start with one table per feature class and sort the data fields by the feature classes that they are associated with. Next, split each table into multiple tables according to any sub groups or data aggregation levels within the data. Identify and add the key fields for each feature class and table (see section 3.1.3.1 below). Draw lines to represent relationships between the feature classes and tables (Figure 15). Ensure that each relationship is based on the appropriate key fields and has the proper cardinality that supports the data within the objects being related (see section 3.1.3.2 below).

Relationships modeled in Visio will be converted into relationship classes within the geodatabase. Joins and relates are not defined within the conceptual schema in Visio—they should be created manually within the geodatabase or a map project after the geodatabase has been developed.

Further instructions on determining relationships within a geodatabase and defining them within Visio can be found at ESRI's Web site.

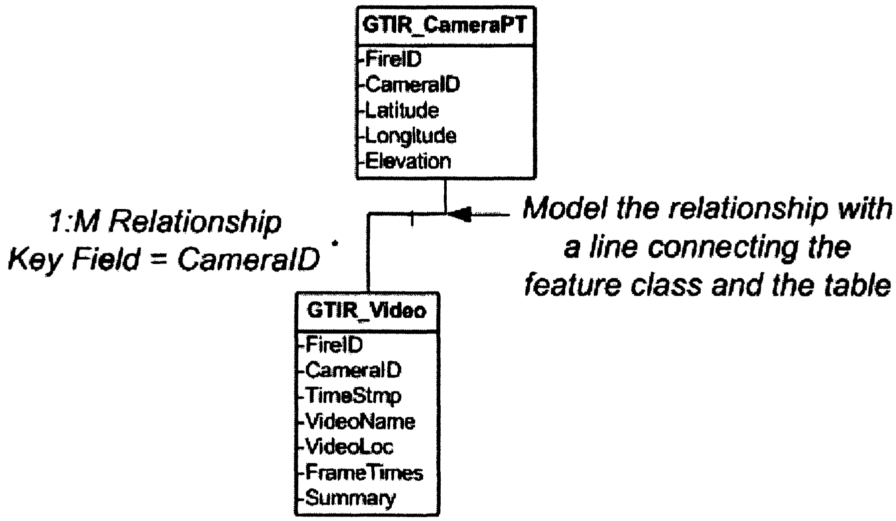

Figure 15—Drawing relationships in UML.

3.1.3.1. **Key Fields**[141]—Relationships depend on the presence of key fields on which the relationship can be based. A key field is a column that contains attribute values that are unique to the object being described. When two tables have key fields containing the same attribute values, they can participate in a relationship. In a geodatabase, all feature class and table objects contain a key field known as OBJECTID.

The OBJECTID field is automatically added to the feature classes and tables as they are created in ArcSDE. As each row of data is added, the OBJECTID field will automatically generate a numeric value that will be unique for every row within a given feature class or table. This field does not work well as a key field on which to base

[141] See page 18.

relationships because a feature in feature class A with an OBJECTID of 10 will probably not correspond to the data in table B that has an OBJECTID of 10. Instead, a more meaningful key field should be created to relate data objects together. First, create a key field that will uniquely identify each feature in a feature class. Next, include that same field in each additional table that will be related to that feature class. Assign values to these fields such that corresponding records in each object contain the same identifying value. These are your key fields.

3.1.3.2. **Cardinality**[142]—Relationships within a database can be described by their cardinality. Cardinality refers to the number of rows that represent each object in a table. The relationships between tables are often expressed as a cardinality ratio; i. e., how many objects participate in the relationship from each table.

There are three cardinality relationships expressed in a geodatabase: one-to-one (1:1), one-to-many (1:M), and many-to-many (M:M). A one-to-one relationship occurs when there is only one row in table A that contains attributes of object X, and only one row in table B that contains additional attributes of object X. A one-to-many relationship occurs when there is only one row in table A that describes object X and in table C, multiple rows contain information about object X. A many-to-many relationship occurs when many rows

[142] See pages 18-21

in table C refer to object X and they correspond to many rows in table D that refer to object X.

The goal for structuring tables within a geodatabase is to maintain the functionality groupings of data, while dividing the data into tables that maintain a given cardinality and keep data duplication to a minimum (preferably, only the key fields will have duplicates).

3.1.4. **Naming Conventions**[143]—A good naming convention is one that uses terminology commonly used throughout a discipline, can be understood by someone not in that discipline, and conforms to the constraints of the computer software. It is important to use terminology that is not only common to a small group of researchers but also common to an entire discipline. This provides a level of interoperability where an individual not part of the immediate research teams can readily decipher the contents of the geodatabase.

Feature class and table names should indicate the contents of the object, yet be brief, as ArcSDE places a 25-character limit on the combined length of the owner's name and the object's name. Field names should be descriptive, brief, and avoid the use of words reserved for use by the DBMS. SQL Server has hundreds of reserved words including date, time, timestamp, zone, and count. It is common to abbreviate words within a field name; however, it is important to ensure that a full description of the contents of each field appear in the metadata. Lastly, keep your naming

[143] See pages 22-23 and 60.

convention consistent. If you abbreviate the word height as "ht" be sure to use that same abbreviation throughout the geodatabase.

3.1.5. **Conceptual Schema**[144]—A conceptual schema is a graphical representation of the geodatabase structure including all of the data objects, their attributes, and their relationships to each other. There are several ways to approach designing a conceptual schema. This protocol recommends creating two diagrams: an inheritance diagram (Figure 16) and a relationship diagram (Figure 17). The inheritance diagram starts with the ESRI Object class. All feature classes and tables in the geodatabase must be connected to this object class in order to inherit the properties of this class, namely, the OBJECTID field. Tables are shown connected directly to the Object class. Feature classes are simply tables with the addition of spatial information that can be read by GIS software. Feature classes are shown connected to the ESRI Feature class, which is in turn shown connected to the ESRI Object class (Figure 16).

The inheritance diagram distinguishes the feature classes from the tables within the geodatabase. The relationship diagram, conversly, shows the relationships that exist between feature classes and tables within the geodatabase. This drawing should be organized by project, with the relationships shown starting at the feature classes and going down through the tables (Figure 17).

[144] See pages 22, 52-53, 59-63, and 78-91.

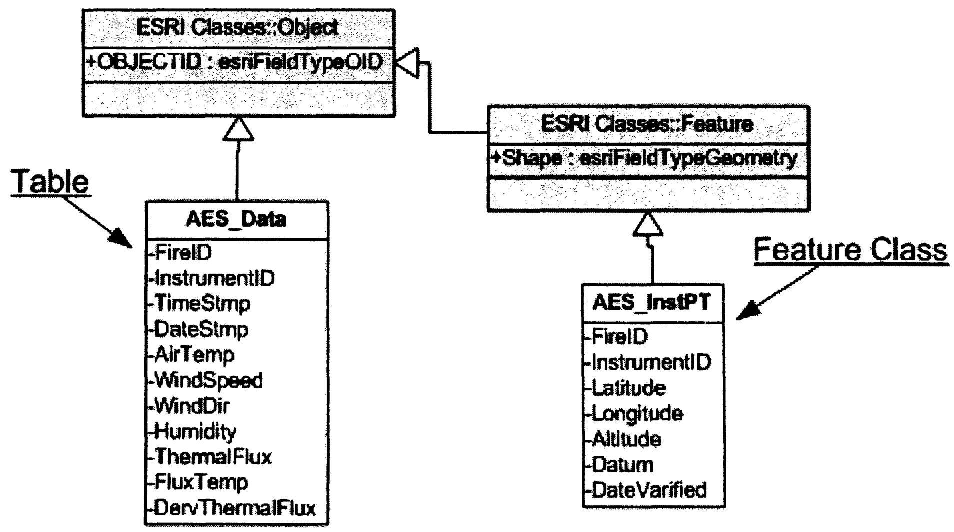

Figure 16—An example inheritance diagram in UML.

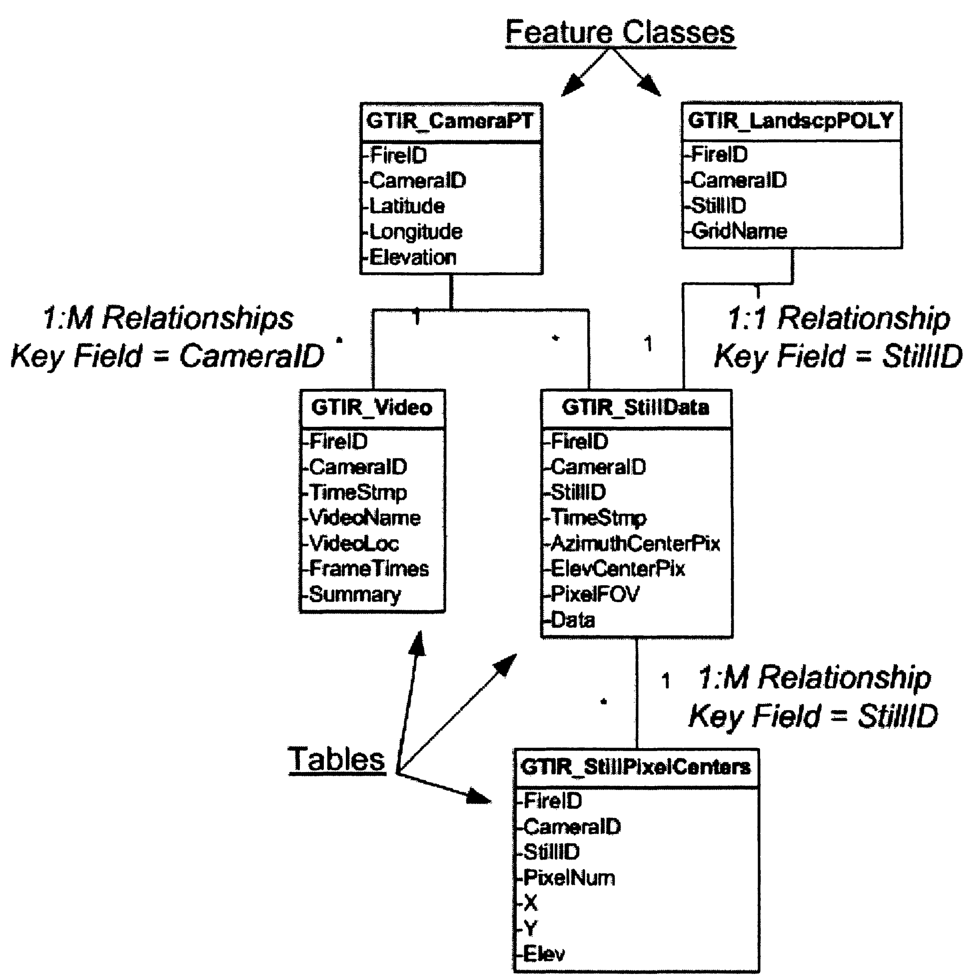

Figure 17—An example relationship diagram in UML.

Reminder: the relationships created within the UML diagram will be translated into relationship classes within the geodatabase.

3.1.5.1. **Microsoft Visio and CASE Tools**[145]—The conceptual schema can be represented using a UML diagram created within Visio. If it is created properly, this UML diagram can be used to automatically create an empty geodatabase structure within a personal or enterprise geodatabase. ESRI has created a set of CASE tools that check the structure of the schema within Visio and convert a schema from a UML diagram to an XML document. Next, the schema wizard is used within ArcCatalog to convert the XML document into the geodatabase structure.

Instructions on the installation and use of ESRI's CASE Tools and the design requirements for creating a UML schema in Visio can be found on ESRI's Web site at: *http://www.esri.com*. The conceptual schema can be created within Visio by creating classes for each feature class and table to be included in the geodatabase. These classes are then given attributes to represent the data fields within the feature classes and tables. The attributes are assigned data types and tagged values, which serve to define the properties of the data field such as length, precision, and scale. Tagged values are also used to assign properties to relationships. There are several sources on ESRI's Web

[145] See pages 15 and 64.

site providing assistance in creating the proper conceptual schema structure in Visio.

3.1.6. **Review Accuracy and Completeness of Data Dictionary**[146]—Once the master data dictionary has been compiled and the initial conceptual schema completed, review the accuracy and completeness of the data structure. This is best accomplished by meeting with each of the research teams (either individually or in a group) and discussing the conceptual schema. Use the UML diagram of the conceptual schema to get the researchers to identify missing or excess elements. The goal is to make sure each research team's portion of the data dictionary is complete and modeled correctly.

3.2. **Logical Design**[147]—The purpose of the logical design stage is to refine the conceptual schema, define the spatial elements, and fill-in the details. The geographical representations of each object should be addressed as well as their topology. The data types for each data field need to be defined along with the length, precision, and scale for each field. Value domains are also defined and created during the logical design stage. The result will be a modified conceptual schema.

3.2.1. **Geographical Representations**[148]—The geographical representation of each data object in the geodatabase should be identified and defined by addressing the following questions: Does the object contain tabular data, vector data, or raster data? If it contains vector data, does it contain points, lines, or polygons? Should certain feature classes be organized into feature

[146] See pages 53-54.
[147] See pages 22, 23-25, and 59-62.
[148] See pages 26 and 59.

datasets? All vector data within a geodatabase is stored within feature classes. Feature classes can only contain one type of vector data (points, lines, or polygons) per feature class. Feature datasets offer a means of grouping related feature classes. For example, feature classes containing hydrography data such as stream lines, lake polygons, and water monitoring station points could be grouped into a feature dataset called *Hydro*.

Feature classes must share the same spatial reference if they are to be placed together within a feature dataset. Feature classes that participate in topology rules or geometric networks must be contained within feature datasets. Ensure that each object is properly defined and organized within the conceptual schema.

Tables are defined by connecting them to the ESRI Object class as discussed above. Feature classes are defined by connecting them to the ESRI Feature class and setting their *GeometryType* tagged value. Feature classes are defined by placing the desired feature classes within a common workspace that has been stereotyped as a feature dataset. Rasters are not defined in the UML diagram, but their existence needs to be recognized and tracked so they may be added to the geodatabase during the implementation stage.

3.2.2. **Spatial Validation**[149]—The geodatabase data model provides spatial validation tools through the use of topology and geometric networks. Topology is the spatial relationship between adjacent features within a geographic data set. The use of topology tools allows you to define the

[149] See pages 26-27.

spatial relationships that can exist between features. Topology rules can enforce the adjacency, coincidence, or connectivity of related features. These rules can be created to govern the features within one or more feature classes.

If multiple feature classes will participate in a set of topology rules, they must be placed together within a feature dataset. Topology rules cannot be created within the UML conceptual schema. They should be identified and written out at this time and created within the geodatabase during the implementation stage. Spatial validation can also occur within a geodatabase through the creation of a geometric network.

A geometric network models the connectivity and direction of flow between features. The feature classes that participate in a geometric network must exist within the same feature dataset. Feature classes cannot participate in topology rules and a geometric network at the same time. Geometric networks can be created within a UML diagram. See ESRI's Web site (*http://www.esri.com*) for more information on establishing topology rules and creating geometric networks.

3.2.3. **Attribute Validation**[150]—The geodatabase data model supports attribute validation through the use of subtypes and value domains. Subtypes provide the ability for features within a feature class to be grouped on a basis of attribute values. The implementation of subtypes validates the feature class by requiring all features to belong to an established subtype. ArcGIS applications use subtypes to support additional feature functionality. Value

[150] See pages 26-27 and 62-63.

domains are structures within the geodatabase that store lists of valid attributes for specific data fields. Coded value domains contain specific values that can be entered in a given data field, and only those values can be entered in that field. Range domains specify a range of values within which every entry in a given data field must fall.

Domains are very effective in limiting data entry error when dealing with known, repetitive values. Both subtypes and value domains can be set up in the UML conceptual schema. Again, documentation and instructions can be found on ESRI's Web site.

3.2.4. **Data Types**[151]—Determine the data types and field lengths for all of the data fields in the geodatabase. This can be done by answering the following questions for each field: Are the attributes text, alphanumeric, or numeric? If they are numeric, does the number include a decimal? What is the maximum length for each text field? What is the precision and scale for each numeric field? Precision is the total number of numeric characters (including decimals), and scale is the number of decimal places (integers do not have scale). These properties can be added as tagged values to each field in the geodatabase. However, at this time (using Visio 2003 and Arc 8.x technologies) the values for precision and scale will not be retained in the geodatabase through the use of the schema wizard.

3.2.5. **Modified Conceptual Schema**[152]—The result of the logical design stage should be a modified conceptual schema. This schema is still in UML

[151] See pages 60-62.
[152] See pages 25 and 59.

diagram form within Visio. You should also have a listing of properties and topology rules that cannot be set within the UML diagram.

3.3. **Physical Design**[153]—There are three parts to the physical design stage. First, any custom interfaces, tools, and applications are designed and developed. The development processes for these custom tools should mirror the basic process described in this protocol. Second, the geodatabase undergoes testing and modification. The testing phase includes the determination of the spatial reference for each feature class, the creation of a test environment, data acquisition, the population of the test environment, the validation of the user requirements, and modifications. Repeat the cycle of creating the geodatabase structure, testing, and modifications until the geodatabase structure satisfies the users' requirements. Third, the installation and set-up of the DBMS and ArcSDE are completed, and user names and permissions are addressed.

3.3.1. **Spatial Reference and Projection**[154]—The spatial reference and projection in which feature classes will be stored within the geodatabase is an important issue. Projecting spatial data will alter one or more of the spatial properties. This alteration is predicable and can be managed by choosing an appropriate projection for the data. The following questions can be useful when determining the projection that will be used: Where is your study area located? How large is your study area? Will the study area increase in size over time? Who will be sharing the data and what

[153] See pages 25 and 64-68.
[154] See page 63.

projections will they be using? Which spatial properties are most critical to maintain within the feature class?

If feature classes are stored in different projections, they can still be displayed in the same coordinate space because ESRI's ArcMap software has the capability for on-the-fly data projection. However, you can only do analysis on layers that have the same projection as that of the data frame.

When deciding on a projection, consider the growth of a given feature class over time. For example, if you are considering storing the feature class in UTM Zone 12, consider the probability of the future addition of features that lie in UTM Zone 11. When developing a geodatabase that includes, or has the potential to include data covering a large geographic area, designers often elect to store feature classes in unprojected geographic coordinates. The datum should be consistent for all feature classes within the geodatabase. The standard datum recommended for Federal agencies is NAD83.

3.3.2. **Conversion of Conceptual Schema into a Test Environment**[155]— Testing the structure of the geodatabase as a personal geodatabase is recommended before creating the geodatabase within ArcSDE. A personal geodatabase provides most all of the functionality of an enterprise geodatabase, just on a smaller, single-user scale. The personal geodatabase does not provide a test environment for the inclusion of raster data or for multiple users. These should be tested in the implementation stage, before the geodatabase is released to the users.

---

[155] See pages 64-65.

If the conceptual schema was created as a UML diagram within Visio, first use the semantics checker tool provided by ESRI's CASE tools to check the schema for errors. This tool will produce a list of errors and warnings that should be addressed within the UML diagram before it is used to create a geodatabase.

Once the errors within the schema have been corrected, export the diagram into an XML document and use the schema wizard within ArcCatalog to import it into an existing, empty, personal geodatabase. The schema wizard will allow each object's properties to be changed during the import process. The spatial reference and projection should be established for feature classes at this time.

3.3.3. **Data and Metadata Acquisition**[156]—If the data have not already been acquired, gather a representative sample of data—if not the entire data set—from each research team. This data should be accompanied by complete metadata. Check to make sure the metadata is complete and accurate. Once the data has been submitted, it is recommended that it be more formally prepared before it is loaded into the geodatabase.

3.3.3.1. **Data Preparation**[157]—Tabular data will likely be submitted in various formats including text documents, spreadsheets, and database tables. Spatial data may be submitted in any of the above forms as well as coverages or shapefiles. However, data can only be imported to a geodatabase through ArcCatalog from properly formatted text files

---

[156] See pages 61-62.
[157] See page 65.

(.txt), comma-delimited files (.csv), database tables, coverages, shapefiles, or feature classes. The file must contain column headings and the data must be assigned the proper data type.

Data preparation may be time-consuming, but if the time is spent formatting the data once at the beginning, the loading process will be smoother, quicker, and more accurate. It is recommended that the data be imported into database tables (Access is a good software program to use here) that mimic the tables within the geodatabase. There are several reasons for this recommendation. First, the data fields can be grouped into the proper table structures. Second, the column headings can be easily changed to reflect those within the geodatabase. Third, the data types can be set and that information will be embedded within each data field. Fourth, it makes loading data into the geodatabase very simple.

3.3.4. **Population**[158]—Next, it is time to populate the trial geodatabase. Population should occur in the same manner in which the users will load data. There are three issues to consider when loading spatial data into a geodatabase: the conversion of the data to a compatible format, correcting any spatial errors and adding attribute information, and the aggregation of individual pieces of data into a complete representation of the study area.

3.3.4.1. **Spatial Data**[159]—Vector spatial data should be loaded into feature classes from shapefiles or coverages. There are a few issues to address

[158] See pages 65-66.
[159] See pages 28-30 and 65-66.

when loading spatial data. The spatial reference and projection should already be set in the destination feature class. The spatial reference and projection of the source data should match that of the destination feature class.

At the current time, the geodatabase data model does not maintain the same level of precision for spatial coordinates as shapefiles or coverages. To prevent loss of accuracy, maximize the precision value when importing spatial data into the geodatabase. The first feature class that is loaded into the geodatabase will establish the spatial domain for the geodatabase as a whole. Calculate the spatial domain such that the maximum spatial extent of the geodatabase will accommodate all current data and any expected expansion. Raster data cannot be loaded into a personal geodatabase. It will be necessary to load and test raster data functionality in the ArcSDE environment during the implementation stage.

3.3.5. **Geodatabase Testing/Validation**[160]—Once the geodatabase structure has been created and populated, systematically test all functionality identified in the requirements document. As errors occur, modify the geodatabase structure appropriately, and retest. It is important to test and validate every user requirement to ensure that the geodatabase structure will support the users' needs.

---

[160] See pages 15 and 66.

3.3.6. **Modifications**[161]—Modifications can be made to the geodatabase structure in the UML diagram. The schema can be re-exported as an XML document and the schema wizard used to apply the changes to the existing geodatabase structure. The modifications will overwrite the existing geodatabase structure only where changes have been made. The schema wizard will overwrite feature classes and tables that contain data and that data will not be lost.

3.3.7. **DMBS Installation and Configuration**[162]—The DBMS that will be used to store the geodatabase should be installed and configured on the designated server.

3.3.8. **ArcSDE Installation and Configuration**[163]—There are several ways to configure ArcSDE as it is installed. These configurations are optimized for various functionality requirements. For example, an ArcSDE environment that will be primarily managing raster data may be configured differently than an ArcSDE environment that will be primarily managing vector data. The installation and configuration of ArcSDE is a complex task and it is recommended that an experienced ArcSDE administrator be consulted for assistance during the installation process.

3.3.9. **ArcSDE User Scenarios**[164]—ArcSDE user accounts and permissions should be carefully designed. Within ArcSDE, the user that creates an object within the geodatabase is its owner. The owner inherently has full

---

[161] See pages 66-67.
[162] See page 67.
[163] See page 67.
[164] See pages 67-68.

permissions to the object and by default is the only user that can view, update or delete this object. It is the responsibility of the owner to assign permissions for other users to be able to view, update, and delete a given object. Permissions can be set such that a given user can only view a given object, or that user can be given view, update, and delete permissions.

It is necessary at this stage to know who is going to be maintaining the geodatabase. Is it going to be a distributed system where the users maintain their own data, or is it going to be centralized with a database administrator maintaining the system? In a research scenario, it is also necessary to decide if each individual will receive their own account or if each research team will receive an account that they will share.

If the geodatabase is going to be a distributed system, the users will require sufficient permissions to allow them to load new objects into the geodatabase. In this scenario, accounts should be established for each user and these accounts should be assigned at least *CreateTable* and *StoredProcedures* permissions. The benefits of this scenario are that a database administrator is not required to maintain the system and the users have the ability to import derived products into the geodatabase. The negative aspects of this scenario are that the users are not held to the designed geodatabase structure, as they can add new objects, and the users may fail to assign the proper permissions for all the other users to view and update or delete their objects.

If the geodatabase is going to be a centralized system, the users will likely only require *CreateTable* permissions. This will allow them to view, update, and delete objects but will not allow them to load new objects. An administrative account should be created through which all objects should be loaded. Thus, the administrator is the owner of all the objects within the geodatabase and is responsible for assigning permissions so the users can view, update, and delete any given object. The users are strictly held to the geodatabase structure as it was designed and a database administrator will ensure that the proper viewing and editing permissions are given to each user in a timely fashion. However, although the users can append new data to existing objects, they cannot add derived products without going through the administrator. If the responsibility of maintaining the entire geodatabase is put on the shoulders of the administrator, it could overwhelm his available resources. These are important decisions and each scenario should be tested and validated against the requirements document.

4. **Implementation**[165]—The implementation of the ArcSDE geodatabase is the process of creating and populating the final geodatabase structure and training the users how to access and use the system. If someone other than the development team will maintain the geodatabase, the transfer of ownership of the system occurs at the end of the implementation stage.

4.1. **Preparation of ArcSDE/DBMS Environment**[166]—The ArcSDE and DBMS environments were installed and tested during the development stage. All test

[165] See pages 14 and 68-74.
[166] See page 68.

data and user accounts must be cleared from both ArcSDE and the DMBS before proceeding with the implementation stage. In order to remove users from the system, all objects owned by the user must be deleted first and then all log files associated with the user must be deleted. After these files are removed, the user account can be deleted from both ArcSDE and the DBMS. Ensure that any modifications identified during the development stage have been made and that the programs have been installed properly.

4.2. **Creation of ArcSDE User Accounts**[167]—The new user accounts are first added to the DBMS and then permissions are assigned within ArcSDE. Adhere to the account structure identified during the development stage. Specific instructions for creating ArcSDE user accounts within specified DBMS' can be found at ESRI's Web site.

4.3. **Creation of Geodatabase Structure within ArcSDE**[168]—The geodatabase structure should be created within the ArcSDE environment either through the use of ESRI's CASE tools, or manually. The use of ESRI's CASE tools to convert a UML schema from Visio to a geodatabase structure works the same in ArcSDE as it does in a personal geodatabase. Simply make a connection to the enterprise geodatabase through the *Database Connections* dialogue in ArcCatalog and use the schema wizard as before.

Unfortunately, because the schema wizard does not maintain the settings for the precision and scale of numeric fields, it may be necessary to create the geodatabase structure manually through ArcCatalog. In this situation, use the

[167] See pages 67-68.
[168] See pages 68-70.

final conceptual schema as a guide as you create the geodatabase structure. If the ArcSDE environment is being created as a distributed system, it is important that each data object be created from the appropriate owner's user account. If it is a centralized system, create the entire structure from the administrator's user account. Remember to assign every user the proper permissions (view or view/ update/delete) for each object.

4.4. **Population of Geodatabase**[169]—Populate the empty geodatabase structure with data that has been properly prepared for loading. All data should be loaded at this time. It is important that the proper spatial domain and precision for the geodatabase be established. The first feature class that is loaded into the geodatabase will set the maximum spatial extent for the geodatabase as a whole. After the first feature class has been loaded, subsequent spatial data can be loaded with differing spatial extents and precisions, as long as the data fall within the maximum spatial extents of the geodatabase.

4.5. **Metadata**[170]—Use ArcCatalog to enter the appropriate metadata as identified in the requirements document. ArcCatalog provides several pre-defined style sheets that conform to the various metadata standards, including the FGDC metadata content standard. New metadata can be entered directly into these style sheets; preexisting metadata can be imported into the ArcSDE geodatabase if it is stored in the proper XML format. The complete and comprehensive provision of metadata allows users to understand the format and origins of the data stored

[169] See pages 70-71.
[170] See pages 71-72.

within the geodatabase. If the metadata information is not captured, the source of the data is questionable and the utility of the geodatabase diminishes.

4.6. **Custom Interfaces and Applications**[171]—If customized interfaces or applications were developed, they must be fully implemented with the final geodatabase before it is released to the user. Ensure that all interfaces and applications are functional and accessible to each target user. If custom products must be run from the users' computers, create an installation package and distribute it to the users.

4.7. **User Training**[172]—It is important to recognize that the target audience for this advanced geodatabase-based GIS may or may not have the knowledge and skill to use the system. The learning curve for an enterprise geodatabase can be steep. Thus, it is essential that the researchers receive training in the use of their new geodatabase. Training can be conducted on an individual level, with each separate research team, or for the group as a whole. Training should be conducted by a person or persons who understand the functionality of the geodatabase, are experienced in GIS, and have the ability to communicate these concepts to individuals with varying levels of experience. The moral of the story: If users do not know how to get at the functionality they require, or are intimidated by the system, they will not use the geodatabase, and it will fail.

5. **Evolution**[173]—Any software application must have the ability to evolve in order to continue to meet the needs of its users over time. It is not uncommon for a system to become out-dated almost immediately after its release. As soon as the users are able

---

[171] See page 15.
[172] See page 72.
[173] See page 15.

to accomplish one task, the results will often leave them wanting to perform another task that was not part of the original functionality of the system. Thus, the system must evolve—it must be scalable to meet increasing data and demands. Good database design will allow for increasing amounts of data. Therefore, as derived data are added to the geodatabase, it is important that they conform to these same design principals. The functionality and interoperability inherent in ESRI's GIS products will provide the geodatabase with compatible software that will allow the functionality of the geodatabase to evolve with advancements in technology. However, this evolution will not be easy unless the geodatabase receives regular, systematic maintenance.

5.1. **Maintenance**[174]—Because of the complexity of the enterprise geodatabase data model, regular maintenance is necessary to keep the data current and the structure clean. Unless the users' access to the geodatabase is very limited and controlled, the structure of the geodatabase will change over time. Most of this change is beneficial as the system evolves with the needs of the users, but some change can be detrimental. Whether or not the ArcSDE is a centralized system, someone should fill the role of "administrator" and monitor the growth of the geodatabase in order to identify and eliminate detrimental practices. Practices that can be detrimental to the geodatabase include things that alter the basic relational structure linking the tables together within the geodatabase, or errors in data entry (specifically errors in key fields). An enterprise geodatabase has the ability to be versioned so multiple users can edit the geodatabase at the same time. These versions need to be reconciled periodically to ensure that all users

---

[174] See page 45.

are viewing the most current data. Reconciliation can be performed by each user or by an administrator. It is recommended that an administrator perform regularly-scheduled reconciliations to ensure that the maintenance is performed.

6. **Process Assessment**[175]—An important stage in any development process is the process assessment. Gathering information about the process itself will allow the developer to identify the successes and failures throughout the lifecycle of the project. Essentially, if the time is taken to identify mistakes, the process can be improved the next time it is implemented. Also, if elements that led to success are identified, they can be improved upon.

A process assessment can be conducted by an individual or a group. When few individuals are involved in the development process, it is likely that one person has been exposed to all of the stages of the development process. That person is likely knowledgeable enough to analyze the efficiencies and inefficiencies of the process through first-hand experience with the project. When the development process is conduced by a larger group, it is unlikely that any one person has been exposed to all of the stages of the development process. In this case, it is important to involve a group of people during the process assessment in order to analyze the project at its fullest extent.

Several methods can be employed when conducting a process assessment with a group of people. A survey can be used to capture responses from a large group of people. Another method is to conduct individual interviews with team leaders and selected team members. This method is especially useful when interviews are conducted throughout the course of the project, allowing successes and failures to be

[175] See pages 16 and 72-74.

assessed as they occur. A process assessment can also be conducted as a workshop where team leaders and team members are invited to discuss the process in an open forum. This method may not capture all of the successes and failures relevant to the process because people may be reluctant to discuss these issues openly with their supervisors and peers.

The following is a list of question that can be addressed, regardless of the method in which the process assessment is conducted: What tasks were effective? What tasks were not effective? How much time and effort did each task take? Were there tasks performed that were not in the process document? Were all of the requirements met? If not, why not? What areas can be improved upon for the future? How do the risks identified during the risk analysis compare with those that affected the project lifecycle? Where the pre-identified risks actually sources of failure? Why did the associated mitigation techniques identified in the risk analysis succeed or fail?

This protocol places the process assessment as the last stage of the development process. However, this does not mean that you should wait until the project has been completed before you address this issue. A process assessment is more effective if it is addressed continually throughout the duration of the project. If one keeps the issue of process assessment in mind throughout the project, answering the above questions as each task is completed, the result will be a comprehensive assessment that will be very useful for future projects. If it is put off until the end of the project, many situations that could be learned from will have been forgotten, developers will likely be involved in other projects and unavailable for comment, and the motivation for completing a comprehensive assessment diminishes. In sum, take some time to learn from the project's successes and failures, and the next geodatabase development project will benefit.

Chapter Summary

This protocol is the result of a combination of the software process theory, database design principals, and spatial database design theory discussed in this thesis. These theories have been woven together with the practical lessons learned from the rapid response geodatabase lifecycle as documented in this thesis.

CHAPTER 6

CONCLUSION

In the beginning of 2003, the National Center for Landscape Fire Analysis (NCLFA), through Colin Hardy of the USFS Fire Sciences Laboratory, became involved in a geodatabase development project sponsored by the Joint Fire Sciences Program (JFSP). The rapid response geodatabase development project was the result of a request made by the JFSP board of governors for several researchers to investigate a common database architecture. The purpose of this project was to create a multi-user enterprise geodatabase in which multiple rapid response research teams could store, retrieve, and analyze spatial and non-spatial data. The purpose of this thesis was to document the lifecycle of the rapid response geodatabase from conception through implementation. The lifecycle was presented in Chapters 3 and 4 of this thesis. The result of the lifecycle is a protocol for geodatabase development for Federal fire sciences research, which was presented in Chapter 5.

In order to capture the lifecycle of the rapid response geodatabase development project, the author participated in, and documented, all of the tasks in the development process. The development process began with a hardware and software assessment. The process continued through the pre-design, design, development, and implementation phases, and ended with a process assessment. The result of the rapid response geodatabase development project was a fully functional enterprise geodatabase based on ESRI technology (ArcSDE and ArcGIS) and a document detailing the project lifecycle (this thesis).

The project lifecycle was then supplemented and integrated with the development processes presented by software process theory, database design theory, spatial database design theory, and federal geospatial standards. The result of this integration was the protocol for geodatabase development presented in this thesis. The protocol guides the reader step-by-step through the process of designing and developing an enterprise geodatabase beginning with the conceptualization of the project and ending with the process assessment. The protocol is directed toward federal researchers, but can be adapted for use by a wide range of interested parties. In creating the protocol for geodatabase development, the intent of the author was to further the use of geospatial database technology in the federal research arena by providing an accessible "how-to" for geodatabase development.

The use of state-of-science geospatial database technology can enhance the efforts of Federal fire sciences research by allowing researchers to share resources and data and ultimately "connect the dots of the big picture." The JFSP funds numerous research projects each year; however, each of the projects that they fund are islands of research, data, and results. From the perspective of the JFSP the big picture often looks like just a bunch of dots.

Zooming in to one of those dots—an individual research project—the perspective of the researcher can be seen. Typically, the researchers only focus on their own projects. They are very aware of the other research being done in their field, but the mindset is to focus on the project at hand and protect their data. The researchers involved in the rapid response geodatabase project are prime examples of this. Each research team had been requested by the JFSP to exploit the linkages between their projects by investigating and contributing to a shared database. However, prior to the data committee workshop, no contact between the research teams had been made. During the data committee workshop, the researchers began to understand that through the use of a geodatabase, they could access each other's data to broaden their research, develop new research questions, and have a larger impact on the field of fire sciences as a whole.

Currently, the field of Federal fire research is lagging in its use of state-of-science geospatial database technology. Researchers may be implementing geodatabase technologies individually, but there is no overarching database framework unifying the individual researchers. Theoretically, through the use of a shared geodatabase both the researchers and the JFSP can "connect the dots and see the bigger picture." But in reality, the question becomes: How do we get there from here?

There is a relatively steep learning curve when the geodatabase data model is introduced to both GIS and non-GIS professionals. For the non-GIS professional, the difficulty lies in the use of the data stored within the geodatabase. However, this same learning curve would exist regardless of the geospatial data model being presented and can be mitigated through training with the appropriate software package. For the GIS professional, the difficulty is not in the use of the data stored within a geodatabase, but in the design of the geodatabase itself. GIS professionals are trained in the manipulation of spatial data within a GIS. Most GIS professionals are not trained in database design. With the emergence of the geodatabase data model, GIS professionals must come to realize that it is critical that relational database design principals be used in combination with the principals of the geodatabase data model when developing a geodatabase.

A geodatabase created without due attention to database design principals will serve its immediate functions, but it will not easily evolve with the needs of its users. The geodatabase will likely contain redundant data and data integrity may be lost to update and deletion errors. If good relational database design principals are applied during the design phase, the geodatabase will be functional, scalable, interoperable, and will easily evolve with the needs of its users.

As the importance of relational database design principals in geodatabase design becomes recognized, project leaders are turning to database design specialists to design their geodatabase structures. However, just as GIS professionals are not often trained in database design, database designers are not often trained in GIS. The designer must understand that a relational database created without consideration of the geodatabase data model will not support spatial data.

Thus, it becomes apparent that an effective approach to answering the question of how to achieve a shared database framework is to write step-by-step instructions to guide the developer through the entire geodatabase development lifecycle. The protocol for geodatabase development that is presented in this thesis was created for this purpose.

The protocol incorporates the principals of software engineering, relational database design, and ESRI's geodatabase data model. The use of the principals of software process from the field of software engineering provides needed structure to the geodatabase development lifecycle. By creating and following a defined development process, project managers can ensure that the resulting product meets the users' requirements. The use of relational database design principals in combination with the geodatabase data model provides the structure of the geodatabase itself. The geodatabase data model ensures the proper storage of spatial data, while the application of relational database design ensures the integrity of the data being stored. The merging of these principals makes this protocol for geodatabase design a very robust development tool.

The protocol for geodatabase development will benefit from future research. Each subsequent use of the protocol should result in modifications and improvements to the protocol as a result of the recommended process assessment task. Future research in this area could also include the application of the spiral model for software process rather than the waterfall model that was employed in this thesis.

APPENDIX A

GEOSPATIAL STANDARDS

This appendix contains a listing of existing geospatial standards. Additional Open GIS Consortium (OGC) standards are too numerous to be listed here, they can be viewed on the Web at: *http://www.opengis.org*.

FGDC Geospatial Standards[176]

Name	Number	Status	Publicly Available?
Content Standard for Digital Geospatial Metadata	FGDC-STD-001-1998	Final	Yes
Content Standard for Digital Geospatial Metadata, Part 1: Biological Data Profile	FGDC-STD-001.1-1999	Final	Yes
Metadata Profile for Shoreline Data	FGDC-STD-001.2-2001	Final	Yes
Spatial Data Transfer Standard (SDTS)	FGDC-STD-002	Final	Yes
SDTS Raster Profile and Extensions	FGDC-STD-002.5-1999	Final	Yes
SDTS Point Profile	FGDC-STD-002.6-1998	Final	Yes
SDTS Computer Aided Drafting and Design Profile	FGDC-STD-005.7-2000	Final	Yes
SDTS, Part 5: Raster Profile and Extensions	FGDC-STD-002.5	Final	Yes
SDTS, Part 6: Point Profile	FGDC-STD-002.6	Final	Yes
SDTS, Part 7: Computer-Aided Design and Drafting Profile	FGDC-STD-002.7-2000	Final	Yes
Cadastral Data Content Standard	FGDC-STD-003	Final	Yes

[176] *http://www.fgdc.gov/standards/status/textstatus.html*

FGDC Geospatial Standards (Continued)

Name	Number	Status	Publicly Available?
Classification of Wetlands and Deepwater Habitats of the United States	FGDC-STD-004	Final	Yes
Vegetation Classification Standard	FGDC-STD-005	Final	Yes
Soil Geographic Standard	FGDC-STD-006	Final	Yes
Geospatial Positioning Accuracy Standard, Part 1: Reporting Methodology	FGDC-STD-007.1-1998	Final	Yes
Geospatial Positioning Accuracy Standard, Part 2: Geodetic Control Networks	FGDC-STD-007.2-1998	Final	Yes
Geospatial Positioning Accuracy Standard, Part 3: National Standard for Spatial Data Accuracy	FGDC-STD-007.3-1998	Final	Yes
Geospatial Positioning Accuracy Standard, Part 4: Architecture, Engineering Construction, and Facilities Management	FGDC-STD-007.4-1998	Final	Yes
Content Standard for Digital Orthoimagery	FGDC-STD-008-1999	Final	Yes
Content Standard for Remote Sensing Swath Data	FGDC-STD-009-1999	Final	Yes
Utilities Data Content Standard	FGDC-STD-010-2000	Final	Yes
U.S. National Grid	FGDC-STD-011-2001	Final	Yes
Content Standard for Digital Geospatial Metadata: Extensions for Remote Sensing Metadata	FGDC-STD-012-2002	Final	Yes
Content Standard for Framework Land Elevation Data		Review	Yes
Digital Cartographic Standard for Geologic Map Symbolization		Review	Yes
Facility ID Data Standard		Review	Yes
Geospatial Positioning Accuracy Standard, Part 5: Standard for Hydrographic Surveys and Nautical Charts		Review	Yes
Hydrographic Data Content Standard for Coastal and Inland Waterways		Review	Yes
NSDI Framework Transportation Identification Standard		Review	Yes
Address Content Standard		Review	Yes

FGDC Geospatial Standards (Continued)

Name	Number	Status	Publicly Available?
Earth Cover Classification System		Draft	No
Encoding Standard for Geospatial Metadata		Draft	No
Geologic Data Model		Draft	No
Governmental Unit Boundary Data Content Standard		Draft	No
Biological Nomenclature and Taxonomy Data Standard		Draft	No
Federal Standards for Delineation of Hydrologic Unit Boundaries		Proposed	No
National Hydrography Framework Geospatial Data Content Standard		Proposed	No
National Standards for the Floristic Levels of Vegetation Classification in the United States: Associations and Alliances		Proposed	No
Revisions to the National Standards for the Physiognomic Levels of Vegetation Classification Standards, FGDC-STD-005-1997		Proposed	No
Riparian Mapping Standard		Proposed	No

ANSI / ISO Geospatial Standards[177]

Name	Number	Status	Publicly Available?
Representation of Geographic Point Locations for Information Interchange	ANSI INCITIS 61-1986 (R2002)	Final	Yes
SDTS Base Specifications	ANSI NCITS 320-1998	Draft	Yes
SDTS Topological Vector Profile	ANSI NCITS 320-1998	Final	Yes
Standard representation of latitude, longitude and altitude for geographic point locations	ISO 6709: 1983	Final	Yes

[177] *http://webstore.ansi.org/ansidocstore/default.asp* Search on "geographic"

ANSI / ISO Geospatial Standards (Continued)

Name	Number	Status	Publicly Available?
Reference model	ISO 19101: 2002	Final	Yes
Conformance and testing	ISO 19105: 2000	Final	Yes
Spatial Schema	ISO 19107: 2003	Final	Yes
Temporal schema	ISO 19108: 2002	Final	Yes
Spatial referencing by coordinates	ISO 19111: 2003	Final	Yes
Spatial referencing by geographic identifiers	ISO 19112: 2003	Final	Yes
Quality principals	ISO 19113: 2002	Final	Yes
Quality evaluation procedures	ISO 19114: 2003	Final	Yes
Metadata	ISO 19115: 2003	Final	Yes
Functional Standards	ISO/TR 19120: 2001	Final	Yes
Imagery and gridded data	ISO/TR 19121: 2000	Final	Yes
Access to Simple Features: Common Architecture	ISO 19125-1	Draft	No
Geography Markup Language (GML)	ISO/TC 211/WG 4/PT 19136	Draft	Yes

APPENDIX B

SURVEY

Geodatabase Use in Fire Sciences Research
Rapid Response Geodatabase Survey

Please complete this survey and return it to Lee Macholz at the National Center for Landscape Fire Analysis by Monday, February 9th. You may email the completed survey to macholz@ntsg.umt.edu or send it via regular mail to: Lee Macholz, University of Montana, NCLFA, SC442, Missoula, MT 59812. If you have any questions, please contact Lee Macholz (406-243-6777) or Don Helmbrecht (406-243-6244, donh@ntsg.umt.edu).

Please provide detailed answers and use additional pages where needed. A list of data objects identified from the data dictionaries you provided has been included. Please use this list to answer questions and modify the list where necessary. Thank you for your help!

- What are your expectations of the rapid response geodatabase project?

- What functionality would you require from the geodatabase?

- What are the spatial elements of your data? Do you have points, lines, polygons, or plain coordinates associated with your data? What do these spatial features represent? Do you have raster data?

- What are the data types for each of your data objects? (Fill answers in on the attached list of data objects identified from the data dictionaries you provided.)

- What type of analysis will you be doing with your data?

- Will you be using other researchers' data in your analysis? If yes, who's and how?

- Will there be a need to edit your data?

- Are there any secondary products that you will derive and want imported to the database?

- What software packages will you be using for analysis? What formats do these packages require?

APPENDIX C

RAPID RESPONSE FEATURE CLASSES AND TABLES

Automated Environmental Sensor Research Area

Feature Class: AES_InstPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Point
FireID	String		Fire ID
InstrumentID	String		Instrument ID
Latitude	Double	Observed	Instrument location (decimal degrees)
Longitude	Double	Observed	Instrument location (decimal degrees)
Altitude	Double	Observed	Instrument altitude (m)
Datum	Text	Observed	Datum of Lat/Long coordinates
DateVerified	Date		Last date verified by PI

Table: AES_Data

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
InstrumentID	String		Instrument ID
TimeStmp	Double	Observed	Z Date
DateStmp	Long Integer	Observed	Z Time
AirTemp	Float	Observed	Air temperature (°C)
WindSpeed	Float	Observed	Wind speed (m/s)
WindDir	Float	Observed	Wind direction (degrees, True N)
Humidity	Float	Observed	Relative humidity
ThermalFlux	Long Integer	Observed	Raw thermal flux
FluxTemp	Float	Observed	Thermal flux sensor temp (°C)
DervThermalFlux	Float	Derived	Calibrated thermal flux (kW/m^2)

Fire Behavior Package Research Area

Feature Class: FBP_CameraPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Point
FireID	String		Fire ID
FBPCameraID	String		Camera ID
Latitude	Double	Observed	Camera location (decimal degrees)
Longitude	Double	Observed	Camera location (decimal degrees)
TimeStmp	Text	Observed	Time stamp of observation

Feature Class: FBP_InstPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Point
FireID	String		Fire ID
PackageID	String		Package ID
Latitude	Double	Observed	Instrument location (decimal degrees)
Longitude	Double	Observed	Instrument location (decimal degrees)
Datum	Text	Observed	Datum of Lat/Long coordinates
CalTotHeatFlux	Double	Observed	Calibration factor for total heat flux sensor (kW/m^2/mV)
CalRadHeatFlux	Double	Observed	Calibration factor for radiant heat flux sensor (kW/m^2/mV)
CalVertAirVel	Double	Observed	Calibration factor for vertical air velocity sensor (m/s/mV)
CalHorizAirVel	Double	Observed	Calibration factor for horizontal air velocity sensor (m/s/mV)
CalFlameEmissive	Double	Observed	Calibration factor for flame emissivity sensor (kW/m^2/mV)
InstOrientation	Text	Observed	Instrument orientation

Table: FBP_Data_Raw

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PackageID	Text		Package ID
JulianDay	Long Integer	Observed	Date of observation (Julian date)
HrMin	Long Integer	Observed	Time (hour and minute) of observation
Seconds	Float	Observed	Time (second) of observation
AirTemp	Float	Observed	Air temperature (°C)
TotHeatFlux	Double	Observed	Raw total heat flux
RadHeatFlux	Double	Observed	Raw radiant heat flux
VertAirVel	Double	Observed	Raw vertical air velocity
HorizAirVel	Double	Observed	Raw horizontal air velocity
FlameEmissive	Double	Observed	Raw flame emissivity

Table: FBP_Data_Calibrated

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PackageID	String		Package ID
TimeStmp	String	Derived	Date/time stamp
AirTemp	Float	Derived	Air Temperature (°C)
TotHeatFlux	Double	Derived	Calibrated total heat flux (kW/m^2)
RadHeatFlux	Double	Derived	Calibrated radiant heat flux (kW/m^2)
VertAirVel	Double	Derived	Calibrated vertical air velocity (m/s)
HorizAirVel	Double	Derived	Calibrated horizontal air velocity (m/s)
FlameEmissive	Double	Derived	Calibrated flame emissivity (kW/m^2)

Table: FBP_FireBehav_Derv

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
DateStmp	Date	Derived	Date/time stamp
FlameHt	Float	Derived	Flame height (m)
FlameDepth	Float	Derived	Flame depth (m)
FlameAngle	Float	Derived	Flame angle
RateSpread	Double	Derived	Rate of spread (m/s)

Fuel Loading Research Area

Feature Class: Fuel_PlotPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Point
FireID	String		Fire ID
PlotID	String		Plot ID
Latitude	Double	Observed	Plot center location (decimal degrees)
Longitude	Double	Observed	Plot center location (decimal degree)
Datum	String	Observed	Datum
Elev	Long Integer	Observed	Elevation of plot center
UTMZone	String	Observed	UTM zone
Northing	Double	Observed	Plot center location (UTM)
Easting	Double	Observed	Plot center location (UTM)
Aspect	Long Integer	Observed	Aspect of plot
SlopePct	Long Integer	Observed	Percent slope of plot
Units	String / dFuel_Units	Observed	Unit system for observations
HabitatCvrType	String	Observed	Habitat cover type in plot
Comments	String	Observed	Examiner comments
FireTime	String	Observed	Time of fire
AirTemp	Long Integer	Observed	Air temperature during fire
RH	Long Integer	Observed	Relative humidity during fire
FlameLength	Double	Observed	Flame lengths
FireType	String / dFuel_FireType	Observed	Fire type
ROS	Long Integer	Derived	Rate of spread
WindDir	Long Integer	Observed	Wind direction (0-360)
WindSpeed	Long Integer	Observed	Wind speed (mph)
Moist_1Hr	Long Integer	Derived	Pre-fire 1-hour fuel moisture
Moist_10Hr	Long Integer	Derived	Pre-fire 10-hour fuel moisture
Moist_100Hr	Long Integer	Derived	Pre-fire 100-hour fuel moisture
Moist_1000Hr	Long Integer	Derived	Pre-fire 1000-hour fuel moisture
DuffMoist	Long Integer	Derived	Pre-fire duff fuel moisture
TreePlotSz	Double	Observed	Size of tree plot
SapPlotSz	Double	Observed	Size of sapling plot
SeedPlotSz	Double	Observed	Size of seedling plot
BreakPntDia	Double	Observed	Break point diameter
TranLen_1Hr	Long Integer	Observed	Transect length for 1-hr fuels
TranLen_10Hr	Long Integer	Observed	Transect length for 10-hr fuels

Feature Class: Fuel_PlotPT (Continued)

Field Name	Data Type / Domain	Data Level	Description
TranLen_100Hr	Long Integer	Observed	Transect length for 100-hr fuels
TranLen_1000Hr	Long Integer	Observed	Transect length for 1000-hr fuels
NumTran	Long Integer	Observed	Number of transects

Table: Fuel_PlotInfo

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Examiner	String	Observed	Name of examiner
NPhotoID	String	Observed	N photo name
EPhotoID	String	Observed	E photo name
SPhotoID	String	Observed	S photo name
WPhotoID	String	Observed	W photo name
Comments	String	Observed	Examiner comments

Table: Fuel_MoistData

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleDate	Date	Observed	Sample date
SampleNum	String	Observed	Sample number
SizeClass	String / dFuel_TimeLag	Observed	Size class
MoistTareWt	Double	Observed	Moist tare weight
DryTareWt	Double	Observed	Dry tare weight
BottleWt	Double	Observed	Bottle weight
MoistSample	Double	Observed	Moist sample
DrySample	Double	Observed	Dry sample
PctMoist	Float	Derived	Percent moisture

Table: Fuel_MoistStats

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SizeClass	String / dFuel_TimeLag	Observed	Size class
Mean	Float	Derived	Mean fuel moisture for size class
StdError	Float	Derived	Standard error of fuel moisture for size class
Median	Float	Derived	Median fuel moisture for size class
Mode	Float	Derived	Mode fuel moisture for size class
StdDev	Float	Derived	Standard deviation of fuel moisture for size class
SampleVar	Float	Derived	Sample variance of fuel moisture for size class
Kurtosis	Float	Derived	Kurtosis of fuel moisture for size class
Skewness	Float	Derived	Skewness of fuel moisture for size class
Range	Float	Derived	Range of fuel moistures for size class
Minimum	Float	Derived	Minimum fuel moisture for size class
Maximum	Float	Derived	Maximum fuel moisture for size class
SumMoist	Float	Derived	Sum fuel moisture for size class
CountMoist	Float	Derived	Count of fuel moisture samples for size class
CI95Pct	Float	Derived	95% confidence interval of fuel moisture for size class

Table: Fuel_FuelsMacro

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Examiner	String	Observed	Examiner
FuelLd_1Hr	Float	Derived	1-hr fuel load
FuelLd_10Hr	Float	Derived	10-hr fuel load
FuelLd_100Hr	Float	Derived	100-hr fuel load
FuelLdSnd_1000Hr	Float	Derived	1000-hr sound fuel load
FuelLdRot_1000Hr	Float	Derived	1000-hr rotten fuel load

Table: Fuel_FuelsMacro (Continued)

Field Name	Data Type / Domain	Data Level	Description
TotWdyFuelLd	Float	Derived	Total woody fuel load
Duff	Float	Derived	Duff fuel load
Litter	Float	Derived	Litter fuel load
TotFuelLd	Float	Derived	Total fuel load
DuffLitterConsump	Float	Derived	Duff/Litter consumption
CntMoist_1Hr	Long Integer	Observed	1-hr fuel moisture count
CntMoist_10Hr	Long Integer	Observed	10-hr fuel moisture count
CntMoist_100Hr	Long Integer	Observed	100-hr fuel moisture count
CntMoist_1000Hr	Long Integer	Observed	1000-hr fuel moisture count
CntLitterMoist	Long Integer	Observed	Litter moisture count
CntDuffMoist	Long Integer	Observed	Duff moisture count

Table: Fuel_Fuels1000Hr

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Transect	Long Integer	Observed	Transect observed
NumLog	Long Integer	Observed	Number of log
Dia	Float	Observed	Diameter of log
DecayCl	String / dFuel_DecayCl	Observed	Decay class of log
Dist	Float	Observed	Distance of log to plot center

Table: Fuel_FuelsFineDL

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Transect	Long Integer	Observed	Transect observed
SlopePctSeg1	Long Integer	Observed	Percent slope of segment 1
SlopePctSeg2	Long Integer	Observed	Percent slope of segment 2
Hr1	Long Integer	Observed	1-hr fuels
Hr10	Long Integer	Observed	10-hr fuels

Table: Fuel_FuelsFineDL (Continued)

Field Name	Data Type / Domain	Data Level	Description
Hr100	Long Integer	Observed	100-hr fuels
AvgTreeShrubHt1	Float	Observed	Average height of trees/shrubs on segment 1
DuffLitterDepth1	Float	Observed	Depth of duff/litter on segment 1
LitterPct1	Long Integer	Observed	Percent litter on segment 1
AvgTreeShrubHt2	Float	Observed	Average height of trees/shrubs on segment 2
DuffLitterDepth2	Float	Observed	Depth of duff/litter on segment 2
LitterPct2	Long Integer	Observed	Percent litter on segment 2
Pin1	Float	Observed	Pin 1 measurement
Pin2	Float	Observed	Pin 2 measurement
Pin3	Float	Observed	Pin 3 measurement
Pin4	Float	Observed	Pin 4 measurement
PinDist1	Float	Observed	Distance of pin 1 to plot center
PinDist2	Float	Observed	Distance of pin 2 to plot center
PinDist3	Float	Observed	Distance of pin 3 to plot center
PinDist4	Float	Observed	Distance of pin 4 to plot center
DuffRemain1	Float	Observed	Duff remaining at pin 1
DuffRemain2	Float	Observed	Duff remaining at pin 2
DuffRemain3	Float	Observed	Duff remaining at pin 3
DuffRemain4	Float	Observed	Duff remaining at pin 4

Table: Fuel_SCComp

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
ItemCode	String	Observed	Item observed
Status	String	Observed	Status of item observed
PctCover	Float	Observed	Percent cover
PctDead	Long Integer	Observed	Percent dead
Ht	Float	Observed	Height

Table: Fuel_TreeMacro

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Examiner	String	Observed	Examiner
TreePerAcre	Long Integer	Derived	Trees per acre
TreeBasalArea	Long Integer	Derived	Tree basal area
TreeAvgLiCrBHt	Long Integer	Derived	Average live crown base height
TreeAvgHt	Long Integer	Derived	Average height of trees
TreeQMnDia	Float	Derived	Quartile mean diameter
SapPerAcre	Long Integer	Derived	Saplings per acre
SeedPerAcre	Long Integer	Derived	Seedlings per acre
TotPerAcre	Long Integer	Derived	Total trees, saplings, seedlings per acre
SnagsPerAcre	Long Integer	Derived	Snags per acre
AvgCrVolSc	Long Integer	Derived	Average crown volume scorched

Table: Fuel_TreeMature

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
TagNo	Long Integer	Observed	Tree tag number
Species	String	Observed	Tree species
DBH	Float	Observed	Breast height diameter
CrBsHt	Long Integer	Observed	Crown base height
Ht	Long Integer	Observed	Tree height
Mort	String / dFuel_Mort	Observed	Tree mortality
Damage	Long Integer	Observed	Damage
ScortchHt	Long Integer	Observed	Scorch height
CrScPct	Long Integer	Observed	Percent crown scorched
CrBkPct	Long Integer	Observed	Percent crown black

Table: Fuel_TreeSap

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Species	String	Observed	Sapling species
SizeClDia	Float / dFuel_SapSxCl	Observed	Size class diameter
SapCount	Long Integer	Observed	Count of saplings (per species, per size class)
AvgHt	Float	Observed	Average height (per species, per size class)
AvgLiCr	Long Integer	Observed	Average live crown (per species, per size class)
TreeStat	String / dFuel_TreeStatus	Observed	Tree status

Table: Fuel_TreeSeed

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
PlotID	String		Plot ID
SampleEvent	String	Observed	Sample event (pre- or post-fire)
SampleDate	Date	Observed	Sample date
Species	String	Observed	Seedling species
SizeClHt	Float / dFuel_SeedSzCl	Observed	Seedling height size class
TreeStat	String / dFuel_TreeStatus	Observed	Tree status
SeedCount	Long Integer	Observed	Seedling count
PctLiGrassCvr	Long Integer	Observed	Percent cover of live grass
PctDdGrassCvr	Long Integer	Observed	Percent cover of dead grass
PctLiForbCvr	Long Integer	Observed	Percent cover of live forbs
PctDdForbCvr	Long Integer	Observed	Percent cover of dead forbs
PctLiLowShrub	Long Integer	Observed	Percent cover of live low shrubs
PctDdLowShrub	Long Integer	Observed	Percent cover of dead low shrubs
PctLiHighShrub	Long Integer	Observed	Percent cover of live high shrubs
PctDdHighShrub	Long Integer	Observed	Percent cover of dead high shrubs
PctLiSeedCvr	Long Integer	Observed	Percent cover of live seedlings

Burn Severity Research Area

Feature Class: Sev_SubPlotPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometery		Point
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SubPlotID	String		Subplot ID
Latitude	Double	Derived	Subplot center location (decimal degrees)
Longitude	Double	Derived	Subplot center location (decimal degrees)
Easting	Double	Derived	Subplot center location (UTM)
Northing	Double	Derived	Subplot center location (UTM)

Table: Sev_SoilSubPlot

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SubPlotID	String		Subplot ID
SampleDate	Date	Observed	Sample date
SubPlotPhotoID	String	Observed	Subplot photo name
PctGrNew	Long Integer	Observed	Percent new green
TypeGrNew	String	Observed	Type new green
PctGrOld	Long Integer	Observed	Percent old green
TypeGrOld	String	Observed	Type old green
PctCvrNew	Long Integer	Observed	Percent cover new
PctNewChar	Long Integer	Observed	Percent new charred
PctMisc	Long Integer	Observed	Percent miscellaneous
PctMiscChar	Long Integer	Observed	Percent miscellaneous charred
TypeMisc	String	Observed	Type miscellaneous
PctCvrOld	Long Integer	Observed	Percent cover old
PctOldChar	Long Integer	Observed	Percent old charred
PctAsh	Long Integer	Observed	Percent ash
PctRock	Long Integer	Observed	Percent rock
PctRockChar	Long Integer	Observed	Percent rock charred
PctSoil	Long Integer	Observed	Percent soil

Table: Sev_SoilSubPlot (Continued)

Field Name	Data Type / Domain	Data Level	Description
PctSoilChar	Long Integer	Observed	Percent soil charred
PctCharD	Long Integer	Observed	Percent deep char
PctCharM	Long Integer	Observed	Percent moderate char
PctCharL	Long Integer	Observed	Percent light char
PctCharU	Long Integer	Observed	Percent Unchar
PctOrgChar	Float	Derived	Percent organic charred
PctOrgUnChar	Float	Derived	Percent organic uncharred
PctInOrgChar	Float	Derived	Percent inorganic charred
PctInOrgUnChar	Float	Derived	Percent inorganic uncharred
PctAllOldChar	Float	Derived	Percent all old char
PctAllOldUnChar	Float	Derived	Percent all old uncharred
NewOldGr	Float	Derived	New and old green
NewOldChar	Float	Derived	New and old char
NewOldUnChar	Float	Derived	New and old uncharred

Table: Sev_WaterSubPlot

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SubPlotID	String		Subplot ID
SampleDate	Date	Observed	Sample Date
WDPTLowStartT	Long Integer	Observed	WDPT low char start time
WDPTLowEndT	Long Integer	Observed	WDPT low char end time
WDPTLowTime	Float	Derived	WDPT low char time
WDPTModStartT	Long Integer	Observed	WDPT moderate char start time
WDPTModEndT	Long Integer	Observed	WDPT moderate char end time
WDPTModTime	Float	Derived	WDPT moderate char time
WDPTDeepStartT	Long Integer	Observed	WDPT deep char start time
WDPTDeepEndT	Long Integer	Observed	WDPT deep char end time
WDPTDeepTime	Float	Derived	WDPT deep char time
WDPTUnStartT	Long Integer	Observed	WDPT unchar start time
WDPTUnEndT	Long Integer	Observed	WDPT unchar end time
WDPTUnTime	Float	Derived	WDPT unchar time
InfiltLowStartT	Long Integer	Observed	Infiltrometer low char start time
InfiltLowEndT	Long Integer	Observed	Infiltrometer low char end time
InfiltLowBubbleT	Long Integer	Observed	Infiltrometer low char bubble time
InfiltLowTime	Float	Derived	Infiltrometer low char time

Table: Sev_WaterSubPlot (Continued)

Field Name	Data Type / Domain	Data Level	Description
InfiltModStartT	Long Integer	Observed	Infiltrometer moderate char start time
InfiltModEndT	Long Integer	Observed	Infiltrometer moderate char end time
InfiltModBubbleT	Long Integer	Observed	Infiltrometer moderate char bubble time
InfiltModTime	Float	Derived	Infiltrometer moderate char time
InfiltDeepStartT	Long Integer	Observed	Infiltrometer deep char start time
InfiltDeepEndT	Long Integer	Observed	Infiltrometer deep char end time
InfiltDeepBubbleT	Long Integer	Observed	Infiltrometer deep char bubble time
InfiltDeepTime	Float	Derived	Infiltrometer deep char time
InfiltUnStartT	Long Integer	Observed	Infiltrometer unchar start time
InfiltUnEndT	Long Integer	Observed	Infiltrometer unchar end time
InfiltUnBubbleT	Long Integer	Observed	Infiltrometer unchar bubble time
InfiltUnTime	Float	Derived	Infiltrometer unchar time
InfiltLowStartV	Long Integer	Observed	Infiltrometer low char start volume
InfiltLowEndV	Long Integer	Observed	Infiltrometer low char end volume
InfiltLowBubbleV	Long Integer	Observed	Infiltrometer low char bubble volume
InfiltLowRate	Float	Derived	Infiltrometer low char rate
InfiltModStartV	Long Integer	Observed	Infiltrometer moderate char start volume
InfiltModEndV	Long Integer	Observed	Infiltrometer moderate char end volume
InfiltModBubbleV	Long Integer	Observed	Infiltrometer moderate char bubble volume
InfiltModRate	Float	Derived	Infiltrometer moderate char rate
InfiltDeepStartV	Long Integer	Observed	Infiltrometer deep char start volume
InfiltDeepEndV	Long Integer	Observed	Infiltrometer deep char end volume
InfiltDeepBubbleV	Long Integer	Observed	Infiltrometer deep char bubble volume
InfiltDeepRate	Float	Derived	Infiltrometer deep char rate
InfiltUnStartV	Long Integer	Observed	Infiltrometer unchar start volume
InfiltUnEndV	Long Integer	Observed	Infiltrometer unchar end volume
InfiltUnBubbleV	Long Integer	Observed	Infiltrometer unchar bubble volume
InfiltUnRate	Float	Derived	Infiltrometer unchar rate

Feature Class: Sev_PlotPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Point
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
Latitude	Double	Observed	Plot center location (decimal degrees)
Longitude	Double	Observed	Plot center location (decimal degrees)
Easting	Double	Observed	Plot center location (UTM)
Northing	Double	Observed	Plot center location (UTM)

Table: Sev_SoilPlot

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SampleDate	Date	Observed	Sample date
PlotPhotoID	String	Observed	Plot photo name
Examiner	String	Observed	Examiner
Severity	String	Observed	Plot severity
SlopePct	Long Integer	Observed	Percent slope of plot
SlopePosition	String	Observed	Slope position of plot
VertSlopeShape	String	Observed	Vertical slope shape of plot
HorizSlopeShape	String	Observed	Horizontal slope shape of plot
Aspect	String	Observed	Plot aspect
NewLitterDepth	Long Integer	Observed	New litter depth
OldLitterDepth	Long Integer	Observed	Old litter depth
DuffDepth	Long Integer	Observed	Duff depth
DensN	Long Integer	Observed	Densiometer N
DensE	Long Integer	Observed	Densiometer E
DensS	Long Integer	Observed	Densiometer S
DensW	Long Integer	Observed	Densiometer W
CanopyN	Float	Derived	Canopy N
CanopyE	Float	Derived	Canopy E
CanpoyS	Float	Derived	Canopy S
CanopyW	Float	Derived	Canopy W
MnCanopy	Float	Derived	Mean plot canopy
SDCanopy	Float	Derived	Standard deviation of mean plot canopy
MnGrNew	Float	Derived	Subplot mean new green

Table: Sev_SoilPlot (Continued)

Field Name	Data Type / Domain	Data Level	Description
SDGrNew	Float	Derived	Standard deviation of subplot mean new green
MnGrOld	Float	Derived	Subplot mean old green
SDGrOld	Float	Derived	Standard deviation of subplot mean old green
MnCvrNew	Float	Derived	Subplot mean new cover
SDCvrNew	Float	Derived	Standard deviation of subplot mean new cover
MnNewChar	Float	Derived	Subplot mean new char
SDNewChar	Float	Derived	Standard deviation of subplot mean new char
MnMisc	Float	Derived	Subplot mean miscellaneous
SDMisc	Float	Derived	Standard deviation of subplot mean new char
MnMiscChar	Float	Derived	Subplot mean miscellaneous charred
SDMiscChar	Float	Derived	Standard deviation of subplot mean miscellaneous charred
MnCvrOld	Float	Derived	Subplot mean old cover
SDCvrOld	Float	Derived	Standard deviation of subplot mean old cover
MnOldChar	Float	Derived	Subplot mean old char
SDOldChar	Float	Derived	Standard deviation of subplot mean old char
MnAsh	Float	Derived	Subplot mean ash
SDAsh	Float	Derived	Standard deviation of subplot mean ash
MnRock	Float	Derived	Subplot mean rock
SDRock	Float	Derived	Standard deviation of subplot mean rock
MnRockChar	Float	Derived	Subplot mean rock charred
SDRockChar	Float	Derived	Standard deviation of subplot mean rock charred
MnSoil	Float	Derived	Subplot mean soil
SDSoil	Float	Derived	Standard deviation of subplot mean soil
MnSoilChar	Float	Derived	Subplot mean soil charred
SDSoilChar	Float	Derived	Standard deviation of subplot mean soil charred
MnCharD	Float	Derived	Subplot mean deep char
SDCharD	Float	Derived	Standard deviation of subplot mean deep char

Table: Sev_SoilPlot (Continued)

Field Name	Data Type / Domain	Data Level	Description
MnCharM	Float	Derived	Subplot mean moderate char
SDCharM	Float	Derived	Standard deviation of subplot mean moderate char
MnCharL	Float	Derived	Subplot mean light char
SDCharL	Float	Derived	Standard deviation of subplot mean light char
MnCharU	Float	Derived	Subplot mean unchar
SDCharU	Float	Derived	Standard deviation of subplot mean unchar
MnOrgChar	Float	Derived	Subplot mean organic charred
SDOrgChar	Float	Derived	Standard deviation of subplot mean organic charred
MnOrgUnChar	Float	Derived	Subplot mean unorganic char
SDOrgUnChar	Float	Derived	Standard deviation of subplot mean unorganic char
MnInOrgChar	Float	Derived	Subplot mean inorganic char
SDInOrgChar	Float	Derived	Standard deviation of subplot mean inorganic char
MnInOrgUnChar	Float	Derived	Subplot mean inorganic unchar
SDInOrgUnChar	Float	Derived	Standard deviation of subplot mean inorganic unchar
MnAllOldChar	Float	Derived	Subplot mean all old charred
SDAllOldChar	Float	Derived	Standard deviation of subplot mean all old charred
MnAllOldUnChar	Float	Derived	Subplot mean all old uncharred
SDAllOldUnChar	Float	Derived	Standard deviation of subplot mean all old uncharred
MnNewOldChar	Float	Derived	Subplot mean new and old charred
SDNewOldChar	Float	Derived	Standard deviation of subplot mean new and old charred
MnNewOldUnChar	Float	Derived	Subplot mean new and old uncharred
SDNewOldUnChar	Float	Derived	Standard deviation of subplot mean new and old uncharred

Table: Sev_WaterPlot

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SampleDate	Date	Observed	Sample date
MnWDPTLowTime	Float	Derived	Subplot mean WDPT low char time
SDWDPTLowTime	Float	Derived	Standard deviation of subplot mean WDPT low char time
MnWDPTModTime	Float	Derived	Subplot mean WDPT moderate char time
SDWDPTModTime	Float	Derived	Standard deviation of subplot mean WDPT moderate char time
MnWDPTDeepTime	Float	Derived	Subplot mean WDPT deep char time
SDWDPTDeepTime	Float	Derived	Standard deviation of subplot mean WDPT deep char time
MnWDPTUnTime	Float	Derived	Subplot mean WDPT unchar time
SDWDPTUnTime	Float	Derived	Standard deviation of subplot mean WDPT unchar time
MnInfiltLowBubbleT	Float	Derived	Subplot mean infiltrometer low char bubble time
SDInfiltLowBubbleT	Float	Derived	Standard deviation of subplot mean infiltrometer low char bubble time
MnInfiltModBubbleT	Float	Derived	Subplot mean infiltrometer moderate char bubble time
SDInfiltModBubbleT	Float	Derived	Standard deviation of subplot mean infiltrometer moderate char bubble time
MnInfiltDeepBubbleT	Float	Derived	Subplot mean infiltrometer deep char bubble time
SDInfiltDeepBubbleT	Float	Derived	Standard deviation of subplot mean infiltrometer deep char bubble time
MnInfiltUnBubbleT	Float	Derived	Subplot mean infiltrometer unchar bubble time
SDInfiltUnBubbleT	Float	Derived	Standard deviation of subplot mean infiltrometer unchar bubble time

Table: Sev_WaterPlot (Continued)

Field Name	Data Type / Domain	Data Level	Description
MnInfiltLowRate	Float	Derived	Subplot mean infiltrometer low char rate
SDInfiltLowRate	Float	Derived	Standard deviation of subplot mean infiltrometer low char rate
MnInfiltModRate	Float	Derived	Subplot mean infiltrometer moderate char rate
SDInfiltModRate	Float	Derived	Standard deviation of subplot mean infiltrometer moderate char rate
MnInfiltDeepRate	Float	Derived	Subplot mean infiltrometer deep char rate
SDInfiltDeepRate	Float	Derived	Standard deviation of subplot mean infiltrometer deep char rate
MnInfiltUnRate	Float	Derived	Subplot mean infiltrometer unchar rate
SDInfiltUnRate	Float	Derived	Standard deviation of subplot mean infiltrometer unchar rate

Table: Sev_Veg750

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SampleDate	Date	Observed	Sample date
PctGrassGr	Long Integer	Observed	Percent cover grass green
PctGrassBr	Long Integer	Observed	Percent cover grass brown
PctGrassBl	Long Integer	Observed	Percent cover grass black
GrassPrimarySp	String	Observed	Grass primary species
GrassSecondarySp	String	Observed	Grass secondary species
PctForbGr	Long Integer	Observed	Percent cover forb green
PctForbBr	Long Integer	Observed	Percent cover forb brown
PctForbBl	Long Integer	Observed	Percent cover forb black
ForbPrimarySp	String	Observed	Forb primary species
ForbSecondarySp	String	Observed	Forb secondary species
PctLowShrubGr	Long Integer	Observed	Percent cover low shrub green
PctLowShrubBr	Long Integer	Observed	Percent cover low shrub brown
PctLowShrubBl	Long Integer	Observed	Percent cover low shrub black
LowShrubPrimarySp	String	Observed	Low shrub primary species

Table: Sev_Veg750 (Continued)

Field Name	Data Type / Domain	Data Level	Description
LowShrubSecondarySp	String	Observed	Low shrub secondary species
CntSeedLive	Long Integer	Observed	Count live seedlings
PctCvrSeedLive	Long Integer	Observed	Percent cover live seedlings
CntSeedDead	Long Integer	Observed	Count dead seedlings
PctCvrSeedDead	Long Integer	Observed	Percent cover dead seedlings
PctSeedDead	Long Integer	Observed	Percent seedlings dead
SeedPrimarySp	String	Observed	Seedlings primary species
SeedSecondarySp	String	Observed	Seedlings secondary species

Table: Sev_Veg100

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID
SampleDate	Date	Observed	Sample date
CntHighShrubLive	Long Integer	Observed	Count high shrub live
MnHtHighShrubLive	Long Integer	Observed	Mean height high shrub live
CntHighShrubDead	Long Integer	Observed	Count high shrub dead
MnHtHighShrubDead	Long Integer	Observed	Mean height high shrub dead
PctHighShrubDead	Long Integer	Observed	Percent high shrub dead
HighShrubPrimarySp	String	Observed	High shrub primary species
HighShrubSecondarySp	String	Observed	High shrub secondary species
CntSapLive	Long Integer	Observed	Count saplings live
MnHtSapLive	Long Integer	Observed	Mean height saplings live
CntSapDead	Long Integer	Observed	Count saplings dead
MnHtSapDead	Long Integer	Observed	Mean height saplings dead
PctSapDead	Long Integer	Observed	Percent saplings dead
SapPrimarySp	String	Observed	Saplings primary species
SapSecondarySp	String	Observed	Saplings secondary species
Comments	String	Observed	Examiner comments

Table: Sev_Tree50

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID

Table: Sev_Tree50 (Continued)

Field Name	Data Type / Domain	Data Level	Description
SampleDate	Date	Observed	Sample date
CntLiveTree	Long Integer	Derived	Count of live trees
CntOldSnag	Long Integer	Derived	Count of old snags
CntNewSnag	Long Integer	Derived	Count of new snags
SumBasalArea	Double	Derived	Sum tree basal area
MnBasalArea	Double	Derived	Mean tree basal area
SDBasalArea	Double	Derived	Standard deviation of mean tree basal area
SumTreePerHa	Double	Derived	Sum of trees per hectare
MnTreePerHa	Double	Derived	Mean trees per hectare
SDTreePerHa	Double	Derived	Standard deviation of mean trees per hectare
MinHt	Double	Derived	Minimum tree height
MaxHt	Double	Derived	Maximum tree height
MnHt	Double	Derived	Mean tree height
SDHt	Double	Derived	Standard deviation of mean tree height
MinHtCrownBase	Double	Derived	Minimum crown base height
MaxHtCrownBase	Double	Derived	Maximum crown base height
MnHtCrownBase	Double	Derived	Mean crown base height
SDHtCrownBase	Double	Derived	Standard deviation of mean crown base height
PlotBAGr	Double	Derived	Plot basal area green
PlotBABr	Double	Derived	Plot basal area brown
PlotBABl	Double	Derived	Plot basal area black
PlotBATotal	Double	Derived	Plot basal area total
PctCrownGr	Double	Derived	Percent crown green
PctCrownBr	Double	Derived	Percent crown brown
PctCrownBl	Double	Derived	Percent crown black
TreePrimarySp	String	Observed	Tree primary species
TreeSecondarySp	String	Observed	Tree secondary species
TreeTertiarySp	String	Observed	Tree tertiary species
Comments	String	Observed	Examiner comments

Table: Sev_Tree50Data

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
PlotID	String		Plot ID

Table: Sev_Tree50Data (Continued)

Field Name	Data Type / Domain	Data Level	Description
SampleDate	Date	Observed	Sample date
SampleNum	String	Observed	Sample number
ObjectMeasured	String / dSev_TreeMeas	Observed	Type of object measured
DBH	Float	Observed	Breast height diameter
Ht	Float	Observed	Height
HtCrownBase	Double	Observed	Crown base height
PctCrownGr	Long Integer	Observed	Percent crown green
PctCrownBr	Long Integer	Observed	Percent crown brown
PctCrownBl	Long Integer	Observed	Percent crown black
BasalArea	Float	Observed	Basal area
TreesPerHa	Float	Observed	Trees per hectare
WtPctCrownGr	Long Integer	Derived	Weighted percent crown green
WtPctCrownBr	Long Integer	Derived	Weighted percent crown brown
WtPctCrownBl	Long Integer	Derived	Weighted percent crown black
Species	String	Observed	Species

Table: Sev_SoilSite

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
SampleDate	Date	Observed	Sample date
MnNewLitterDepth	Float	Derived	Plot mean new litter depth
SDNewLitterDepth	Float	Derived	Standard deviation of plot mean new litter depth
MnOldLitterDepth	Float	Derived	Plot mean old litter depth
SDOldLitterDepth	Float	Derived	Standard deviation of plot mean old litter depth
MnDuffDepth	Float	Derived	Plot mean duff depth
SDDuffDepth	Float	Derived	Standard deviation of plot mean duff depth
MnCanopyN	Float	Derived	Plot mean canopy N
SDCanopyN	Float	Derived	Standard deviation of plot mean canopy N
MnCanopyE	Float	Derived	Plot mean canopy E
SDCanopyE	Float	Derived	Standard deviation of plot mean canopy E
MnCanopyS	Float	Derived	Plot mean canopy S

Table: Sev_SoilSite (Continued)

Field Name	Data Type / Domain	Data Level	Description
SDCanopyS	Float	Derived	Standard deviation of plot mean canopy S
MnCanopyW	Float	Derived	Plot mean canopy W
SDCanopyW	Float	Derived	Standard deviation plot mean canopy W
MnCanopy	Float	Derived	Mean plot canopy
SDCanopy	Float	Derived	Standard deviation of mean plot canopy
MnGrNew	Float	Derived	Plot mean new green
SDGrNew	Float	Derived	Standard deviation of plot mean new green
MnGrOld	Float	Derived	Plot mean old green
SDGrOld	Float	Derived	Standard deviation of plot mean old green
MnCvrNew	Float	Derived	Plot mean new cover
SDCvrNew	Float	Derived	Standard deviation of plot mean new cover
MnNewChar	Float	Derived	Plot mean new char
SDNewChar	Float	Derived	Standard deviation of plot mean new char
MnMisc	Float	Derived	Plot mean miscellaneous
SDMisc	Float	Derived	Standard deviation of plot mean new char
MnMiscChar	Float	Derived	Plot mean miscellaneous charred
SDMiscChar	Float	Derived	Standard deviation of plot mean miscellaneous charred
MnCvrOld	Float	Derived	Plot mean old cover
SDCvrOld	Float	Derived	Standard deviation of plot mean old cover
MnOldChar	Float	Derived	Plot mean old char
SDOldChar	Float	Derived	Standard deviation of plot mean old char
MnAsh	Float	Derived	Plot mean ash
SDAsh	Float	Derived	Standard deviation of plot mean ash
MnRock	Float	Derived	Plot mean rock
SDRock	Float	Derived	Standard deviation of plot mean rock
MnRockChar	Float	Derived	Plot mean rock charred
SDRockChar	Float	Derived	Standard deviation of plot mean rock charred
MnSoil	Float	Derived	Plot mean soil

Table: Sev_SoilSite (Continued)

Field Name	Data Type / Domain	Data Level	Description
SDSoil	Float	Derived	Standard deviation of plot mean soil
MnSoilChar	Float	Derived	Plot mean soil charred
SDSoilChar	Float	Derived	Standard deviation of plot mean soil charred
MnCharD	Float	Derived	Plot mean deep char
SDCharD	Float	Derived	Standard deviation of plot mean deep char
MnCharM	Float	Derived	Plot mean moderate char
SDCharM	Float	Derived	Standard deviation of plot mean moderate char
MnCharL	Float	Derived	Plot mean light char
SDCharL	Float	Derived	Standard deviation of plot mean light char
MnCharU	Float	Derived	Plot mean unchar
SDCharU	Float	Derived	Standard deviation of plot mean unchar
MnOrgChar	Float	Derived	Plot mean organic charred
SDOrgChar	Float	Derived	Standard deviation of plot mean organic charred
MnOrgUnChar	Float	Derived	Plot mean unorganic char
SDOrgUnChar	Float	Derived	Standard deviation of plot mean unorganic char
MnInOrgChar	Float	Derived	Plot mean inorganic char
SDInOrgChar	Float	Derived	Standard deviation of plot mean inorganic char
MnInOrgUnChar	Float	Derived	Plot mean inorganic unchar
SDInOrgUnChar	Float	Derived	Standard deviation of plot mean inorganic unchar
MnAllOldChar	Float	Derived	Plot mean all old charred
SDAllOldChar	Float	Derived	Standard deviation of plot mean all old charred
MnAllOldUnChar	Float	Derived	Plot mean all old uncharred
SDAllOldUnChar	Float	Derived	Standard deviation of plot mean all old uncharred
MnNewOldChar	Float	Derived	Plot mean new and old charred
SDNewOldChar	Float	Derived	Standard deviation of plot mean new and old charred
MnNewOldUnChar	Float	Derived	Plot mean new and old uncharred
SDNewOldUnChar	Float	Derived	Standard deviation of plot mean new and old uncharred

Table: Sev_WaterSite

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
SiteID	String		Site ID
SampleDate	Date	Observed	Sample date
MnWDPTLowTime	Float	Derived	Plot mean WDPT low char time
SDWDPTLowTime	Float	Derived	Standard deviation of plot mean WDPT low char time
MnWDPTModTime	Float	Derived	Plot mean WDPT moderate char time
SDWDPTModTime	Float	Derived	Standard deviation of plot mean WDPT moderate char time
MnWDPTDeepTime	Float	Derived	Plot mean WDPT deep char time
SDWDPTDeepTime	Float	Derived	Standard deviation of plot mean WDPT deep char time
MnWDPTUnTime	Float	Derived	Plot mean WDPT unchar time
SDWDPTUnTime	Float	Derived	Standard deviation of plot mean WDPT unchar time
MnInfiltLowBubbleT	Float	Derived	Plot mean infiltrometer low char bubble time
SDInfiltLowBubbleT	Float	Derived	Standard deviation of plot mean infiltrometer low char bubble time
MnInfiltModBubbleT	Float	Derived	Plot mean infiltrometer moderate char bubble time
SDInfiltModBubbleT	Float	Derived	Standard deviation of plot mean infiltrometer moderate char bubble time
MnInfiltDeepBubbleT	Float	Derived	Plot mean infiltrometer deep char bubble time
SDInfiltDeepBubbleT	Float	Derived	Standard deviation of plot mean infiltrometer deep char bubble time
MnInfiltUnBubbleT	Float	Derived	Plot mean infiltrometer unchar bubble time
SDInfiltUnBubbleT	Float	Derived	Standard deviation of plot mean infiltrometer unchar bubble time
MnInfiltLowRate	Float	Derived	Plot mean infiltrometer low char rate
SDInfiltLowRate	Float	Derived	Standard deviation of plot mean infiltrometer low char rate

Table: Sev_WaterSite (Continued)

Field Name	Data Type / Domain	Data Level	Description
MnInfiltModRate	Float	Derived	Plot mean infiltrometer moderate char rate
SDInfiltModRate	Float	Derived	Standard deviation of plot mean infiltrometer moderate char rate
MnInfiltDeepRate	Float	Derived	Plot mean infiltrometer deep char rate
SDInfiltDeepRate	Float	Derived	Standard deviation of plot mean infiltrometer deep char rate
MnInfiltUnRate	Float	Derived	Plot mean infiltrometer unchar rate
SDInfiltUnRate	Float	Derived	Standard deviation of plot mean infiltrometer unchar rate

Airborne Thermal Infrared Research Area

Table: ATIR_Info

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
DateStmp	Date	Observed	Date stamp
TimeStmp	Date	Observed	Time stamp
Latitude	Double	Observed	Observation location (decimal degrees)
Longitude	Double	Observed	Observation location (decimal degrees)
ImageName	String		Image file name

Ground Thermal Infrared Research Area

Feature Class: GTIR_CameraPT

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Point
FireID	String		Fire ID
CameraID	String		Camera ID
Latitude	Double	Observed	Camera location (decimal degrees)

Feature Class: GTIR_CameraPT (Continued)

Field Name	Data Type / Domain	Data Level	Description
Longitude	Double	Observed	Camera location (decimal degrees)
Elevation	Double	Observed	Camera elevation

Feature Class: GTIR_LandscpPOLY

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
Shape	Geometry		Polygon
FireID	String		Fire ID
CameraID	String		Camera ID
StillID	String		Still ID
GridName	String		Grid file name

Table: GTIR_Video

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
CameraID	String		Camera ID
TimeStmp	Date		Time stamp
VideoName	String		Video file name
VideoLoc	String		Video file location
FrameTimes	Long Integer		Frame time
Summary	String		Summary

Table: GTIR_StillData

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
CameraID	String		Camera ID
StillID	String		Still ID
TimeStmp	Date	Observed	Time stamp
AzimuthCenterPix	Long Integer	Derived	Azimuth of center pixel
ElevCenterPix	Long Integer	Derived	Elevation of center pixel
PixelFOV	String	Derived	Pixel FOV
Data	String	Derived	Data

Table: GTIR_StillPixelCenters

Field Name	Data Type / Domain	Data Level	Description
OBJECTID	ObjectID		
FireID	String		Fire ID
CameraID	String		Camera ID
StillID	String		Still ID
PixelNum	Long Integer		Pixel number
X	Double	Derived	Pixel X location
Y	Double	Derived	Pixel Y location
Elev	Double	Derived	Pixel elevation

APPENDIX D

RAPID RESPONSE CODED VALUE DOMAINS

Coded Value Domain: dFuel_DecayCl

Field Type: String	
Description	**Code**
All bark intact. All but smallest twigs present. Only needles prob still present. Hard when kicked.	1
Some bark and many smaller branches missing. No old needles on branches. Hard when kicked.	2
Most bark and branches less than 1 in. dia missing. Still hard when kicked.	3
Looks like a class 3 log but sapwood rotten. Hollow when kicked.	4
No limbs or limb stubs.	5
Not Assessed	X

Coded Value Domain: dFuel_FireType

Field Type: String	
Description	**Code**
Crown	C
Flanking	F
Backing	B
Head	H
Not Assessed	X

Coded Value Domain: dFuel_LiveCrPct

Field Type: Double	
Description	**Code**
0%	0
Trace or 0-1%	0.5
Present or 2-5%	3
6-15%	10
16-25%	20
26-35%	30

Coded Value Domain: dFuel_LiveCrPct (Continued)

Field Type: Double	
Description	**Code**
36-45%	40
46-55%	50
56-65%	60
66-75%	70
76-85%	80
86-95%	90
96-100%	98

Coded Value Domain: dFuel_Mort

Field Type: String	
Description	**Code**
Fire caused	F
Insect caused	I
Disease caused	D
Abiotic	A
Unable to determine	U
Not Assessed	X

Coded Value Domain: dFuel_Plants

Field Type: String	
Description	**Code**
Grass	GRAS
Forb	FORB
High shrub	SHHI
Low shrub	SHLO
Seedling	SEED

Coded Value Domain: dFuel_SapSzCl

Field Type: Double	
Description	**Code**
>0 - 1 in	0.5
>1 - 2 in	1.5
>2 - 3 in	2.5
>3 - 4 in	3.5
>0.0 - 2.5 cm	1.2
>2.5 - 5.0 cm	3.8
>5.0 - 7.5 cm	6.2
>7.5 - 10.0 cm	8.8

Coded Value Domain: dFuel_SeedSzCl

Field Type: Double	
Description	**Code**
>0.0 - 0.5 ft	0.2
>0.5 - 1.5 ft	1
>1.5 - 2.5 ft	2
>2.5 - 3.5 ft	3
>3.5 - 4.5 ft	4
>0.0 - 0.2 m	0.1
>0.2 - 0.5 m	0.3
>0.5 - 0.8 m	0.6
>0.8 - 1.0 m	0.9
>1.0 - 1.4 m	1.2

Coded Value Domain: dFuel_TimeLag

Field Type: String	
Description	**Code**
1-Hr	1Hr
10-Hr	10Hr
100-Hr	100Hr
1000-Hr	1000Hr
Duff	Duff
Litter	Litter

Coded Value Domain: dFuel_TreeStatus

Field Type: String	
Description	**Code**
Healthy	H
Unhealthy	U
Sick	S
Dead	D
Not Assessed	X

Coded Value Domain: dFuel_Units

Field Type: String	
Description	**Code**
English	E
Metric	M

Coded Value Domain: dSev_TreeMeas

Field Type: String	
Description	**Code**
Tree	Tree
Old Snag	Old Snag
New Snag	New Snag

APPENDIX E

RAPID RESPONSE RELATIONSHIP CLASSES

Relationship Class: rAES Inst Points to AES Data Table

Cardinality:	1:M
Origin Table / Feature Class:	AES_InstPT
Destination Table / Feature Class:	AES_Data
Primary Key:	InstrumentID
Foreign Key:	InstrumentID

Relationship Class: rFBP Inst Points to FBP Calibrated Data Table

Cardinality:	1:M
Origin Table / Feature Class:	FBP_InstPT
Destination Table / Feature Class:	FBP_Data_Calibrated
Primary Key:	PackageID
Foreign Key:	PackageID

Relationship Class: rFBP Inst Points to FBP Raw Data Table

Cardinality:	1:M
Origin Table / Feature Class:	FBP_InstPT
Destination Table / Feature Class:	FBP_Data_Raw
Primary Key:	PackageID
Foreign Key:	PackageID

Relationship Class: rFuel Plot Points to Fuel Moisture Data Table

Cardinality:	1:M
Origin Table / Feature Class:	Fuel_PlotPT
Destination Table / Feature Class:	Fuel_MoistData
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel Plot Points to Fuel Moisture Stats Table

Cardinality:	1:M
Origin Table / Feature Class:	Fuel_PlotPT
Destination Table / Feature Class:	Fuel_MoistStats
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel Plot Points to Fuel Plot Info Table

Cardinality:	1:M
Origin Table / Feature Class:	Fuel_PlotPT
Destination Table / Feature Class:	Fuel_PlotInfo
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel Plot Points to Fuel SC Comp Table

Cardinality:	1:M
Origin Table / Feature Class:	Fuel_PlotPT
Destination Table / Feature Class:	Fuel_SCComp
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel Plot Points to Fuel Tree Macro Table

Cardinality:	1:M
Origin Table / Feature Class:	Fuel_PlotPT
Destination Table / Feature Class:	Fuel_TreeMacro
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel Plot Points to Fuels Macro Table

Cardinality:	1:M
Origin Table / Feature Class:	Fuel_PlotPT
Destination Table / Feature Class:	Fuel_FuelsMacro
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel_FuelsMacroTOFuels1000Hr

Cardinality:	M:M
Origin Table / Feature Class:	Fuel_FuelsMacro
Destination Table / Feature Class:	Fuel_Fuels1000Hr
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel_FuelsMacroTOFuelsFineDL

Cardinality:	M:M
Origin Table / Feature Class:	Fuel_FuelsMacro
Destination Table / Feature Class:	Fuel_FuelsFineDL
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel_TreeMacroTOFuel_TreeMature

Cardinality:	M:M
Origin Table / Feature Class:	Fuel_TreeMacro
Destination Table / Feature Class:	Fuel_TreeMature
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel_TreeMacroTOFuel_TreeSap

Cardinality:	M:M
Origin Table / Feature Class:	Fuel_TreeMacro
Destination Table / Feature Class:	Fuel_TreeSap
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rFuel_TreeMacroTOFuel_TreeSeed

Cardinality:	M:M
Origin Table / Feature Class:	Fuel_TreeMacro
Destination Table / Feature Class:	Fuel_TreeSeed
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rGTIR Camera Points to GTIR Still Data Table

Cardinality:	1:M
Origin Table / Feature Class:	GTIR_CameraPT
Destination Table / Feature Class:	GTIR_StillData
Primary Key:	CameraID
Foreign Key:	CameraID

Relationship Class: rGTIR Camera Points to GTIR Video Table

Cardinality:	1:M
Origin Table / Feature Class:	GTIR_CameraPT
Destination Table / Feature Class:	GTIR_Video
Primary Key:	CameraID
Foreign Key:	CameraID

Relationship Class: rGTIR Landscape Polygons to GTIR Still Data Table

Cardinality:	1:1
Origin Table / Feature Class:	GTIR_LandscapePOLY
Destination Table / Feature Class:	GTIR_StillData
Primary Key:	StillID
Foreign Key:	StillID

Relationship Class: rGTIR Still Data Table to GTIR Still Pixel Centers Table

Cardinality:	1:M
Origin Table / Feature Class:	GTIR_StillData
Destination Table / Feature Class:	GTIR_StillPixelCenters
Primary Key:	StillID
Foreign Key:	StillID

Relationship Class: rSev PlotPoints to Sev Soil Plot Table

Cardinality:	1:M
Origin Table / Feature Class:	Sev_PlotPT
Destination Table / Feature Class:	Sev_SoilPlot
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rSev Plot Points to Sev Tree50 Plot Data

Cardinality:	1:M
Origin Table / Feature Class:	Sev_PlotPT
Destination Table / Feature Class:	Sev_Tree50
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rSev Plot Points to Sev Veg100 PlotTable

Cardinality:	1:M
Origin Table / Feature Class:	Sev_PlotPT
Destination Table / Feature Class:	Sev_Veg100
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rSev Plot Points to Sev Veg750 Plot Table

Cardinality:	1:M
Origin Table / Feature Class:	Sev_PlotPT
Destination Table / Feature Class:	Sev_Veg750
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rSev Plot Points to Sev Water Plot Table

Cardinality:	1:M
Origin Table / Feature Class:	Sev_PlotPT
Destination Table / Feature Class:	Sev_WaterPlot
Primary Key:	PlotID
Foreign Key:	PlotID

Relationship Class: rSev SubPlot Points to Sev Soil SubPlot Table

Cardinality:	1:M
Origin Table / Feature Class:	Sev_SubPlotPT
Destination Table / Feature Class:	Sev_SoilSubPlot
Primary Key:	SubPlotID
Foreign Key:	SubPlotID

Relationship Class: rSev SubPlot Points to Sev Water SubPlot Table

Cardinality:	1:M
Origin Table / Feature Class:	Sev_SubPlotPT
Destination Table / Feature Class:	Sev_WaterSubPlot
Primary Key:	SubPlotID
Foreign Key:	SubPlotID

Relationship Class: rSevTree50_SevTree50Data

Cardinality:	M:M
Origin Table / Feature Class:	Sev_Tree50
Destination Table / Feature Class:	Sev_Tree50Data
Primary Key:	PlotID
Foreign Key:	PlotID

GLOSSARY

Cardinality. The number of rows that represent each object in a table. The relationships between tables are often expressed as a cardinality ratio; or how many objects participate in the relationship from each table. There are three cardinality relationships expressed in a database: one-to-one (1:1), one-to-many (1:M), and many-to-many (M:M).

Conceptual Schema. A graphical representation of the geodatabase structure including all of the data elements and their relationships.

Feature Class. A feature class is a "collection of features with the same type of geometry: point, line, or polygon."[178]

Feature Dataset. In a geodatabase, a feature dataset is a grouping of feature classes that share a common spatial reference. The feature dataset includes spatial features, non-spatial entities, and the relationships between them.[179]

Geodatabase. ESRI defines a geodatabase as "a physical store of geographic information inside a database management system (DBMS)."[180] The geodatabase data model allows spatial data (objects and their attributes) to be stored together in a way that supports advanced rules and relationships between the data. The geodatabase is built within a non-proprietary DBMS, making it more interoperable than other data models. A geodatabase can exist at two levels: the personal geodatabase and the multi-user geodatabase. The personal geodatabase is intended for use by a single user and has size, editing, and storage restrictions. The multi-user geodatabase provides for larger data volumes and allows multiple users to view and edit data simultaneously because it is served through a spatial data engine (SDE).

Geographic Data Model. In his book, Modeling Our World, Zeiler defines a geographic data model as "an abstraction of the real world that employs a set of data objects that support map display, query, editing, and analysis."[181] The data model also

---

[178] Zeiler, 64.

[179] Ibid, 8.

[180] Environmental Systems Research Institute, Inc. Geodatabase Web site: *http://www.esri.com/software/arcgis/geodatabase.html.*

[181] Zeiler, 4.

provides the vocabulary and structure that is used to represent and refer to objects and places on the earth.[182] The most basic form of geographic data model is the paper map. As map-making and analysis has become computerized, geographic data models that you might be familiar with include CAD, coverage, shapefile, TIN, and cell-based raster models. The geodatabase is the most recent geographic data model available.

Geographic Information System (GIS). The ESRI Press Dictionary of GIS Terminology defines GIS as "a collection of computer hardware, software, and geographic data for capturing, storing, updating, manipulating, analyzing, and displaying all forms of geographically referenced information".[183]

Geospatial Technologies. Those electronic technologies that capture data describing the spatial and non-spatial properties of geographic features on the earth. Geospatial technologies include remote sensing, geographic positioning systems (GPS), and geographic information systems (GIS).

Interoperability. There are many definitions of interoperability in the literature. Bishr defines interoperability as the "ability of a system, or components of a system, to provide information portability and inter-application cooperative process control."[184] ESRI focuses its definition from a general system to a GIS saying, "an open GIS system allows for the sharing of geographic data, integration among different GIS technologies, and integration with other non-GIS applications. It is capable of operating on different platforms and databases and can scale to support a wide range of implementation scenarios."[185] The definition that will be used for the rapid response SDE geodatabase project is: the ability to share disparate data sets among multiple platforms, databases, development languages, and applications.

Key. A field containing attributes which uniquely identify rows within a table.[186]

Normalization. The process where tables within a relational database are changed—either split or combined—to reduce editing and deletion errors.[187]

Precision. The number of decimal places that will be stored for each spatial coordinate within a geodatabase.[188]

SDE. ESRI's spatial data engine, ArcSDE, is a platform that facilitates the management of geospatial data in a DBMS.[189] ArcSDE supports geodatabases built in IBM

---

[182] Ibid, vii.

[183] Heather Kennedy, ed. *The ESRI Press Dictionary of GIS Terminology* (Redlands: Environmental Systems Research Institute, Inc., 2001).

[184] Bishr, 299.

[185] Environmental Systems Research Institute, Inc. *Spatial Data Standards and GIS Interoperability.*

[186] Kroenke, 116.

[187] Kroenke, 113.

[188] Environmental Systems Research Institute, Inc. *Introduction to ArcSDE.*

DB2, IBM Informix, Microsoft SQL Server, or Oracle. ArcSDE essentially provides the infrastructure that links ESRI GIS software to a geodatabase and controls access, querying, editing, and versioning of the geodatabase. For most users, the SDE is an invisible string that connects their desktop GIS with the data they require. For more advanced users, the SDE allows them to view different versions of the same geodatabase; to edit the same feature that another user is editing and choose which edit to save; and to check-out portions of the geodatabase to view and edit on their desktop or a mobile unit and check those edits back in to the geodatabase.[190] ArcSDE can be used to establish a connection between a geodatabase and a host of ESRI GIS software including ArcGIS (desktop GIS), ArcIMS (Web-enabled GIS), and ArcPad (mobile GIS).

Schema. A graphical representation of the spatial database structure including all of the data objects, their attributes and their relationships to each other.[191]

Spatial Domain. The allowable coordinate range for x,y coordinates within a personal or enterprise geodatabase.[192]

Topology. The spatial relationship between adjacent features within a geographic data set.

Unified Modeling Language (UML). UML is a standardized modeling language providing graphical notation and syntax for software processes and methods.[193] UML is independent from a software process as it can be used with any process to record and represent analysis and design models.[194]

---

[189] Environmental Systems Research Institute, Inc. ArcSDE Web site: *http://www.esri.com/software/arcgis/arcinfo/arcsde/index.html.*

[190] Environmental Systems Research Institute, Inc. *Introduction to ArcSDE: Lectures.*

[191] Rigaux et al., 7.

[192] Environmental Systems Research Institute, Inc. *Introduction to ArcSDE.*

[193] Fowler and Scott, 1.

[194] Ibid, 14.

BIBLIOGRAPHY

Able, David J., Beng Chin Ooi, Kian-Lee Tan, and Soon Huat Tan. "Towards Integrated Geographical Information Processing." *International Journal of Geographical Information Science* 12, no. 4 (1998): 353-371.

Arctur, David, David Hair, George Timson, E. Paul Martin, and Robin Fegeas. "Issues and Prospects for the Next Generation of the Spatial Data Transfer Standard (SDTS)." *International Journal of Geographical Information Science* 12 no. 4 (1998): 403-425.

Bishr, Yaser. "Overcoming the Semantic and Other Barriers to GIS Interoperability." *International Journal of Geographical Information Science* 12 no. 4 (1998): 299-314.

Devogele, Thomas, Christine Parent, and Stefano Spaccapietra. "On Spatial Database Integration." *International Journal of Geographical Information Science* 12 no. 4 (1998): 335-352.

Elmarsi, Ramez, and Shankant B. Navathe, *Fundamentals of Database Systems*, 4th ed. Boston: Pearson Education, Inc., 2004.

Environmental Systems Research Institute, Inc. *Spatial Data Standards and GIS Interoperability*. Redlands: Environmental Systems Research Institute, Inc., January 2003. Available from: *http://www.esri.com/library/whitepapers/pdfs/spatial-data-standards.pdf*

__________. *Introduction to ArcSDE: Lectures*. Redlands: Environmental Systems Research Institute, Inc., April 2003.

__________. ArcSDE Web site: *http://www.esri.com/software/arcgis/arcinfo/arcsde/index.html* (Updated 13 October 2003)

__________. Geodatabase Web site: *http://www.esri.com/software/arcgis/geodatabase.html* (Updated 13 October 2003)

__________. Frequently Asked Questions Web site: *http://support.esri.com/knowledgeBase/documentation/FAQs/sde_/WebHelp/faq.htm*

Fernandez, R. Norbeto, Marek Rusinkiewicz, Lucia Morais da Silva, and Chris J. Johannsen. "Design and Implementation of a Soil Geographic Database for Rural Planning and Management." *Journal of Soil and Water Conservation* 48 no. 2 (1993): 140-145.

Fowler, Martin, and Kendall Scott. *UML Distilled: Applying the Standard Object Modeling Language*. Reading: Addison Wesley Longman, Inc., 1997.

Hardy, Colin C., and Philip J. Riggan. "Demonstration and Integration of Systems for Fire Remote Sensing, Ground-Based Fire Measurement, and Fire Modeling." Proposal to the Joint Fire Science Program submitted under JFSP RFP-2003-2 Task 1. (2003)

Helmbrecht, Don. "Demonstration and integration of systems for fire remote sensing, ground-based fire measurement, and fire modeling project: Database development." Missoula: National Center for Landscape Fire Analysis, 2003.

Helmbrecht, Don, and Lee Macholz. "Plan of Work for the Rapid Response Project Within the National Center for Landscape Fire Analysis." Missoula: National Center for Landscape Fire Analysis, 2003.

Henry, Joel. *Software Project Management*. Boston: Addison-Wesley, 2003.

Joint Fire Science Program. Joint Fire Science Plan Web site: *http://jfsp.nifc.gov/JointFire.html*

Kennedy, Heather. ed. *The ESRI Press Dictionary of GIS Terminology*. Redlands: Environmental Systems Research Institute, Inc., 2001.

Kroenke, David M. *Database Processing Fundamentals, Design and Implementation*, 7th ed. Upper Saddle River: Prentice Hall, 2000.

Maitra, Julie Binder, and Norman Andersen. *Geospatial Standards (Article 1 of 4)*. Federal Geographic Data Committee. Available at: *http://www.fgdc.gov/publications/documents/standards/geospatial_standards_part1.html*

Murad-al-shaikh, Makram, Krista Page, Mark Stewart, and Marnel Taggart. *Introduction to ArcGIS I*. Redlands: Environmental Systems Research Institute, Inc., 2003.

__________. *Introduction to ArcGIS II*. Redlands: Environmental Systems Research Institute, Inc., 2003.

National Wildfire Coordinating Group. *Firefighters Guide*. NFES 1571 PMS 414-1. Boise, 1986.

__________. Information Resource Management Working Team. *Geospatial Task Group Charter*. Boise, 1999. Available at: *http://www.nwcg.gov/teams/irmwt/gtg/Charter.pdf*

Rigaux, Philippe, Michel Scholl, and Agnes Voisard. *Spatial Databases: With Application to GIS*. San Francisco: Morgan Kaufmann, 2002.

Sommerville, Ian. *Software Engineering*. Harlow, England: Pearson Education Limited, 2001. Sixth Edition.

U.S. Department of the Interior. General Accounting Office. *Geospatial Information: Technologies Hold Promise for Wildland Fire Management, But Challenges Remain*. GAO-03-1047. Washington, D.C., September 2003.

U.S. Federal Geographic Data Committee. *FGDC Standards Reference Model*. Washington, D.C., March 1996. Available at: *http://www.fgdc.gov/standards/refmod97.pdf*

__________. John D. Evans, ed. *A Geospatial Interoperability Reference Model, v1.0*. Washington, D.C, May 2003. Available at: *http://gai.fgdc.gov/girm/*

U.S. Office of Management and Budget. *Federal Participation in the Use of Voluntary Consensus Standards and in Conformity Assessment Activities*. Circular No. A-119 Revised. Washington, D.C., 10 February 1998.

__________. *Coordination of Geographic Information and Related Spatial Data Activities*. Circular No. A-16 Revised. Washington, D.C. 19 August 2002.

U.S. President. 1994. Executive Order 12906. "Coordinating Geographic Data Acquisition and Access: The National Spatial Data Infrastructure." *Federal Register* 59, no. 71 (13 April 1994): 17671-17674.

Zeiler, Michael. *Modeling Our World: The ESRI Guide to Geodatabase Design*. Redlands: Environmental Systems Research Institute, Inc., 1999.