Technical Report: isda06-001
January 11, 2006

Peter Bajcsy, pbajcsy@ncsa.uiuc.edu
David Clutter, dclutter@ncsa.uiuc.edu

National Center for Supercomputing Applications
1205 West Clark, Urbana, IL 61820

Information Gathering about Decision Processes Using Geospatial Electronic Records

Abstract

We address the problems of gathering, archiving and analyzing information about decision making processes using geospatial electronic records (e-records). Our ultimate goals are to gather all observable variables about decision making processes using input geospatial e-records so that (a) a decision making process can be documented in the future, (b) audits and authentications of records can be executed, and (c) accountability of decisions can be achieved. Our objectives are (1) to evaluate the tradeoffs between the amount of information about decision making processes, e-record authentication and audits, and the associated storage and computer performance cost, (2) to develop prototype software for gathering information about the use of computers in high assurance decision making and high confidence application scenarios, such as in military, medicine, or law-enforcement, and (3) to provide guidelines about the future uses of computers in government decision making. To meet the above objectives, one has to perform studies addressing several high performance computing problems related to information gathering, data storage, management, access and retrieval, and novel computer architectures.

We report our preliminary results obtained for a class of decision processes using geospatial electronic records that are related to emergency response decision processes (e.g., foreign plant disease, hurricane Katrina scenarios), land use decisions (e.g., urban development in the third-world countries) or regulatory decision processes (e.g., nutrient levels allowed by EPA). For this purpose, we have prototyped an information gathering system using multiple geospatial analytical tools that support this class of decision processes. Our tradeoff analyses are performed at several levels of information granularity and the results are reported for three distinct decision process types, such as data intensive, computationally intensive, and high complexity decision processes. The preliminary experimental results indicate that unique signatures of decisions processes based on gathered information could be established for understanding of gathering, archiving and retrieval requirements to provide guidelines about the future uses of computers in government decision making.

1. Introduction

We address some of the key computer science problems related to the use of computers in government decision making to improve the capability of documenting these processes based on current information and provide direction for improving this capability in the future in verifiable ways. There is a need to characterize government decision making by providing information about "who knew what and when" during a decision making process in high confidence environments. The motivation for our research comes from the need to research improvements to the current systems for archiving and preserving final documents about government decisions and any decision supporting data sets that might not provide sufficient information about (a) who contributed to important decisions, (b) what was known to decision makers, (c) when was information available to decision makers, and (d) how were government decisions derived from known input information and data sets.

A few hypothetical application scenarios of government decision processes are provided to illustrate the need for our research and development. For example, if the tools for information gathering would be available at the time of Cuban missile crisis or at the time of the accidental bombing of Chinese embassy in Yugoslavia then today we could re-constitute the government decision-making processes. Another example would be the documentation of the space shuttle Challenger and Columbia disasters. Military flight tests and automobile tests could benefit from application specific tools that would gather all information about instruments, human participation and surroundings. Similarly, emergency planning and responses for epidemic breakouts could be other application scenarios for our work.

From a of computer science viewpoint, the application scenarios could be presented as follows. For instance, geospatial data analysis techniques from the NCSA Im2Learn library in their current form are applied to a federal sector analysis in autumn 2004. Results and outcomes accrue and current government decisions are made based on those Im2Learn geospatial analyses. The input data and the Im2Learn geospatial data analysis techniques are retained. Some number of years later the business activities of government compel that the foundational Im2Learn analysis be re-visited to understand previous decisions. *What input data, methodology and work flow, results and technology components should be retained and how should they be retained in order to enable those Im2Learn analyses be robustly re-constituted in that future year?* The answers to this question could be viewed as the long term goals of our research.

In this paper, we constrain our focus only on electronic (digital) geospatial data analysis and decisions derived from them. We chose geospatial e-records because the records have a large file size, vary over time, are likely stored in distributed systems and are characterized by multiple levels of datum uncertainty. Our specific research objectives are formulated as a set of basic questions that have to be answered in order to understand the preservation mechanisms.

- What information has to be gathered?

- How would software tools collect information about decision making processes without any intrusion to the current use of computers in government decision making processes?
- What are the tradeoffs between relevant information and computer performance cost?

Figure 1 shows a sequence of these questions as they would be addressed in the presented work. As illustrated in **Figure 1**, we are also interested in understanding additional issues depending on a concrete application scenario, including:

- What software is needed for authentication of e-records and for documenting decision making processes based on all archived information and preserved technology?
- How would one retrieve information about "who knew what and when" from archived data?
- Can one compare decisions derived by multiple people, using different technology and with temporally changing input information?

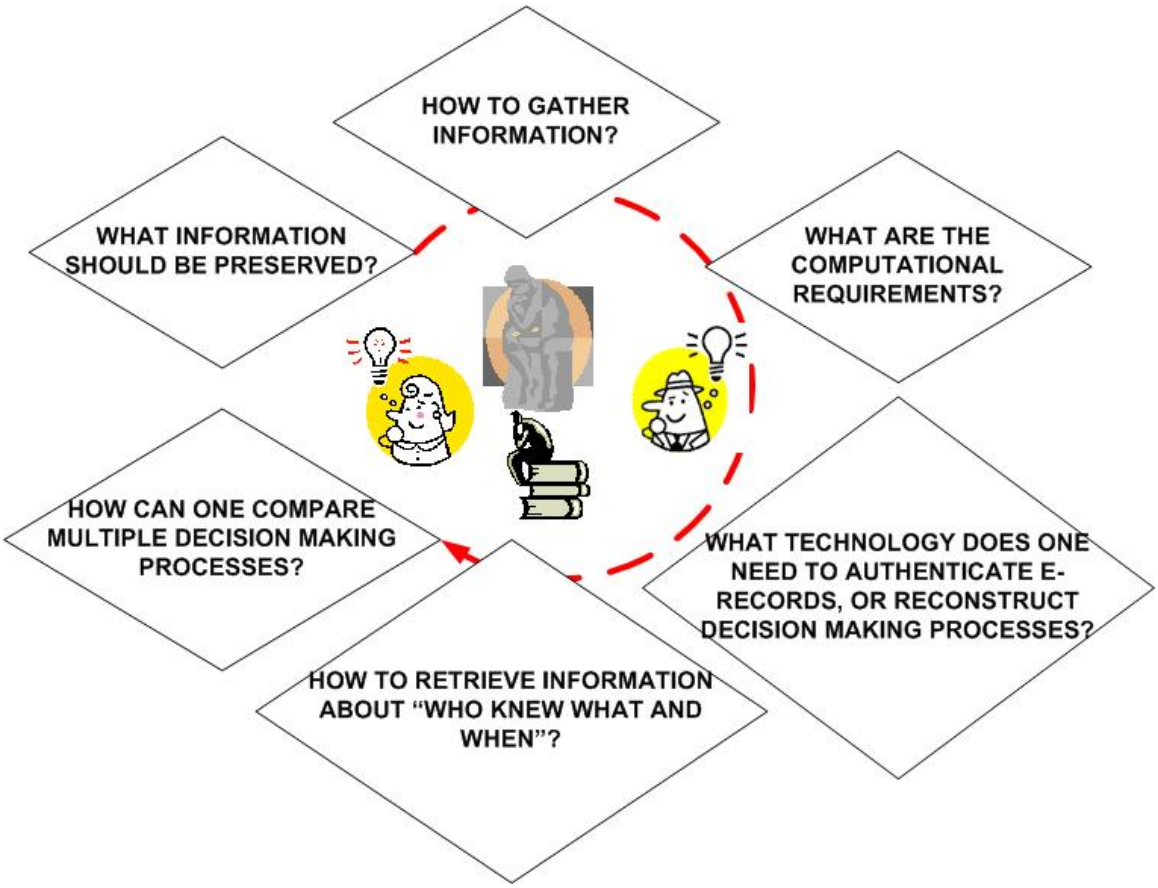

Figure 1: A sequence of questions to address the problem of preservation mechanisms about data analysis and decision making processes.

Our approach in this paper is based on analyzing mechanisms of gathering, archiving and analyzing information about decision making processes using geospatial electronic records (e-records) and their tradeoffs. With our approach, the goals are to gather all

observable variables about decision making processes using input geospatial e-records so that (a) a decision making process can be documented in the future, (b) audits and authentications of records can be executed, and (c) accountability of decisions can be achieved. Our end products are (1) the tradeoff evaluations between the amount of information about decision making processes, e-record authentication and audits, and the associated storage and computer performance cost, (2) the prototype software for gathering information about the use of computers in high assurance decision making and high confidence application scenarios, such as in military, medicine, or law-enforcement, and (3) the guidelines about the future uses of computers in government decision making.

We describe previous work and the information gathering challenges that one has to overcome in Section 2. Section 3 presents the information gathering framework and addresses some basic computer science issues. Next, Section 4 outlines our developed experimental prototype and the preliminary results. In Section 5, we discuss the lessons learnt and summarize our work in Section 6.

2. Previous Work and Challenges

2.1 Terminology

The topic of this report is the information gathered about decision making processes while using computers. The information is related to (a) the origin of geospatial data sets, processing tools and computational environments (hardware and software), (b) the sequence of processing steps including user interactions with data sets, and (c) humans interacting with data sets. There have been several terms used in the past to denote some parts of the information we are interested in.

First, the term lineage refers to origin and subsequent processing history of a data set for computer-based processing with limited human interventions [39]. The computer-based processing (also called *in-silico* processing [29], [44]) is loosely defined as (a) script- or program-based, (b) query-based, (c) workflow-based, or (d) service-based. The terms data provenance and data pedigree have been used for query- and service-based processing [27], [28]. Other terms, such as derivation history, data set dependence, execution trace, log/logbook, filiation, data genealogy, data archeology and audit trails, are referenced in the literature (see the survey paper [39]).

In some papers, the term provenance is more elaborated. For example, in [44], two forms of provenance are introduced, such as (a) the derivation path that records the process by which results are generated and (b) annotations that describe collections of objects. In [38], provenance of a piece of data is defined as the process that led to the data, and it is viewed as a concept (source or derivation of an object) and a record of such derivation. The term knowledge provenance was introduced in [33] to denote both meta-information as a description of knowledge source, and knowledge process information as a description of a reasoning process to generate the answer. The difference between knowledge provenance and data provenance is in its inclusion of proof-like information about the process by which knowledge is extracted. In [37], data provenance is described

as meta-data relating data to other data, with multiple scales and levels of details. This definition accommodates the fact that the cost of recording provenance depends on type, purpose, discipline and project specifications, as it is of interest to us.

We will use the term decision process provenance to refer to the information we are interested in gathering. The meaning of the term is different from the past definitions by including the pieces of information about computational environments, user interactions with data sets and humans interacting with data sets.

2.2 Previous Work and Challenges

The problem of designing preservation mechanisms to document events using electronic information has been of interested not only to the National Records and Archive Administration (NARA) ([4], [5], [6], [7], [10], and see the reports on the US-InterPARES project [9], [36]), but also to NASA (aircraft safety), DARPA (LifeLog [12] and the "existential technology" by Steve Mann), the American Food and Drug Administration [26] and NIH [11]. In addition, the scientific communities (specifically the GRID communities [39] and corporations like Microsoft (MyLifeBits [45]) have invested into researching technologies that would address many of the provenance issues in order to capture knowledge and provide commercial solutions.

The general problem of building a provenance system has been approached by several projects [39] in the domains of environmental analysis [31], hydrologic and climate modeling, molecular biology [44], [30], physics and chemistry [34. Among these projects, we are interested in those that used geospatial data sets [31] and related informatics issues [13]. It is known that a provenance system has to deal with large amounts and a wide variety of provenance information and hence the system has to be scalable, general and customizable, to include trust, security and preservation [26], as well as to provide automatic and systematic provenance gathering [44]. To our knowledge, there is not a provenance system that would meet all above requirements, as well as specific requirements defined by multiple application domains.

As it was stated in [26], there is a paucity of standards, components and techniques for recording provenance. The issue of standards might not have been addressed yet since the first provenance systems were reported only about a decade ago (Geolineus in 1993 to track ArcInfo GIS operations, GOOSE in 1994, and Geo-Opera in 1997 [39]). In addition, the lack of standards in semantic and syntactic descriptions about data leads to difficulties when the data sets are used, stored and retrieved. For instance, the georeferencing information stored in three different places in GeoTiff files can lead to conflicting values in real applications [22], [23]. The issue of components and techniques for recording provenance has been approached from the perspective of a single human or a community. From the perspective of a single human (Gordon Bell and his MyLifeBits [45]), electronic provenance information could be automatically captured by recording email messages, computer keystrokes, phone calls, images, videos, web pages, etc. The recording is triggered by smart sensors detecting light, heat and position, and the data are stored in a searchable, indexable and portable database that is offloaded on a regular basis. However, this type of a provenance system is not applicable to the case of

government decision making processes. From the perspective of a community, the past work in the areas of collaborative environments and experiment management [39] might be viewed as a closer match to the case of government decision making processes. Nonetheless, the type and scope of scientific collaborative or experiment management provenance information is different from the type and scope of government decision making provenance information.

There are typically three components of scientific information systems: (1) design experiment (workflow systems, computational modeling systems, information management systems), (2) conduct experiment (experiment management systems, electronic notebooks), and (3) analyze results (scripting and programming environments) [39]. In the case of government decision making systems, these three components are viewed as the path to answering a question (or defining a question according to the Japanese understanding [46]), where the path (a) usually involves a diverse team membership (multiple agencies with a broad spectrum of expertise), (b) might lack of policy guidance, (c) could be characterized by a low team authority, (d) involves internal politics and organizational inertia, (e) typically lacks of integration, and (f) has gaps and ambiguities in the process [47, Chapter 11] . Thus, the provenance information types and scopes gathered during government decision making are much more limited in comparison with the scientific provenance information. Furthermore, the path to making a government decision might not have a collaborative or experimental nature as in the most scientific systems.

The majority of the past work on scientific provenance systems has excluded interactive processes, as well as human-centric provenance information, because many of the scientific processes are scripts without any human interaction required. It is just a recent trend to design cyber-collaboratories to enable information sharing, and explore social network aspects [48]. It is assumed that most of the *in-silico* processing to support government decisions is performed with commercial software packages providing interactive interfaces. The inclusion of provenance information about human interactions might be critical not only for understanding who knew what and when, but also for addressing the problems of process reconstructions from parallel streams of provenance information (e.g., synchronization of computational sequence and human interaction sequence). For example, two auditing questions and one process reconstruction question could be raised: Did a user scroll the image to view the right lower image area containing an object of interest? Was a user interrupted during the analysis related to a decision making? Would we be able to reconstruct user interactions with data and sequential computational processes if the hardware had changed?

The biggest challenge of designing a provenance system for government decision processes comes from the fact that the task is usually about retrofitting provenance mechanisms into already existing software rather than designing provenance into the software from the beginning. Thus, the architectures for designing provenance systems vary a lot depending on the initial project requirements and whether the focus is on data/metadata management (scientific application middleware built on the Web distributed authoring and versioning standards [37] or virtual data system [49]),or

service oriented architectures (client-service scenarios) [26] [30], workflows running on the GRID [44], archiving [28] or logical roles of provenance providers and consumers [38]. In this work, we did not design or adopt the architecture of a provenance system but rather we attempted to evaluate the cost of preservation based on multiple approaches. We leveraged our past work on designing information gathering systems [1], [2] and evaluating the tradeoffs about geospatial data representations [14], [15].

In addition to the issues presented in the past work, one is interested in understanding the cost of data storage, management, access and retrieval as a function of the scope (scale or granularity) of provenance information. Ideally, solutions should scale linearly with increasing data size, and should be independent of specific hardware and software generations. There is also a need to research and develop novel software for gathering, organizing and storing information, and to explore the use of high performance computing (HPC) or novel computer architectures for the preservation purposes. Finally, there is the problem of provenance information organization and retrieval, statistical analysis/summaries of gathered data, navigation, browsing, and visualization of all gathered information that are of our interest.

3. Information Gathering Framework

We outline an information gathering framework by discussing information gathering issues in the next sections. The main issues include (1) what information should be extracted about decision processes, (2) at what computer system level to gather information, (3) how to gather information at user program computer system level, and (4) how to save gathered information.

3.1 What Information Should Be Extracted about Decision Processes

We have focused on answering the question what information should be extracted about decision processes. The granularity of information maps directly to preservation cost and to the preservation value of information although the mapping is not know currently. We list information grains as follows: (1) all output data, (2) input data, (3) intermediate data, (4) image interaction information (sub-area selection, zoom operations, selection of image display from the stack of raster files, selection of boundary display from the stack of vector files, boundary selection), (5) parameter selection (selection of input and output variables for the decision tree plus all parameters), (6) all input, output and intermediate data that were shown to a decision maker (who knew what and when), (7) hardware and software information (version numbers and patches), (8) computation operations (what classes were called and executed), (9) video, audio and mouse movement during a decision making process, and (10) image drawings and text annotations. Figure 2 illustrates the mappings between information grains, preservation cost and preservation value.

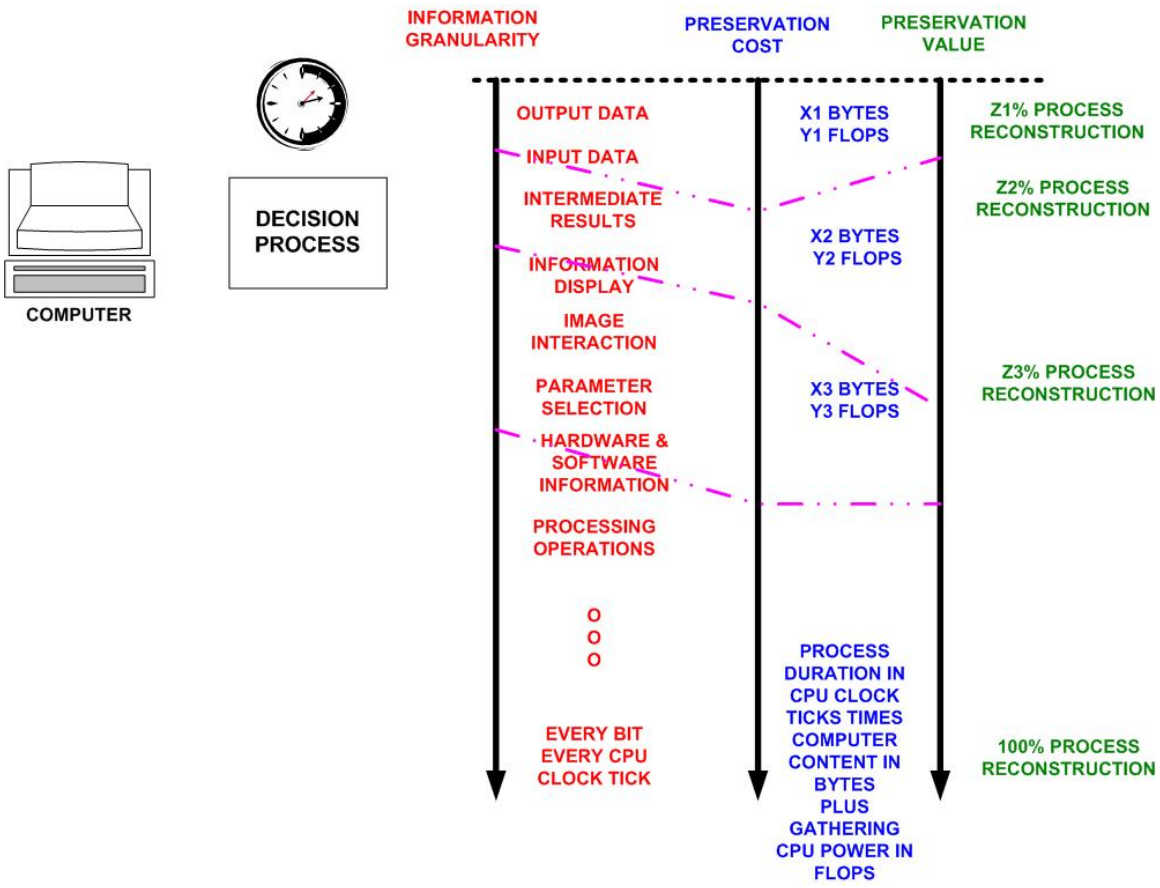

Figure 2: Illustration of the mappings between information granularity, preservation cost and preservation value.

3.2 At what computer system level to gather information

When it comes to extracting information about computer operations, one should be aware of multiple levels of computer systems. A schema of levels in computer systems is presented in Figure 3. How to gather information depends on the computer system level at which one would log information.

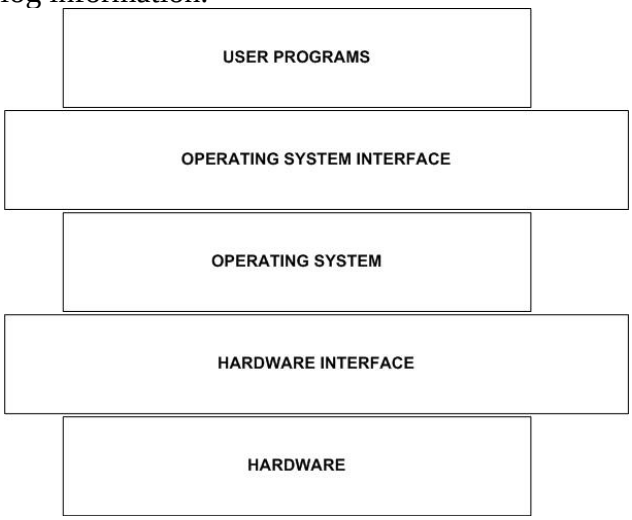

Figure 3: Levels in a computer system (according to Fig. 1.1 in Crowley C. Operating Systems, A design oriented approach, Irwin book team, a Times Mirror Higher Education Group. Inc., 1997, 844p.)

In general, information gathering can be executed by modifying (customizing) computer system levels either to directly gather information or to communicate through a middle layer that intercepts and gathers information. For example, one can modify user programs (Java source code), operating system (e.g., Berkeley tinyOS) or hardware (e.g., MICA motes from Crossbow Inc.) to perform information logging or use a middle layer (e.g., java-based visual programming environment D2KSL [43] or CyberIntegrator [41]) to intercept input and output of source modules. Our current approach has focused on information gathering at the level of computer systems denoted as user programs in Figure 3.

Information gathering at the user program level can be described by Figure 4. There are multiple entry points that one could consider and the information gathering would be executed by (a) modifying the source code of programs by application developers, (b) creating a middle layer that every program has to go through (e.g., custom application framework such as D2KSL), (c) developing a custom compiler (e.g., custom javac or gcc), (d) building custom run time libraries (e.g., custom jre.jar, libc.so (unix) or stdc.dll (windows)), or (e) developing custom archiving, linking and loading code. While the entry points labeled as (a) and (d) would be used for direct information gathering about each executed class/method (e.g., inserted logging functions), the other entry points (b), (c) and (e) would be used for information gathering about any intercepted communication between classes/methods. Although Figure 4 shows C/C++ and Java branches of the source code, the illustration represents two paths for any programming language that is compiled to (1) operating system (OS) dependent machine code, for example, C/C++, Fortran, Algol, Pascal, etc. or (2) a non-OS dependent byte code interpreted at the run time, for example, Java or Basic.

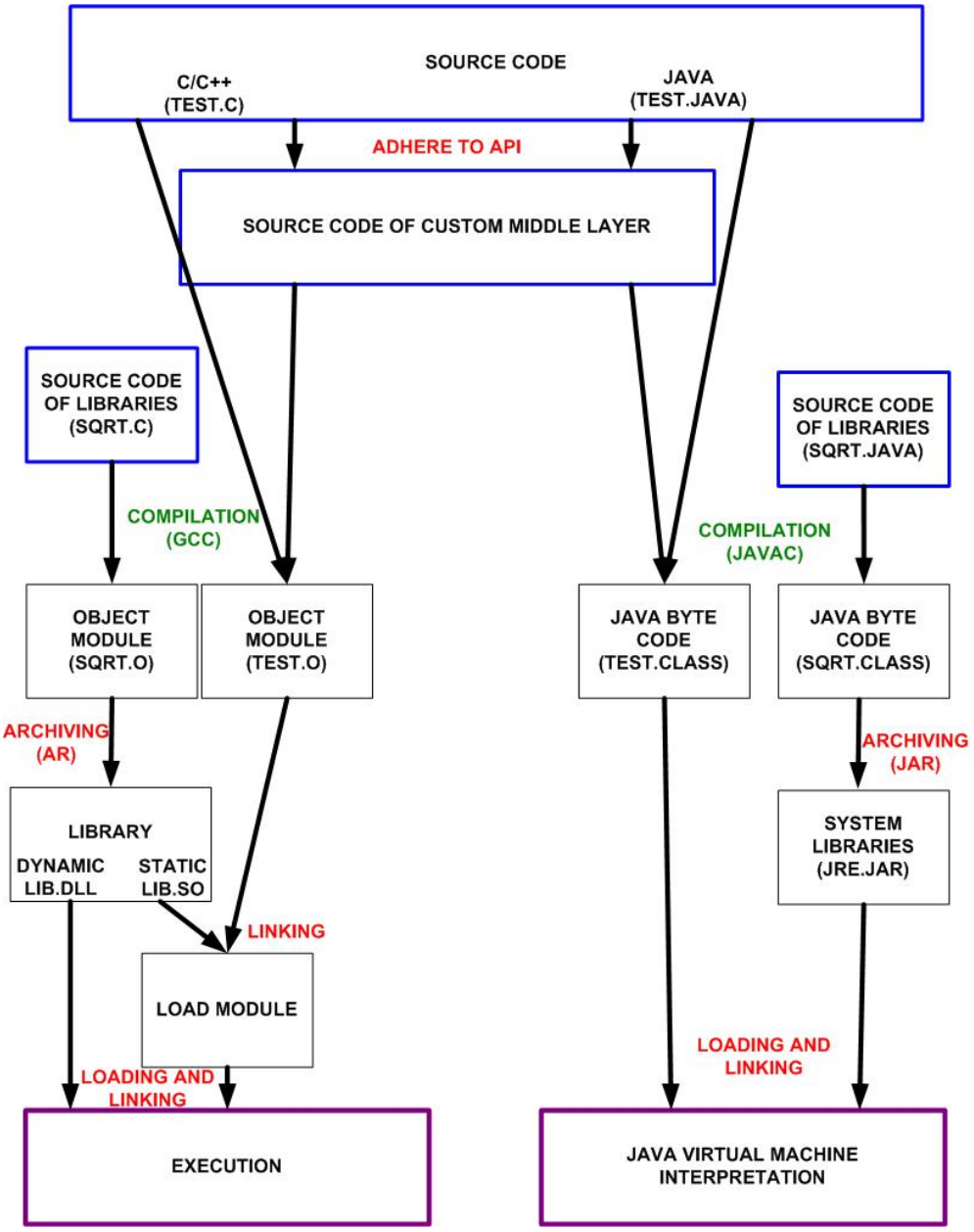

Figure 4: Entry points for information gathering at the user program level.

It is conceivable to create custom executers for every platform (e.g., modified Java virtual machine (JVM)) but it would lead to gathering at other computer system levels. For instance, JVM is a user application that is compiled to OS-dependent machine code and will interpret java byte code. The JVM and the user program source code compiled to OS-dependent machine code will be executed in the user program box of Figure 3 by using the OS interface layer.

Based on our available resource for this work, we decided to explore information gathering at the source code and middle layer entry points.

3.3 How to Gather Information at User Program Computer System Level

In order to perform information gathering at the source code and middle layer entry points, we would like leverage already existing software packages for quality control, Java visual programming development and information logging. Based on the existing software, we present our evaluations of three different approaches, one with quality assurance software packages for testing graphical user interface (GUI), one with Java visual programming development (known as D2KSL) and one with logging functions.

First, we have investigated the use of already existing software packages for quality control (QC) and quality assurance (QA). The following software packages were considered: (a) Abbot http://abbot.sourceforge.net/, (b) jfcUnit http://jfcunit.sourceforge.net/, and (c) Jacareto http://jacareto.sourceforge.net/. These software tools are designed for recording and play-back of user interface interactions with mouse and keyboard, for example, mouse movement, mouse clicks, dialog menu selection, keyboard interactions and so on. The problem with these tools is that they record only the user interface interactions and ignore recording data processing and data modifications. The use of these tools would enable easy decision process reconstruction based on preserving user interactions with the software GUI. One could think of this preservation mechanism as recording a movie of the events on a computer screen. During the playback, the recorded process is re-executed and any dependencies on the state of the recording machine would be needed (and hence would have to be preserved). One example of such dependencies is the file system. The file location could not change between the recording time and the playback time because the playback would interact with the file system. Thus, the underlying assumptions of these QA/QC tools are that there are no changes between the recording and playback times with respect to (a) a file system, (b) GUI layouts and (c) GUI functions.

From the preservation viewpoint, there are three drawbacks in the use of these QA/QC tools. First, the preserved information would not provide direct answers to the data related questions, such as “what is the input data set?” or “what data operations took place during decision processes?” Second, to answer questions about what image part was viewed (“who knew what and when”), one would have to replay the entire recording rather than obtaining a direct answer to the query “what image parts had been viewed during a decision process?” Third, one would have to preserve an entire file system in order to reconstruct the recorded process. If the recorded process could not be replayed then there would be no way to possibly answer questions related to processed data (e.g., see what file was loaded) and viewed data (e.g., what image parts were viewed and at what zoom level).

Second, we considered using the source code of Java visual programming environment D2KSL as the middle layer that intercepts communication between linked code modules. The advantage of this approach lies in the fact that if everyone would adhere to the D2KSL API then information gathering could be embedded into the middle layer and it would not require source code modifications of existing user programs. The D2KSL approach was not adopted for this study, although it might be considered in the future. The challenges with the use of the current D2KSL prototype are in its ability to gather

only information about input and output variables of each module. The information about (a) image data interaction ("who knew what and when"), (b) operations inside a module, and (c) intermediate data cannot be gathered.

Third, we investigated the approach by inserting a logging function into critical code locations. This approach has its own pros and cons. The pros of this approach include (a) a flexible selection of information granularity (we can insert logging functions anywhere), (b) an easy prototyping (it will allow us to perform the preliminary cost study in a short period of time), and (c) a partial utilization of java virtual machine functionalities (we can use the JVM methods for tracing the execution by recovering the function call stack). The cons of this approach are in (a) the lack of software interchangeability (we cannot take other software and run the information recording without inserting the logging functions), (b) the amount of work related to scrutinizing the code for critical locations and inserting the logging functions, and (c) the extra development of decision process reconstruction tools (we cannot directly reuse the QA/QC playback capabilities but it would be possible to leverage the packages in the future).

We decided to adopt to explore a combination of all three approaches to the problem of information extraction. The motivation for our decision is based on our intention to fully understand the cost and value of preserved information at multiple granularity levels that would not be possible with the first and second approaches, and would be extremely labor intensive with the third approach. Given enough time and resources, the use of the third approach allows us to answer all questions about information cost and value while the use of the first and second approaches would not.

3.4 How to Save Gathered Information

Among many information storage mechanisms, we have investigated the use of HDF, XML and free text formats. We implemented the support of these three file formats for saving any gathered information. In our preliminary experiments, we evaluated XML and free text formats. We decided to prefer the use of XML file format because of the abundance of software packages to process XML files. Currently, we are exploring the use of HDF and XML to efficiently store the information gathered. Figure 5 shows the type of information stored in XML files.

```
<xml>
<event type="1" class="ncsa.im2learn.ext.geo.AlignGeoTiles"
method="align" description="beginning of align geo tiles"
timestamp="1120233580574"
arguments="H4sIAAAAAAAAAFvzloG1uIhBOCuxLFGvtCQzR8+xqCix0iezuKSi8ZLszOOJ
c5kZGD0ZWIozq1Ir
ChgYGJjKWYAkVylQU7QPWFtOYl66XnBJUWZeuvXaS2HPX8pWuzMxMEBUlzBY6Wfk56bqJ+e
UlpSk
FumnJJYkFqeWFOv7ubrEe/rEO7sY6icWpSYaGOoXlyYVJ+YW5KQmVRob6GWkFBGh2Qin5oI
KABX6
Ap3dAAAA"/>
```

```
<event type="0" class="ncsa.im2learn.ext.geo.AlignGeoTiles"
method="toArray" description="returning result of align"
timestamp="1120233585084"
arguments="H4sIAAAAAAAAANy7dXQb6Z6u6zAzp9OhDnaYGZ8wOczkmJnZsS1Llm3ZFlps
xhjiMHW4k3SYmTvM
```

Figure 5: An example of XML stored recordings of two processing methods executed during geospatial alignment (registration).

4. Experimental Prototype and Preliminary Results

The experimental prototype presented in this section is driven by decision making scenarios described first. Next, we outline the choices made in our current experimental prototype to address the previously discussed information gathering issues.

4.1 Decision Making Scenarios

While there is an abundance of application-specific decision processes [16], [18], we performed performance study for the class of those decision processes that would involve processing supported by our developed image analysis software tools Im2Learn [19], GeoLearn [21] and I2K [20]. In order to quantitatively evaluate the CPU and storage costs of information gathering (CPU) and preservation (storage), we constructed a decision making scenario that is based on geospatial data, and the developed software tools to execute such a decision making scenario. Our tradeoff analysis study about the cost of gathering and preserving information is based on evaluating the developed software in the contrived decision making scenario. We devised the following contrived emergency response decision making scenario that would represent a class of government high-confidence decision making scenarios.

Example emergency response decision process: The government was notified that a foreign plant disease has been released by air in one of the states in the USA. The disease will spread in those parts of the country where elevation, slope, accumulation flow, precipitation, land surface temperature (LST) and soil type meet certain criteria. It is known that after the disease attacks plants, the vegetation changes its enhanced vegetation index (EVI), albedo, and leaf area index (LAI). The government makes a decision which geographical parts of the state should receive a financial aid to cope with the disease by spraying plants with appropriate chemicals. The decision is based on (1) clustering geographical raster data that represent variables known to promote disease dissemination and growth (called input variables), (2) developing a data-driven model between input variables and output variables that represent the observable changes in vegetation appearance, and (3) partitioning suspected geographical areas contaminated by the disease into man-defined boundaries, such as counties, zip codes or US Census Bureau blocks, in order to deliver the financial aid. A few decades later, a historian would like to reconstruct the emergency response decision making process. He or she would like to relate temporal human death statistics per county with the fact that the plant disease was or was not treated in certain geographical regions depending on the government financial aid.

While there are many challenging preservation issues in this scenario, we would like to quantitatively evaluate the cost of sufficient information gathering to document, preserve and reconstruct such a decision making process later in time. The cost is understood as the need for using high-performance computers with appropriate software tools and sufficient amount of CPU, storage and IO for information gathering, preservation and reconstructions.

Within this supported class of decision processes, we defined three types of decision processes, such as (1) data intensive, (2) computationally intensive and (3) high complexity decision processes. We believe that these three types of decision processes form distinct distributions of gathered information represented by information categories, such as (a) input data representation, (b) output data representation, (c) intermediate data, (d) user interface information, (e) sequence of processing operations, (f) processing environment characteristics, (g) human-centric information (e.g., user login) and (h) execution duration.

For example, data intensive decision processes would demonstrate large amounts of gathered information stored as input data and possibly as intermediate data. High complexity decision processes would be expected to lead to large file size labeled as sequence of operations, and as intermediate data, as well as very likely long process execution duration. Computationally intensive decision processes would be indicated by long execution duration and short sequence of operations.

4.1.1 Support of Decision Making Scenarios

To support the above class of scenarios, we have prototyped software tools that are able to execute the aforementioned emergency response decision making process according to Figure 6. The software algorithms can (1) ingest HDF EOS files since many of the input variables are disseminated by NASA in the HDF file format, (2) derive relevant features from raster elevation images, for example, slope, aspect, curvature, flow direction, accumulation flow [17] (see Figure 7), (3) integrate multiple raster files by adjusting their spatial resolution and re-projecting geographic projections in a common one, (4) visualize a stack of multiple integrated raster files and vector files representing various boundaries, (5) select boundaries for data-driven modeling, (6) develop decision tree based supervised model between input and output variables [24], [25], (7) visualize results and model residuals geographically, (8) cluster raster values based on the developed model or in an unsupervised fashion, (9) visualize the clusters and report statistics that are significant for the final decision making step, and (10) save the results. The decision process is limited to a linear workflow with ‘next’ and ‘back’ buttons defining the step of processing. One can think of the decision process user interface as a geospatial data analysis wizard (called GeoLearn [21]) as illustrated in Figure 8.

.

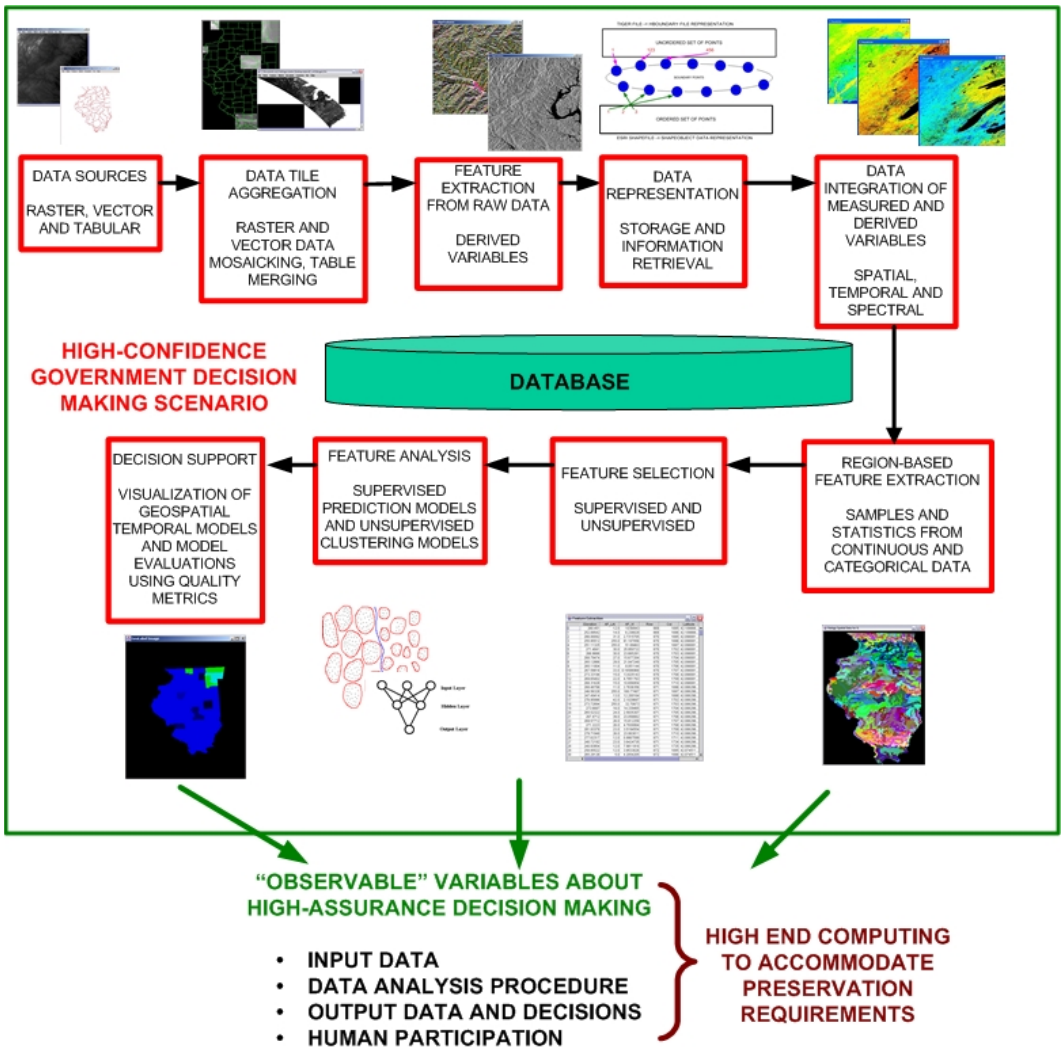

Figure 6: Illustration of the processing steps that take place in the contrived emergency response decision making process.

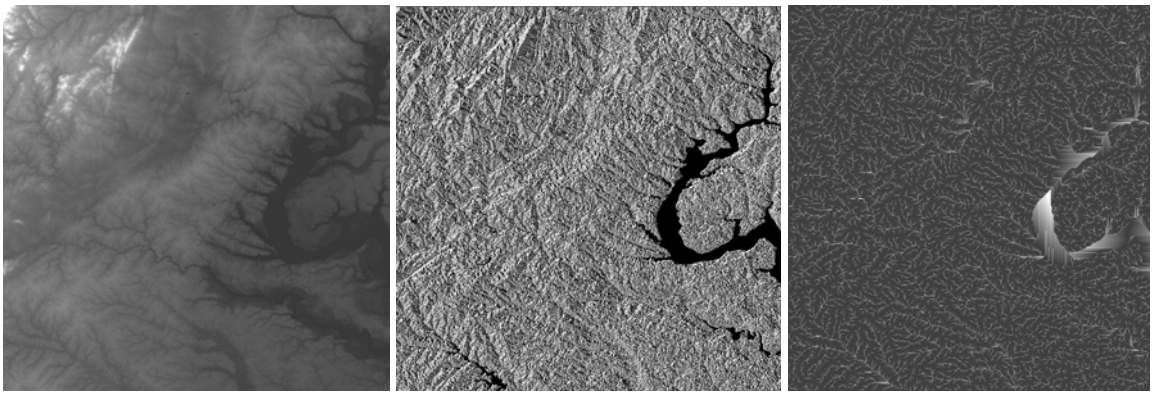

Figure 7: Feature extraction examples. Left – original digital elevation map; middle – aspect feature; and right – flow accumulation feature.

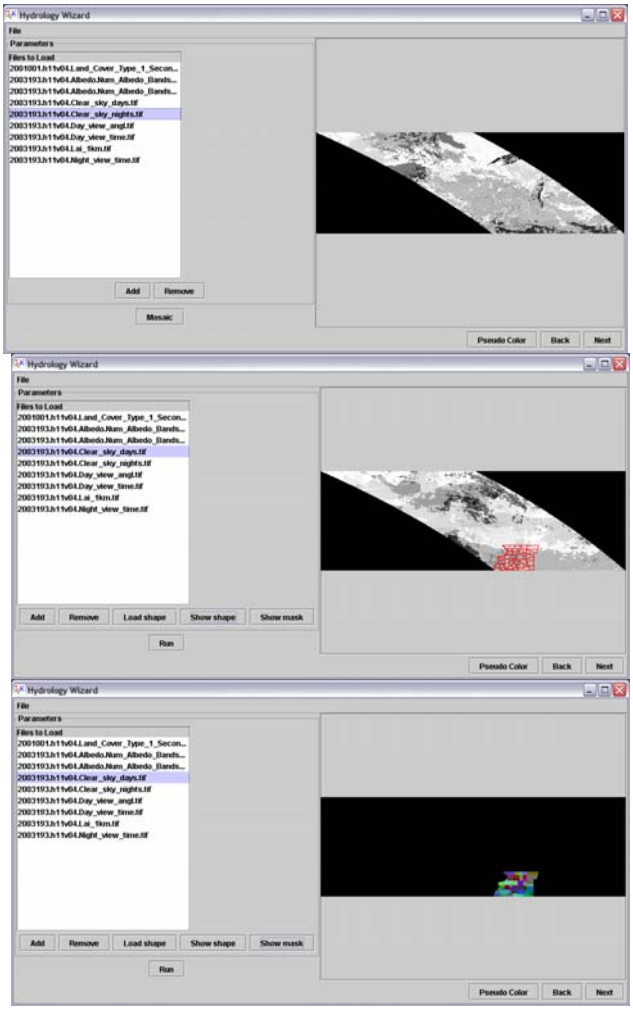

Figure 8: Screen captures of the developed geospatial data analysis. From top to bottom – loaded and re-projected raster file, vector boundary overlaid with raster data, and mask defining a pixel membership based on the boundary information.

4.1.2 Implementation of Information Gathering

Given the tools for executing the above scenarios, we focused on developing software mechanisms that would allow us to gather information about (1) all input data, (2) output data, (3) intermediate data, (4) image interaction information (sub-area selection, zoom operations, selection of image display from the stack of raster files, selection of boundary display from the stack of vector files, boundary selection), (5) parameter selection (selection of input and output variables for the decision tree plus all parameters), (6) all input, output and intermediate data that were shown to a decision maker (who knew what and when), (7) hardware and software information (version numbers and patches), (8) computation operations (what classes were called and executed), (9) video, audio and mouse movement during a decision making process, and (10) image drawings and text annotations. An overview of our information gathering system is presented in Figure 9.

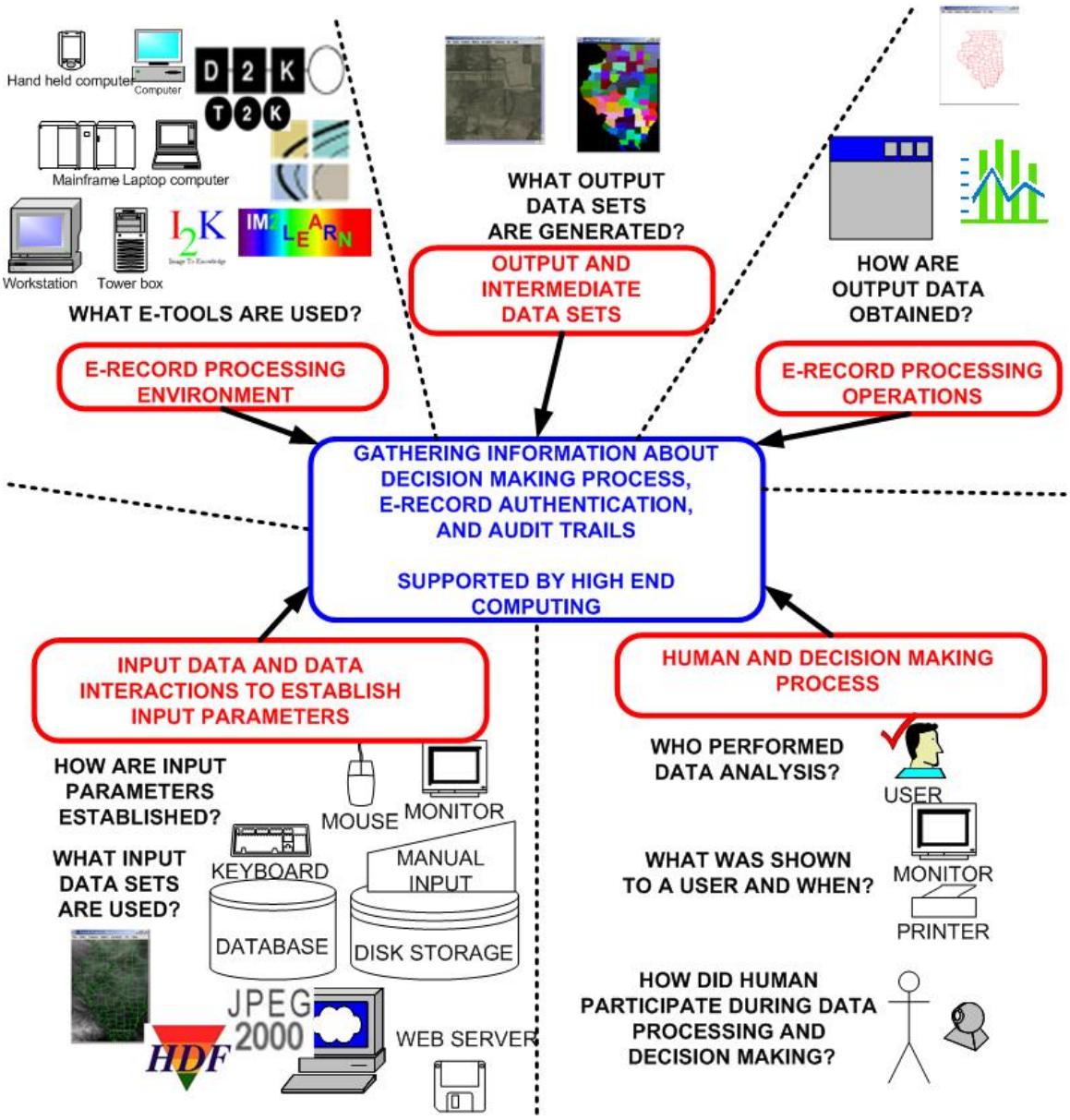

Figure 9: An information gathering system overview decomposed into several types of information.

4.2 Experimental Prototype

We present the implementation choices to answer questions about: (1) what information should be extracted about decision processes, (2) at what computer system level to gather information, (3) how to gather information at user program computer system level, and (4) how to save gathered information

4.2.1 What Information Should Be Extracted About Decision Processes

In our current work, we analyzed several levels of information granularity in order to answer the question “who knew what and when”. In the prototype implementation, we

focused on information categories, such as (a) input data representation, (b) output data representation, (c) intermediate data, (d) user interface information, (e) sequence of processing operations, (f) processing environment characteristics, (g) human-centric information (e.g., user login) and (h) execution duration. The recording options about gathered information are available for experimental evaluations as shown in Figure 10.

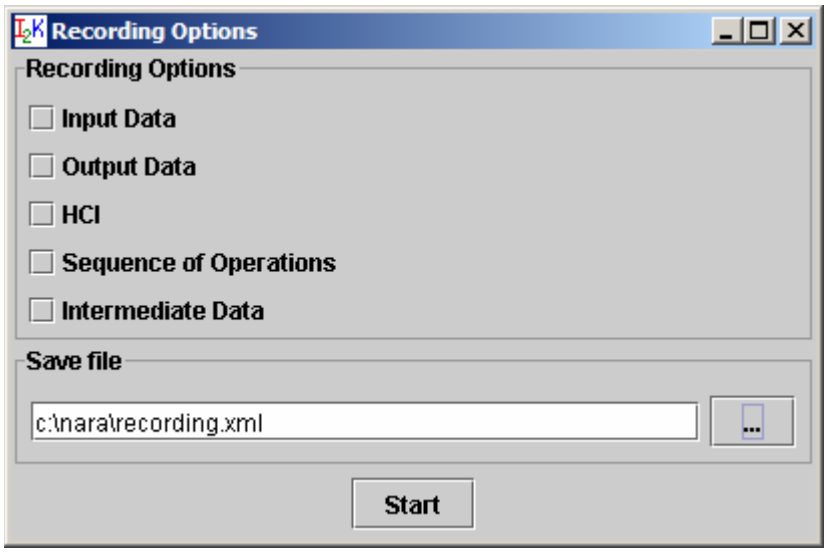

Figure 10: Recording options available for experimental evaluations. Recording evaluations will be performed with multiple combinations of information types about decision processes.

4.2.2 At What Computer System Level To Gather Information

Based on our available resource for this work, we decided to explore information gathering at the source code and middle layer entry points (see the analysis in Section 3.2).

4.2.3 How to Gather Information at User Program Computer System Level

We decided to use the following mechanisms for gathering information.

Table 1: Information gathering mechanisms

Information category	Gathering mechanism
Input Data Representation	Triggered by ‘Add’ and ‘Load’ button executions, saved data structures of unprocessed input data
Intermediate Data	Triggered by logging functions, saved data structures of partially processed input data

Sequence of Processing Operations	Triggered by any button, saved properties of operations
Human-centric Information (e.g., user login)	Triggered by 'Start" button, saved values from java System calls
Output Data Representation	Triggered by 'Save' button execution, saved data structures of fully processed input data
User Interface Information	Triggered by mouse and keyboard, saved events intercepted by Java listeners
Processing Environment Characteristics	Triggered by 'Start" button, saved values from java System calls
Execution Duration	Triggered by 'Start' and 'Stop' buttons, saved values from java System_current_time calls

The information gathering mechanisms presented in Table 1 are a combination of multiple approaches, such as (a) graphical user interface (GUI) monitoring approach, (b) the approach using a middle layer (linear workflow implementation), and (c) logging function approach.

4.2.4 How to Save Gathered Information

In our preliminary experiments, we stored all gathered information in XML files. In the near future, we plan to use the HDF file format or any other mechanism for semantic content storage.

Each information category in Table 1 is saved into a separate file. We plan on analyzing each file for further information statistics, for example, what percentage of user interface information was due to (a) mouse movement, (b) mouse clicks, and (c) keyboard strokes.

4.3 Preliminary Results: Cost of Information Preservation

We present the description of (a) data, (b) decision process, and (c) gathered information for the three types of decision processes separately in Appendices A, B and C. The quantitative preliminary results are summarized next in terms of the storage and CPU load.

Figure 11 illustrates the storage cost (file size in bytes) divided among the seven types of gathered information (information granules), such as input, output, intermediate, user interface (UI) events, sequence of operations, environment variables and user profile

variables. The three curves in Figure 11 correspond to the three classes of decision making processes (Data intensive, CPU intensive and high complexity). One can conclude that (a) the storage cost of the intermediate data is the largest, and (b) the storage cost of user profile, environmental variables, sequence of operations and user interface events is much smaller that the storage cost of input, output and intermediate data.

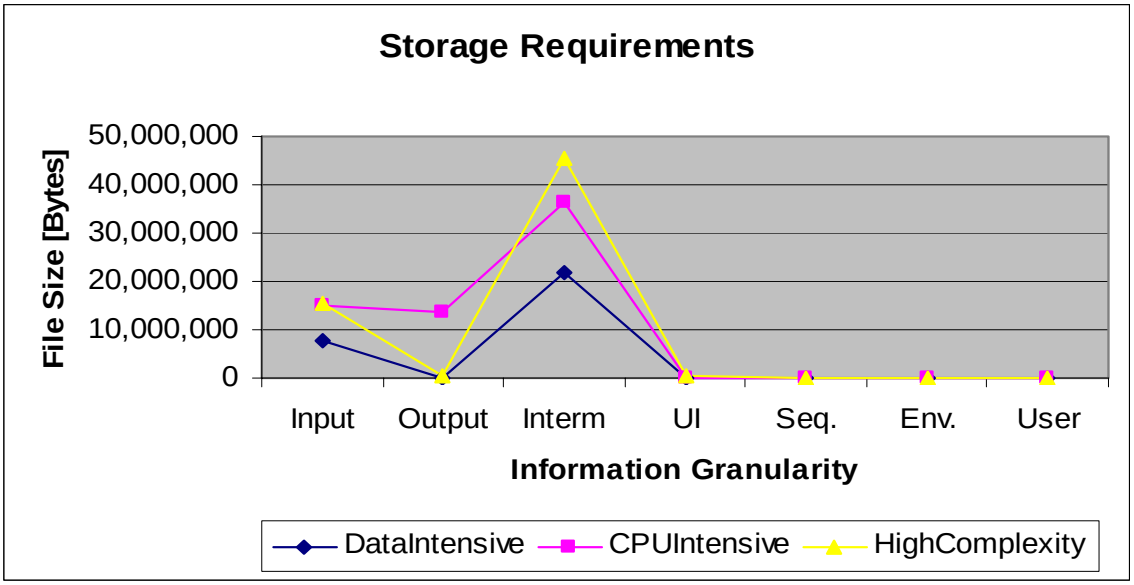

Figure 11: Storage requirements (file size in bytes) for the seven information granules, such as input, output, intermediate, user interface (UI) events, sequence of operations, environment variables and user profile variables.

It is apparent that gathering information about decision making process will take certain amount of CPU. Unfortunately, the experimental prototype written in Java does not allow us to collect information about the exact number of CPU cycles that have been utilized during execution because of the nature of the Java virtual machine. Codes written in Java can run on various architectures with different instruction sets. So, the number of CPU cycles needed to perform an operation can vary on different architectures. Thus, we tried to measure the elapsed time since the execution started to collect a very rough estimate of the CPU utilization (no other jobs were executed on the computer used during the experimental runs). Furthermore, we manually turned on and off those components of the experimental prototype that (a) monitored user interface interactions (called Event Listeners) and (b) recorded all gathered information into a hard drive. **Figure 12** shows the elapsed time for three on/off configuration combinations with Event Listeners and recording. The three curves in **Figure 12** correspond to the three classes of decision making processes (Data intensive, CPU intensive and high complexity). Based on **Figure 12**, one can quantify the computer performance cost due to information gathering (the difference between LisYesRecYes and LisNoRecNo). Furthermore, one can also demonstrate that the performance cost of event listening is much smaller that the performance cost of information recording (the difference between LisYesRecNo and

LisNoRecNo in comparison with the difference between LisYesRecYes and LisYesRecNo).

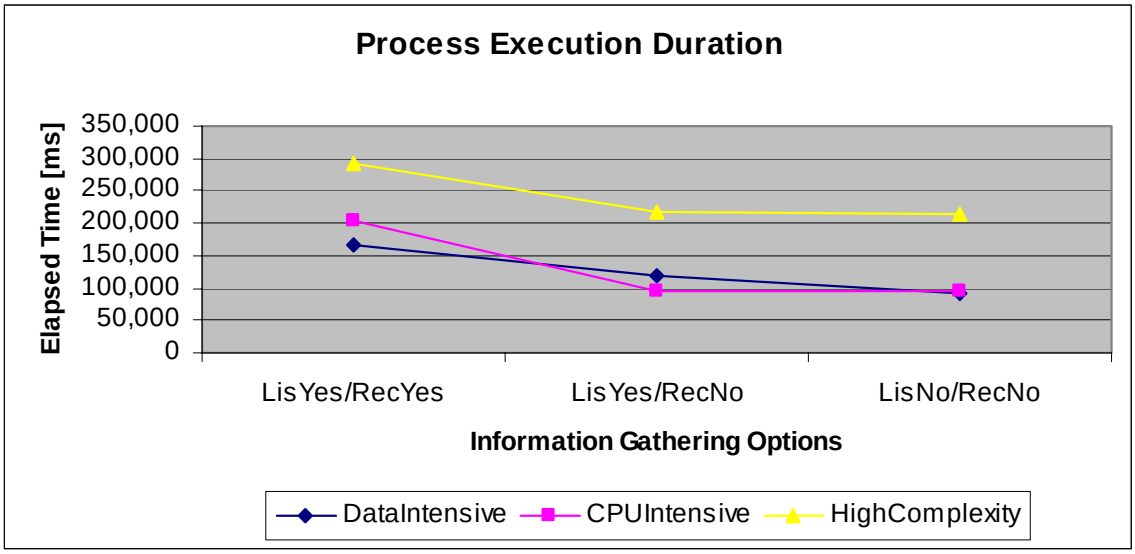

Figure 12: Process execution duration (elapsed time in ms) for the three monitoring options. LisYes/RecYes – Event Listeners are on and recording is on, LisYes/RecNo – Event Listeners are on and recording is off, and LisNoRecNo – Event Listeners are off and recording is off.

5. Discussion

Repeatability of experiments: The current experimental prototype allowed us to gather preliminary results about the cost of information about decision making processes using geospatial electronic records. We incorporated into our experimental prototype user interactions with software since the use of computers in high confidence decision processes is usually a very interactive process. However, while monitoring user interactions was added, the repeatability of our experiments was sacrificed. In all experiments, the duration of the entire decision process consists of the time for a user to select files and parameter, and the time to perform each processing step. Thus, the elapsed time represents only an approximate number since we could not replicate the user-driven selections when running the same sequence of steps with event listeners and recording option on or off. We believe that this problem could be addressed by using an environment where processing flows are formed by example so that the processing sequence of any decision making process can be saved and re-run.

Constraints on information gathering: When addressing the question "What information should be gathered about decision making processing using geospatial electronic records without any intervention to the current analysis and decision processes?" we encountered constraints about monitoring humans using cameras. While we have developed systems that would gather computer-centric and human-centric information [1], [2], the use of cameras in the government decision making processes is

constrained legally and the information cannot be gathered. Thus, the original question "what information should be gathered …" becomes "what information can be gathered …"

Software design to accommodate information gathering: It is prohibitively complex to achieve provenance by inserting logging functions into existing code. Provenance works best when the operations can be represented in a flow chart. In a flow chart the inputs, outputs, and parameters can be easily identified. Legacy codes do not necessarily adhere to this paradigm. Also, many pieces of software are delivered in a binary format. The insertion of logging functions into existing code requires the original source code and the ability to re-compile the code into an executable format. This is not always an option. In the cases where it is an option, it is better to design provenance into the software from the beginning rather than retrofitting it later.

A peculiar technical issue relevant to object-oriented languages manifested itself in the development of a provenance system. In the Java language there is a look-and-feel API that determines how a UI component will appear to the user. There are instances of classes that when executed are part of a different class hierarchy than when compiled. For example, in the code developed, a listener was used that processes each HCI event. The Java source code filtered these events based on the type of the classes (e.g., if instance of JButton then …, if instance of JComboBox then …). This was not sufficient, however, because the classes used to display these components on the screen are not necessarily equal to the classes used in the Java source code. In this case, the screen component of a javax.swing.JComboBox was javax.swing.plaf.metal.MetalComboBoxButton, which is a descendant of javax.swing.JButton. This component was then treated as a JButton instead of as a JComboBox, and the incorrect action was taken. This provenance issue occurs due to the nature of object-oriented programming. The discrepancy between run-time and compile-time classes needs to be understood when designing future provenance systems.

A future direction for this research could be to embed provenance into a workflow engine [40]. Workflow processing lends itself well to provenance. Each unit of processing can be thought of as a block in a flow chart. The inputs and outputs are well-defined, and the parameters should also be defined. The HCI components of the workflow architecture should be decoupled from the workflow engine, making it simpler to perform the two types of provenance. The Tupelo semantic content repository [50] is used by our currently developed workflow engine [40] to save provenance data. The provenance data are represented currently as triples with (subject, predicate and object) [51]. More elaborate taxonomy and ontology representations [52], as well mechanisms for interfacing the provenance meta-data [53], [54] would be considered in the future. In our current meta-workflow prototype, the triplets are extended with a timestamp and the name of the user that created the triplet. The editor will receive the provenance information and make it visible to an end user in the provenance view. While it is important to have unique persistent names for each object and action, we plan on mapping the terms to more user friendly titles for display. The current provenance view uses the following output format:

Time: Function [Subject (Engine), Predicate (action), Object (tool & input)] by User

Reconstruction of dynamic information: In any experimental run, the information gathered about the decision process can be divided into static and dynamic with respect to the decision execution period. While researching what dynamic information is feasible to gather, two distinct, non-overlapping types of provenance emerged: data and human computer interactions (HCI). Data provenance records the state of the data in the process, including the inputs, outputs, and parameters to a computational unit. Data provenance is needed to interpret the state of the user's data in a recorded process. HCI provenance is mainly concerned with the user's interaction with the program. HCI provenance records a timestamp and an action. For example, the quality assurance/quality control (QA/QC) tools described in Section 3.3 only handle HCI provenance issues.

It is insufficient to use only HCI to play back a recorded process. It does not take into account the variations in time needed to complete a unit of computation. There is no guarantee that the amount of time needed to process data is deterministic. For example, if a user starts a process with the duration of one minute and then interacts with the results when the process completes, the HCI recording will record the interaction needed to start the process, and then interacting with the results one minute later. If the recording is to be played back on a different machine, that one minute of processing time can vary due to differences in hardware and/or software. Then the timestamps of the user interface interactions will no longer be in sync with the processing, rendering it useless.

The union of data and user interaction provenance works best when they are thought of as a linear, non-overlapping flow. The HCI recordings should not overlap with the data recordings. Playback can accurately occur only if all variables dependent on processing time are removed. Thus, it is best to record the state of the data before and after each processing unit to remove the time spent processing altogether when playing back a recorded operation. This affects the duration of the original process and the amount of disk space needed as described in Section 4.3.

6. Summary

In this report, we presented preliminary hard numbers about the cost of preservation for three types of decision processes, such as data intensive, CPU intensive and high complexity decision processes. We built a prototype system that has significant storage and data management requirements, as well as high-performance access and retrieval requirements, and hence it provided a suitable test bed for performing our preliminary research study.

We have performed preliminary analyses of the issues related to (1) decision process scenarios, (2) what information should be extracted about decision processes, (3) at what computer system level to gather information, (4) how to gather information at user program computer system level, and (5) how to save gathered information.

By leveraging multiple funded projects and collaborations, we have implemented the following functionalities to support (a) government decision processes using geospatial electronic records and (b) gathering information about the decision processes:

- Raster data preprocessing (Feature extraction, QA/QC screening, Spatial and temporal resolution adjustment, Geographic coordinate adjustment, Mosaicking of spatial tiles)
- Raster and vector data integration (Geographic coordinate adjustment, Visualization and boundary selection, Mask formation for feature extraction)
- Data-driven predictive modeling (Feature selection, Decision tree modeling, Visualization)
- Spatial and temporal representation of predictive models (Data mapping, Visualization)
- Interpretation of predictive models (Analysis of predictive models, Visualization)
- Information gathering about inputs and outputs triggered by selected buttons and their implementations
- Information gathering about intermediate results triggered and executed by logging functions
- Information gathering about user interface information executed with Event Listeners
- Information gathering about sequence of processing operations, processing environment characteristics, human-centric information (e.g., user login) and execution duration triggered and executed by “Start” and “Stop” buttons.

In a summary, adding government decision process provenance to existing applications turns out to be a difficult problem with limited human resources. The extracted provenance information becomes often only the input and output of each application. In order to collect government decision process provenance information, new applications will have to be created with provenance gathering in mind. We believe that new workflow environments that have provenance built into their design, interface heterogeneous tools, accommodate information gathering not only about computation but also about user interactions and allow varying the granularity of gathered provenance information might be explored as the new software applications for harvesting provenance information.

In the future, we would like (a) to provide more hard numbers about the tradeoffs between the cost of preservation and the value of preserved information, (b) to continue our study of the decision-making process, (c) to do parallel performance studies of large datasets on both distributed and shared-memory systems, and (d) to continue investigation of techniques to improve ingestion of heterogeneous geospatial records at high accretion rates. We currently believe that the government decision process provenance

Appendix A: Evaluations of Data Intensive Decision Processes

Data Description:
Raster input: 9 files, cumulative raw file size: 24,300,574 bytes

Predicted raster output: 1 file, raw file size: 2,210,474 bytes
Raster input and predicted output file format: HDF EOS
Spatial coverage: 849 rows x 2595 columns, Tile: H11, V04 (see Figure 13 left)
Spatial resolution=0.0011785 degrees of lat or longitude per pixel
County of interest: Lee (see Figure 13 right)

Raster input variables:
(1) fraction of Photosynthetically Available Radiation (PAR) absorbed by the vegetation canopy (FPAR) "LAI/Fpar_1km",
(2) Land cover (LC) type 1 "LC/Land_Cover_Type_1_Secondary",
(3) land surface temperature (LST) clear sky days "LST/Clear_sky_days",
(4) land surface temperature (LST) clear sky nights "LST/Clear_sky_nights",
(5) land surface temperature (LST) Day view angle "LST/ Day_view_angle",
(6) land surface temperature (LST) Day view time "LST/ Day_view_time",
(7) land surface temperature (LST) Night view angle "LST/ Night_view_angle",
(8) albedo "albedo/Albedo_bands_01.Num_albedos_01",
(9) albedo "albedo/Albedo_bands_02.Num_albedos_01"
Night view time, day view time, day view angle, clear sky nights, clear sky days, land cover type, fraction of Photosynthetically Available Radiation (PAR) absorbed by the vegetation canopy (FPAR), albedo I, albedo II.

Raster output variable:
(1) Leaf Area Index (LAI) "LAI/Lai_1km",

Vector input: 1 file, raw file size: 1,100,000 bytes
Vector input file format: ESRI Shapefile
Vector input variable: boundary points of counties in Illinois defined in latitude and longitude

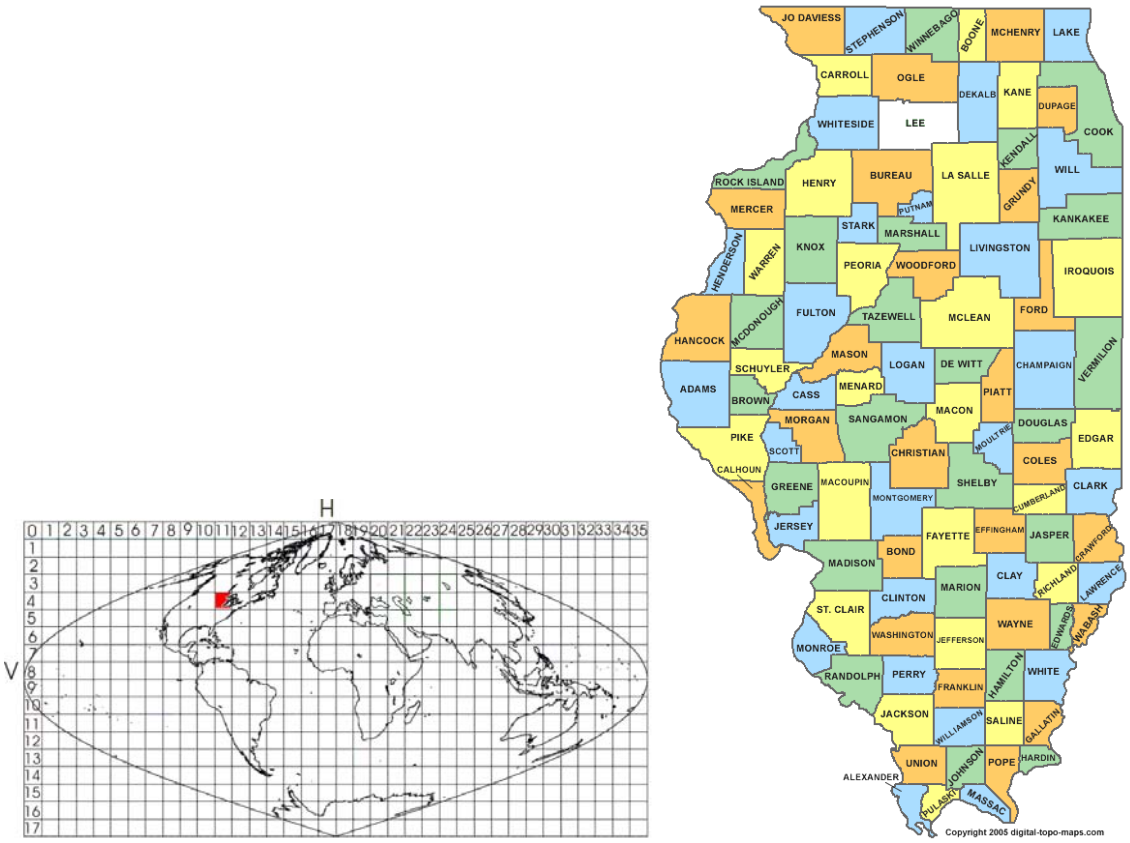

Figure 13: Left – spatial coverage shown by a red tile represented by HDF EOS files. Right –the selected area of interest is the white colored Lee county in Illinois defined by an ESRI Shapefile file with county boundaries.

Decision Process Description:

Step 1: Launch application, Open Recording Options, Start recording, Close recording options

Step2: Load raster input files, Load raster output feature to be predicted, View every loaded raster file, View in pseudo color every other raster file

Step 3: Integrate all raster files, View all integrated images, Load Vector input file, Integrate the stack of raster files with vector file and view an overlay of raster and vector files (button Show), Create mask image to assign county membership to each pixel, View mask, Select Lee county,

Step 4: Select input and output variables for prediction, Create table with all variables and all pixels in Lee county, Predict LAI from other variables using regression tree, View regression tree model, View spatial results, View spatial error, View table, Save results, from File menu Open Recording Options, Stop recording, Close Recording Options, Close application

Description of Gathered Information:

Table 2: Storage statistics collected from a data intensive decision process.

Information Type	File Size [bytes]	Description
Loaded data representation	7,849,608	Data structure representations of all loaded files
Saved data representation	16,512	Table, boundary image, prediction model, result image, error image
Intermediate data representation	21,809,640	Change view of data, pseudo-color view, integrated data, mask image
User interface operations	126,601	Mouse operation, keyboard strokes
Sequence of processing operations	2,500	All buttons (add, back, next, view, …)
Processing environment	6996	Computer hardware parameters, operating system, Java version, software package name and version
Human centric information	343	User login ID and profile

Table 3: Time statistics collected from a data intensive decision process.

Description	Elapsed Time (ms)
Listeners On Recording On	165,936
Listeners On Recording Off	119,651
Listeners Off Recording Off	93,236

Note#1: We did not record pseudo-colored images as output right now. If they would become selected as output then one pseudo-colored ImageObject adds to a file size about 631,913 bytes for this particular decision process scenario.

Note#2: The serialized object is Base64-encoded and therefore the loaded data representation is smaller than the cumulative raw file size.

Appendix B: Evaluations of Computation Intensive Decision Processes

Data Description:
Input: 1 file, raw file size: 1, 902, 642 bytes
File format: NASA SRTM
Spatial coverage: 1201 rows x 1201 columns, Tile: west longitude = 78, north latitude 38 (upper left corner)
Spatial resolution= the entire image represents 1 degree of latitude and longitude
Input variable: elevation
Extracted output variables: (1) slope, (2) curvature, (3) aspect, (4) flow direction, and (5) flow accumulation

Decision Process Description:
Step 1: Launch application, Open Recording Options, Start recording, Close recording options,
Extract Hydro Features, Select slope, Show slope, Close Show Window, Select Aspect, Show aspect, Close Show Window, Select Curvature, Show curvature, Close Show Window, Select Flow Direction, Show flow direction, Close Show Window, Select Flow Accumulation, Show Flow accumulation, Close Show Window, Close DEM Features window

Step2: Select one by one the extracted features and view them in pseudo color, from File menu Open Recording Options, Stop recording, Close Recording Options, Close application

Description of Gathered Information:

Table 4: Storage statistics collected from a computation intensive decision process.

Information Type	File Size [bytes]	Description
Loaded data representation	15,206,003	Serialized objects of all loaded files
Saved data representation	13,547,098	Table, prediction model,

		result image, error image
Intermediate data representation	36,321,305	Change view of data, pseudo-color view, integrated data, mask image
User interface operations	63,732	Mouse operation, keyboard strokes
Sequence of processing operations	1,885	All buttons (add, back, next, view, ..)
Processing environment	6996	Computer hardware parameters, operating system, Java version, software package name and version
Human centric information	343	User login ID and profile

Table 5: Time statistics collected from a computation intensive decision process.

Description	Elapsed Time (ms)
Listeners On Recording On	202,678
Listeners On Recording Off	94,623
Listeners Off Recording Off	95,496

Appendix C: Evaluations of High Complexity Decision Processes

Data Description:
Raster input: 10 files, cumulative raw file size: 43,110,000 bytes
Predicted raster output: 1 file, raw file size: 2,210,474 bytes
Raster input and predicted output file format: HDF EOS
Spatial coverage: 849 rows x 2595 columns, Tile: H11, V04 (see Figure 13 top)
Spatial resolution=0.0011785 degrees of lat or longitude per pixel
County of interest: Lee (see Figure 13 bottom)

Raster input variables:
(1) albedo “albedo/Num_Albedo_Bands_01.Num_Albedos_01”
(2) albedo “albedo/Num_Albedo_Bands_02.Num_Albedos_01”
(3) albedo “albedo/Num_Albedo_Bands_03.Num_Albedos_01”
(4) albedo “albedo/Num_Albedo_Bands_04.Num_Albedos_01”
(5) albedo “albedo/Num_Albedo_Bands_05.Num_Albedos_01”
(6) albedo “albedo/Num_Albedo_Bands_06.Num_Albedos_01”
(7) albedo “albedo/Num_Albedo_Bands_07.Num_Albedos_01”
(8) albedo “albedo/Num_Albedo_Bands_08.Num_Albedos_01”
(9) albedo “albedo/Num_Albedo_Bands_09.Num_Albedos_01”
(10) albedo “albedo/Num_Albedo_Bands_10.Num_Albedos_01”

Raster output variable:
(1) Leaf Area Index (LAI) “LAI/Lai_1km”,

Vector input: 1 file, raw file size: 1,100,000 bytes
Vector input file format: ESRI Shapefile
Vector input variable: boundary points of counties in Illinois defined in latitude and longitude

Decision Process Description:
Step 1: Launch application, Open Recording Options, Start recording, Close recording options,

Step 2: Load input raster files and the output raster variable , View every second loaded file in pseudo color

Step 3: Integrate all raster files, View Each integrated image, Load ESRI shapefile of county boundaries, View boundaries, Compute mask and show it, select Lee County, IL (Maroon mask color)

Step 4: Select input attributes (albedo/Num_Albedo_Bands_01.Num_Albedos_01 and albedo/Num_Albedo_Bands_02.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 5: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 6: Select input attributes (albedo/Num_Albedo_Bands_03.Num_Albedos_01 and albedo/Num_Albedo_Bands_04.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 7: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 8: Select input attributes (albedo/Num_Albedo_Bands_05.Num_Albedos_01, albedo/Num_Albedo_Bands_06.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 9: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 10: Select input attributes (albedo/Num_Albedo_Bands_07.Num_Albedos_01, albedo/Num_Albedo_Bands_08.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 11: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 12: Select input attributes (albedo/Num_Albedo_Bands_09.Num_Albedos_01, albedo/Num_Albedo_Bands_10.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 13: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 14: Select input attributes (albedo/Num_Albedo_Bands_01.Num_Albedos_01, albedo/Num_Albedo_Bands_02.Num_Albedos_01, albedo/Num_Albedo_Bands_03.Num_Albedos_01, albedo/Num_Albedo_Bands_04.Num_Albedos_01, albedo/Num_Albedo_Bands_05.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 15: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 16: Select input attributes (albedo/Num_Albedo_Bands_06.Num_Albedos_01, albedo/Num_Albedo_Bands_07.Num_Albedos_01, albedo/Num_Albedo_Bands_08.Num_Albedos_01, albedo/Num_Albedo_Bands_09.Num_Albedos_01, albedo/Num_Albedo_Bands_10.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 17: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step

Step 18: Select input attributes (albedo/Num_Albedo_Bands_01.Num_Albedos_01, albedo/Num_Albedo_Bands_02.Num_Albedos_01, albedo/Num_Albedo_Bands_03.Num_Albedos_01, albedo/Num_Albedo_Bands_04.Num_Albedos_01, albedo/Num_Albedo_Bands_05.Num_Albedos_01,

albedo/Num_Albedo_Bands_06.Num_Albedos_01,
albedo/Num_Albedo_Bands_07.Num_Albedos_01,
albedo/Num_Albedo_Bands_08.Num_Albedos_01,
albedo/Num_Albedo_Bands_09.Num_Albedos_01,
albedo/Num_Albedo_Bands_10.Num_Albedos_01) and output attribute (LAI/2003193.h11v04.Lai_1km.tif), compute regression tree

Step 19: View Boundary (default), View Model, View Result, View spatial Error, View table, Go back to previous step
from File menu Open Recording Options, Stop recording, Close Recording Options, Close application

Description of Gathered Information:

Table 6: Storage statistics collected from a high complexity decision process.

Information Type	File Size [bytes]	Description
Loaded data representation	15,592,207	Serialized objects of all loaded files
Saved data representation	284,149	Table, prediction model, result image, error image
Intermediate data representation	45,296,011	Change view of data, pseudo-color view, integrated data, mask image
User interface operations	265,294	Mouse operation, keyboard strokes
Sequence of processing operations	6378	All buttons (add, back, next, view, ..)
Processing environment	6996	Computer hardware parameters, operating system, Java version, software package name and version
Human centric information	343	User login ID and profile

Table 7: Time statistics collected from a high complexity decision process.

Description	Elapsed Time (ms)
Listeners On Recording On	291,390
Listeners On Recording Off	217,223
Listeners Off Recording Off	213,328

Acknowledgement

This research was partially supported by a National Archive and Records Administration (NARA) supplement to NSF PACI cooperative agreement CA #SCI-9619019. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the National Science Foundation, the National Archives and Records Administration, or the U.S. government.

References

[1] Lee Y-J. and P. Bajcsy, “An Information Gathering System For Medical Image Inspection,” Proceedings of SPIE Conference on Medical Imaging, PACS and Imaging Informatics, Vol. #5748, 12-17 February 1005, San Diego, CA.
[2] Lee Y-J. and P.Bajcsy. “Software Tools for Recording Image Inspection Processes,” Technical Report NCSA-ALG-04-0006, December 2004.
[3] The list of USGS Water Discipline data files:
http://water.usgs.gov/lookup/getgislist
[4] The current guidelines for archival description used at the National Archives:
http://www.archives.gov/research_room/arc/arc_info/lifecycle_data_requirements.doc
[5] John T. Phillips, “Implementing an Electronic Records Program,” ECURE 2000: Preservation and Access for Electronic College and University Records, October 5-6, 2000 Mesa, Arizona, ppt presentation available at URL:
http://www.infotechdecisions.com/downloads/ECURE%202000.PDF
[6] Robert Chadduck, “Overview and Update of NARA's Electronic Records Archive (ERA) Program,” Informal Remarks Presented to the Government Information Preservation Working Group, 14 October 2004, ppt presentation available at URL:
http://www.itl.nist.gov/div895/gipwg/oct04/Chadduck%20Oct04.ppt
[7] John Garrett and Donald Waters, “Preserving digital information: Report of the Task Force on Archiving Digital Information, May 1, 1996,
http://www.rlg.org/ArchTF/tfadi.index.htm
[8] InterPARES (International Research on Permanent Authentic Records in Electronic Systems) www.interpares.org/
[9] US-InterPARES Project, Findings on the Preservation of Authentic Electronic Records, final report to the National Historical Publications and Records Commission, September 2002.

[10] William Roberts, "Archiving Electronic Information", Tessella Support Services PLC, Issue V1.R2.M1, September 2004.
[11] Clinical Information Systems and Electronic Records and e-Health 2004, http://www.ehealthexpo.co.uk/pdf/programme.pdf
[12] DARPA's dreams, http://www.wired.com/news/business/0,1367,58909,00.html
[13] Bajcsy P., "Data Processing and Analysis," Section IV (pp. 258-378) of the book "Hydroinformatics: Data Integrative Approaches in Computation, Analysis, and Modeling," by Kumar P., J. Alameda, P. Bajcsy, M. Folk and M. Momcilo, published by CRC Press LLC, October, 2005, 534p.
[14] Clutter D. and P.Bajcsy. "Storage and Retrieval Efficiency Evaluations of Boundary Data Representations for LLS, TIGER and DLG Data Structures," Technical Report NCSA-ALG-04-0007, October 2004
[15] Clutter D. and P. Bajcsy, "Tradeoff Studies about Storage and Retrieval Efficiency of Boundary Data Representations for LLS, TIGER and DLG Data Structures," The 2005 Symposium on Document Image Understanding Technology at College Park, Maryland, November 2-4, 2005 (URL:http://lamp.cfar.umd.edu/meetings/SDIUT05/index.htm
[16] Kumar P., P. Bajcsy, D. Tcheng, D. Clutter, V. Mehra, W-W Feng, P. Sinha and A. White, "Using D2K Data Mining Platform for Understanding the Dynamic Evolution of Land-Surface Variables," the 2005 Earth-Sun System Technology Conference, University of Maryland, MD, June 28-30, 2005.
[17] Feng W. and Bajcsy P., "Extracting Topographic Features From Shuttle Radar Topography Mission (SRTM) Images,"Technical Report NCSA-ALG-05-002, July 18, 2005.
[18] Bajcsy P., "Supporting Decision Making Processes Based on Geographic Information," NCSA PSP meeting on May 16, 2005, URL: http://www.ncsa.uiuc.edu/Conferences/2005Meeting/agenda.html.
[19] Bajcsy, P., Kooper, R., Clutter, D., Lee, S-C., Ferak, P. (2005), Image to Learn user manual, Tech report ALG-2005-005, URL: http://i2k.ncsa.uiuc.edu/Im2Learn/Im2LearnManual_v1.pdf.
[20] Bajcsy P. et al. (2003), Image to Knowledge user manual, http://i2k.ncsa.uiuc.edu/i2kmanual/
[21] GeoLearn user manual (software tools for data-driven modeling from remote sensing data) http://i2k.ncsa.uiuc.edu/GeoLearn/GeoLearn_userGuide_v8.pdf
[22] Alumbaugh T.J. and Bajcsy P., "Georeferencing Maps with Contours in I2K," ALG NCSA technical report, alg02-001, October 11 2002.
[23] Bajcsy P. and Alumbaugh T.J., "Georeferencing Maps with Contours," accepted to the 7th world multi-conference on systemics, cybernetics and informatics, July 27-30, 2003, Orlando, FL.
[24] Han, J., and Kamber, M., 2001. "Data Mining: Concepts and Techniques, Morgan Kaufmann Publishers, San Francisco, CA.
[25] Duda, R., P. Hart and D. Stork, 2001. Pattern Classification, Second Edition, Wiley-Interscience.
[26] Groth P., L. Moreau, and M. Luck. Formalising a protocol for recording provenance in grids. In Proceedings of the UK OST e-Science Third All Hands Meeting, Nottingham, UK, Sept. 2004. http://citeseer.ist.psu.edu/groth04formalising.html

[27] Buneman, P., Khanna, S., Tan, W.: Data provenance: some basic issues. In: Foundations of Software Technology and Theoretical Computer Science, 20th Conference, (FST TCS) New Delhi, India. Volume 1974 of Lecture Notes in Computer Science., Springer (2000) 87–93
[28] Buneman P., S. Khanna, K.Tajima, and W. Tan. Archiving scientific data. In Proc. of the 2002 ACM SIGMOD International Conference on Management of Data, pages 1–12. ACM Press, 2002.
[29] Groth P., M. Luck, and L. Moreau. A protocol for recording provenance in service-oriented grids. In Proceedings of the 8th International Conference on Principles of Distributed Systems (OPODIS'04), Grenoble, France, Dec. 2004.
[30] Miles S., P. Groth, M. Branco, and L. Moreau. The requirements of recording and using provenance in e-science experiments.Technical report, University of Southampton, 2005.
[31] Lanter D., Design of a lineage-based meta-data base for GIS. Cartography and Geographic Information Systems, 18(4):255–261, 1991.
[32] Groth P., S. Miles, W. Fang, S. C. Wong, K-P. Zauner, and L. Moreau. Recording and Using Provenance in a Protein Compressibility Experiment. In Proceedings of the 14th IEEE International Symposium on High Performance Distributed Computing (HPDC'05), July 2005
[33] da Silva, P. Pinheiro, D. L. McGuinness and R. McCool. Knowledge Provenance Infrastructure. IEEE Data Engineering Bulletin Vol.26 No.4, pages 26-32, December 2003. http://citeseer.ist.psu.edu/pinheirodasilva03knowledge.html
[34] Pancerella C., J. Myers, L. Rahn, "Data Provence in the CMCS," Workshop on the Collaboratory for Multiscale Chemical Science, 2005
[35] Talbott T., M. Peterson, J. Schwidder, J. D. Myers, "Adapting the Electronic Laboratory Notebook for the Semantic Era," Proceedings of the 2005 International Symposium on Collaborative Technologies and Systems (CTS 2005), May 15-20, 2005, St. Louis, MO
[36] Gilliland-Swetland A. J., "Enduring Paradigm, New Opportunities: The Value of the Archival Perspective in the Digital Environment," published by Council of Library and Information Resources, Washington DC, February 2000, p.37
[37] Myers J.D., C. Pancerella, C. Lansing, K. Schuchardt, B. Didier, "Multi-scale Science: Supporting Emerging Practice with Semantically-Derived Provenance," Semantic Web Technologies for Searching and Retrieving Scientific Data Workshop at the 2nd International Semantic Web Conference Oct. 20, 2003, Sanibel Island, FL
[38] Moreau, L., Chen, L., Groth, P., Ibbotson, J., Luck, M., Miles, S., Rana, O., Tan, V., Willmott, S. and Xu, F. (2005) Logical architecture strawman for provenance systems. Technical Report, ECS, University of Southampton.
[39] Bose R. and J. Frew, (2005) "Lineage Retrieval for Scientific Data Processing: A Survey," ACM Computing Surveys, Vol. 37, No. 1, March 2005, pp. 1–28.
[40] Bajcsy P., R. Kooper, L. Marini, B. Minsker and J. Myers, "A Meta-Workflow Cyber-infrastructure System Designed for Environmental Observatories," Technical Report: NCSA Cyber-environments Division, ISDA01-2005, December 30, 2005.
[41] Kooper R, L. Marini, J. Myers and P. Bajcsy, "A Meta-Workflow System Designed for Solving Complex Scientific Problems using Heterogeneous Tools," the 2006 Winter

Federation of Earth Science Information Partners ("Federation") Conference, poster, January 4-6, 2006 in Washington, DC.
[42] Bajcsy P and D. Clutter, "Gathering and Analyzing Information about Decision Making Processes Using Geospatial Electronic Records," the 2006 Winter Federation of Earth Science Information Partners ("Federation") Conference, poster, January 4-6, 2006 in Washington, DC.
[43] Data to Knowledge (D2K) and D2K Streamline Software Description, Automated Learning Group, NCSA, UIUC, URL: http://alg.ncsa.uiuc.edu/do/tools/d2ksl
[44] Greenwood M., C. Goble, R. Stevens, J. Zhao, M. Addis, D. Malvin, L. Moreau, T. Oinn, "Provenance of e-Science Experiments – experience from Bioinformatics," In Proceedings of the UK e-Science All Hands Meeting. Nottingham, UK. 223–226., EPSRC e-Science Pilot Project myGrid, http://www.mygrid.org.uk
[45] S. Cherry, "Total Recall," IEEE Spectrum, November 2005, pp. 25-30. (also see http://research.microsoft.com/users/GBell/)
[46] Drucker, P.F. Management: Tasks, Responsibilities, Practices. New York: Harper and Row, 1974, 466-467.
[47] Cowings J. S., "Strategic Leadership and Decision Making," the Industrial College of the Armed Forces (ICAF), http://www.au.af.mil/au/awc/awcgate/ndu/strat-ldr-dm/cont.html
[48] CLEANER (Collaborative Large-scale Engineering Analysis Network for Environmental Research), URL: http://cleaner.ncsa.uiuc.edu/home/
[49] Virtual Data System (formerly Chimera and including Pegasus), tools for expressing, executing, and tracking the results of workflows, URL: http://www.globus.org/grid_software/computation/chimera.php; http://vds.uchicago.edu/twiki/bin/view/VDSWeb/WebMain
[50] Tupelo, data and metadata archiving system supporting object-oriented metadata schemas based on RDF-OWL, logical file naming, version and access control, and Grid services, URL: http://dlt.ncsa.uiuc.edu/wiki/index.php/Main_Page
[51] Resource Description Framework (RDF) , http://www.w3.org/TR/rdf-concepts/#section-Graph-Literal
[52] Semantic Web for Earth and Environmental Terminology (SWEET) http://sweet.jpl.nasa.gov/ontology/
[53] Pfister R., R. Ullman, and K. Wichmann, "ECHO Responds to NASA's Earth Science User Community", available at URL: http://www.prism.washington.edu/bloodstream/NASA-ECHO.html
[54] ECHO: Earth observing system ClearingHOuse, URL: http://www.echo.eos.nasa.gov/