Data Conservancy CyberInfrastructure Early Development Case Study

Virgil E. Varvel Jr., Ph.D.

Center for Informatics Research in Science and Scholarship
Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign

This study funded in part by the Data Conservancy (NSF Award Number OCI-0830976)

CIRSS

Data Conservancy Case Overview and Focus

Under the Office of Cyberinfrastructure (OCI), the National Science Foundation (NSF) solicited applications for 'Sustainable Digital Data Preservation and Access Network Partners (DataNet) funding. One collaboration among several sponsored under this award is the Data Conservancy (NSF DataNet Award OCI0830976). The purpose of this collaboration is:

> "Through collection, preservation, and semantic integration of data that are now very difficult to assemble and analyze, the Data Conservancy will transform the ability of scientists to answer grand challenge questions that are important to the nation and the world. The Data Conservancy will research, design, implement, deploy, and sustain data curation infrastructure for cross-disciplinary discovery with an emphasis on observational data. Initial efforts of the project will highlight astronomy, earth sciences, life sciences, and social sciences." (http://dataconservancy.org/)

The Data Conservancy (DC) will promote access to data across disciplines and provide opportunities for new and bigger questions through cross-dataset queries. Data is the currency of science. "To be able to exchange data, communicate it, mine it, reuse it, and review it is essential to scientific productivity, collaboration, and to discovery itself." (Gold, 2007) The larger DC vision entails scientific data curation as a means to collect, organize, validate, and preserve data so that researchers can address research challenges facing society as a whole. The DC addresses its goals through a comprehensive program comprising four inter-dependent objectives and teams: Infrastructure Research and Development (IRD), information science and computer science research (IS/CS), broader impacts (BIDC), and sustainability (SDC). The DC infrastructure development occurs in an iterative manner with testing and prototyping of systems informed directly by user-centered design and information science and computer science research. At every stage of development, the DC engages domain scientists, generates substantive broader impacts and focuses on sustainability planning. Furthermore, the initial DC prototyping efforts were successfully demonstrated during their eighteen month NSF review.

This case study focuses on lessons learned during the DC architecture and infrastructure development in its initial phases from 2009-2011. This study explores how the IRD team functioned, what the architecture underlying the DC infrastructure was, what compelling issues presented themselves during the project, and how those issues were resolved in terms of the context and reasoning of decisions made in the initial phases of the DC infrastructure development.

Method

This DC study is presented as a descriptive case study where the purpose is to elucidate the core facets of the case and make the unfamiliar familiar following a phenomenological approach outline by Stake (1995). Two data sources used included interviews of the present and past IRD team leaders and document analysis of the contents of the IRD team wiki. The primary data analysis was one of narrative structuring including when decisions and issues occurred and who was involved. This structure was used to supply a first person account of how various issues affected the infrastructure development and was chosen because the various technical and social issues, which were a central focus to the infrastructure

development, could be used to structure the account. To cross analyze the interview data, meaning condensation was employed to determine themes within the organized structure and text of the IRD wiki. The text within these themes was recontextualized in terms of the issues involved in this case. All issues were grounded in the interview data. Following this analysis, the current IRD team leader provided additional insight into the architecture for inclusion in the final draft written as a narrative around the primary contextual issues.

The Data Conservancy CyberInfrastructure Case

Infrastructures in General

Infrastructures are not separate systems unto themselves, but rather conglomerations of systems that enable loci of control and maintenance over subordinate systems that interoperate transparently. Historically, infrastructures are seen as conglomerations that should become ubiquitous, accessible, reliable, and increasingly transparent as they mature. Gateways within the larger system allow for the dissimilar smaller systems to be linked through standardization and integration. Modern infrastructures are a form of second order large technical systems (LTS) that allow a digital convergence where better information technology increases the capacity for distributing control from a central point to the nodes throughout the network.

Social, technological, and organizational issues help to shape infrastructure development. In some ways, infrastructure development is determined by organizations acting as social agents in determining technological change (Edwards, *et al.*, 2007). Tensions within infrastructure development can therefore include such issues as time scales, data scales, and the other needs of the agencies and systems involved. Technology related issues can include scaling digital content transfer, management, and storage processes, particularly as they increase in quantity, size, and diversity (Littman, 2009). These processes include acquiring, quality review of, moving, inventorying, storing, and manipulating or transforming content. No single process or application programming interface (API) can be used for every scenario. Differences include the ordering of steps, the presence or absence of steps, and what function performs a task when and by whom. Workflow services enable the management of these complex processes.

CyberInfrastructure

The DC is not just an infrastructure, but a cyberinfrastructure. There are many ways to define cyberinfrastructure. Atkins *et al.* defined cyberinfrastructure as a comprehensive and integrated system of hardware, software, networks, and social practices supporting a range of activities including computation, storage, and communication over computer networks: “cyberinfrastructure refers to infrastructure based upon distributed computer, information, and communication technology. If infrastructure is required for industrial economy, then we could say that cyberinfrastructure is required for a knowledge economy.” (2003, pg. 5) The NSF’s vision sees the complementary areas that make up cyberinfrastructure as computing systems, data information resources, networked, digitally enabled-sensors, instruments, virtual organizations, and observatories, along with an interoperable suite of software services and tools (NSF, 2007).

NSF saw the need for supporting human-centered, sustainable, stable but extensible, petascale resource sharing in science and engineering. As such, the grand challenge of cyberinfrastructure is to "provide continued access to an authentic copy of electronic data in perpetuity," according to the former project lead for the DC IRD team. One difficulty in this challenge is the realization that electronic data is growing in quantity exponentially while also diversifying in size and format of data files. Furthermore, curation of digital data entails managing multiple encodings of data and its provenance, dealing with increasingly complex data types, handling complex relationships within and across data, mitigating against risk of loss of data and unwanted access to data, obsolescence of data type, and even changing user interactions with data. The DC cyberinfrastructure is formulated in terms of supporting the lifecycle of data in all of its complexities, running a wide range of services upon and with regards to that data, and connecting that data with people across many domains with an emphasis on preservation and reuse of data.

The DC Infrastructure Research and Development (IRD) Team

Infrastructure development involves many interwoven issues including social, technological, organizational, economical, and political aspects, and the DC infrastructure development was no exception. The development of the DC cyberinfrastructure was tasked to the IRD team. The IRD team included partners from Johns Hopkins University, Cornell University, the National Snow and Ice Data Center, the University Corporation for Atmospheric Research, the Marine Biological Laboratory, DuraSpace, Portico, and Tessela. Since the IRD team was comprised of institutionally and geographically separated partners, social and organizational issues could be expected. The digital nature of the project, costs involved, and National Science Foundation funding added technological, economical, and political issues.

Initial socio-organizational issues associated with infrastructure development were addressed by the IRD team at a first meet-and-greet session in Ithaca, New York in 2009. In some way, the dissociated members of the IRD team needed to come together physically and metaphorically as a team and discuss the issues that they faced and begin to make decisions they knew would impact the future developments of the project. According to the current head of the IRD team, the long list of decisions that needed to be made included:

- who would be involved in what aspects of the infrastructure development and at what resource level;
- what software stacks and tools would be utilized by the infrastructure IRD team to write code and manage the build process of the technical architecture;
- the processes of task and issue tracking and management;
- what particular source code repository would be used;
- agreements on implementation frameworks and software development guides;
- best practices and principles to follow in configurations; and
- how priorities would be set and to what extent resources would be allocated to certain tasks.

While most infrastructures develop over time as other systems interface, DC was being conceptualized from the beginning to meet goals of sustainable digital data access and preservation. Therefore, the IRD

team was able to make decisions initially with these goals in mind. Through scholarly discussions whereby each partner outlined their expertise and available options from possible alternatives, decisions were made such as using Maven for managing the build process, using Subversion as the source code repository and Jira for task management. The primary programming language for the entire DC software stack including the Application Programming Interface (API) was also chosen as Java. Each decision included implications too such as acceptance of open source versus proprietary formats to retain open coding and ease of access and feature integration but increased need for security and front-end development. A key was to build on the strengths present within the team and to keep building.

It was important to continually bring the group together face-to-face twice a year at the start of each six-month development cycle as well to set new broad goals and priorities for the group and to re-center their focus. Although task prioritization, organization, and management were the primary purposes, these meetings also provided a social outlet for the group. Large bucket tasks that were set at these meetings included priorities such as the Feature Extraction Framework. For each aspect, or component, of the infrastructure framework, code had to be designed and written. The meetings helped to decide who would work on each component, who would lead the team, who would be responsible for given tasks, and who could report to the overall team. Each component maintained a lead inside the IRD team to shepherd the process to fruition. These six-month meetings also allowed the group to look beyond the single projects in a broad sense to set overall project goals in the long term such as cyber security big picture views. All of the various pieces of the infrastructure had to continually fit together.

The communication within the IRD team has been constant through weekly information calls as well. These calls have enabled updates as well as the introduction of new topics and ideas that percolate between different development teams within DC. Sometimes these may lead to high level, core topic meetings such as discussions about whether the team is using the right data model.

The Data Model

The IRD had a mandate to have a prototype system by the end of the first year of development. The former IRD team lead saw settling on a data model as part of that system as one of the primary hurdles of the IRD team partly due to the time constraints and due to finding a model upon which the whole team could agree. A model was required that could embrace the evolution of the system, support long-term preservation, have the ability to support the implementation of control services over the preservation system, and accommodate different domains to insure preservation and curation of content and secondarily to procure buy-in from the scientific community that would be using the system. Elements within the system needed to be coordinated across disciplines to facilitate discovery. The team already had in-roads with the Planets Consortium through their first team leader, so they leveraged this connection and chose a simplified version of the mature Planets data model that supported long-term preservation as the basis for the DC data model.

The main purpose of the PLANETS model is to describe the actual preservation objects that need to be preserved or to explicitly define different types of information objects in need of preservation actions within an interoperability framework (http://www.planets-project.eu/about/). PLANETS explicitly identifies four types of information objects with different levels of aggregation as well as three types of

digital objects with different levels of aggregation. These information objects tie into the Open Archival Information System (OAIS) model, thus tying into the overall architecture of the system. The PLANETS conceptual data model includes several additional features of note: separation of logical and physical objects (or conceptual and physical views), allowances for several manifestations of objects, and many-to-many migration accommodations of concrete parts such as files and bitstreams (Farquhar & Hockx-Yu, 2007). The main conceptual entities present within the DC year-one prototype are shown in Figure 1. This model as implemented by DC has not undergone any major changes within their current architecture. However, the Planets data model was developed around a primarily document-based data standard and needed to be adapted to a more robust and data-oriented environment. The simplified DC version of the PLANETS model is still comprised entirely of PLANETS entities, but they have introduced novel relationships between those entities. To a simplified version of the Planets model was accreted their own digital attribution called a deliverable unit, providing their notion of a manifestation, which is the technical representation of that entity, and a notion of the digital objects, which are the bit streams of technical representations.

The DC have a number of current uses that stretch the data model, specifically around versioning of content, metadata handling, and certain forms of data (i.e. databases). The current Application Programming Interfaces (APIs) and data model may therefore have to change at some point to accommodate the scientific use cases and needs. One of the attractions of the DC approach with a simplified version of the Planets model is that they can evolve the model as their use cases and needs change. Small changes have occurred with the model over time already, highlighting a need to apply versioning to the data model. Continued adherence to the PLANETS model could reflect an example of satisficing behavior whereby early adoption of the one model is interfering with optimization (Edwards, *et al.*, 2007), but it is equally as possible that alternative models have not yet presented themselves as superior.

The Architecture

The overarching guidelines of the DC integrated within infrastructure development in general, i.e., they wanted to put in place a reliable, modular architecture with small, loosely coupled services that could be controlled through a central mechanism as needed. Therefore, numerous architectural considerations were taken into account in the design of the DC architecture. The system needed capacity and scalability to grow and link to other systems while handling a large number of objects of varying sizes and types. In data curation terms the system needed to handle large volumes and types of data and metadata while taking into consideration varying data complexity. Complexity within data as well as complex data interconnections needed to be accounted for. The system needed to be evolvable so as to allow new services to attach as available and old services to improve without affecting the methods of invocation. Flexibility was important to allow services to work together and to maintain a policy neutral approach.

Not only were they concerned with linking to other systems, but they also believed that integrating or embedding DC in scientific systems and workflows were a necessary strategy for success. Since what works for one community may not work for another, standards needed to be considered in terms of data, metadata, data models, and interchange.

The architecture could not be file-centric, for the content of the file was also important; however, a valid criticism of the current DC model is that it tends to be file-centric. Challenges early in the DC project with preserving databases validated the importance of understanding the need to understand the features within a data object. To help interoperability, an open design avoiding proprietary lock-in and dependencies was preferred. Performance and availability needed to be maintained at necessary levels as well as an acceptable level of redundancy and failover.

In practice, DC needed a range of curation services for data and metadata across the whole life cycle of the data. DC also needed to enable the content to be discoverable and decipherable. Lower level services needed to be in place for data ingestion and migration, content extraction, and quality checking and semantics, and features were extracted from the content to support key integrations (i.e. spatial and temporal properties). At the top level, APIs defined the contract for other systems and other components of the cyberinfrastructure allowing them to use the service, for the data needed to be connected with communities of users. Each framework or API then was as independent as possible from each other.

With all of these needs, the Open Archival Information System (OAIS) was used as an abstract reference framework to help define the functional components and to inform concrete implementation of the DC architecture. The OAIS framework includes terminology and concepts for describing and comparing architecture components and system requirements. It does not necessarily tell one how to build the preservation system, but it was used as a guide to the bookings and services needed by the DC. OAIS was useful for providing a shared understanding of archive semantics, functionality, and actors while focusing on long-term access and preservation. It afforded a service-oriented architecture with well-defined APIs and loosely coupled, minimally dependent system components. These principles were adhered to throughout the DC to facilitate interoperability and promote sustainability thus allowing independent evolution and extension of DC modules while also reducing potential risks of adhering to a single service or system approach should it fail.

A block architecture diagram of the DC envisioned how the various aspects of the DC architecture tied together and could be reframed into OAIS (See Figure 2). In OAIS, a data object is interpreted using its representation information (all of the information that maps a data object into more meaningful concepts) to yield an information object that is aggregated into information packages. Ingest of an information package can include its acceptance by the system, performing quality assurance, generating fixity, extracting metadata, or generating an archival information package. In DC, these functions are managed through a central ingest service accessed through public ingest API's. OAIS archival storage involves physically storing, maintaining, and retrieving archival information packages. Within DC, archival storage API's and associated services run as a separate service with separate and cloud based dataset storage linked to other services through a client. OAIS access involves both the discovery and access to records and archives. Access is handled by DC in the same manner as ingest but with different APIs acting on behalf of the consumer as opposed to the producer.

From a physical and computating standpoint, the DC architecture was designed with a variety of components. DC applications were considered external to DC services. Applications communicated

with DC over HTTP using DC public APIs. Since most DC applications were Java based, a typical DC application required Java and a servlet container such as Tomcat. There were two DC user interfaces; one for deposit, and one for access. Since the software for each was similar, these were collocated on a single machine inside a single Tomcat instance. DC services related to process orchestration, ingest API and services, storage API and services, search and access API and services, and other common services could run on a single instance requiring Java, Tomcat, PostgreSQL for databases, Lucene for indexing, and characterization tools such as JHOVE along with standard tools including virus scanning. Batch ingest machines within DC could be command line or web driven. Hosts of the archival storage API and associated services were linked to the DC services though http.

OAIS provided the right set of functions while being a well understood and adopted framework suitable for adaption to DC. The overall modular design was fundamental to overcoming barriers to infrastructures in general and those specific to this project such as time for prototyping. The key was the high degree of integration among the various components. Individual infrastructure components allowed the interaction of architecture elements and the accomplishment of DC goals. Producers/Consumers needed to be able to switch between data publication systems and the data itself in a system that supported data deposit, ingest, access, management, and archiving too.

The Data Unit

Linking technical and conceptual issues is the determination of what the actual data unit is within the data conservancy (and in any scientific endeavor for that matter), as the DC deals with data from a wide range of sources and types that must be treated in a wide range of manners for ingest, manipulation, and delivery. At this time, collaboration among research groups within DC has been focused on this issue and no clear answer has yet been reached, but much research has been centered on a feature extraction framework that addresses an extrapolation of the data unit. Technological issues such as the data model and feature extraction cut across the entire IRD team and how the entire DC system functions. Services within the system utilize the data model and ultimately define the management of data within any infrastructure today or historically. Since the system must accommodate a variety of data formats, no assumptions can be made regarding data input or output, and processes cannot be strictly coupled to specific execution modules. These accommodations also require data granularity and scalability to facilitate reuse, analysis, subsetting, indexing, etc. While the IRD team chose the data model and its implementation, the data unit is an issue addressed by the entire DC team, particularly the IS/CS Data Concepts team. As they have shown, while a shared terminology framework is needed, no common understanding of basic concepts of a dataset yet seems to exist with wide ranging definitions including content, grouping, purpose, and relatedness (Renear, Sacchi, & Wicket, 2010). The DC continues therefore to focus on a clearer delineation of the data unit within the DC architecture.

The Prototype

Although not discussed as an issue in the interviews, but present as a primary element in the IRD wiki and a primary output in the initial IRD development cycle, early IRD work was influenced by the need to produce a working prototype system. Initial prototyping efforts through Dry Valleys data illuminated the complex socio-technical dimensions of infrastructure development, particularly as it relates to multi-

disciplinary fourth paradigm science or data-intensive scientific discovery. The social dimension related to the interplay between those producing the system and the scientists wanting to use the system and providing the data. On the technical dimension, diverse data types from physical objects to images and text required diverse curatorial tasks and processes, which had associated costs that needed to be taken into account.

Sociological Aspects

While technological issues could be overcome with careful planning and professional programming, Elliot Metsger made a point to cite sociological aspects as a primary issue facing the IRD team in the DC infrastructure development. He separated the issue into three aspects: (1) serving the user's needs, (2) inter-team communication, and (3) intra-team communication.

For instance, as Elliot stated, "We are not writing the DC code base for IT people but to serve the needs of the scientific and archival community, and that is the challenge." Any system is created to serve the needs of a given user, and the DC users make up a diverse community with equally diverse needs that may not be fully understood by the technical community creating the interface. Pilot system testing and working with experts from the target communities have been methods used to help alleviate this issue.

Integration of other DC teams in order to get their feedback into the IRD process to date has mostly been ad hoc and informal; however, it has been seen as important and purposeful to the project design to get this feedback for the most effective system's design. They have tried to lower barriers to communication such as requiring the use of a management system like Contour to the use of emails and wiki pages, but have not arrived at a universally applied mechanism as of yet save inter-team meetings.

Finally, the IRD team had no formalized process or structure to their design, implementation, and process documentation. Although it was present on a wiki, there existed issues with production, organization, and consistency. While they hoped to hire someone to formally take control of the process, they still needed to formally define the expectations of the process. There existed a large amount of documentation on the wiki that could be difficult for an outsider to decipher, leaving no good way to quickly learn the technical or other aspects of the DC infrastructure.

Conclusions

Investigation of the DC infrastructure development revealed social, organizational, and technological influences on the Infrastructure Research and Development team in the early development phase. Each of these influences impacted the others while still clearly categorizing themselves. Early team development, management and decision structures, and cohesion structured the social and organization dynamics of the group while laying the foundations of the infrastructure technical framework and development. Communication was a noticeable factor in several directions including inter-DC, intra-IRD, and between DC and the science community that helped shape infrastructure evolution. Technical issues included selection and manipulation of the data model, defining the data unit, and the myriad issues involved in the actual architecture vision. Keeping in mind an historical infrastructure model, the IRD team's overall infrastructure architecture was based on flexibility, openness, and modularity with a

decentralized design. As Aaron Birkland stated, “It became clear early on that the framework should specifically avoid being designed around any particular execution model. Scalability, analysis, and computing services were positioned as an area of evolving DC research. [We] wanted to embrace that, not stifle it.” (2011, Feature Extraction Framework Presentation) The IRD was able to successfully demonstrate their first year prototype and continue to development new aspects of the infrastructure as they move ahead with the DC project. As with all infrastructures, development takes time, with complex multimodal issues involving many stakeholders that the IRD teams have so far been successful at overcoming as they move towards a defined vision.

Looking forward, new changes in the mandates put forward by the NSF will afford new political influences upon the IRD team towards more production and artifact output that may alter the parameters set forth in this case study. Further analysis at the end of the project could prove interesting in terms of how changes put forth by policy, including new requirements for biweekly reporting and greatly inter-team and inter-institutional communication, will influence the IRD. Furthermore, the potential for technological and conceptual changes such as data model shifts and new code development and projects as the DC develops into a working model will prove interesting, especially in terms of the interactions among scientists and programmers. Future case study analysis in other avenues should continue to address these areas.

References

Atkins, D. E., Droegemeier, K. K., Feldman, S. I., Garcia-Molina, H., Klein, M. L., Messerschmitt, D. G., Messina, P., Ostriker, J. P., & Wright, M. H., (2003, January). *Revolutionizing science and engineering through cyberinfrastructure: Report of the National Science Foundation blue-ribbon advisory panel on cyberinfrastructure*. Arlington, VA: Office of Cyberinfrastructure, National Science Foundation. Retrieved April 19, 2011, from http://www.nsf.gov/od/oci/reports/toc.jsp

Consultative Committee for Space Data Systems, (2002, January). *Reference model for an open archival information system (OAIS)*. (Reference Manual CCSDS 650.0-B-1.) Washington, DC: National Aeronautics and Space Administration. Retrieved April 19, 2011, from http://public.ccsds.org/publications/archive/650x0b1.PDF

Edwards, P. N., Jackson, S. J., Bowker, G. C., & Knobel, C. P. (2007, January). Understanding infrastructure: Dynamics, tensions, and design. (Final Report of the Workshop "History & Theory of Infrastructure: Lessons for New Scientific Cyberinfrastructures). Washington, DC: National Science Foundation. Retrieved April 19, 2011, from http://deepblue.lib.umich.edu/bitstream/2027.42/49353/3/UnderstandingInfrastructure2007.pdf

Farquhar, A., & Hockx-Yu, H. (2007). Planets: Integrated services for digital preservation. *International Journal of Digital Curation, 2*(2), 88-99, Retrieved September 29, 2011, from http://www.ijdc.net/index.php/ijdc/issue/view/3

Gold, A. (2007, September/October). Cyberinfrastucture, data, and libraries, part 1. *D-Lib Magazine, 13*(9/10). Retrieved April 19, 2011, from http://www.dlib.org/dlib/september07/gold/09gold-pt1.html

Littman, J. (2009, January/February). A set of transfer-related services. *D-Lib Magazine, 15* (1/2). Retrieved April 19, 2011, from http://www.dlib.org/dlib/january09/littman/01littman.html

National Science Foundation Cyberinfrastructure Council (2007, March). *Cyberinfrastructure vision for 21st century discovery.* (Report NSF 07-28). Arlington, VA: Author.

Planets (2010, July). *An emerging market: Establishing demand for digital preservation tools and services.* (Vendor White Paper V4:Layout 1). Author. Retrieved April 19, 2011, from http://www.planets-project.eu/docs/reports/Planets-VENDOR-White-Paperv4.pdf

Renear, A. H., Sacchi, S., & Wickett, K. M. (2010, October). Definitions of dataset in the scientific and technical literature. *Proceedings of the 73rd ASIS&T Annual Meeting on Navigating Streams in an Information Ecosystem*, *47*, Retrieved July 9, 2011, from http://www.asis.org/asist2010/proceedings/proceedings/ASIST_AM10/submissions/240_Final_Submission.pdf

Stake, R. E. (1995). *The art of case study research*. Thousand Oaks, CA: Sage Publications.

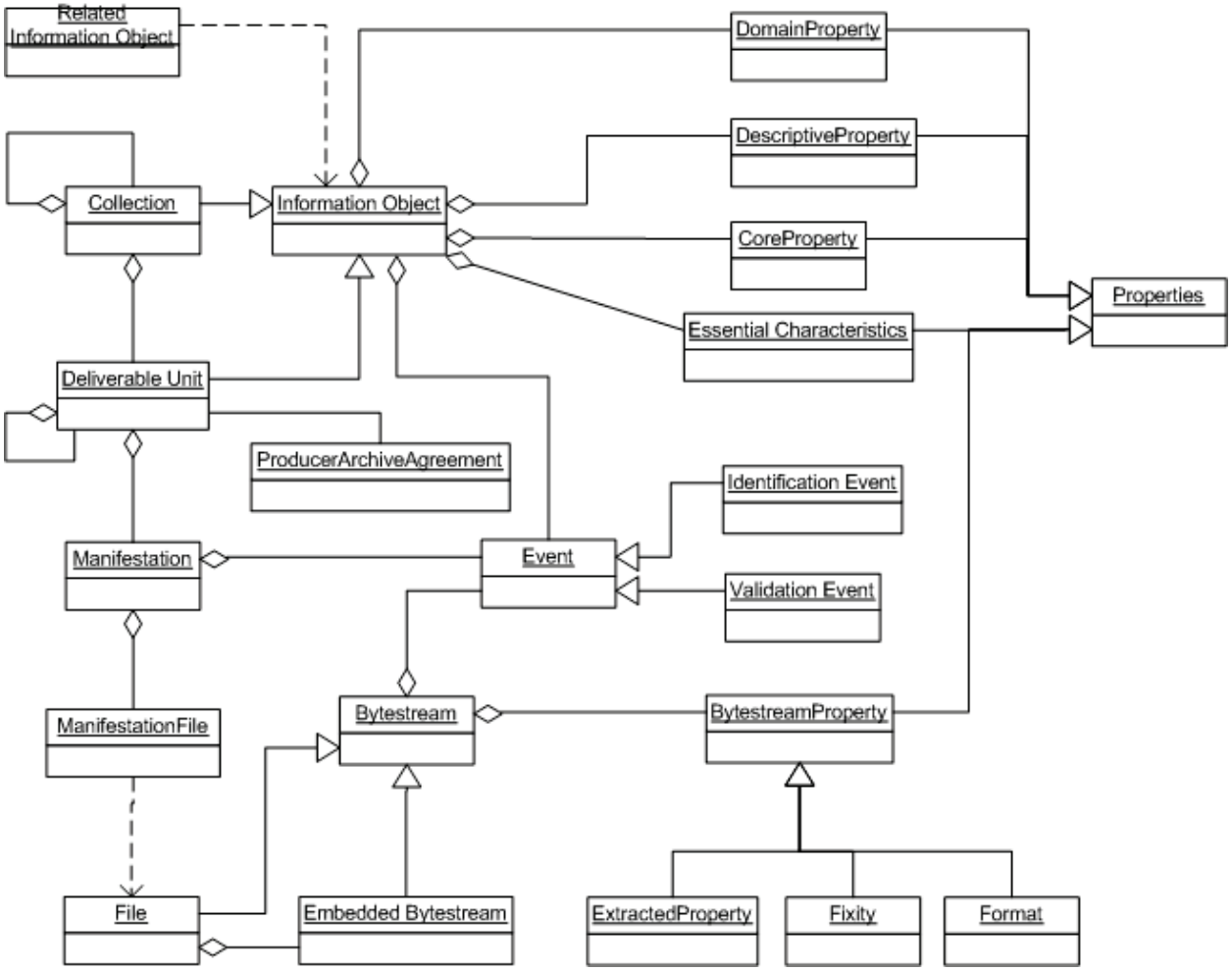

Figure 1: Data Conservancy early conceptual data model. (Image courtesy of Data Conservancy)

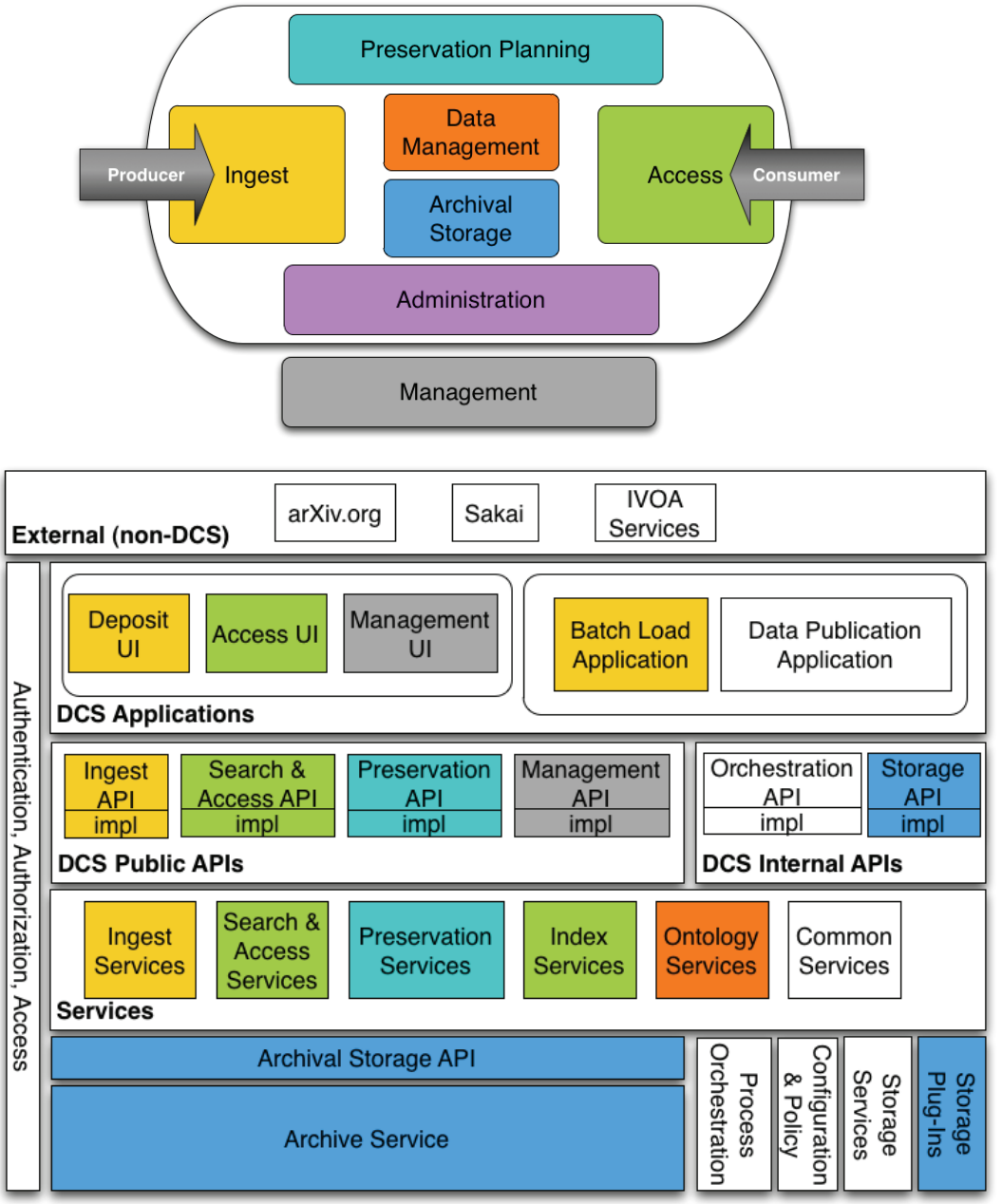

Figure 2. Data Conservancy block architecture diagram mapped to OAIS Reference Model. (Image courtesy of Data Conservancy)