NISTIR 5792

Organization of the Manufacturing Systems Integration Division's On-line Information – Experiences and Recommendations

Michelle Potts Steves
Don Libes

April 1996

U.S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Manufacturing Engineering Laboratory
Manufacturing Systems Integration Division
Gaithersburg, MD 20899

Table of Contents

Organization of the Manufacturing Systems Integration Division's On-line Information
–
Experiences and Recommendations

Michelle Potts Steves
Don Libes
Manufacturing Information Technology Transfer Team, MSID, MEL

"Anyone who starts a Web server has to be aware that maintenance is where the cost is. Just putting the information up in the first place is very easy. Updating is a serious effort. But it's very valuable, it's what the public needs."
— Tim Berners-Lee

I. Introduction

This paper presents our observations and recommendations on the organization of the's Manufacturing Systems Integration Division's (MSID) on-line information. The motivation for this is:

- to reduce data redundancy and facilitate configuration management
- to improve ease of use
- to maintain quality

The scope of this paper is MSID's publicly accessible information, which lives in the /proj/elib/online/pub directory structure. The objective of this paper is two-fold: (1) to document the current structure and formative rationales and (2) to design the structure for the near future. We address issues of configuration management and maintenance of the information, as well as issues of presentation and navigation of the information. Finally, we provide guidelines for MSID staff in organizing information for external dissemination.

A. Delivery Mechanisms

This paper makes reference to a large number of information dissemination mechanisms. These include:

- File Transfer Protocol (FTP)
- private FTP
- email archive server
- gopher
- WAIS

- HyperText Transport Protocol (HTTP)
- kermit

B. Conventions

An *information provider (IP)* is the person responsible for adding or maintaining information that other users can access. Since we use the term so frequently, we use *IP* as a convenient abbreviation.

An *information user* makes use of the information provided by IPs. For convenience, we often refer to information users as simply *users*.

II. What We Found

A. Casual and Sporadic Oversight

The general oversight of information population and organization of the externally accessible information space has traditionally been casual and sporadic. Staff were allowed write access to the directory structure and added information as they deemed appropriate. Ad hoc reviews were performed and out-of-date material removed. Information organization guidelines were, and remain, very general and therefore, there is little consistency among branches maintained by different people. There have been recent efforts to use a locally-developed approval procedure for web (HTTP served) material, with some success. However, with appropriate write access, staff can bypass the approval procedure for some existing directory branches. A general tightening of write access for the entire externally accessible information area has been occurring over the last year.

Within the directory structure /proj/elib/online/pub, branches of the structure are created and may be maintained by different people or projects. Generally, support for that creation and maintenance varies with funding levels and project managers' priorities and deliverables. As such, information organization structures often reflect a very provincial view. Therefore, their information dissemination mechanism(s) of choice and how up-to-date their information is, may vary greatly. With funding-driven allocation of effort, this situation may be unavoidable.

B. Information Structure Changes Slowly Despite New Dissemination Mechanisms

We have found that, once in place, directory structures do not change often except at the ends of the branches, mainly because of legacy issues. One common legacy issue revolves around users and their familiarity with existing structures. Another prevalent issue revolves around the IP, where even small changes must be propagated through potentially large information sets.

1. New Tools/Views/Requirements Bring New Demands

MSID's externally-available information repository has evolved from a relatively small collection of directories of selected software tools and publications available to the public for FTP access into a conglomeration of software tools, program and project descriptions, the division's publications from the past ten years, standards committees' file sharing repositories, online software services and software testing services, etc. As access mechanisms and services have been added, the directory structure has not been restructured to accommodate its new scope, but added to with little consideration for the larger information organization picture. Each new mechanism added its own view and requirements of the directory structure.

2. Historically – Information Organized by Hierarchical Directories

a) FTP (including private FTP and kermit)

MSID's externally available information repository was originally designed for FTP access; kermit access was an afterthought. These information dissemination mechanisms are closely tied to the physical file system structure from which they serve information. README files are an FTP convention, used to describe a directory's contents and were used in MSID's directory structure, although sporadically. The use of README files also compensates for the use of 8.3 file name limitations for MS DOS-based IPs and users by providing a mechanism for describing a file's contents other than the use of the file's name.

b) Email Archive Server And Gopher

Some of the subsequently-deployed information dissemination access mechanisms, e.g., email archive server and gopher, are also closely tied to the use of directory structures as an information organization mechanism. An email archive server delivers functionality similar to FTP, but with an email interface; it does not require any support files in the directory structure it is serving (although the README files are useful for users). The gopher information dissemination mechanism uses a menu file in each directory it serves. Currently, the menu files are automatically generated by a script from the contents of the directory and in their current form are not particularly useful, however, we would like to have each menu file have the potential to be customized while being updated automatically (if its maintenance is continued – see recommendations).

3. Future – Organization Becomes More Web-like

With the advent of information dissemination mechanisms which are not closely tied to a directory structure in terms of presentation to the user, the function of information organization via the directory structure mechanism becomes of lesser importance to the information requester. The directory structure still remains an important issue for the IP regarding the previously noted legacy issues, however many other organization principles are able to be used concurrently.

a) HTTP

HTTP is such a mechanism, when used in its most commonly used mode to browse documents via embedded hyperlinks. HTTP departs from previously employed dissemination mechanisms on MSID's external server, in that the user's common view is of documents connected by hypertext links, rather than files in directory structures. However, HTTP allows directory browsing functions, if so configured, and as such, really is a hybrid mechanism. Additionally, the HTTP dissemination mechanism departs from other mechanisms in that information presentation is more central to functionality and attractiveness of the client side of this mechanism, whereas previously mentioned mechanisms focus on file transfer and directory/information hierarchy traversal. The central role of information presentation necessitates that formatting information is embedded in the same file as the information content. This embedded formatting information is not meaningful to other servers, unless gateways (programs that can decipher other protocols) are used.

b) WAIS

The WAIS protocol is another information dissemination mechanism that is not closely tied to a physical directory structure in terms of information presentation. WAIS is being provided as an MSID service to facilitate searching and retrieval of selected information sets. In MSID, it is primarily being used via a www-WAIS gateway, where the search is initiated from a HyperText Mark-up Language (HTML) form and the results are presented to the requestor as a hyperlinked HTML-formatted page. The use of this mechanism, requires index files to be generated of the searchable material, but there is no requirement that they physically reside with their corresponding information sets. This mechanism provides enhanced accessibility, while not increasing the burden of organizing disparate information sets and types.

C. IPs May Be Ignorant Of The Issues

Currently, some IPs may not even be aware of the many different mechanisms by which their information is available, and therefore, are not configuring views for those audiences out of ignorance rather than choice. This scenario is easy to envision: an IP requests to have their information made available for FTP access, which is granted. The IP may be unaware that their information is now also accessible by gopher and www clients. To further compound this issue of presentation for multiple mechanisms, once aware that information has several access mechanisms, an IP may choose not to provide facilitating views for some mechanisms, leaving gaps in the presentation of the whole publicly available information set via any particular mechanism.

D. Information Dissemination Demands Rigor

As information access becomes easier for information seekers, the impetus for IPs to provide more on-line information via new mechanisms grows. This desire to provide information via many different mechanisms with their attendant disparate requirements has created an information organization crisis. Currently, MSID's directory structure is growing in untamed and often unconsidered ways to accommodate the demand for information dissemination via the entire spectrum of available information dissemination mechanisms. This untamed growth is the motivation for this effort: to take a step back and to carefully consider how MSID's externally available information should be organized so that it is maintainable, expandable, and easy to use.

III. Recommendations

This section of the paper contains recommendations for IPs and infrastructure configuration and maintenance recommendations.

Our primary goals are as follows:

- To provide a system that is intuitively easy for users and IPs to use.
- To provide a robust and reliable system.
- To rationalize and document our practices.
- To keep the maintenance low-cost.

Our secondary goals are:

- Improve upon our current practices.
- Reuse existing practices.

All of these goals may not be mutually compatible, but our recommendations nonetheless strive to meet them.

A. Dissemination Mechanism Support

Due to the limitations of individual dissemination mechanisms, there is no single dissemination mechanism suitable for our information accessibility requirements and it is unlikely that there will be one in the near future. These requirements are: to facilitate information presentation and accessibility using non-proprietary, widely accepted mechanisms for users with and without Internet connectivity. Therefore a combination of mechanisms needs to be employed.

1. We recommend supporting the following information services (unless otherwise noted):

a) HTTP

HTTP will be the usual choice for most IPs. Private HTTP is necessary in special cases, such as the MSID internal web.

b) FTP

FTP is the natural choice for data file transfer. Private FTP is necessary in special cases. This may change in the future as secure HTTP may offer a better solution.

c) **Gopher**

We believe Gopher usage is lessening and we recommend not providing it as an information dissemination mechanism. This would reduce server suite maintenance and impose more consistency in MSID's information presentation. Existing menu files can be used as the basis for README files (see below).

d) **Email archive servers**

The Email Archive Server is useful for users without direct TCP/IP connectivity or other experimental purposes. We currently have four, they are:

(1) **Library server**

This is a basic replacement for FTP service for users who have email but without TCP/IP connectivity.

(2) **NIST EXPRESS Server**

This is documented by a report in the references. It shows a well thought-out alternative to traditional information dissemination services, providing collaboration services and the delivery of X window service without the limitations of HTML.

(3) **National PDES Testbed Mail Server**

This is essentially a duplication of the Email archive server, albeit with a modified greeting. This mail server should be phased out.

(4) **Agora email web server**

This server provides an email interface to our on-line information which allows the user to use Universal Resource Locators (URLs) to specify files for retrieval.

e) **WAIS**

WAIS provides a widely-used searching mechanism with an interface to HTML. We recommend its continued support until a mechanism with more functionality (e.g., better searching or query specification capabilities) is available. We are currently testing alternative search systems such as Glimpse.

f) **Ad hoc**

Some projects have successfully used ad hoc methods of information dissemination. We must recognize that diverging from traditional methods is expensive, while at the same time can lead to high payoff. We must require that such departures be fully justified and financed.

g) **Help documentation**

We recommend the creation of a help document which describes how the materials contained in the publicly accessible directories are best accessed and who to contact for help. This document need not be lengthy, but must be thoughtfully written to be helpful.

Such documentation should be located in the top level of the public directory structure and should be in several of the most common formats such as HTML and unformatted text.

2. IP impact

Dissemination mechanism selection impacts directory structure and/or content. Unfortunately, IPs must at times be aware of this. Additionally, specific dissemination mechanism support requires labor and this must be factored into the time estimated to prepare the information itself.

The following are provisions that must be made for the different mechanisms.

a) README

FTP, http, and email servers provide explicit support for README files. README files should exist in all directories browsable by these services where the file contents are not otherwise obvious from the pathname. The README file should contain a heading describing the directory in general terms and then a list of files and descriptions of each file. If the directory is maintained by a single maintainer, this should be noted at the top. If the files are individually maintained, this should be noted on each description.

The maintainer of the README and the directory should be noted within.

MS DOS is case-insensitive and such IP maintainers must take extra steps to force uppercase in the filename.

b) index.html And Other .html Files

index.html is the file returned by the HTTP server when no file is explicitly specified in a requested URL. If index.html does not exist, the server attempts to deliver a directory listing of the specified directory. IPs can use this mechanism to control the information users' view of selected directories.

Other .html files may be useful for HTML browsers. These files can exist in and amidst other non-HTML files. They do not require dedicated directories.

The maintainer of .html files (and directory it describes if appropriate) should be noted within.

c) WAIS

Indices are updated periodically, IPs must be aware that their new or updated materials will only be available through this searching capability when the indices are updated.

d) private FTP

Private FTP repository instances are set-up on an as needed basis; as such, IPs should give support personnel adequate time to prepare individual repositories. IPs are responsible for the repository as long as it is in existence. When the effort or project which prompted the creation of the repository is finished, the private FTP repository should be removed, this action should be prompted by the IP.

3. Future

Because of changing technology and mechanism use trends, dissemination mechanism support should be reviewed periodically; ideally, as necessary, but for practical purposes, every one to two years.

B. File System Conventions

In the past, our file system conventions followed UNIX conventions because we used UNIX. With the increase of MS DOS and MacOS users and providers, we are modifying our practices to accommodate them.

Shown below are the current file system structure and our proposed file system structure with some specific recommendations for each; more general recommendations follow.

1. Structure

Files should be organized into directories of related areas. This makes sense from both the IP's and the user's point of view as will be noted elsewhere. The most noticeable structure is the directory /proj/elib/online/pub. This contains several dozen files and directories. They are not organized identically because they do not have identical requirements. However, they are organized in a small number of ways. These are formalized below.

Ideally, the directories should be organized so that it is always possible for a visitor to decide which subdirectory to visit next. This aim is especially helpful in the directory-based browsers such as FTP.

In the future, we anticipate the increased need for more sophisticated information organization mechanisms than the directory structure mechanism to facilitate information dissemination. We expect that, increasingly, information dissemination servers will be interfaced to database management systems. This type of information organization and presentation system has the potential to provide better security and configuration management than the current directory structure mechanism. We recommend that MSID investigate this technology as it matures for its use.

a) Information Dissemination Mechanism Requirements

Some information dissemination mechanisms require their own layouts or auxiliary directories for which we have little choice in placement. Currently, the primary requirements are for files rather than top-level directories. Files are covered in "Dissemination mechanism support".

Unless otherwise stated, all file and directory names are relative to /proj/elib/online/pub.

(1) Current structure

A partial view of the current structure is shown in the following figure:

```
/proj/elib/online
        |--bin/
        |--dev/
        |--etc/
        |--icons/
        |--logs/
        |--pub/ ---------- |- README
        |                  |- atp/
        |                  |- ats/
        |                  |- cm/
        |                  |- depot/
        |                  |- download
        |                  |- expect/
        |                  |- fasd@
        |                  |- helpdoc/
        |                  |- images/
        |                  |- lost+found/
        |                  |- ls-lR.Z
        |                  |- mei/
        |                  |- menu
        |                  |- msid/
        |                  |- nipde/
        |                  |- oiw/
        |                  |- pptb/
        |                  |- private/
        |                  |- sc4/
        |                  |- sendmail/
        |                  |- sima/
        |                  |- src/
        |                  |- stdoview/
        |                  |- step/
        |--usr/
        |--index.html@
```

Figure 1: Current Directory Structure

Below are several items that should be addressed in the current structure.

(a) /proj/elib/online/icons

The "icons" directory should be removed or renamed. It is not required by FTP nor is it documented.

(b) /proj/elib/online/logs

The "logs" directory should be removed or renamed. It is used by kermit and should not be available to public view.

(2) Proposed directory structure

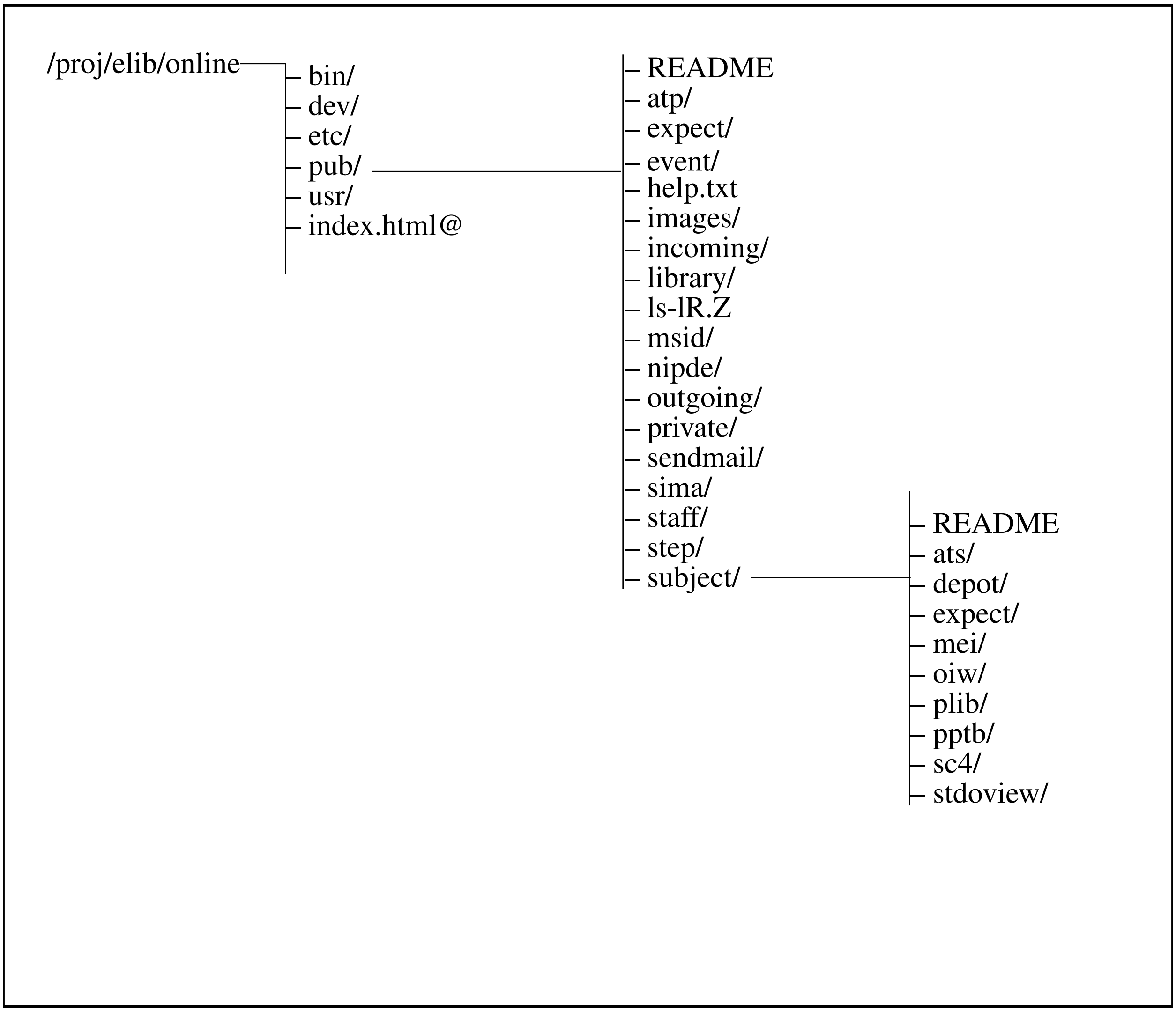

Figure 2: Proposed Directory Structure

(a) /proj/elib/online

/proj/elib/online is the name of the top of our externally accessible electronic library structure. It should not be changed because of the effort required to do so and legacy issues.

(b) /proj/elib/online/pub

The pub directory contains all "interesting" information that users want. This organization is not a requirement of FTP but it is so standard that people would be confused if any other organization would be used.

Most of the directories in /proj/elib/online are required by FTP. Only the pub directory is useful to FTP users. However, all of the files and directories are visible to users.

(c) junk

We recommend the creation of a "junk" (or other suitably named) directory outside of the publicly accessible directory structure for the purpose described in "Maintenance" below.

b) Broad Topics

Broad groupings of disjoint information make appropriate top-levels. "step" and "sima" are examples of this; they are programmatic categories. Suggestions for specific top-levels follow:

(1) Generic subject Top-Level

We recommend adding a "subject" top-level to contain directories of subject-related information. We believe this would make the task of finding subject-related information easier than any other organization. For example, a person looking for SIP-related information should not have to guess whether that is part of STEP or SIMA or yet a different project or perhaps not even a project at all.

The subject top-level provides a place for the traditional projects as well as completed projects, unfunded projects, future projects, speculation, and affinities.

Completed projects show some of our best work and we continue to have the knowledge and competence earned on those projects. Unfunded or future projects are similar in that they advertise our competencies and let others know what we are interested in working on. Lacking of funding is quite often only a temporary situation. Affinity areas allow us to express interest in areas that may not be specifically related to projects (i.e., "The following staff are interested in AI"). Speculative areas allow a place to present thoughts far afield from traditional projects.

The msid/proj directory can point back to the subject top-level entries as appropriate. However, the subject top-level should not be constrained by the proj organization. Project titles are often not particularly meaningful and grouped in unhelpful ways, often more for political, administrative or budgetary reasons that mean nothing to the outside user.

(2) library Top-Level

We recommend adding a "library" top-level directory to contain all publications (their official, browsable version) regardless of format (HTML, text, PostScript, Portable Document Format, etc). This would make the task of finding publications easier than any other organization. For example, a person looking for EXPRESS Toolkit-related papers should not have to guess whether they were published as part of STEP or SIMA or yet a different project.

Access should be provided to these and other documents from appropriate viewpoints. For instance, views showing publications by date, by author, and by topic seem like reasonable candidates. All groupings by date should start from our most recent publications and work backwards. Although dates should be indicated, breaks for years or months are not meaningful here. (Old-style identifiers such as "libes90q" should not be the primary mechanism for identifying publications to users.) Maintaining synchronized multiple views can be expensive. To keep maintenance costs low, we recommend making views using HTML and not (hard) links for FTP access; the maintenance costs would far outweigh the benefits when browser use trends are considered.)

In the future, we believe it will become appropriate to also use this repository for non-sensitive pre-publications. For example, reports that are in WERB review could be listed here. (Of course, links for the full report would only work for staff inside NIST.) Status reports or abstracts could be made available for reports that are in progress. For example, it might be useful to make available the information that the staff is working on a particular report.

(3) event Top-Level

We recommend an "event" top-level to contain information where time is of an important nature. In particular, conference, workshop, demonstration, and call for participation dates are appropriate. This can be a good place to link monthly progress reports and other reports whose primary motivation is a particular date. Conference trip-reports should be made available through the conference information

Events should be organized most-recently-first. This obviates the need of an expiration mechanism. In fact, there is no reason to expire events.

(4) staff Top-Level

We recommend a "staff" top-level to contain information on our staff. This should include staff that has left. As with projects, some of our ex-staff are good advertisements for our results.

The existing msid staff directory can continue to indicate the current staff however, it should be augmented with a pointer to the full staff.

(5) Specific Subject Top-Levels

Existing subject-style directories such as "step" and "sima" are major groupings of substantial amounts of our information. These are appropriate at the top-level because these are all that many people care about. The internal structure of these directories may resemble the structure of other directories described in this section. However, this is only a recommendation and not a requirement.

For similar reasons, contracted-for services such as Advanced Technology Program (ATP) and National Product Data Exchange Resource Center are also appropriate at this level.

(6) Administrative Organization

We recommend avoiding administrative organizations as a structure. Historically, this structure has a poor payoff. Administrative organizations change frequently – much more so than underlying projects or other information.

c) Common Gateway Interface (CGI) scripts

CGI scripts are not stored in the public directory described here. In part, as a simple way of preventing people from seeing the script source. However, CGI URLs form a hierarchy which is evident to users.

Our present use of CGI scripts is minimal, but it is already apparent that this will increase substantially. We plan to revisit the CGI script file hierarchy issue in the near future.

d) Legacy Issues

Several files/directories do not fall into other categories but remain because they are "well known". For instance, Expect is a popular utility that we have distributed for years and its place at the top-level is documented by many other README files and similarly static documents. Moving it would cause sufficient pain that it isn't worth doing. Fortunately, only a few file/directories fall into this category and the group is not getting any larger.

e) Future Issues

Unless you come up with a really convincing rationale, it is unlikely that new file and directories will be added here. We believe that many unrelated files at the top level increases the difficulty of finding information.

2. Hard Links

UNIX hard links allow multiple names for the same file. This is convenient for allowing files to be found in multiple ways when file directory based presentations are used, however, since anyone who has a FTP connection can have a HTTP connection, these links are rarely necessary.

When hard links are needed, the filenames can appear in different directories. For example, the same publication can be found in the pub directory as well as a specific project directory. No extra space is required. Such links are automatically counted by the system so that if the last link disappears so does the file. This particular aspect of hard links facilitates good maintenance practices, however, the links themselves must be maintained, which is a trickier configuration management problem.

Hard links cannot span file system partitions, however this not be a problem unless the size of the data grows substantially.

Hard links are created with the ln command. See the man page for more info.

3. **Symbolic Links**

   UNIX symbolic links may be used to create links to directories. Otherwise symbolic links should not be used. They are confusing to users who go down one path, try to return, and find themselves somewhere else. Cyclic paths are especially confusing. There are rare situations where these kinds of links are helpful but try to avoid them if possible.

   Symbolic links are created with the ln -s command. See the man page for more info.

4. **File and directory names**

   Choose brief and meaningful names where possible. For example, the name "modeling_dynamic_surfaces_with_octrees" is much more obvious than "libes93f", and while not brief, the name immediately conveys relevant information about its content. Choosing good names is a one-time, low-cost, high-payoff activity.

   a) **length**

      MS DOS only supports 8.3 filename lengths. We expect this to evolve in the near future, and eventually no longer be a concern. However, in the interim, we address it as follows:

      Information users should not be particularly adversely affected by longer filenames as they will generally have GUI browsers which allow point and click access.

      IPs using MS DOS on the other hand, will have difficulty accessing files with longer names for some time yet. Therefore, we recommend files which will be maintained using MS DOS, should be limited to 8.3 filename lengths.

   b) **directory names**

      Name directories in singular form. For example "doc" is preferred over "docs". The "s" is almost always redundant. Indeed, this can be found in many places already, such as /home, /depot, /proj, etc.

   c) **case-sensitivity**

      Case-sensitive names are not recommended unless required. MS DOS does not support case sensitivity; therefore the distinction is meaningless to this platform. There are some instances where case-sensitive suffixes indicate special file formats – these names will be recognizable on the platforms on which they are meaningful; see File Formats for information about indicating formats with names.

d) filename selection impact on information retrieval

Filename selection can impact information cataloging mechanisms such as Archie. Archie relies on filenames to help catalog information; unmeaningful names render Archie useless. Other mechanisms use different mechanisms which do not rely on filenames, however, up-to-date README files and, where applicable, hyperlinked description files, are encouraged to facilitate information searching.

5. File Formats

This section describes common formats and gives recommendations. Some of the formats are used together. For example, uuencoded compressed tar files are common. An example file might be called foo.tar.Z.uu which indicates a tar file which was compressed and in turn uuencoded.

a) compress

Compressed files are indicated by a .Z extension.

b) gzip

gzipped files are indicated by a .gz extension. gzip compresses much better than compress, however fewer people have taken the effort to get gzip itself, so this can increase user's efforts.

c) tar

Tar files are indicated by a .tar extension. tar is a format for grouping multiple files together.

d) shar

Shar is a format for grouping multiple files together. Use tar instead.

e) uuencode

uuencoded files are indicated by a .uu extension. uuencode translates files into a form that is less likely to be corrupted by unknown network transmission mechanisms. However, there is no need to use it with information mechanisms such as FTP and HTTP as they handle this problem automatically. Nonetheless, you may see these files occasionally.

f) zip

zipped files are indicated by a .zip extension. This is a common MS DOS format for grouping multiple files.

g) HTML

HyperText Mark-up Language (HTML) files are indicated by a .html or .htm extension. HTML files are designed to be viewed by HTML browsers.

Many style guides are available and dedicated buttons are provided by many browsers. One recommendation that is nonetheless commonly violated is worth mentioning here: pages with simple lists mean following a link rather than simply scrolling. This is particularly bad for remote users, who incur a long latency due to the poor Internet bandwidth. This latency is likely to continually increase.

h) CGI

CGI scripts are indicated by a .cgi extension. CGI scripts are executable programs that produced HTML output, as well as have the potential to perform other functions such as send email, spawn processes, etc.

i) PDF

Adobe Acrobat's portable document exchange format, indicated by a .pdf extension, is becoming quite popular. Viewers with print capability are freely available and the rendering is better than PostScript with ghostview.

j) PostScript

PostScript files indicated by a .ps extension. PostScript files are intended for printing on a PostScript printer. This is the most common format for high-quality printable documentation. However, there is actually a range of issues that cloud this as a standard. In particular, PostScript files define paper sizes and fonts which may not always be available for outside users. Embed any unusual fonts in your PostScript documents so that they will be available. Provide instructions for converting to other common formats (A4) if appropriate.

k) text

Unformatted text files are sometimes indicated by a .txt extension.

l) binary files

UNIX binary files generally have no extension. MS DOS binary files generally have an .exe extension.

m) other

There are many other formats (gif, jpeg, mpeg, etc.), but the ones covered here are the ones that we use most frequently.

Several formats may be conspicuous by their absence. For example, Word Perfect is a common format among the NIST administrative staff. However, the Word Perfect format is not commonly used outside NIST, in part because there is no freely available program to display Word Perfect files. Thus, we discourage use of Word Perfect format as a distribution format.

6. File Ownership And Protection

Files should be owned by their maintainer. When maintainers relinquish their maintenance role (e.g., leave NIST), the file ownerships should be changed to a new maintainer.

File protection should permit reading. Directories should generally have traversal permission. By design, the outgoing (to-the-public) area (pub/download) does not enable users to list the files in it. (All such files are removed after four days.) We recommend renaming this directory “outgoing”.

Files in the incoming (from-the-public) area (pub/upload) initially arrive with root ownership. Every hour, permissions are changed to enable IPs to remove the files. (All such files are removed at 2AM each night.) We recommend renaming this area “incoming”.

C. Maintenance

In general, as much as possible should be automated. In addition to the descriptions provided here, no documents should be added without provisions for their maintenance. That means that documents must either be fully automated or that their manual maintenance must be accounted for out of project funds.

Information that could conceivably require regular maintenance must include funding plans. One possible funding plan may be “none” in which case the pages must be so marked to the viewer and approved by management.

1. Syntax

All public HTML files should be regularly checked for conformance to a standard. The actual choice of standard is not clear at this time.

2. Permissions

A nightly daemon should check and if necessary reset permissions on all files. Except for the incoming area, files and directories should not be writable by outside users.

3. Dates

Daemons described in this section should avoid changing dates except when specifically appropriate. For example, if a file permission must be changed, the date should not be. However, if a descriptor file such as README has been substantively changed, its date should reflect that.

A nightly daemon should update an indicator in the events page showing what events are before and after the present date. This should be a simple matter of moving an icon through the file.

4. README

A nightly daemon should check that all appropriate directories contain a README and that it is at least as up-to-date as the directory and files that it describes.

5. Publications

Publications should automatically be added to the top-level library directory as publications are approved. Entries should be made to the other dissemination mechanisms such as the index.html file and the individual staff pages. The links can then be copied freely to other pages.

Document authors should not have to take any action for these entries to be created. Links can be made to the publications from other directories, but these links must be created by other IPs.

Publication URLs should include the source location in an HTML comment. It is easier to cut and paste from the source than from PostScript. A nightly daemon can check the presence of these comments. To avoid annoying authors who have prepared links before the date of this report, the daemon should not complain about particularly early papers.

6. **Statistics**

   It is useful to know whether information is being accessed, how frequently, and by whom. For example, if the information is not being accessed, it should be removed. Statistics should be available for this purpose. Useful statistics include:

   - **Most frequently accessed pages**
   - **Access by file name, by hierarchy, by URL**
   - **Access by domain**
   - **Access by host**
   - **Access by time**

7. **WAIS and other Search Indices**

   WAIS and other search indices should be generated regularly to account for substantive information changes.

8. **Information Retirement**

   An information retirement policy should be formalized and automated. For example, every directory with an .expire file should contain entries which list the file and the date on which it should be removed. Owners should be notified before files are removed. When file owners leave MSID, their files should be inherited by someone else.

9. **Garbage**

   A automatic nightly daemon should check for garbage. For example, editor backups are garbage. So are unreferenceable files. For example, an HTML file that can not be reached from our initial page is unreferenceable and should be moved to a "junk" directory. Mail should be sent to the owner. If the file is not moved, it should be deleted in some time period, such as three months.

10. **Bogus References**

    References that are not valid should be corrected. A nightly daemon should check for such references and mail the author requests to fix them. The daemon should be able to account for pages that are temporarily unavailable.

11. What's New/Hot

What's New/Hot sections are useful on many pages. These should all be automated by nightly daemons.

12. CGI Scripts

CGI scripts allow the ability to substantially lower the maintenance cost of preparing HTML pages. For example, HTML pages produced by CGI scripts are easy to move around from host to host or directory to directory. Similarly, it is possible to update headers/footers on a large number of pages without actually having to physically edit the pages.

We see CGI scripts as an important and frequently overlooked tool in reducing the manual maintenance of HTML pages. On the other hand, CGI scripts require some moderate programming skills. (Without sufficient skill, CGI scripts can just make the maintenance problem even worse.)

13. Relative vs. absolute URLs[1]

An absolute URL contains a complete path specification.

Relative URLs are generally used in links between related documents; and each relative URL contains a partial path specification relative to the file in which the link is contained. Within a group of related documents, relative URLs are preferred because they:

- allow movement of a group of documents to a new location without change to the contained links,
- require less typing

Absolute URLs should be used to link "less closely" related documents or groups of documents.

IPs must decide what constitutes a "group" of related files. We recommend that divisions first be made based on topic, and secondly on control. For instance, if there is a group of closely related documents on one topic but some of the documents are maintained by separate organizations, teams and/or individuals, then the secondary split should be based on who maintains the information, that way local, coordinated control is maintained so that the links remain valid.

14. Server configuration

Information server configurations should be periodically examined for incremental improvements which will reduce manual information maintenance. HTTP server side includes are a good example.

Additionally, URL maintenance can be reduced by using HTTP server defined aliases. This HTTP server feature allows changes to the upper levels of the

1. See http://www.w3.org/pub/WWW/Addressing/rfc1738.txt for the complete specification for URLs.

directory structure (what's defined in the alias) without affecting URL specifications. Therefore, we recommend the use of HTTP server defined aliases.[1]

D. Duplication & Gaps

Due to the differing formats for different dissemination mechanisms, there is a potential for duplication of information. To a lesser extent, duplication can also exist due to different views. Duplication tends to increase maintenance costs as well as disk space.

1. Incompatible Formats

Incompatible formats are a necessary evil which we accept. For example, it may be appropriate for some directories to contain both index.html and README files.

2. Logical Views

It is often convenient to present the same information in different ways. For example, one user may find it more helpful to find a publication by looking in a common publications page. Another user may want to find publications in associated project page.

Multiple views are most easily created by using HTML pages. However, it may occasionally be convenient to create different directories for each view with multiple links to the same files to support FTP access. Different names for the same files may also occasionally be appropriate.

3. Divergence

It may be useful for a file to diverge into multiple files where one file changes while another remains the same. For instance, published documents are expected to remain the same even if they contain errors. Of course, corrected documents are useful too. However, both should remain available and should be clearly identified.

4. Gaps

We should strive to avoid gaps in our information. This is almost certainly unachieveable, however it is made easier by logical and orthogonal organizations.

E. Security

There are restrictions on who can read and who can write information. These restrictions are not meant to be onerous but they are complicated, in part, due to the multitude of dissemination mechanisms and the interactions between them. Inadvertent information exposure is a fact of life and for this reason, it is a healthy attitude to assume that all information is not private, and may be read by anyone. Fortunately, accidental writing is much less likely than accidental reading.

1. For information regarding MSID's HTTP server aliases, email: web-questions@cme.nist.gov

All of the dissemination mechanisms allow reading of files in the /proj/elib/online structure which have world-read permissions. Directory listings require world-executable permissions. Additional restrictions are provided by particular dissemination mechanisms. Symbolic links cannot be used to circumvent protections.

1. **FTP**

   Files may be uploaded to the /proj/elib/pub/upload area by anyone. See the section "File Ownership and Protection" for more information.

2. **HTTP**

   Our HTTP server allows read-access to files in /proj/elib/internal to some NIST networks and users outside of MSID.

   A page should be accessible to all MSID staff indicating the accessibility of MSID files. Ideally, it should be generated directly from the srm.conf server configuration file and any .htaccess files (potentially) located throughout the directory structure.

F. Documentation

Local practices concerning information dissemination should be documented. For instance, nowhere is it written down what MSID's STEP On-line Information System (SOLIS) or FTP administrators do. Changes to practices described in this document should be made to this document. The "References" section (below) is a start at a set of references.

All of these documents should be made easily accessible. Physical copies should be available in our library. On-line copies should be available through the Web.

IV. Acknowledgments

This document includes input from many people. Thanks to the following people for suggestions, feedback, and proofreading: Debbie Fowler, Peter Hart, Josh Lubell, Julie Parker, Steve Ray, Gaylen Rinaudot, Carolyn Rowland, Craig Schlenoff, and Selden Stewart.

V. References and Bibliography

[1] Libes, Don, "*The NIST EXPRESS Server – Usage and Implementation*", NISTIR 5323, Gaithersburg, MD, May 11, 1994.

[2] Liu, Cricket, et al, "*Managing Internet Information Services: World Wide Web, Gopher, FTP, and more*", O'Reilly & Associates, Inc., ISBN: 1-56592-062-7, December 1994.

[3] Potts, Michelle and Hart, Peter, *"The Manufacturing Information Technology Transfer Project Electronic Library: An Implementation Description"*, NISTIR 5656, Gaithersburg, MD, March, 1995.

[4] Rinaudot, Gaylen, "*The IGES/PDES ORGANIZATION STEP On-Line Information Service (SOLIS)*", NISTIR 5511, October, 1994.

[5] Schlenoff, Craig, "*World Wide Web and Mosaic: User's Guide*", NISTIR 5453, June 1994.