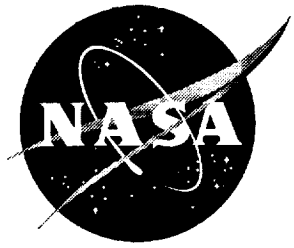
NASA Technical Memorandum 112877



Buckets, Clusters and Dienst

Michael L. Nelson
Langley Research Center, Hampton, Virginia

Kurt Maly and Stewart N. T. Shen
Old Dominion University, Norfolk, Virginia

July 1997

National Aeronautics and
Space Administration
Langley Research Center
Hampton, Virginia 23681-0001

Buckets, Clusters and Dienst

Michael L. Nelson
NASA Langley Research Center
MS 157A
Hampton, VA 23607
m.l.nelson@larc.nasa.gov

Kurt Maly
Old Dominion University
Computer Science Department
Norfolk, VA 23529
maly@cs.odu.edu

Stewart N. T. Shen
Old Dominion University
Computer Science Department
Norfolk, VA 23529
shen@cs.odu.edu

Abstract

In this paper we describe NCSTRL+, a unified, canonical digital library for scientific and technical information (STI). NCSTRL+ is based on the Networked Computer Science Technical Report Library (NCSTRL), a World Wide Web (WWW) accessible digital library (DL) that provides access to over 80 university departments and laboratories. NCSTRL+ implements two new technologies: cluster functionality and publishing "buckets". We have extended the Dienst protocol, the protocol underlying NCSTRL, to provide the ability to "cluster" independent collections into a logically centralized digital library based upon subject category classification, type of organization, and genres of material. The concept of "buckets" provides a mechanism for publishing and managing logically linked entities with multiple data formats. The NCSTRL+ prototype DL contains the holdings of NCSTRL and the NASA Technical Report Server (NTRS). The prototype demonstrates the feasibility of publishing into a multi-cluster DL, searching across clusters, and storing and presenting buckets of information. We show that the overhead for these additional capabilities is minimal to both the author and the user when compared to the equivalent process within NCSTRL.

1 Introduction

In aerospace engineering, Multidisciplinary Design Optimization (MDO) is a growing field concerned with the integrated design and analysis of applications using a combination of mathematical, engineering, and economic models and tools. Surpassing its early analysis and optimization roots, MDO now also includes "functions of interdisciplinary communication" [21]. To hasten the desired adoption of MDO methodology by other engineering research communities such as electrical and chemical engineering [21], NASA is acutely interested in the creation of a unified, canonical digital library of Scientific and Technical Information (STI).

Spurred by recent advances in network information systems such as the World Wide Web (WWW), digital libraries (DLs) are the topic of research in many scientific communities. However, digital library

projects are partitioned by both the discipline they serve (computer science, aeronautics, physics, etc.) and by the format of their holdings (technical reports, video, software, etc.). A recent survey found over 10 existing or recent different WWW-oriented digital library projects spanning over 5 different disciplines [4]. In short, each scientific community is hand crafting their own digital library infrastructure.

There are two significant problems with current digital libraries. First, multidiscipline research is difficult because the collective knowledge of each discipline is stored in incompatible DLs that are known only to the specialists in the field. In the MDO example above, current DL practices leave no good method for the structural engineers to be aware of computer or mathematical tools developed by another discipline that might be relevant to their research. The second significant problem with digital libraries is that although technical information created consists of manuscripts, software, datasets, etc., the manuscript receives the majority of attention, and the other components are often discarded [22]. Past format restrictions have forced an artificial partitioning of the STI output along format lines (software tapes, photo negatives, printed reports, etc.). Although non-manuscript digital libraries such as the software archive *Netlib* [2] are successful, they still placed the burden of STI reintegration on the customer. NASA customers desire to have the entire set of manuscripts, software, data, etc. available in one place [20]. With the increasing availability of all-digital storage and transmission, the re-integration of the STI output back to its original state is possible.

NASA Langley Research Center and Old Dominion University have established NCSTRL+ to address both of these problems. NCSTRL+ is based on the Networked Computer Science Technical Report Library (NCSTRL) [3], which is a highly successful digital library offering access to over 80 different university departments and laboratory since 1994, and is implemented using the Dienst protocol [11]. NCSTRL+ will initially include selected holdings from the NASA Technical Report Server (NTRS) [16] and NCSTRL, providing *clusters* of collections along the dimension of disciplines such as aeronautics, space science, mathematics, computer science, and physics, as well as clusters along the dimension of publishing organization and genre, such as project reports, journal articles, theses, etc. NCSTRL+ holdings will be published in *buckets* [18], an object-oriented construct for creating and managing collections of logically related information units as a single object. A bucket can contain both different data

syntax (PostScript, PDF, Word, etc.) and different data semantics (manuscripts, data files, images, software, etc.)

The outline of the rest of the paper is as follows: section 2 provides a discussion of DL background material. Sections 3 and 4 introduces clusters of Dienst servers and buckets. In section 5 we discuss how NCSTRL+ is used from both the searcher and author's perspective. Section 6 discusses the architecture and implementation of NCSTRL+. Section 7 discusses the current status and future work, and we conclude in section 8.

2 Background

NCSTRL+ has a long lineage (Figure 1). In 1992, the ARPA-funded CS-TR project began [6] as did the Langley Technical Report Server (LTRS) [17]. In 1993, the Wide Area Technical Report Server (WATERS) [13] shared a code base with LTRS. In 1994, LTRS launched the NTRS, and the CS-TR and WATERS projects formed the basis for the current NCSTRL. In 1997, NTRS and NCSTRL formed the basis for NCSTRL+.

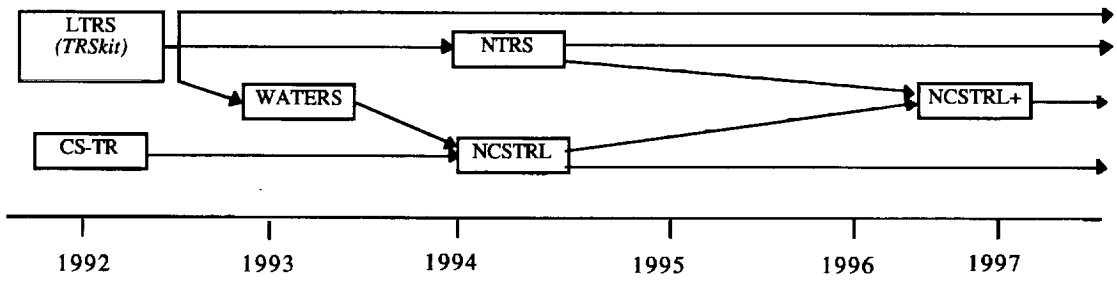


Figure 1: NCSTRL+ Lineage

We chose to implement NCSTRL+ using Dienst instead of other digital library protocols such as TRSkit [19] because of Dienst's success in several years of production in NCSTRL. Dienst appears to be the most scalable, flexible, and extensible of digital library systems we surveyed [4]. Dienst also serves as the basis for other digital library projects, including: the Electronic Thesis and Dissertation Project [5], the University of Virginia undergraduate engineering thesis project [23] and the ACM SIGIR conference proceedings project (which requires ACM authentication) [1].

While Dienst is discipline independent, it is currently discipline monolithic. It makes no provision for knowledge of multiple subjects within its system. While it is possible to set up a collection

of Dienst servers independent of NCSTRL, there is no provision for linking such collections of servers into a higher level meta-library. Dienst consists of 5 components: 1) Repository Service; 2) Index Service; 3) Meta-Service; 4) User Interface Service; and 5) Library Management Service. Dienst names objects in collections using the CNRI Handle system [7]. Meta-data for objects is stored in the RFC-1807 format [12].

Our buckets are similar in concept to the "digital objects" first proposed in [8]. It is important to note that many services have had "proto-buckets" in operation for some time, including the NACA Report Server [15] and NCSTRL. In both of the above servers, each logical entity in the archive actually consisted of a "wrapper" entity providing access to multiple formats of the same manuscript (PostScript, scanned images, PDF, etc.). However, each of the above servers provide only different formats of a single manuscript, or in the case of NCSTRL it also supports the concept of separate pages within a manuscript. But neither supports an interface to a collection of related objects such as the manuscript, software, datasets, etc. We chose the term "buckets" because related terms such as "objects", "packages" and "containers" are greatly overloaded in the computer science realm and because "buckets" provide a clear visual metaphor for the concept when speaking with non-computer scientists.

3 Clusters of Dienst Servers

Clusters are a way of aggregating logically grouped sub-collections in a DL along some criteria. NCSTRL already has a single default cluster of publishing authority, which in practice generally maps to the author's organization. NCSTRL+ provides 3 clusters: organization, data genre, and subject category (see Figure 7 for an example). *Genre* is a term provided by E. Fox in a private communication and refers to distinguishing between journal articles, technical reports, theses and dissertations, etc. For the purposes of this paper, we illustrate the concept of clusters by discussing the subject category cluster. Other clusters are implemented similarly.

Dienst currently carries no concept of subject category in its protocol, despite having provisions for specifying keywords from the title, authors, and abstract. In fact, digital libraries using the Dienst

protocol such as NCSTRL have the implicit assumption that all holdings are computer science related. We propose to modify Dienst by providing *subject* arguments to existing message verbs (Table 1).

Service	Message Verb	Argument	Argument Type
Index	List-Contents	subject=	optional
Index	SearchBoolean	subject=	optional
Meta	Publishers	subject=	optional
Meta	Indices	subject=	optional
Meta	Repositories	subject=	optional
Meta	Lite	subject=	optional
User Interface	Search	none; would modify default output to include subject selector	N/A
User Interface	QueryNF	subject=	optional
User Interface	BrowseYears	subject=	optional
User Interface	ListYears	subject=	optional
User Interface	BrowseAuthors	subject=	optional
User Interface	ListAuthors	subject=	optional
Library Management	ListSubjects (proposed)	none	none
Library Management	DescribeSubjects (proposed)	none	none

Table 1: Proposed Modifications of Dienst 4.0

The proposed message verb modifications are to be used to demonstrate the concept of subject category based server cluster functionality. The term "clusters" for this purpose is due to Carl Lagoze, who in a private communication proposed a new Dienst service that would provide a separate cluster service allowing the creation of clusters of Dienst servers along arbitrary criteria. The new clustering service will solve the general case of the problem, where our Dienst modifications will support the specific clustering around subject categories in the early stages of the NCSTRL+ prototype. The purpose of our cluster prototype is to perform experiments with an initial set of clusters and determine user response.

For the NCSTRL+ prototype, we adopted the NASA STI subject categories. A full listing can be found in [14]. The NASA STI topics are attractive since they are familiar with the majority of our customer base, and they also provide over 100 subtopics while producing only a small number of high level topics (Table 2).

AERONAUTICS	LIFE SCIENCES
ASTRONAUTICS	MATHEMATICAL AND COMPUTER SCIENCES
CHEMISTRY AND MATERIALS	PHYSICS
ENGINEERING	SOCIAL SCIENCES
GEOSCIENCES	SPACE SCIENCES

Table 2: NASA STI Main Topics

NCSTRL+ reads its known subject categories from a preference file, so future augmentation or replacement of this list should not be difficult. The NASA STI topics are not meant to replace an institution's use of any subject specific categories, such as the ACM CR categories. Rather, NCSTRL+ will maintain a mapping of how various specialized classification schemes map into the larger NASA STI topics (Figure 2). The NASA STI topics for NCSTRL+ will be implemented as a new optional and repeatable field in RFC-1807 format.

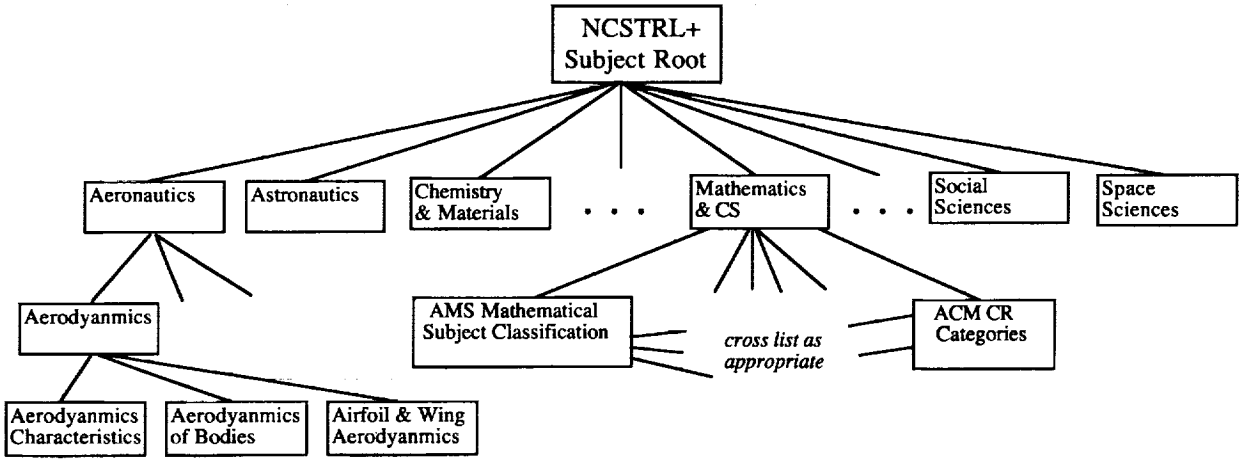


Figure 2: NCSTRL+ Subject Tree

4 Buckets

We define buckets as a construct for creating publishing and archival entities for digital libraries. A bucket corresponds to a single logical collection of information. Buckets are designed to be highly customizable and unique. It would be possible for large archives to not have any buckets with exactly the same functionality. Not all bucket types or applications are known at this time. However, we can describe a generalized buckets as containing many formats of the same data item (PS, Word, Framemaker, etc.) but more importantly, it can also contain collections of related non-traditional STI materials (manuscripts, software, datasets, etc.) Thus, buckets allow the digital library to address the long standing problem of ignoring software and other supportive material in favor of archiving only the manuscript [22] by providing a common mechanism to keep related STI products together. A single bucket can have multiple *packages*.

Packages can correspond to the semantics of the information (manuscript, software, etc.), or can be more abstract entities such as the metadata for the entire bucket, bucket terms and conditions, pointers to other buckets or packages, etc. A single package can have several *elements*, which are typically different file formats of the same information, such as the manuscript package having both PostScript and PDF elements. Figure 3 illustrates the architecture of a typical bucket.

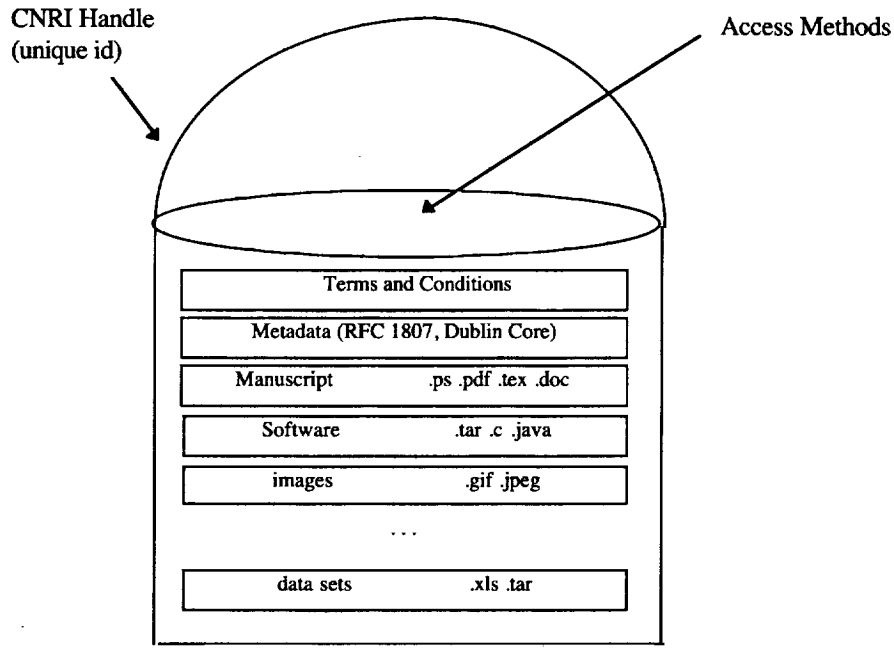


Figure 3: A Typical Bucket Architecture

4.1 Bucket Requirements

Buckets are intended to be either standalone objects or to be placed in digital libraries. They have unique ids (CNRI handles) associated with them. Buckets are intended to be useful even in repositories that are not knowledgeable about buckets in general, or possibly just not about the specific form of buckets. Buckets should not lose functionality when removed from their repository. The envisioned scenario is that NCSTRL+ will eventually have moderate numbers of (10s - 100s of thousands) of intelligent, custom buckets instead of large numbers (millions) of homogenous buckets. Figure 4 contrasts the traditional architecture of having the repository interface contain all the intelligence and functionality with that of the architecture possible with buckets where the repository intelligence and functionality can be split between the repository and individual buckets. This could be most useful when individual buckets require custom terms and conditions for access (security, payment, etc.). Figure 4 also illustrates a bucket gaining some

repository intelligence as it is extracted from the archive en route to becoming a standalone bucket. Table 3 lists some additional bucket requirements, and Table 4 lists the required bucket methods. Note that Table 4 differs from protocols such as the Repository Access Protocol (RAP) [10] in that we have defined actions buckets perform on themselves, not actions a repository performs on buckets. Although the two are not mutually exclusive, our current plan is to not implement RAP for NCSTRL+.

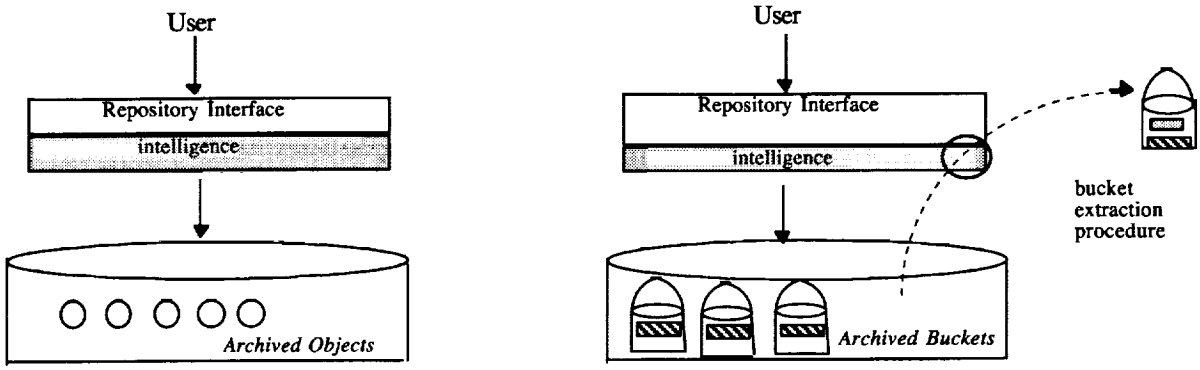


Figure 4: Traditional and Bucket Repository Architectures

a bucket is of arbitrary size
a bucket has a globally unique identifier
a bucket contains 0 or more components, called packages (no defined limit)
a package contains 1 or more components, called elements (no defined limit)
a package can be another bucket (i.e., buckets can be nested)
a package can be a "pointer" to a remote bucket (remote packages can only be accessed through the appropriate access method of the hosting bucket)
buckets can keep internal logs of actions taken on them
interactions with packages are made only through defined methods on a bucket

Table 3: Bucket Requirements

Method	**Description**
`metadata`	returns the bucket's metadata in its native form
`display`	default method; bucket "unveils" itself to requester
`id`	returns the bucket's unique identifier (handle)
`terms and conditions`	describes the nature of the bucket's terms and conditions
`list methods`	list all methods known by a bucket
`add package`	adds a package to an existing bucket
`delete package`	deletes a package from an existing bucket
`add method`	"teaches" a new method to an existing bucket
`delete method`	removes a method from a bucket

Table 4: Required Bucket Methods (other methods can be custom defined)

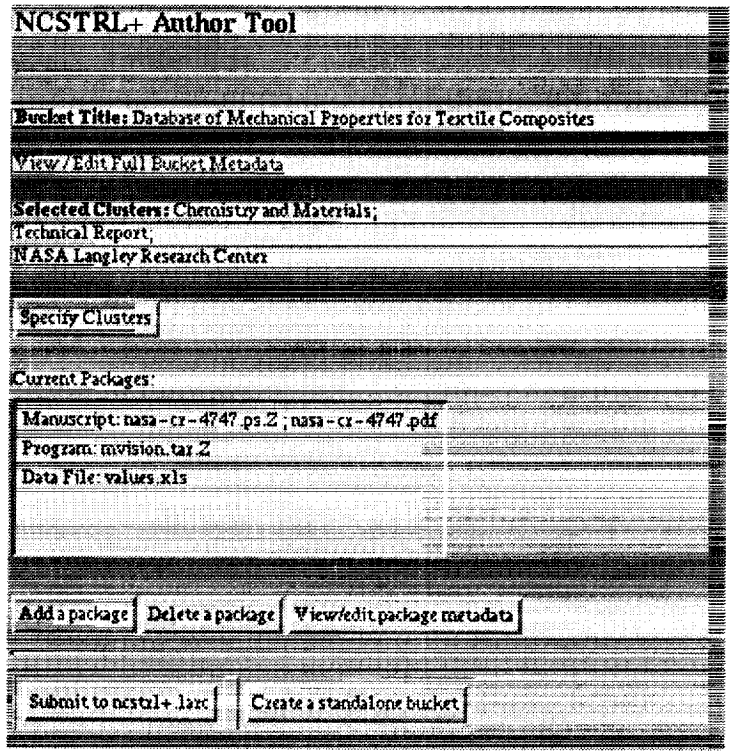


Figure 5: Author Tool

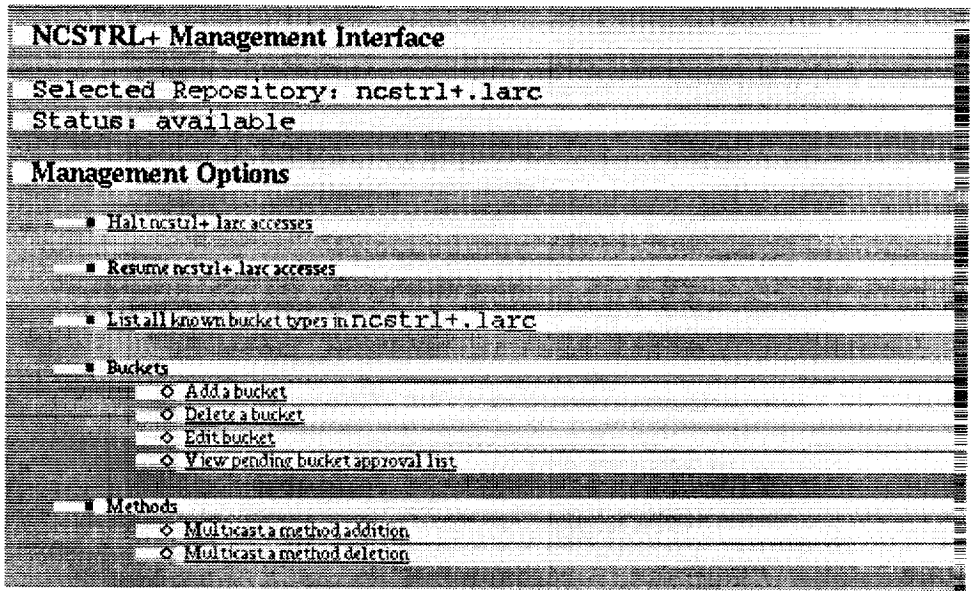


Figure 6: NCSTRL+ Management Tool

4.2 Bucket Tools

There are two main tools for bucket use. One is the *author tool*, which allows the author to construct a bucket with no programming knowledge. Figure 5 shows the author tool. Here, the author specifies the metadata for the entire bucket, adds packages to bucket, adds elements to the packages, provides metadata for

the packages, and selects applicable clusters (which lead to the cluster options available as shown in Figure 7). The author tool gathers the various packages into a single component and parses the packages based on rules defined at the author's site. Many of the options of the author tool will be set locally via the second bucket tool, the *management tool*. The management tool provides an interface to allow site managers to configure the default settings for all authors at that site. The management tool also provides an interface to query and update buckets at a given repository. Additional methods can be added to buckets residing in a repository by invoking the `add_method` on them and transmitting the new code. Figure 6 shows the management tool interface. From this interface, the manager can halt the archive and perform operations on it, including updating or adding packages to individual buckets, updating or adding methods to groups of buckets, and performing other archival management functions.

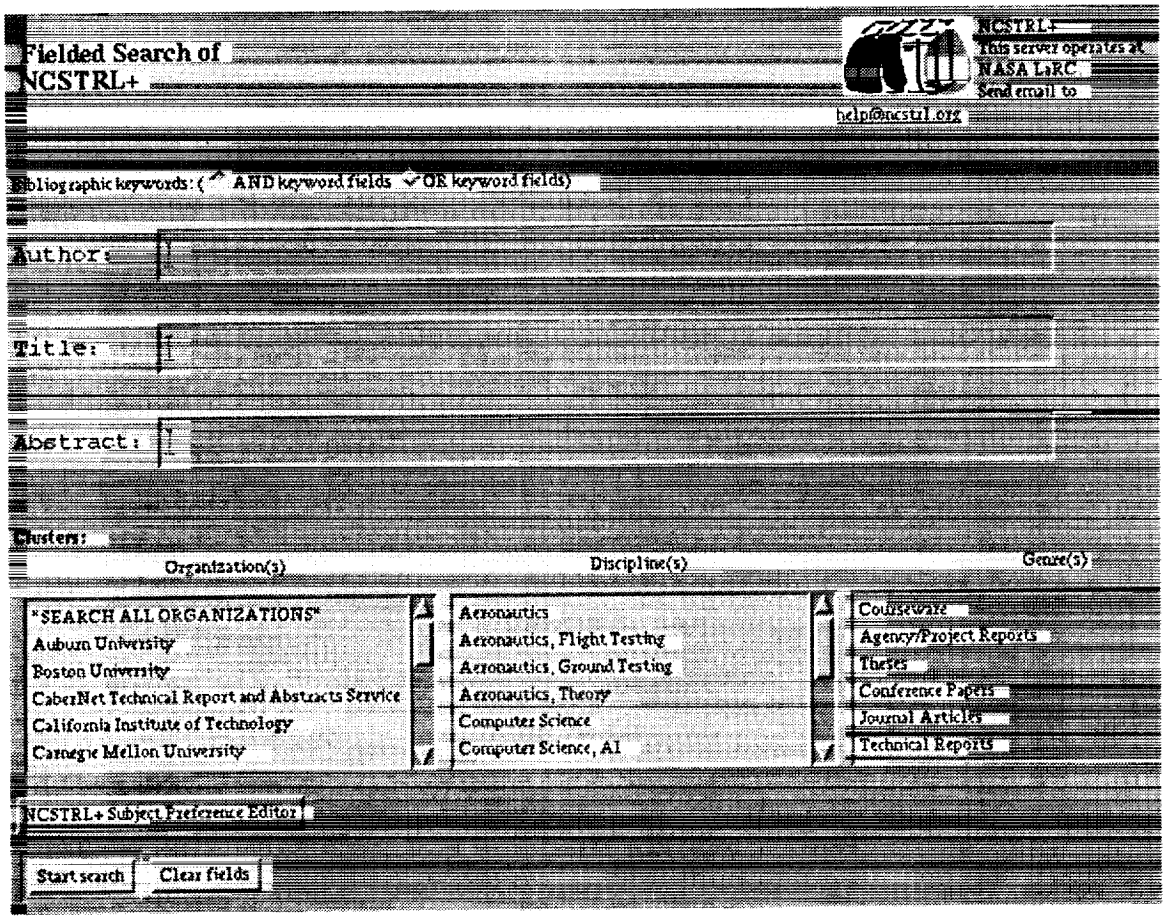


Figure 7: The Fielded Search Screen of NCSTRL+

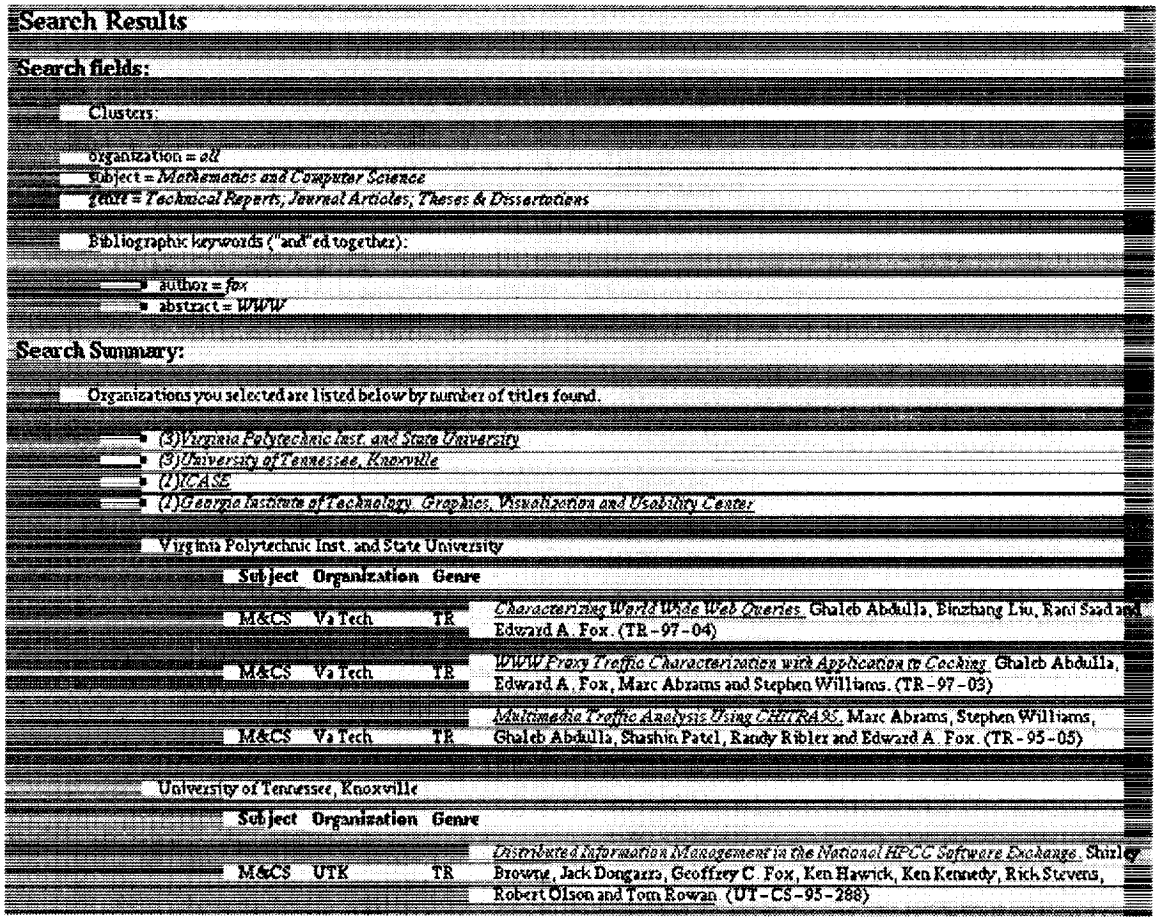


Figure 8: NCSTRL+ Search Results Screen

5 Using NCSTRL+

5.1 Searching NCSTRL+

NCSTRL+ searching is similar to searching with NCSTRL, with the addition of specifying desired clusters to search. Figure 7 shows the how the advanced fielded search form of NCSTRL+ is modified, allowing the selection of desired subject categories and data genres. Figure 8 shows a sample search results page, including the keyword and cluster hit results. The user will select the desired bucket from this page. At that point, the bucket will return the defined default initial interface of the bucket, which will be dependent on the bucket contents and the rules present. In practice, the bucket presentation will look largely similar to the choices available to current users of NCSTRL. This is especially true if the buckets in which they are interested only contain various manuscript formats. However, the real benefit is the richer presentation formats available if the bucket has non-manuscript packages. Figure 9 illustrates a typical bucket with

packages other than a manuscript. The interface is similar to NCSTRL, with the exception that the additional data semantics are presented (software, datasets, etc.).

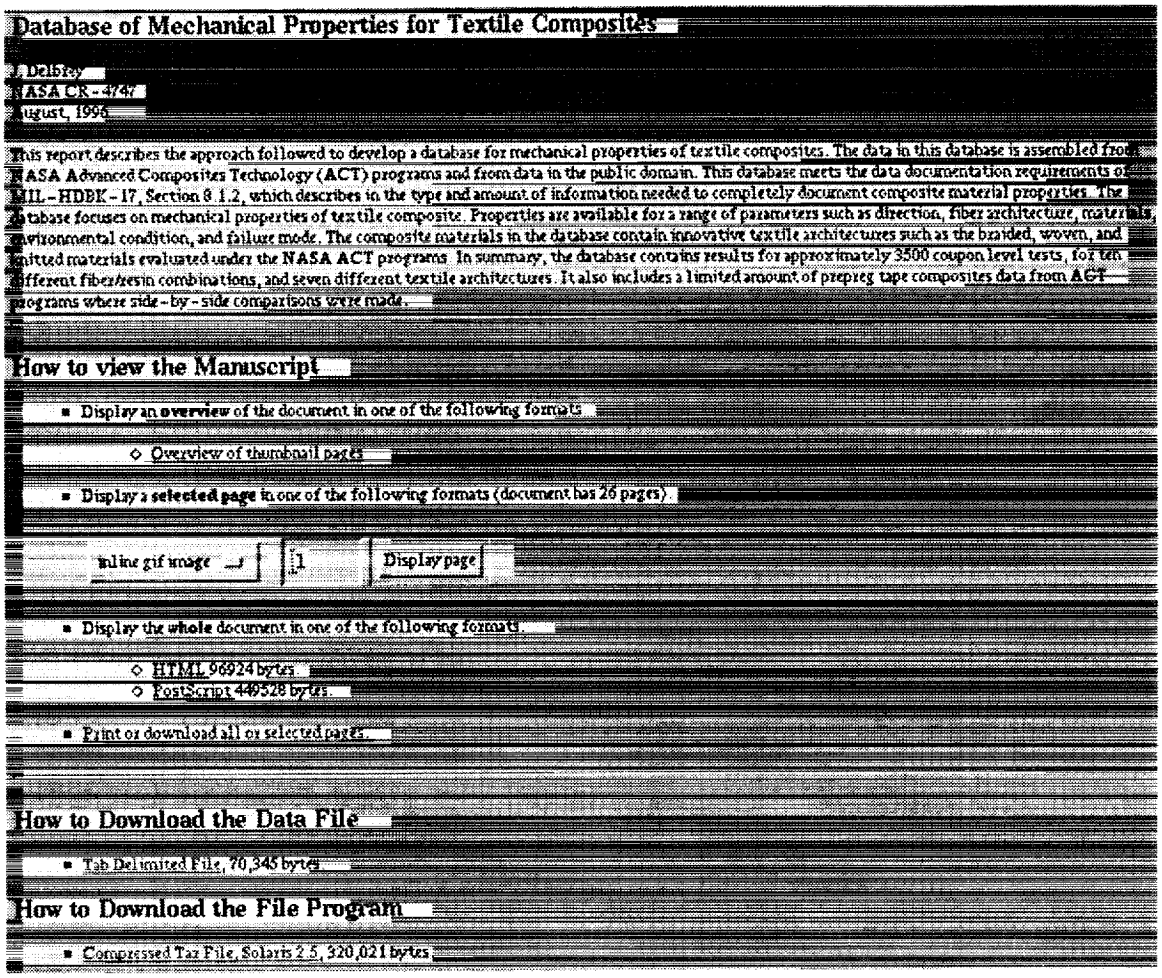


Figure 9: A Typical Bucket Presentation

5.2 Publishing into NCSTRL+

The goal of NCSTRL+ is to produce the least intrusive interface possible to the author. The authoring process for NCSTRL+ is to be as similar to authoring into NCSTRL as possible. Additions include the ability to add to a bucket multiple data semantics and formats through using multiple selection boxes to select local files. Publishing a manuscript in NCSTRL is equivalent to publishing a package in NCSTRL+, and publishing a bucket is the sum of publishing all of its packages. The author also has to choose the appropriate cluster to place the new bucket in. This step can be skipped if the site manager has defined a default, or if authors have saved a value already in their preferences.

6 NCSTRL+ Testbed

6.1 Architecture

Figure 10 shows the architecture of NCSTRL+. Three machines will be employed. The first will be the home page and meta data collection/search machine, and will reside at NASA. NASA will also house a second machine for the aeronautics cluster. Old Dominion will use a third machine to host the computer science cluster. Although similar in appearance, the NCSTRL+ prototype will be operationally independent of NCSTRL.

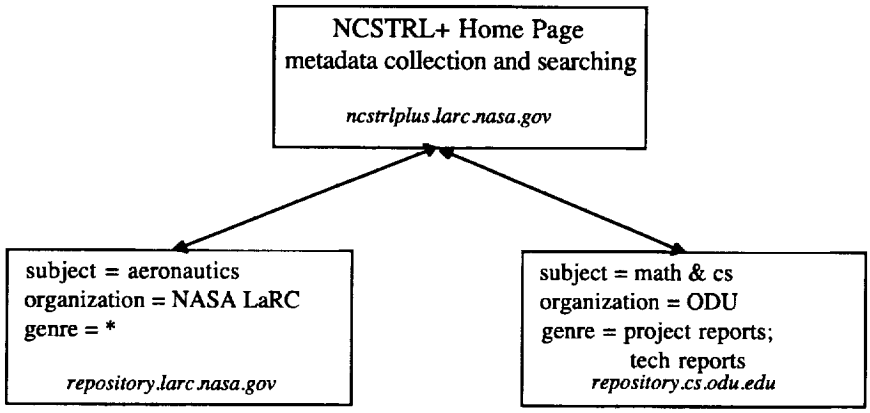


Figure 10: Initial NCSTRL+ Architecture

6.2 Metadata

Currently, all NCSTRL+ buckets use the RFC-1807 metadata format. However, any format can be used and Dublin Core [9] is a likely format to be adopted in the future. There is no reason that multiple metadata formats cannot be simultaneously supported. Although logically, a bucket has its own metadata file, and all its packages have their own separate metadata file, the implementation is that all the package metadata fields be embedded with the single metadata file for the bucket. It is this single metadata file that is indexed. This allows the package metadata to be searched simultaneously, and the linkage is created so that multiple hits across many packages within a single bucket will produce only one bucket to be returned. Figure 11 shows the example from [12] modified for a single metadata file to carry both bucket and package metadata.

BIB-VERSION:: X-NCSTRL+1.0
ID:: OUKS//CS-TR-91-123
ENTRY:: January 15, 1992
ORGANIZATION:: Oceanview University, Kansas, Computer Science
TYPE:: Technical Report
TITLE:: Scientific Communication must be timely
AUTHOR:: Pooh, Winnie The
CONTACT:: 100 Aker Wood
DATE:: December 1991
PAGES:: 48
HANDLE:: hdl:oceanview.electr/CS-TR-91-123
OTHER_ACCESS:: url:http://electr.oceanview.edu/CS-TR-91-123
KEYWORD:: Scientific Communication
CR-CATEGORY:: D.0
CR-CATEGORY:: C.2.2 Computer Sys Org, Communication nets, Net Protocols
NCSTRL+CATEGORY: 59
ABSTRACT::
Many alchemists in the country work on important fusion problems.
All of them cooperate and interact with each other through the
scientific literature. This scientific communication methodology
has many advantages. Timeliness is not one of them.
PACKAGE:: OUKS//CS-TR-91-123/P1
TITLE:: Timeliness Simulator
OTHER_ACCESS:: url: http://electr.oceanview.edu/CS-TR-91-123/timely.java
END-PACKAGE:: OUKS//CS-TR-91-123/P1
PACKAGE:: OUKS//CS-TR-91-123/P2
TITLE:: Timeliness Data Sets
ABSTRACT:: These data sets track averages # years to publish for the department.
OTHER_ACCESS:: url: http://electr.oceanview.edu/CS-TR-91-123/timely.xls
END-PACKAGE:: OUKS//CS-TR-91-123/P2
END:: OUKS//CS-TR-91-123

Figure 11: Bucket + Package Metadata in a Single File (new fields in bold)

7 Status and Future Work

We are using the author tool to populate NCSTRL+ so that we gain insight on how to improve its operation. We are starting with buckets authored at Old Dominion University and NASA Langley Research Center and are choosing the initial entries to be "full" buckets, with special emphasis on buckets relating to NSF projects for ODU and for windtunnel and other experimental data for NASA. Until NCSTRL+ becomes a full production system, we are primarily seeking rich functionality buckets that contain diverse sets of packages.

It is also important to note that adding a subject category mechanism to NCSTRL+ provides the necessary groundwork for additional services for digital libraries using Dienst. These could include subject-based browsing of NCSTRL+ holdings, as well as selected dissemination of information (SDI). This would be most useful if users were offered a subscription option to receive digested updates (i.e., e-mail messages) of new additions to NCSTRL+ in specified subject areas. The initial defined subject categories for NCSTRL+ and cross-listing them with other subject-specific categorization schemes is intended to provide a working framework for evaluating the prototype. As more experience in NCSTRL+'s use is

gained, the fine tuning of the subject categories and appropriate cross listing becomes an area that would benefit from the attention of a professional cataloger.

8 Conclusions

Due to the increased requirements for multidisciplinary activities, NASA is interested in the availability of a canonical, unified digital library for STI to counter the current trend of disciplines developing their own incompatible digital libraries. We have prototypes of NCSTRL+ and are in the process of full implementation. Since our modifications are limited in scope, we have noticed no change in the performance profile of NCSTRL+ versus NCSTRL. NCSTRL+ is forged from the holdings of the NCSTRL and NTRS archives and providing access to aerospace, mathematics, computer science, physics and engineering STI. NCSTRL+ uses the highly successful Dienst protocol, with some extensions for providing clustering functionality around subject category, genre, and organization. These extensions are to gain user feedback on the usefulness of this service while awaiting the development of a generalized clustering service for Dienst. The most significant technology from this project is the concept of buckets as a construct to capture multiple data formats and genres in an intuitive manner. Although the associated social and political problems of changing the nature of an institution's publication vector are not addressed, NCSTRL+ provides a platform for experimentation for testing user response to multidiscipline clusters and logical collections of STI. At this point, we have no data concerning the usefulness of buckets and clusters to the user, or about their cost effectiveness. However, we are in the process of experimenting with users at NASA and Old Dominion University. From the users' perspective, the publishing and searching interfaces are largely unchanged. However, it is unknown what impact the cluster and bucket modifications have on network load, search and retrieval times, the users' perceived quality of searching multiple clusters, etc. To determine these unknowns, NCSTRL+ will have to grow to a large enough size to be considered a useful production system. The authors seek other users and participants for NCSTRL+. Contact information, current status, and related software can be found at: `http://ncstrlplus.larc.nasa.gov/`

References

[1] ACM SIGIR On-Line Conference Proceedings,
`http://turing.acm.org:8071/`

[2] S. Browne, J. Dongarra, E. Grosse, S. Green, K. Moore, T. Rowan, & R. Wade, "Netlib Services and Resources," University of Tennesse Technical Report UT-CS-93-222, 1993.

[3] J. R. Davis, D. B. Krafft, & C. Lagoze. "Dienst: Building a Production Technical Report Server," *Advances in Digital Libraries*, Springer-Verlag, 1995, pp. 211-222.

[4] S. L. Esler & M. L. Nelson. "The Evolution of Scientific and Technical Information Distribution," Submitted to the *Journal of the American Society of Information Science*, 1997.

[5] E. Fox, J. Eaton, G. McMillan, N. Kipp, L. Weiss, E. Arce, & S. Guyer. "National Digital Library of Theses and Dissertations: A Scalable and Sustainable Approach to Unlock University Resources," *D-Lib Magazine, The Magazine of Digital Library Research*, Sep. 1996.
`http://www.dlib.org/dlib/september96/theses/09fox.html`

[6] R. Kahn. "An Introduction to the CS-TR Project," December 1995.
`http://WWW.CNRI.Reston.VA.US/home/describe.html`

[7] R. Kahn. "The Handle System Version 3.0: An Overview."
`http://www.handle.net/docs/overview.html`

[8] R. Kahn & R. Wilensky, "A Framework for Distributed Digital Object Services," cnri.dlib/tn95-01, May, 1995.
`http://WWW.CNRI.Reston.VA.US/home/cstr/arch/k-w.html`

[9] C. Lagoze, C. A. Lagoze, &R. Daniel, "The Warwick Framework: A Container Architecture for Aggregating Sets of Metadata," Cornell University Computer Science Technical Report TR-96-1593, June, 1996.

[10] C. Lagoze & D. Ely, "Implementation Issues in an Open Architectural Framework for Digital Object Services," Cornell University Computer Science Technical Report, TR95-1540, June, 1995.

[11] C. Lagoze, E.Shaw, J. R. Davis, & D. B. Krafft, "Dienst: Implementation Reference Manual," Cornell Computer Science Technical Report TR95-1514, 1995.

[12] R. Lasher, & D. Cohen, "A Format for Bibliographic Records," Internet RFC-1807, June 1995.

[13] K. Maly, J. French, A. Selman, & E. Fox, "Wide Area Technical Report Service," *Proceedings of the Second International World Wide Web Conference*, Chicago, IL, October 21-23, 1994, pp. 523-533.

[14] NASA Scientific and Technical Information Program, "NASA STI Topics".
`ftp://ftp.sti.nasa.gov/pub/scan/SCAN-TOPICS`

[15] National Advisory Committee for Aeronautics (NACA) Report Server.
`http://www.larc.nasa.gov/naca/`

[16] M. L. Nelson, G. L. Gottlich, D. J. Bianco, S.S. Paulson, R. L. Binkley, Y. D. Kellogg, C. J. Beaumont, R. B. Schmunk, M. J. Kurtz & A. Accomazzi, "The NASA Technical Report Server," *Internet Research: Electronic Networking Applications and Policy*, vol. 5, no. 2, 1995, pp. 25-36.

[17] M. L. Nelson , G. L. Gottlich & D. J. Bianco, "World Wide Web Implementation of the Langley Technical Report Server," NASA TM-109162, September 1994.

[18] M. L. Nelson, K. Maly & S. N. T. Shen, "Building a Multi-Discipline Digital Library Through Extending the Dienst Protocol," *Proceedings of the Second International ACM Conference on Digital Libraries*, Philadelphia, PA, July 20-23, 1997, pp. 262-263.

[19] M. L. Nelson & S. L. Esler, "TRSkit: A Simple Digital Library Toolkit," *Journal of Internet Cataloging*, 1(2), 1997, pp. 41-55.

[20] D. G. Roper, M. K. McCaskill, S. D. Holland, J. L. Walsh, M. L. Nelson, S. L. Adkins, M. Y. Ambur & B. A. Campbell, "A Strategy for Electronic Dissemination of NASA Langley Technical Publications," NASA TM-109172, December 1994.

[21] J. Sobieszczanski-Sobieski & R. T. Haftka, "Multidisciplinary Aerospace Design Optimization: Survey of Recent Developments," 34th AIAA Aerospace Sciences Meeting and Exhibit, Reno, Nevada, AIAA Paper No. 96-0711, January 15-18, 1996.

[22] J. Sobieszczanski-Sobieski, "A Proposal: How to Improve NASA-Developed Computer Programs," NASA CP-10159, 1994, pp. 58-61.

[23] UVa SEAS Electronic Undergraduate Thesis Pilot, `http://univac.cs.virginia.edu:3066/SEAS_ETD.html`

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	July 1997	Technical Memorandum

4. TITLE AND SUBTITLE
Buckets, Clusters and Dienst

5. FUNDING NUMBERS

6. AUTHOR(S)
Michael L. Nelson, Kurt Maly, Stewart N. T. Shen

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
NASA Langley Research Center
Hampton, VA 23681-2199

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Washington, DC 20546-0001

10. SPONSORING/MONITORING AGENCY REPORT NUMBER
NASA TM-112877

11. SUPPLEMENTARY NOTES
Michael L. Nelson, NASA Langley Research Center, Hampton, VA; Kurt Maly and Stewart N. T. Shen, Old Dominion University, Norfolk, VA. Also appeared as Old Dominion University Computer Science Technical Report TR-97-30.

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Unclassified–Unlimited
Subject Category 82 Distribution: Nonstandard
Availability: NASA CASI (301) 621-0390

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)
In this paper we describe NCSTRL+, a unified, canonical digital library for scientific and technical information (STI). NCSTRL+ is based on the Networked Computer Science Technical Report Library (NCSTRL), a World Wide Web (WWW) accessible digital library (DL) that provides access to over 80 university departments and laboratories. NCSTRL+ implements two new technologies: cluster functionality and publishing "buckets". We have extended the Dienst protocol, the protocol underlying NCSTRL, to provide the ability to "cluster" independent collections into a logically centralized digital library based upon subject category classification, type of organization, and genres of material. The concept of "buckets" provides a mechanism for publishing and managing logically linked entities with multiple data formats. The NCSTRL+ prototype DL contains the holdings of NCSTRL and the NASA Technical Report Server (NTRS). The prototype demonstrates the feasibility of publishing into a multi-cluster DL, searching across clusters, and storing and presenting buckets of information. We show that the overhead for these additional capabilities is minimal to both the author and the user when compared to the equivalent process within NCSTRL.

14. SUBJECT TERMS	15. NUMBER OF PAGES
WWW, Digital Libraries, STI, Distributed Information Retrieval	19
	16. PRICE CODE: A03

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	