

Standards for DSEP

*Standards for the Implementation of a Deposit
System for Electronic Publications (DSEP)*

Table of Contents

Table of Contents	3
1 Introduction	5
1.1 Introduction	5
1.2 Document scope	5
1.3 Structure of this document	5
1.4 References	5
1.5 Glossary	6
2 Standards for Information Packages in a DSEP	9
2.1 Information Package format	9
2.2 Representation information format	10
2.3 Proposed metadata elements	10
3 Standards for the Ingest process	11
3.1 Standards for receiving SIPs	11
3.2 Standards for quality assurance	17
3.3 Standards for AIP generation	18
3.4 Standards for generation of descriptive information	18
3.5 Standards for the co-ordination of updates	19
4 Standards for the Archival Storage process	20
4.1 Standards for receiving AIPs from Ingest	20
4.2 Standards for AIP storage	20
4.3 Standards for management of the storage hierarchy	23
4.4 Standards for media refreshment	24
4.5 Standards for error checking	24
4.6 Standards for disaster recovery	24
4.7 Standards for providing AIPs to Access	25
4.8 Sending AIPs to Preservation	26
5 Standards for the Data Management process	27
5.1 Standards for administration of archive database functions	27
5.2 Standards for handling query requests	27
5.3 Standards for database updates	27
6 Standards for the Access process	29
6.1 Standards for the co-ordination of access activities	29
6.2 Standards for the generation of DIPs	29
6.3 Standards for the delivery of responses to consumers	30
7 Standards for the Administration process	31
7.1 Standards for submission agreements with Producers	31
7.2 Standards for the management of the system configuration	31
7.3 Standards for the development and maintenance of archive standards and policies	31
7.4 Standards for submission audit	32
7.5 Standards for interacting with Management	32
7.6 Standards for the activation of stored requests	32
7.7 Standards for customer service	32
7.8 Standards for monitoring changes in the Designated Communities	32
7.9 Information Technology management	32
8 Standards for the Preservation process	34
8.1 Medium Preservation	34
8.2 Technology Preservation	34
8.3 Intellectual Preservation	34
8.4 Preservation strategies	34
9 Architectural design of a DSEP	36

Standards for DSEP

- 9.1 *Standards for requirements specification*..... 36
- 9.2 *Standards for specifying user and system interaction*..... 36
- 9.3 *Standards for IT architecture specification*..... 36
- 9.4 *All DSEP processes on one machine vs. a networked solution*..... 36
- 10 *Standards for implementation of a DSEP* 38
 - 10.1 *Coding standards* 38
 - 10.2 *Documentation standards for technical documentation*..... 39
 - 10.3 *Standards for software configuration management* 39
 - 10.4 *Standards for testing* 40

1 Introduction

1.1 Introduction

This document describes the standards that can be used in the realisation of a Deposit System for Electronic Publications (DSEP) and should be seen as an addendum to [Ref-2].

The standards in this document are presented as-is; a selection of standards or product selection is not made since this is not the goal of this document. The selection of standards to use should be part of a requirements specification and/or design phase for the implementation of an actual DSEP.

1.2 Document scope

This document gives an overview of standards for a DSEP. It is based on the main functions of the Open Archival Information Systems (OAIS) model [Ref-1]: Ingest, Archival Storage, Data Management, Access and Administration. In addition the Preservation process which is part of the DSEP is also incorporated.

The DSEP processes “Delivery & Capture” and “Packaging & Delivery” are outside the scope of this document. However, the communication interface with these processes (e.g. how a publication is sent from Delivery & Capture to Ingest) is within the scope of this document.

1.3 Structure of this document

In section 2 standards for Information Packages are discussed.

In chapters 3 to 8 standards for the OAIS-processes, as scoped within DSEP, are discussed. For this, each main process is split up in its sub-processes (as defined in [Ref-1]), for which the different available standards are listed. If applicable it will be mentioned how these standards can be used to fulfil the requirements; the possible constraints on the use of standards are also listed.

In section 9 some standards are presented for the architectural design of a DSEP.

Section 10 presents standards that can be used during the implementation of a DSEP.

1.4 References

- [Ref-1] Reference Model for an Open Archival Information System (OAIS), Consultative Committee for Space Data Systems, Draft recommendation for space data system standards, CCSDS 650.0-R-1, May 1999, <http://www.ccsds.org/documents/pdf/CCSDS-650.0-R-1.pdf>.
- [Ref-2] The Deposit System for Electronic Publications (DSEP) – A Process Model. NEDLIB REPORT SERIES, Koninklijke Bibliotheek, Den Haag, 2000.
- [Ref-3] Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation, A Report to the Council on Library and Information Resources, Jeff Rothenberg, January 1999, ISBN 1-887334-63-7, <http://www.clir.org/pubs/abstract/pub77.html>.
- [Ref-4] An Experiment in Using Emulation to Preserve Digital Publications, Jeff Rothenberg, RAND Europe, April 2000, NEDLIB REPORT SERIES, Koninklijke Bibliotheek, Den Haag, 2000. <http://www.kb.nl/coop/nedlib/>.
- [Ref-5] TCP/IP Tutorial and Technical Overview, IBM redbook, sixth edition, October 1998, GG24-3376-05, <http://www.redbooks.ibm.com/abstracts/gg243376.html>.
- [Ref-6] Call for tender, Depot van Nederlandse Elektronische Publicaties, Koninklijke Bibliotheek, The Hague, 2 December 1999.
- [Ref-7] Archival theory and information technologies: the impact of information technologies on

archival principles and methods, C.M. Dollar, Macerata: University of Macerata Press, 1992.

[Ref-8] Metadata for long term preservation, Catherine Lupovici, Julien Masanès, NEDLIB REPORT SERIES, Koninklijke Bibliotheek, Den Haag, 2000.
<http://www.kb.nl/coop/nedlib/>.

[Ref-9] Requirement for the Digital Research Collection, P.S. Graham, College & Research Libraries, July 1995, Vol. 56, No. 4, p. 331-339

1.5 Glossary

This Glossary describes terms which are most relevant in the context of the NEDLIB project and this deliverable. Further information about the terms can be found in the following glossaries:

- List of NEDLIB Terms, Geneviève Clavel, NEDLIB REPORT SERIES, Koninklijke Bibliotheek, Den Haag, 2000.
- A full glossary compiled by NEDLIB : <http://www.kb.nl/coop/nedlib/glossary.pdf>
- A glossary of Internet terms by Internet Literacy Consultants (TM):
<http://www.matisse.net/files/glossary.html>.
- A glossary of computer oriented abbreviations and acronyms called BABEL by Irving Kind:
<http://www.access.digex.net/~ikind/babel96b.html>.
- A glossary for NCSA Mosaic and the WWW World Wide Web users:
<http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/Glossary/GlossaryDL.html>.
- A glossary for the project Communication in Physics:
<http://www.phys.uva.nl/fnsis/onderzoek/comm/papers/glossary.htm>.
- Harrod's Librarians' glossary : 9,000 terms used in information management, library science, publishing, the book trades, and archive management / compiled by Ray Prytherch. - 8th ed. - Aldershot : Gower , c1995

AIP	Archival Information Package
AIX	Advanced Interactive Executive. AIX is an open operating system from IBM that is based on a version of UNIX. AIX/ESA was designed for IBM's System/390 or large server hardware platform. AIX/6000 is an operating system that runs on IBM's workstation platform, the RISC System/6000.
API	Application Programming Interface
ASCII	ASCII is the most common format for text files in computers and on the Internet. In an ASCII file, each alphabetic, numeric, or special character is represented with a 7-binary digit binary number (a string of seven 0s or 1s). 128 possible characters are defined.
base64 encoding	Encoding used to convert data from its binary or bit-stream representation into the 7-bit ASCII set of text characters.
CIFS	Common Internet File Services. A statefull, connection-oriented, network file-sharing protocol developed by IBM and Microsoft as part of LAN Manager. CIFS is the native file sharing protocol for systems running Windows for Workgroups, Windows95 and Windows NT. Sometimes referred to as SMB.
CRC	Cyclic Redundancy Code (see section 3.2.1.2 on page 17)
DAS	Direct Attached Storage (see section 4.2.1.3 on page 21)
DIP	Dissemination Information Package
DRM	Digital Rights Management

DSEP	Deposit System for Electronic Publications
FTP	File Transfer Protocol (see section 3.1.1.4 on page 12)
HP-UX	Hewlett Packard UNIX
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol (see section 3.1.1.1 on page 11)
IEEE	Institute of Electrical and Electronics Engineers
IPC	Inter Process Communication (see section 3.1.1.7 on page 14)
ITIL	IT Infrastructure Library (see section 7.9.1 on page 32)
ITPM	Information Technology Process Model (see section 7.9.2 on page 33)
Linux	Linux is a UNIX-like operating system that was designed to provide personal computer users a free or very low-cost operating system comparable to traditional and usually more expensive UNIX systems.
MIME	Multipurpose Internet Mail Extensions (see section 3.1.1.3 on page 12)
NAS	Network Attached Storage (see section 4.2.1.3 on page 21)
NFS	Network File System. NFS is an ONC application-layer protocol for peer-to-peer, distributed, file system communication. NFS allows a remote file system (often located on a file server) to be mounted transparently by client workstations. The client cannot perceive any functional difference in service between remote and local file systems (with trivial exceptions). NFS is the most popular ONC service, has been licensed to over 300 computer system vendors, runs on an estimated 10 million nodes and is a de facto UNIX standard. See also VFS, ONC, and NFSv3.
OAIS	Open Archival Information System
ONC	Open Network Computing
PDI	Preservation Description Information
RAID	Redundant Array of Independent Disks. RAID is used to increase the reliability of disk arrays by providing redundancy either through complete duplication of the data (RAID 1, i.e., mirroring) or through construction of parity data for each data stripe in the array (RAID 3, 4, 5). RAID 5, which distributes parity information across all disks in an array, is among the most popular means of providing parity RAID since it avoids the bottlenecks of a single parity disk.
RDF	Resource Description Framework (see section 2.2 on page 10)
RFC	Request for Comments. An RFC is an Internet formal document or standard that is the result of committee drafting and subsequent review by interested parties. Some RFCs are informational in nature. Of those that are intended to become Internet standards, the final version of the RFC becomes the standard and no further comments or changes are permitted. Change can occur, however, through subsequent RFCs that supersede or elaborate on all or parts of previous RFCs.
RTF	Rich Text Format. RTF is a file format that lets you exchange text files between different word processors on different operating systems.
SAN	Storage Area Network (see section 4.2.1.3 on page 21)

SET	Secure Electronic Transactions (see section 7.7 on page 32).
SIP	Submission Information Package
SMIL	Synchronised Multimedia Integration Language
SMTP	Simple Mail Transfer Protocol (see section 3.1.1.2 on page 11)
Solaris	Solaris is the computer operating system based on UNIX that Sun Microsystems provides for its family of SPARC-based processors as well as for Intel-based processors.
SQL	Structured Query Language
TCP/IP	Transmission Control Protocol/Internet Protocol
TFTP	Trivial File Transfer Protocol (see section 3.1.1.5 on page 13)
UAF	Underlying Abstract Form
UDP	User Datagram Protocol
UML	Unified Modelling Language (see section 9.1 on page 36)
UNIX	UNIX is an operating system that originated at Bell Labs in 1969 as an interactive time-sharing system. UNIX operating systems are used in widely sold workstation products from Sun Microsystems, Silicon Graphics, IBM, and a number of other companies. The UNIX environment and the client/server program model were important elements in the development of the Internet and the reshaping of computing as centred in networks rather than in individual computers
UVC	Universal Virtual Computer (see section 8.4 on page 34).
Windows NT	Windows NT is the Microsoft Windows personal computer operating system designed for users and businesses needing advanced capability. Windows NT is actually two products: Microsoft NT Workstation and Microsoft NT Server. The Workstation is designed for users, especially business users, who need faster performance and a system a little more fail-safe than Windows 95 and Windows 98). The Server is designed for business machines that need to provide services for LAN-attached computers.
XML	eXtensible Markup Language (see section 2 on page 9)

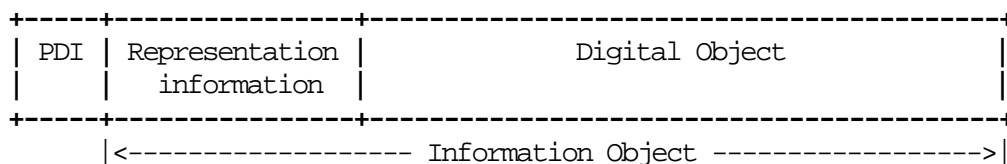
2 Standards for Information Packages in a DSEP

The definition of an Information Package is applicable to the Submission Information Package (SIP), Archival Information Package (AIP) and Dissemination Information Package (DIP):

2.1 Information Package format

In [Ref-1] the Information Package is split up into three parts (see also figure below):

- *Preservation Description Information (PDI)*: contains additional information about the Information Object and is needed to make the Information Object meaningful for the indefinite long-term.
- *Information Object* consisting of:
 - *Representation Information*: enables access to the digital object in a meaningful way; in practice this can include available software that understands (parts of) the Representation Information. This is useful as long as the software executes properly. However for indefinite long-term information preservation, a full and understandable description of the Representation Information is essential [Ref-1] According to [Ref-4], this can be bypassed by recreating the experience of viewing (or even interacting with) the original by use of emulation.
 - *Digital Object*: the actual publication.



The eXtensible Markup Language (XML) can be used to represent Information Packages; more information on XML can be found on the site of the World Wide Web Consortium (<http://www.w3.org/XML/>).

A problem with XML is its inability to easily handle binary data. XML formatted messages only contain human-readable, human-editable text and links to “unreadable” digital data. If an XML message does contain digital data, this can only be done if this data is base64 encoded¹ and placed in the XML message as NDATA (i.e. an unparsed entity with processing instructions).

In a paper “Handling Binary Data in XML Documents” by Lisa Rein (<http://www.xml.com/pub/98/07/binary/binary.html>) more information can be found on handling digital data. She proposes the following methods to include binary data:

- Encoding of binary data, e.g. base64.
- Handle binary data as external objects, this is however not desirable in the DSEP case, since all data for a publication should be contained in one Information Package.
- Use Multipart/related MIME type (as described in RFC 2112, <http://www.rfc-editor.org/rfc/rfc2112.txt>).
- SMIL (Synchronised Multimedia Integration Language) and Binary Data, this does also use external objects and is therefore not suitable.

¹ Note that base64 encoding would increase the size of the digital object by approximately 33%, this could be undesirable. If AIPs are however stored in a compressed form, for instance using standard ZIP-compression, this problem might be less relevant.

2.2 Representation information format

The Representation Information could be structured by use of the Resource Description Framework (RDF). The XML implementation of RDF is described in: “Resource Description Framework (RDF) Schema Specification 1.0”, W3C Candidate Recommendation 27 March 2000, <http://www.w3.org/TR/2000/CR-rdf-schema-20000327>.

RDF is a framework for metadata; it provides interoperability between applications that exchange machine-understandable information on the Web. RDF uses XML to exchange descriptions of Web resources but the resources being described can be of any type, including XML and non-XML resources. RDF emphasises facilities to enable automated processing of Web resources. RDF can be used in a variety of application areas, for example: in resource discovery to provide better search engine capabilities or in cataloguing for describing the content and content relationships available at a particular Web site, page, or digital library.

2.3 Proposed metadata elements

In [Ref-8] a minimal set of metadata elements is proposed. The following are the proposed metadata elements for Representation Information (note that not all these elements are mandatory):

- *Specific hardware requirements*: Description of non-standard platform configuration or hardware requirements.
- *Specific microprocessor requirements*: Description of specific microprocessor instructions set (for instance MMX instructions set) or co-processor.
- *Specific multimedia requirements*: Description of non-standard multimedia hardware requirements.
- *Specific peripheral requirements*: Description of non-standard peripheral devices (for instance a ZIP storage device).
- *Operating system*: The operating system on which the publication can run.
- *Interpreter and compiler*: Piece of program allowing to analyse and execute each statement in a source program or allowing to translate a program expressed in a high level language into machine language.
- *Object format*: Name of the object format.
- *Application*: Name and version of the application.

In [Ref-8] the following are proposed as metadata elements for preservation and description information (note that not all these elements are mandatory):

- *Reference Information*: Information that identifies the Content Information.
- *Assigned Identifier*: Unique identifier that identifies the Content Information.
- *URL*: Location of the document on the World Wide Web.
- *Fixity Information*: Data used to prove the authenticity of an AIP.
- *Checksum*: Information about the use of a checksum.
- *Change History*: Information about every change that has occurred in the digital object and which has implied a change in any metadata for long term preservation.
- *Main metadata concerned*: Information about a change that occurred in the digital object.
- *Tool*: Tool that has been used to operate the transformation on the digital object.
- *Reverse*: Designate either the content or the previous version of the digital object if it has been archived, or the tool to operate the backwards transformation.
- *Other metadata concerned*: Information about other aspects of a change that occurred in the digital object.

3 Standards for the Ingest process

The Ingest process consists of a number of main processes, these are:

- receiving Submission Information Packages (see section 3.1 on page 11);
- performing quality assurance on SIPs (see section 3.2 on page 17);
- generating an Archival Information Package (see section 3.3 on page 18);
- extracting Descriptive Information from AIPs (see section 3.4 on page 18);
- co-ordination of updates (see section 3.5 on 19), i.e. sending AIPs to Archival Storage.

Standards and solutions for these processes are described in section 3.1 to 3.5.

The following documents on Ingest Standards in the OAIS model are available:

- The Archive Ingest Process, JPL Document D-7669, Mike Martin, 10-7-1999, <http://ssdoo.gsfc.nasa.gov/nost/isoas/awiics/ingestmethodology.html>.
- Ingest Standards (and others) in OAIS, CEDARS Project, D. Holdsworth, <http://gps0.leeds.ac.uk/~ecldh/cedars/AWIICS.html>.

3.1 Standards for receiving SIPs

The receive SIP function provides the appropriate storage capability or devices to receive a SIP from the Producer (or from Administration). In the DSEP model only digital SIPs are delivered; these are delivered via some form of electronic transfer. This function provides a *confirmation of receipt* of a SIP to the Producer, which may include a request to *resubmit a SIP* in case of errors resulting from the SIP submission.

The only sources of SIPs in the DSEP model are the Delivery & Capture process and the Preservation process; Ingest receives SIPs from these processes. In the following sections Delivery & Capture and the Preservation are called the 'Producers'.

3.1.1 Available standards

For the communication between a Producer and the Ingest process a number of standard protocols and strategies can be identified, these are described in the following sections.

Most of the information on TCP/IP protocols in this section has been derived from [Ref-5].

3.1.1.1 HTTP/1.1

Hypertext Transfer Protocol (HTTP) 1.1 is described in RFC 2616 (<http://www.rfc-editor.org/rfc/rfc2616.txt>), which obsoletes RFC 2068.

The hypertext transfer protocol is a protocol designed to allow the transfer of hypertext mark-up language (HTML) documents. HTML is a tag language used to create hypertext documents. Hypertext documents include links to other documents that contain additional information about the highlighted term or subject. Such documents may contain other elements apart from text, such as graphic images, audio and video clips, Java applets, active server pages, and even virtual reality worlds.

3.1.1.2 SMTP

Electronic mail (e-mail) is probably the most widely used TCP/IP based protocol. The basic Internet mail protocols provide mail (note) and message exchange between TCP/IP hosts; facilities have been added for the transmission of data that cannot be represented as 7-bit ASCII text.

The Simple Mail Transfer Protocol (SMTP) is a standard protocol for the exchange of mail between two computers, which specifies the protocol used to send mail between TCP/IP hosts. It is described in STD 10/RFC 821 (<http://www.rfc-editor.org/rfc/rfc821.txt>).

SMTP (that is, an STD 10/RFC 821-compliant mailing system) is limited to 7-bit ASCII text with a maximum line length of 1000 characters, which results in a number of limitations:

- SMTP cannot transmit executable files or other binary objects.
- SMTP cannot transmit text data that includes national language characters since these are represented by code points with a value of 128 (decimal) or higher in all character sets based on ASCII.
- SMTP servers may reject mail messages over a certain size.
- SMTP gateways that translate from ASCII to EBCDIC and vice versa do not use a consistent set of code page mappings, resulting in translation problems.
- Some SMTP implementations or other mail transport agents (MTAs) in the Internet do not adhere completely to the SMTP standards defined in RFC 821.

3.1.1.3 MIME

Electronic mail based on SMTP has its problems as described in section 3.1.1.2. Multipurpose Internet Mail Extensions (MIME) is a standard that includes mechanisms to solve these problems in a manner that is highly compatible with existing RFC 822 (<http://www.rfc-editor.org/rfc/rfc822.txt>) standards. Because mail messages are frequently forwarded through mail gateways, it is not possible for an SMTP client to distinguish between a server that manages the destination mailbox and one that acts as a gateway to another network. Since mail that passes through a gateway may be tunneled through further gateways, some or all of which may be using a different set of messaging protocols, it is not possible in general for a sending SMTP to determine the lowest common denominator capability common to all stages of the route to the destination mailbox. For this reason, MIME assumes the worst: 7-bit ASCII transport, which may not strictly conform to or be compatible with RFC 821. It does not define any extensions to RFC 821, but limits itself to extensions within the framework of RFC 822. Thus, a MIME message is one that can be routed through any number of networks that are loosely compliant with RFC 821 or are capable of transmitting RFC 821 messages.

MIME is a draft-standard protocol with a status of elective. It is described in five parts:

- Protocols for including objects other than US ASCII text mail messages within the bodies of messages conforming to RFC 822. These are described in RFC 2045 (<http://www.rfc-editor.org/rfc/rfc2045.txt>).
- General structure of the MIME media typing system and defines an initial set of media types. This is described in RFC 2046 (<http://www.rfc-editor.org/rfc/rfc2046.txt>).
- A protocol for encoding non-US ASCII text in the header fields of mail messages conforming to RFC 822. This is described in RFC 2047 (<http://www.rfc-editor.org/rfc/rfc2047.txt>).
- Various IANA registration procedures for MIME-related facilities. This is described in RFC 2048 (<http://www.rfc-editor.org/rfc/rfc2048.txt>).
- MIME conformance criteria. This is described in RFC 2049 (<http://www.rfc-editor.org/rfc/rfc2049.txt>).

3.1.1.4 FTP

The File Transfer Protocol (FTP) is a standard protocol with STD Number 9. Its status is recommended. It is described in RFC 959 - File Transfer Protocol (FTP) (<http://www.rfc-editor.org/rfc/rfc959.txt>) and updated in:

- RFC 2228 - FTP Security Extensions (<http://www.rfc-editor.org/rfc/rfc2228.txt>).
- RFC 2640 - Internationalisation of the File Transfer Protocol (<http://www.rfc-editor.org/rfc/rfc2640.txt>).

Copying files from one machine to another is one of the most frequently used operations. The data transfer between client and server can be in either direction. The client can send a file to the server machine. It can also request a file from this server.

To access remote files, the user must identify himself or herself to the server. At this point the server is responsible for authenticating the client before it allows the file transfer.

From an FTP user's point of view, the link is connection-oriented. In other words, it is necessary to have *both* hosts up and running TCP/IP to establish a file transfer.

3.1.1.5 TFTP

The Trivial File Transfer Protocol (TFTP) is a standard protocol with STD number 33. Its status is elective and it is described in RFC 1350 - The TFTP Protocol (Revision 2) (<http://www.rfc-editor.org/rfc/rfc1350.txt>). Updates to TFTP can be found in the following RFCs:

- RFC 1785 - TFTP Option Negotiation Analysis (<http://www.rfc-editor.org/rfc/rfc1785.txt>);
- RFC 2347 - TFTP Option Extension (<http://www.rfc-editor.org/rfc/rfc2347.txt>);
- RFC 2348 - TFTP Blocksize Option (<http://www.rfc-editor.org/rfc/rfc2348.txt>);
- RFC 2349 - TFTP Timeout Interval and Transfer Size Options (<http://www.rfc-editor.org/rfc/rfc2349.txt>).

TCP/IP file transfer is a disk-to-disk data transfer, as opposed to, for example, the VM SENDFILE command, a function that is considered in the TCP/IP world as a mailing function, where you send out the data to someone's mailbox (reader in the case of VM).

TFTP is an extremely simple protocol to transfer files. It is implemented on top of UDP (User Datagram Protocol). The TFTP client initially sends read/write request via port 69, then the server and the client determines the port that they will use for the rest of the connection. TFTP lacks most of the features of FTP. The only thing it can do is read/write a file from/to a server.

TFTP has no provisions for user authentication; in that respect, it is an unsecure protocol. Since TFTP does not have any authentication mechanism, the server should protect the host files. Generally, TFTP servers do not allow write access and only allow read access to public directories. Some server implementations also have a host access list.

3.1.1.6 Message queuing

Message queuing products enable applications to exchange information between different applications running on different host platforms, even if the target program or host is not currently running. They take care of network interfaces, communications protocols, and handle recovery after system problems, assuring the delivery of messages.

The problem with message queuing is that messages should not be too large; e.g. it's preferable to handle 1GByte of data as a file rather than as a message.

There is a number of commercial messaging products available:

Product	Supplier	Platforms	Further information
ActiveWorks	Active	Microsoft Windows NT, Sun Solaris, HP-UX, Compaq Tru64, DEC/Digital UNIX, SGI IRIX	http://www.activesw.com/prod/prod.htm
BEA Tuxedo /Q	BEA Systems Inc.	<i>Client:</i> Apple MacOS 7.X, OS/2 3.0, Linux, Slackware, DOS 6.1, Windows 3.1/95/98 <i>Server:</i> Compaq Tru64 Unix, Compaq OpenVMS, HP-UX, IBM AIX, Windows NT, NCR MP-RAS, SCO OpenServer, SCO UnixWare, Sequent DYNIX/ptx, Sun Solaris.	http://www.beasys.com/products/tuxedo/index_tux.html
BusinessWare	Vitria Technologies	Microsoft Windows NT, Sun Solaris, HP-UX, IBM AIX.	http://www.vitria.com/products/businessware.html
DataGate /	Software	Microsoft Windows NT (Intel & DEC	http://www.stc.com/products/eg

Product	Supplier	Platforms	Further information
e*Gate	Technologies Corporation (STC)	Alpha), Sun Solaris, IBM AIX, HP/UX.	ate.html
Mercator	TSI Software	Windows (3.1, 95/98,NT), SCO Open Server, SCO Unixware, OS/400, AIX, HP-UX, SINIX-Reliant, Solaris, Sun OS, Digital Unix, Windows NT for Alpha, OpenVMS for Digital Alpha, DEC VMS, FTX, VOS, MVS, CICS, Unix for the mainframe.	http://www.mercator.com/products/index.html
Microsoft Message Queue Server (MSMQ)	Microsoft Corporation	<i>Client:</i> Microsoft Windows 95/98/2000 and NT. <i>Server:</i> Microsoft Windows NT/2000.	http://www.microsoft.com/msmq/
MQSeries	IBM	Microsoft Windows NT, IBM AIX, IBM AS/400, HP-UX, IBM OS/390, Sun Solaris, MVS	http://www.software.ibm.com/mqseries
Oracle Advanced Queuing	Oracle Corporation	Windows 95/98, NT, Solaris, AIX, Digital Unix, HP/UX, OS/390, OS/2, Intel SCO and Netware.	http://www.oracle.com/database/features/index.html?advque.html
TIB/ActiveEnterprise	TIBCO Software Inc.	Digital UNIX, OpenVMS, NT, Ultrix, Dec VMS, HI-UX, HP/UX, Nextstep, AIX, OS/390, MVS, OS/400, Free BSD, Linux, Win 95, Windows 3.11, NCR UNIX, OS/2, SCO UNIX, Solaris, SRV4, Irix, FTX, SunOS	http://www.tibco.com/products/enterprise.html

3.1.1.7 Inter-process communication

If both the Producers and Ingest process are running on one machine, the process of receiving a SIP by Ingest could be handled using inter-process communication (IPC). This type of communication can either be used to receive the full SIP from Delivery & Capture, or receive a reference to the location of the SIP on the file-system. The last option is preferable, since this can be done much faster, requires less memory and allows for better parallel processing.

The following facilities for inter-process communication are supported by most leading operating systems:

- FIFOs (Named Pipes);
- File locking;
- Message Queues;
- Semaphores;
- Shared Memory;
- Memory Mapped Files;
- Sockets.

3.1.1.8 Security considerations

If the communication channel between Delivery & Capture and Ingest is open to the outside world (i.e. uses a public network), then security measures are needed for this communication channel; here are some solutions to effectively defend oneself against an attack (as mentioned in [Ref-5]). It has to be noted that any of these solutions only solve a single or just a very limited number of security problems. Therefore, a combination of several such solutions should be considered in order to guarantee a certain level of safety and security.

- *Encryption*: to protect data and passwords.
- *Authentication* and *authorisation*: to prevent improper access.

- *Integrity checking and message authentication codes*: to protect against improper alteration of messages.
- *Non-repudiation*: to make sure that an action cannot be denied by the person who performed it.
- *Digital signatures and certificates*: to ascertain a party's identity.
- *One-time passwords and two-way random number handshakes*: to mutually authenticate parties of a conversation.
- *Frequent key refresh, strong keys and prevention of deriving future keys*: to protect against breaking of keys (crypt analysis).
- *Address concealment*: to protect against denial-of-service attacks.

The following protocols and systems are commonly used to provide various degrees of security services in a computer network.

- IP filtering.
- Network Address Translation (NAT).
- IP Security Architecture (IPSec).
- SOCKS.
- Secure Sockets Layer (SSL).
- Application proxies.
- Firewalls, which use a combination of IP filtering, proxy, and SOCKS.
- Kerberos and other authentication systems (AAA servers)
- Secure Electronic Transactions (SET) which uses digital certificates.

The table below summarises the characteristics of some of the security solutions mentioned before and compares them to each other. This should help anyone who needs to devise a security strategy to determine what combination of solutions would achieve a desired level of protection.

	<i>Access Control</i>	<i>Encryption</i>	<i>Authenticat-ion</i>	<i>Integrity Checking</i>	<i>Perfect Forward Secrecy</i>	<i>Address Conceal-ment</i>	<i>Session Monitoring</i>
<i>IP Filtering</i>	y	n	n	n	n	n	n
<i>NAT</i>	y	n	n	n	n	y	y (connection)
<i>IPSec</i>	y	y (packet)	y (packet)	y (packet)	y	y	n
<i>SOCKS</i>	y	n	y (client / user)	n	n	y	y (connection)
<i>SSL</i>	y	y (data)	y (system / user)	y	N/A	n	y
<i>Application Proxy</i>	y	normally no	y (user)	y	normally no	y	y (connection and data)
<i>AAA servers</i>	y (user)	n	y (user)	n	n	n	n

For more information on TCP/IP and security refer to chapter 5 of [Ref-5].

There are many vendors of security software and hardware; the requirements for security are however not yet clear enough to determine which products are most applicable. Below is a, non-exhaustive, list of security products, most of these product suites offer firewalls, encryption, access control and other security measures.

<i>Product</i>	<i>Vendor</i>	<i>Platforms</i>	<i>Further information</i>
Raptor	Axent	Windows	http://www.axent.com/
AccessMaster	Bull	Solaris, Windows NT, AIX.	http://www.bullsoft.com/accessmaster/index.htm
eTrust solutions suite	Computer Associates	Windows NT	http://www.ca.com/etrust/

Product	Vendor	Platforms	Further information
VPN-1, Firewall-1, Cyber Attack Defence System,	Check Point	Windows NT, Solaris, HP-UX, AIX	http://www.checkpoint.com/
Secure	Cisco	Windows	http://www.cisco.com/warp/public/cc/so/neso/sqso/index.shtml
getAccess	enCommerce	Solaris, Windows NT	http://www.encommerce.com/products.asp
SecureWay	IBM	AIX, Windows NT	http://www-4.ibm.com/software/security/
SAFEsuite	Internet Security Systems	Windows NT, Solaris	http://www.iss.net/securing_e-business/security_products/index.php

3.1.2 Requirements and constraints

There were no requirements put down for receipt of SIPs by Ingest. There are however some minimal requirements that could be stated, such as:

- Ability to receive large SIPs; say 18 Gbytes, which is the data capacity of a DVD.
- Ability for Producers, i.e. Delivery & Capture and Preservation, and Ingest to run on different machines.
- etc.

The preferred solution depends on a number of factors:

- Are the Delivery & Capture, Preservation and Ingest processes running on a single machine or on separate machines?
- How reliable should the communication be? I.e., is it allowed/possible to re-send SIPs or should delivery to Ingest be guaranteed.
- What kind of security should be offered?
- Is the availability of the processes guaranteed or can processes temporarily be unavailable?
- etc.

If the processes are located on one machine inter-process communication will suffice. This is however not very likely (see also section 9.4 on page 36). Since the processes will probably be located on different machines, communication should be done using FTP. Message queuing solutions are not desirable for sending SIPs, since SIPs are too large to be considered messages. A combination of message queuing and FTP could however be possible; consider the following example:

1. Producer P that runs on host H^p prepares a SIP s and puts it on temporary storage.
2. Producer P puts a message m^p on the message queue of the Ingest process I that is running on host H^i ; this message contains enough information for I to find s on H^p .
3. I reads m^p from its message queue and starts an FTP session to H^p .
4. s is fetched² and its correctness and completeness are checked by I .
5. A reply message is put on the message queue of P ; this reply indicates if the receipt of s was successful. If correct, s can be removed from P 's temporary storage, if not P could create a corrected SIP and/or (re-)submit m^p to I 's message queue (depending on the error returned in the reply from I).

² Note that this approach is slightly different to the usual OAIS approach in which producers send their SIPs to Ingest and Ingest does not do a fetch. This approach does however allow for a possible temporary unavailability of the Ingest process.

If however a solution is chosen where producers actively put their SIPs on the file-system of the machine where Ingest is running, then an FTP-daemon should run on this same machine. Producers of SIPs can use an FTP-session to put their SIPs in a specific directory (this could be a specific directory for each producer) on the file-system of the “Ingest”-machine; the Ingest process(es) poll this directory, and if a file appears in it, it will pick it up and process it. This does however imply that the machine Ingest runs on should always be up and reachable. If not, the Producer should have some re-try mechanism for putting the SIP on the “Ingest”-machine.

A serious constraint to the protocol is the degree of security it offers; it should for instance not be possible that a third unauthorised party could send SIPs to Ingest, that would be accepted into the system as if they originated from an official source. This particular example could be handled by a communication protocol if it offers authentication. One choice could also be to “place” the full DSEP including Delivery & Capture behind a firewall, thus excluding unauthorised parties.

3.2 Standards for quality assurance

The quality assurance function validates the successful transfer of the SIP to the staging area. For digital submissions, these mechanisms might include cyclic redundancy codes (CRCs) or checksums associated with each data file, or the use of system log files to record and identify any file transfer or media read/write errors.

A check on the syntax of the SIP and a check to determine if it contains the minimal number of components are also necessary.

3.2.1 Available standards

3.2.1.1 Checking the SIP syntax

How the syntactical correctness of a SIP can be determined is highly dependent on the format of a SIP. The exact definition of the SIP format is outside the scope of this document; some information on standards for information packages can however be found in section 2 on page 9.

If, as proposed in section 2, SIPs are formatted using XML, then XML parsers can be used to check the syntax of the SIP.

3.2.1.2 Standards for fault detection

Fault detection is usually achieved by use of checksums; a checksum is added to the data and is the result of a checksum-algorithm applied to the data. The sender computes a checksum for the data, adds the checksum to the data (header). On receipt the receiver re-computes the checksum and checks if this matches the checksum computed by the sender; if these don't match an error has occurred during transmission.

Most communication protocols (see section 3.1.1 on page 11) already have build-in fault detection mechanism. It could however be desirable to add additional checksums to the SIP or individual components of the SIP. Below a short list of checksum algorithms is given:

- *MD5*: MD5 is probably the strongest checksum algorithm most people will need for common use. The official description is in the Internet RFC 1321, The MD5 Message-Digest Algorithm (<http://www.rfc-editor.org/rfc/rfc1321.txt>).
- *CRC*: There are lots of examples of CRC source code floating around the Internet.
- *Unix sum compatible*: The Unix command `sum` comes with every Unix system and therefore the defacto standard. One implementation is at the uunet archives (<ftp://ftp.uu.net/packages/bsd-sources/usr.bin/cksum/>). The GNU textutils (<ftp://prep.ai.mit.edu/pub/gnu/textutils-1.14.tar.gz>) package also has a Unix compatible version.
- *POSIX*: The POSIX.2 group has defined the `cksum` utility, which is a 32-bit CRC implementation. Check the GNU textutils package for an example (<ftp://prep.ai.mit.edu/pub/gnu/textutils-1.14.tar.gz>).

3.2.1.3 Security considerations

Security could be seen as part of quality assurance, since a SIP submitted by a malevolent person might contain erroneous data which compromises the overall quality of the contents of the archive. For information on security refer to section 3.1.1.8 on page 14.

3.2.2 Requirements and constraints

In [Ref-6] it is stated that the completeness of a SIP should be checked. The completeness of a SIP can only be checked if rules have been defined which describe the contents of a SIP.

Another requirement states that it should be detectable if the data in a SIP has been changed. Use of checksums and the authentication of the sources of SIPs can accomplish this. If data could be changed by malicious acts from outsiders, security is needed to assure the quality of SIPs presented to Ingest.

3.3 Standards for AIP generation

In [Ref-1] the following definition is given for the generate AIP function:

“The generate AIP function transforms one or more SIPs into one or more AIPs that conform to the archive’s data formatting and documentation standards. This may involve file format conversions, data representation conversions or reorganisation of the content information in the SIPs. The generate AIP function may issue report requests to Data Management to obtain reports of information needed by the AIP generation function to produce the descriptive information that completes the AIP.”

Note that the producers of SIPs are within the DSEP domain, since these are Delivery & Capture and Preservation; therefore the publication should already be in an acceptable format. AIP generation will be rather trivial since the digital object contained in the SIP can be transferred to the AIP without conversion³.

3.3.1 Available standards

The AIP should contain a digital object: it could consist of any combination of the following:

- the unconverted copy of the original publication.
- a converted/migrated version of the original publication
- software to run the publication
- metadata.

XML can be used to format an AIP; see section 2 on page 9 for AIP standards. It would be desirable to have a checksum computed to check the integrity of the digital object in a later stage (see section 3.2.1.2 on page 17).

3.3.2 Requirements and constraints

AIP generation in DSEP is relatively straightforward and consists mainly of repackaging parts of the SIP content into one or more AIPs.

3.4 Standards for generation of descriptive information

Descriptive information in the DSEP environment is called bibliographic description and is managed outside the DSEP. Other metadata, such as technical, structural and preservation metadata, is extracted

³ Generating the digital object contained in the SIP is by no means trivial, this is however the responsibility of the Delivery & Capture process and therefore outside the scope of this document.

from the SIP and or generated during Ingest and provided to Data Management. This includes metadata to identify and retrieve AIPs.

3.4.1 Available standards

Description of metadata standards is outside the scope of this document; some links for more information on metadata are listed below:

- General links:
 - <http://www.ifla.org/II/metadata.htm>
 - <http://www.ukoln.ac.uk/metadata/>

Error! Hyperlink reference not valid.

Information on metadata specific to the NEDLIB project can be found in [Ref-8]; some of the information contained in this document can also be found in section 2.3 on page 10.

3.4.2 Requirements and constraints

N/A

3.5 Standards for the co-ordination of updates

The co-ordinate updates function is responsible for transferring the AIPs to Archival Storage and the Descriptive Information to Data Management. Transfer of the AIP includes a storage request and may represent an electronic, physical, or a virtual (i.e., data stays in place and only a reference is passed) transfer. The co-ordinate updates function transfers the storage identification information for the AIP to the Data Management entity.

3.5.1 Available standards

For communication standards refer to section 3.1.1 on page 11.

When the AIP is sent to Archival Storage and metadata for that AIP is sent to Data Management both the AIP and metadata should either be stored or not be stored, e.g. an AIP cannot be stored by Archival Storage if Data Management refuses to accept its metadata. What is stated here indicates that sending an AIP and its metadata should be seen as a transaction. For this the two-phase commit protocol should be used in which database changes required by a transaction are initially stored temporarily by each database. The transaction monitor then issues a "pre-commit" command to each database, which requires an acknowledgement. If the monitor receives the appropriate response from each database, the monitor issues the "commit" command, which causes all databases to simultaneously make the transaction changes permanent.

3.5.2 Requirements and constraints

The chosen solution depends highly on the architecture chosen. If processes are implemented on different machines, network communication protocols should be used for communication. Furthermore it is very probable that the AIPs are stored on another machine, and in another database than the metadata.

Since AIPs can be very large these should be transferred using FTP. The size of typical metadata will be much smaller; therefore one of the message queuing solutions (see section 3.1.1.6 on page 13) could be used, the use of FTP is however also possible.

In the OAIS model an AIP is stored using a storage request; requests should have a reply, therefore the used communication protocol should also be usable to send this reply to Ingest.

4 Standards for the Archival Storage process

The Archival Storage process consists of a number of main processes, these are:

- receiving AIPs from Ingest (see section 4.1 on page 20) and adding them to permanent storage (see section 4.2 on page 20);
- managing the storage hierarchy (see section 4.3 on page 23);
- refreshing the media on which archive holdings are stored (see section 4.4 on page 24);
- performing routine and special error checking (see section 4.5 on page 24);
- providing disaster recovery capabilities (see section 4.6 on page 24);
- providing AIPs to Access (see section 4.7 on page 25);
- sending AIPs to Preservation (see section 4.8 on page 26).

Standards and solutions for these processes are described in section 4.1 to 4.8.

4.1 Standards for receiving AIPs from Ingest

4.1.1 Available standards

For possible communication standards between Ingest and Archival Storage refer to section 3.5.1 on page 19.

4.1.2 Requirements and constraints

As stated before, the choice of a communication mechanism is highly dependent on the architecture chosen (see also section 9.4 on page 36). A constraint on this mechanism is that AIPs received from Ingest can be rather large (e.g. a CD-ROM can already contain 650 Mbytes of data and DVDs can contain up to 18 Gbytes of data), therefore file-based communication is preferred above message-based.

4.2 Standards for AIP storage

4.2.1 Available standards

4.2.1.1 Databases

For the storage of large amounts of data, the Relational Database paradigm is currently prevalent. The Object Oriented database paradigm could however be used more and more over the coming years. A number of commercial relational database products is available, some of the most commonly used are shown in the table below.

<i>Product</i>	<i>Supplier</i>	<i>Platforms</i>	<i>Further information</i>
DB2	IBM	Windows, AIX, Linux, Sun's Solaris Operating Environment, OS/2, HP-UX, NUMA-Q, OS/390, VSE and VM, and AS/400	http://www-4.ibm.com/software/data/db2/
Dynamic Server	Informix	Windows NT, UNIX ⁴	http://www.informix.com/ids2000/
Oracle	Oracle	40+ UNIX ⁵ , VMS, MVS, VM, HP MPE/XL, Siemens, ICL, Novell Netware, OS/2, and Windows NT	http://www.oracle.com/database/
SQL Server	Microsoft	Windows	http://www.microsoft.com/sql/d

⁴ More detailed information on platform support was not available.

⁵ The specific UNIX platforms were not specified by Oracle.

<i>Product</i>	<i>Supplier</i>	<i>Platforms</i>	<i>Further information</i>
			efault.htm
Sybase Adaptive Server	Sybase	Sun Solaris, IBM RS6000, Digital UNIX, HP UX, Windows NT	http://www.sybase.com/product/databaseservers/

For more theoretic background information on relational databases refer to:

- An introduction to database systems, C.J.Date, 1 September 1999, Longman Higher Education; ISBN 0-2013-8590-2.
- Fundamentals of Database Systems, R.Elmasri & S.B.Navathe, 3rd ed., 1999, Addison-Wesley, ISBN 0-8053-1755-4.

There is also a number of Object Oriented databases commercially available, these are shown in the table below:

<i>Product</i>	<i>Supplier</i>	<i>Platforms</i>	<i>Further information</i>
Object Database Management System	Versant	SUN Solaris 2.5.1 / 2.6 / 2.7, Windows NT 4.0, Windows 2000, Windows 95, Windows 98, IBM/AIX, HP, SGI, DEC Alpha Unix, Linux RedHat 5.2 / 6.0	http://www.versant.com/
Objectivity /DB	Objectivity	Windows, Intel Linux, Sun Solaris, IBM AIX, Compaq Tru-64, HP-UX, SGI Irix.	http://www.objectivity.com/
ObjectStore	Excelon	Microsoft Windows 2000, NT, 98; Sun Solaris SPARC, Intel; Hewlett-Packard HP-UX; Silicon Graphics IRIX; IBM AIX; Compaq Tru64 UNIX	http://www.odi.com/products/objectstore.html

4.2.1.2 Digital Library products

The databases mentioned in 4.2.1.1 on page 20 can be used for storage of the AIPs; the insertion, management and retrieval of these AIPs is however better handled using a Digital Library product which uses the database (optionally). These Digital Library products offer standard specialised functions for storage management, search and retrieval of digital publications. Some of commercial Digital Library products available are shown in the table below.

<i>Product</i>	<i>Supplier</i>	<i>Platforms</i>	<i>Supported databases</i>	<i>Further information</i>
Agora	Agora	not known	not known	http://www.agora.de/
Digital Library (integrated into Content Manager)	IBM	AIX, Mac OS, Windows 95, Windows 98, Windows NT	DB2 Oracle	http://www-4.ibm.com/software/is/dig-lib/ http://www-4.ibm.com/software/data/cm/
Documentum 4i	Documentum	not known	not known	http://documentum.com
Intranet based Document Library Solution	Seasoft	Windows NT	SQL Server	http://www.seasoft.be/scripts/home.asp
Lifelink	OpenText	UNIX, Windows NT	Sybase, Oracle, SQL Server	http://www.opentext.com/livelink/

4.2.1.3 Mass storage

There are many types of media available for mass storage: e.g. magnetic disc, optical disk, and magnetic tape. Within DSEP multiple media types will be in use:

- magnetic disc could for instance be used for frequently used data and as temporary storage for SIPs, AIPs and DIPs during the processing steps of the DSEP processes,
- optical disks or disk arrays could for instance be used by Archival Storage to store AIPs, and
- magnetic tape could for instance be used for backups or by Preservation to preserve the digital publication.

There are a number of possible types of storage architecture:

- **Direct Attached Storage (DAS)**

The Direct Attached Storage model can be thought of as the way computer systems worked before networks. The DAS model contains three basic software layers: application software, file system software (which is part of the UNIX or NT operating system software) and disk controller software. The elements are usually located close together physically and operate as a single entity. In the DAS model, the UNIX or NT application software makes an I/O request to the file system, which organises files and directories on each individual disk partition into a single hierarchy. The file system also manages buffer cache in UNIX.

When database applications are installed, the database software sometimes bypasses the UNIX buffer cache and provides its own cache as with Oracle's System Global Area (SGA). The file system or database software determines the location of the I/O requested by the application and manages all caching activity. If the data is not in cache, the file system then makes a request to the disk controller software that retrieves the data from its disks or RAID array and returns the data to the file system to complete the I/O process.

- **Network Attached Storage (NAS)**

The NAS model was made possible because NFS for UNIX or CIFS for Windows allows a file system to be located or mounted remotely and accessed over a network, instead of residing on the application server. In the NAS model the application software makes a network request for I/O to the remote file system mounted on a NAS server. The file system on the NAS server determines the location of the data requested by the client application and manages all caching activity. If the data is not in cache, the NAS file system then makes a request to the disk controller software, which retrieves the data from its disks or RAID array and returns the data to the NAS file system, which returns the data to the client across the network. Compared to DAS, NAS servers off-load all of the functions of organising and accessing all directories and data on disk and managing cache. This frees the server's CPU to do additional work, thereby reducing potential CPU bottlenecks.

- **Storage Area Network (SAN)**

In the SAN model, the file system continues to reside on the application server. As in the case of the DAS model, the server performs its normal file system functions of organising and accessing all files and directories on each individual disk partition and managing all caching activity. Reads and writes to the disk controller software, however, must be sent over the Storage Area Network, thereby adding latency to the I/O process and reducing performance. Unlike NAS there is no reduced workload for the server processor to offset added network latencies. However, a SAN does offer the benefits of storage resource pooling and LAN-free backup.

In the paper "A Storage Architecture Guide" by Auspex Systems the storage architectures mentioned above are described in more detail and compared to each other. To obtain a copy of this paper, refer to http://www.auspex.com/storage_guide2k/guide1.htm.

The table below lists some of the vendors of storage products (both hardware and software), most of which support the storage architectures mentioned above. Note that this list is by no means exhaustive.

<i>Supplier</i>	<i>Further information</i>
Compaq	http://www.compaq.com/storage/
Dell	http://www.dell.com
EMC	http://www.emc.com/products/systems/
Hewlett Packard	http://www.hp.com/storage/
Hitachi Data Systems	http://www.hds.com/storage/

<i>Supplier</i>	<i>Further information</i>
IBM	http://www.storage.ibm.com/
Legato	http://www.legato.com/
StorageTek	http://www.stortek.com/StorageTek/hardware/
Sun Microsystems	http://www.sun.com/storage

Not only the storage devices are of importance; so is the storage management system used to manage the storage devices. On <http://www.stormgt.org/> information can be found on storage management, this site contains links to industry white papers and sites of vendors of storage management systems; the table below contains a non-exhaustive list of storage management products:

<i>Product</i>	<i>Vendor</i>	<i>Further information</i>
ARCserveIT	Computer Associates	http://www.cai.com/arcserveit/
StorageWorks	Compaq	http://www.compaq.com/products/storageworks/storage_mgmt_software.html
Tivoli Storage Manager	IBM	http://www.tivoli.com/products/index/storage_mgr/
GEMS	Legato	http://www.legato.com/products/management/
Storage Management Suite	NTP Software	http://www.ntpssoftware.com/products/sm/

4.2.2 Requirements and constraints

The storage of AIPs is constrained by many factors, such as the properties of the Digital Library product, the underlying database product and the supporting storage hardware.

One of the main requirements for the storage mentioned in [Ref-6] is that the storage capacity of the supporting software/hardware should be in the tera-byte range. Note that this is not per definition a requirement for the database, since the database could only contain references to the physical location on a file-system of the AIPs. If however AIPs are stored in a database both maximum storage and maximum transaction size are important factors⁶.

4.3 Standards for management of the storage hierarchy

The function for management of the storage hierarchy positions the contents of the AIPs on the appropriate media based on storage management policies, operational statistics, or directions from Ingest.

4.3.1 Available standards

The storage management product used should manage the storage hierarchy, see section 4.2.1.3 on page 21.

4.3.2 Requirements and constraints

One of the things that could influence the storage hierarchy is that the AIPs and the catalogue used to find AIPs could be located on different servers. If a storage management product is used it should support this.

⁶ The maximum size for an AIP depends on the maximum size of a transaction; if the AIPs are stored in a database the transaction should contain the full AIP.

4.4 Standards for media refreshment

During media refreshment the content of the stored AIPs may not be changed, only the medium on which the AIP is stored may change. The media refreshment strategy must select a storage medium, taking into consideration the expected and actual rates of errors encountered in various media types, their performance, and their costs of ownership. Media-dependent attributes (e.g., tape block sizes, CD-ROM volume information) should not be part of the AIP⁷.

4.4.1 Available standards

The considerations on the preservation of digital information as stated in [Ref-3] should be kept in mind when dealing with media refreshment.

In most cases refreshment will mean copying the data to new media of the same type. Only if the current media will no longer be supported in the foreseen future should the data be migrated to another type of media or if a new media type offers significant improvements to the current media type. Most storage management products offer some form of media refreshment (see section 4.2.1.3 on page 21).

4.4.2 Requirements and constraints

Migrating large amounts of data can be a laborious and time-consuming process, which could lead to prolonged unavailability of service. A DSEP operations department should therefore manage this activity; factors like timing and estimated duration are of importance when planning migration.

4.5 Standards for error checking

The error checking function provides statistically acceptable assurance that no components of the AIP are corrupted during any internal Archival Storage data transfer.

4.5.1 Available standards

Standards for fault detection have been described in section 3.2.1.2 on page 17. For the digital data contained in an AIP the checksum should be calculated and be added to the AIP.

4.5.2 Requirements and constraints

N/A

4.6 Standards for disaster recovery

A disaster recovery function provides a mechanism for duplicating the digital contents of the archive collection and storing the duplicate in a physically separate facility. This function is normally accomplished by copying the archive contents to some form of removable storage media but may also be performed via hardware transport or network data transfers.

4.6.1 Available standards

There are basically three types of backup:

- *Full backup*: a complete backup of every single file.
- *Incremental backup*: Incremental backups include files that were created or changed since the last backup (that is, the last full or incremental backup). To achieve this, the status of each file must be recorded either within the backup software or through the use of the archive attribute of the files. If no previous backup was made, an incremental backup is equivalent to a full backup.

⁷ These media dependent attributes can however be part of the digital data contained in the AIP, since media dependent data should also be seen as part of the publication and may be needed to recreate the publication.

- *Differential backup*: A differential backup includes all files that were created or modified since the last *full* backup. Note the difference between incremental and differential: incremental backups save files changed since the last (incremental or full) backup, whereas differential backups save files changed since the last full backup. In some publications, a differential backup is also called a *cumulative incremental* backup.

If making backups for data recovery in case of disasters, it is advisable to store backups at a different geographical location. Different types of backup strategy are possible for geographically spread backup, these are:

- *No backup*: No provision is made for disaster recovery.
- *Periodic Backup*: At a certain time, a consistent copy of everything required to recover to that point is made, and is sent to a safe location
- *Ready-Roll-Forward*: Data update logs are sent to a safe location. Recovery will be to the last log data set received.
- *Roll-Forward*: A shadow copy of data is maintained at a recovery site. Data update logs are collected there and periodically applied to the shadow copy.
- *Real-time Roll-Forward*: Similar to roll-forward, except that updates are transmitted and applied at the same time they are being logged in the production site.
- *Real-time Remote-Update*: The capability to update both the primary and shadow copy of data, prior to sending the transaction response or completing a program/task.

The most commonly used devices for backup are tape devices. The total backup strategy is however highly dependent on the type of database and storage management product used. Most commercial databases for instance offer facilities for backups, if however the storage management product also offers this facility, only one of the two facilities should be used.

4.6.2 Requirements and constraints

The requirements stated in [Ref-6] indicate that a full backup of data should be available. It is however not feasible to create a full backup every day of all data stored in DSEP. There should therefore at least be daily or hourly incremental backups and maybe one weekly full backup, this can for instance be done on a Sunday, since the number of new publications entered on a Sunday will probably be lower than on other days.

Creating backups and restoring backup can be time consuming, especially if the amount of data is large. Restoring all of the data stored in DSEP within a few hours will be virtually impossible if the only type of backup would for instance be tape. The only way to assure fast restart of service is if the whole of the data stored in DSEP was mirrored on a separate site in a different geographic location.

A more profound study is needed to determine the backup strategy; a full backup strategy can only be designed once the hard- and software architecture for DSEP has been determined.

4.7 Standards for providing AIPs to Access

Copies of stored AIPs are provided to Access. For this the function receives a data request that identifies the requested AIP(s) and provides them on the requested media type or transfers them to a staging area. This function also sends a notice of data transfer to Access upon completion of an order.

4.7.1 Available standards

Internal access to an AIP is always through the identifier therefore searching is not relevant at this stage. Searching is done outside of the DSEP; the result of this “external” search is a list of zero or more identifiers of the publication. Data-Management maps the identifier of the publication to the corresponding AIP-identifiers. The only thing Archival Storage has to do is return the AIP given its identifier. Therefore Archival Storage only has to retrieve the AIP from storage and then transfer it to Access; this can be done using SQL.

The actual transfer of AIPs could be done using one of the standard protocols mentioned in section 3.1.1 on page 11. If the AIP is however retrieved using SQL, the database server could already have its own means of transferring the query result (for more information on databases, refer to section 4.2.1.1 on page 20). Since a notice of data transfer has to be sent to Access, some message-based communication could also be used.

The standard protocol for information retrieval is the Z39.50 protocol (<http://lcweb.loc.gov/z3950/agency/>), of which the most recent is the 1995 version (<http://lcweb.loc.gov/z3950/agency/1995doce.html>), this protocol specifies search and retrieval facilities. Z39.50 is however meant for end-users and may be a little bit too heavy for the realisation of Access.

4.7.2 Requirements and constraints

Since AIPs can be rather large a file-based communication is preferred above message-based communication; this architecture issue should be addressed during the design phase for DSEP.

4.8 Sending AIPs to Preservation.

Any AIP stored by Archival Storage should be sent to the Preservation process, upon request..

4.8.1 Available standards

For possible communication standards between Archival Storage and Preservation refer to section 3.1.1 on page 11.

4.8.2 Requirements and constraints

Since AIPs can be rather large a file-based communication is preferred above message-based communication; this architecture issue should be addressed during the design phase for DSEP.

5 Standards for the Data Management process

The Data Management process consists of a number of main processes, these are:

- administration of the archive database functions; maintaining schema and view definitions, and referential integrity (see section 5.1 on page 27);
- handling query request from Access (see section 5.2 on page 27);
- receiving database updates (see section 5.3 on page 27).

Standards and solutions for these processes are described in section 5.1 to 5.3.

5.1 Standards for administration of archive database functions

The integrity of the Data Management database should be maintained; this database contains descriptive information, system information and preservation information. Descriptive information identifies and describes the archive holdings, system information is used to support archive operations, and preservation information is needed for the preservation process.

5.1.1 Available standards

The used digital library and database products (as mentioned in 4.2.1 on page 20) should offer Administrative functions. These functions should for instance be used for:

- Creating the database; i.e. creating database descriptions, table descriptions, etc.
- Controlling database access.
- Auditing of database activities.
- Backup and recovery of the database contents.
- Database optimisation and performance tuning.

5.1.2 Requirements and constraints

N/A

5.2 Standards for handling query requests

Queries are received from Access and should be executed to generate a *result set* that is transmitted to the requester.

5.2.1 Available standards

All commercial relational databases offer a Structured Query Language (SQL) interface, which can be used to perform queries on the data in the database.

The Z39.50 protocol specifies a search facility, but as mentioned in section 4.7.1 on page 25 the Z39.50 protocol may be a bit too heavy.

5.2.2 Requirements and constraints

N/A

5.3 Standards for database updates

Data base updates are updates or modifications of existing information in the Data Management database; the main sources of these updates are Ingest and Preservation

5.3.1 Available standards

All commercial relational databases offer a Structured Query Language (SQL) interface, which can be used to update the data in the database. They also support two-phase commit for updates (see section 3.5.1 on page 19), in which each update is handled as a transaction.

5.3.2 Requirements and constraints

N/A

6 Standards for the Access process

The Access process consists of a number of main processes, these are:

- co-ordination of access activities (see section 6.1 on page 29);
- generation of Dissemination Information Packages (see section 6.2 on page 29);
- delivery of responses (DIPs, result sets, reports) to consumers (see section 6.3 on page 30).

Standards and solutions for these processes are described in sections 6.1 to 6.3.

6.1 Standards for the co-ordination of access activities

The interface that co-ordinates the access activities handles consumer requests. These consumer requests can be divided into three types:

- query requests, which are executed in Data Management and return immediate result sets for presentation to the user;
- report requests, which may require a number of queries and produce formatted reports for delivery to the consumer;
- orders, which may access either or both Data Management and Archival Storage to prepare a format DIP for on- or of-line delivery.

In the DSEP model the Packaging & Delivery process instead handles much of these tasks.

6.1.1 Available standards

The standard protocol for information retrieval is the Z39.50 protocol (see section 4.7.1 on page 25). Z39.50 is however meant for end-users and may be a little bit too heavy for this purpose. It is more probable that Packaging & Delivery presents Access with a list of “publication-identifiers” of publications to be retrieved, simple SQL queries could be used to handle these requests.

An obvious interface for consumers to submit requests would be using the HTTP protocol via a web-browser. Other interfaces such as FTP or message queuing could also be considered (see section 3.1.1 on page 11). Security is also a factor in the co-ordination of access activities (see section 3.1.1.8 on page 14).

6.1.2 Requirements and constraints

N/A

6.2 Standards for the generation of DIPs

For the generation of DIPs, dissemination requests are accepted, AIPs are retrieved from Archival Storage and copied to the staging area for further processing (e.g. conversion to a different output format).

6.2.1 Available standards

How an AIP is obtained from storage depends on the facilities offered by Archival Storage (see section 4.7 on page 25).

When generating a DIP, conversion may be needed depending on the format the user requires. Some form of Digital Rights Management (DRM), e.g. digital watermarking, will also be needed.

6.2.2 Requirements and constraints

Not every format can be converted to any other format without any loss of information; e.g. if a web-page containing graphics is converted to ASCII the graphics can't be converted. There may therefore be constraints on the form in which certain DIPs can be delivered to a user.

6.3 Standards for the delivery of responses to consumers

For the delivery of responses to consumers the intended recipient has to be identified and the transmission procedure has to be determined.

6.3.1 Available standards

Standards for communication have been described in section 3.1.1 on 11. The Open Archives initiative (<http://www.openarchives.org/>) aims to set some standards for this.

6.3.2 Requirements and constraints

The most likely standards to be used for delivery of responses to consumers will be via HTTP and/or FTP, if it can be assumed that the primary interface for consumers to DSEP is a web-browser. If however the time between the request of the consumer and the reply by Access is too long, the sending of replies using mail (i.e. MIME format) should be considered.

Again, much stays uncertain here since not enough is known about the hardware architecture (see also section 9.4 on page 36).

7 Standards for the Administration process

The administration process consists of a number of main processes, these are:

- soliciting and negotiating submission agreements with Producers (see section 7.1 on page 31);
- maintaining configuration management of system hardware and software (see section 7.2 on page 31);
- developing and maintaining archive standards and policies (see section 7.3 on page 31);
- auditing submissions to ensure that they meet archive standards (see section 7.4 on page 32);
- interacting with Management (see section 7.5 on page 32);
- activating stored requests (see section 7.6 on page 32);
- providing customer support (see section 7.7 on page 32);
- monitoring changes in the Designated Communities (see section 7.8 on page 32).

There are almost no technical standards available to fill in the Administration processes, since most administration processes consist of non automated procedures. Furthermore the only task it seems to have within DSEP is the generation of reports. It is however possible to map the Information Technology Process Model (ITPM) from IBM onto most OAIS Administration functions; this is shown in section 7.9.2. Another model which could be used is the IT Infrastructure Library (ITIL), this is presented in section 7.9.1.

7.1 Standards for submission agreements with Producers

With Producers submission agreements and submission schedules have to be agreed upon. This process is a process of human-interaction and cannot be automated.

For DSEP a standard submission agreement could be devised, which would be used as template for submission agreements with specific Producers. This standard submission agreement also describes the information from the Producers that is needed to create the metadata for a publication (see also [Ref-8]).

7.2 Standards for the management of the system configuration

7.2.1 Configuration management

Part of the management of the system configuration is the maintenance of the integrity and tractability of the configuration. It is not entirely clear what this means in the OAIS model, but if it refers to standard Configuration Management practices as known in Software Engineering then there are standards tools available for this.

There are non-commercial tools available such as RVS and CVS (<http://www.cyclic.com/>), and commercial tools are also available (see <http://www.cmtoday.com/yp/commercial.html>).

7.2.2 System and performance monitoring

How system and performance monitoring should be implemented depends on the architecture and implementation used to realise the DSEP.

7.3 Standards for the development and maintenance of archive standards and policies

The development and maintenance of archive standards and policies comprises determining format standards and document standards, and the definition of storage management policies and migration policies. On these standards and policies formal agreements have to be made; this process cannot be automated.

7.4 Standards for submission audit

Submissions should be audited to determine whether they meet archive standards. . The correctness of the submission includes the correctness of the packaging format according to agreed standards, and the correctness and completeness of the metadata supplied with the publication.

The audit of submissions can be computer aided; humans should however always be involved in the audit.

7.5 Standards for interacting with Management

The interaction with management consists of setting the overall policies for the DSEP; automation of these processes is not feasible.

7.6 Standards for the activation of stored requests

Most relational database systems (see section 4.2.1.1 on page 20) offer some means to define stored procedures. These stored procedures can contain a sequence of SQL commands and can be activated at any time.

7.7 Standards for customer service

Within the OAIS model customer service refers to the billing and payment process.

One of the standards for on-line billing is the Secure Electronic Transaction (SET) protocol (see <http://www.setco.org/>). The SET Specification is an open technical standard for the commerce industry developed by Visa and MasterCard as a way to facilitate secure payment card transactions over the internet. Digital certificates create a trust chain throughout the transaction, verifying cardholder and merchant validity, a process unparalleled by other internet security solutions. A full listing of SET certified products is to be found at <http://www.setco.org/cgi-bin/vsm.cgi>.

7.8 Standards for monitoring changes in the Designated Communities

Monitoring changes in the Designated Communities actually implies that new user requirements are gathered. Gathering user requirements is a process that involves interviews with users, workshops, creation of use-cases, etc., and can therefore not be automated (it can however be aided by tools). There are however many standards for gathering user requirements. Note however that heavy requirements analysis is not always needed; users should have the ability to submit change requests to request minor changes (e.g. support of a new file format).

7.9 Information Technology management

7.9.1 The IT Infrastructure Library (ITIL)

The IT Infrastructure Library was initially developed, and is still owned by, the CCTA, the UK Government's Central Computer and Telecommunications Agency.

The ITIL philosophy embraces standards, guidelines and best practices in the IT Service Management industry, and a range of ITIL-related products and services have grown up including:

- training,
- qualifications,
- consultancy,
- software tools, and
- user groups (the itSMF).

From its beginnings in the UK and Europe, the ITIL has rapidly become a world standard for measuring and improving standards of IT service delivery. The ITIL is a major non-proprietary set of

IT service standards, crossing the boundaries of infrastructure type and industry, to provide objective measurement of service quality across the whole IT service spectrum.

In short ITIL covers the following areas:

- Service Delivery;
- Software Support;
- Computer Operations;
- Security Management.

For more information on ITIL refer to <http://www.proactive-sv.com.au/public.htm>.

7.9.2 The Information Technology Process Model (ITPM)

Early in 1994, IBM initiated a project which goal it was to develop a common industry model that would assist in the identification and management of functions and resources to positively exploit IT within a business environment. The basic hypothesis was that a fundamental set of processes was necessary to manage any IT environment, regardless of the organisation structure or technology used.

The IT Process Model contains Eight Process Groups:

1. Satisfy customer relationships
2. Provide enterprise information technology management system
3. Manage information technology business value
4. Realise solutions
5. Deploy solutions
6. Deliver operational services
7. Support information technology service and solutions
8. Manage information technology assets and infrastructure

In the table below the ITPM process groups are mapped onto the OAIS Administration processes. The Archival Information Update and Activate Requests processes cannot be mapped.

		OAIS									
		<i>Negotiate Submission Agreement</i>	<i>Manage System Configuration</i>	<i>Archival Information Update</i>	<i>Physical Access Control</i>	<i>Develop Standards and Policies</i>	<i>Audit Submission</i>	<i>Interact with Management</i>	<i>Activate Requests</i>	<i>Customer Service</i>	<i>Monitor Designated Community</i>
ITPM	<i>Satisfy Customer relationships</i>									X	X
	<i>Provide enterprise information technology management system</i>		X								
	<i>Manage information technology business value</i>					X	X				X
	<i>Realise solutions</i>					X					
	<i>Deploy solutions</i>					X					
	<i>Deliver operational services</i>	X						X			
	<i>Support information technology service and solutions</i>	X	X					X			
	<i>Manage IT assets and infrastructure</i>				X					X	

The ITPM is however not much more detailed than the OAIS model in its descriptions and it is by no means a perfect fit between the two models.

8 Standards for the Preservation process

The impact of electronic information in today's society is ever increasing. As more information becomes available in digital form, along with greater use for research and study, questions and issues of long term access become more critical. Preservation of digital objects needs to be looked at from at least three perspectives: medium preservation, technology preservation and intellectual preservation [Ref-9].

8.1 Medium Preservation

The artefact or medium can decay. Medium preservation is the concern for preserving the medium on which information is stored, such as tapes, disks, optical disks, CD-ROMs and the like (see sections 4.3-4.6 on pages 23-25).

8.2 Technology Preservation

More problematic than medium decay are the rapid changes in the storage formats and in the software that allows electronic information to be of use. We need to be aware that technology obsolescence is a bigger problem than medium decay, and undertake steps for technology preservation.

In addition to refreshing, we need to take other measures, such as migration and emulation:

- Migrating information forward through technology / format stages as they become available and as the old technologies / formats cease to be supported by vendors and the user community.
- Emulating old and obsolete technologies / formats on current technology platforms

8.3 Intellectual Preservation

There remains a third preservation requirement, intellectual preservation, which addresses the integrity and authenticity of the information as originally recorded. The need for intellectual preservation arises because in a digital environment it is not always easy to identify modifications / tampering with the original. Numerous initiatives in digital watermarks / encryption to protect copyrights are currently underway. Verance Digital, for instance, provides high-end digital media copy protection solutions which have been adopted as the world-wide industry standard for DVD Audio copy control and for the Secure Digital Music Initiative (URL: <http://www.verance.com/digital/index.html>).

8.4 Preservation strategies

An effective preservation strategy must take all three perspectives of which the first and the last are already addressed in other sections of this report. Technology preservation, however, is still largely an uncharted area. Although some de facto standards for migration to widely accepted and supported formats exist, for example PDF, not much research has yet been done in the area of emulation.

Without migration of obsolete formats we need to be able to restore, in year 2100, on a machine M2100, data generated in 2000, on an M2000 machine. The basic approach with emulation is to provide in 2100 an emulator running on the M2001 machine capable of running the software and therefore accessing the digital objects of the M2000 machine.

In the document by Jeff Rothenberg [Ref-3] an emulation strategy, based on emulation of the hardware platform is proposed by providing a formal description of the platform which will enable the generation of an emulator in the future, as elaborated in [Ref-3]. Though it requires further research and proof of feasibility, it appears to have many conceptual advantages over the other migration-based approaches suggested and is offered as a promising candidate solution for technology preservation.

In relation to hardware emulation the Verilog Hardware Description Language (HDL) could be looked at. This HDL is described in a publication by the IEEE: 1364-1995 IEEE Standard Description Language Based on the Verilog(TM) Hardware Description Language, ISBN: 1-55937-727-5.

Raymond Lorie, from the IBM research department at Almaden, San Jose CA, USA, takes a different approach. He describes his approach and related technical aspects of long term archiving of digital information in his paper “Long-term Archiving of Digital Information” . Central in his approach is a Universal Virtual Computer (UVC). The UVC is a Computer in its functionality; it is Virtual because it will never have to be built physically; it is Universal because its definition is so basic that it will endure for a very long time.

The UVC can than be used in two modes depending on whether or not it necessary to also preserve the application to access the data in future. When only the data has to be preserved a restore application has to be written on the UVC that extracts the logical data (meaningful for the client) from the physical data (just bits in the bit stream) in order for the UVC to be able to display the information. Only in more complex environment like CD-ROMs with integrated application logic the whole application has to be emulated on the UVC, which is synonymous with the previous described approach by Jeff Rothenberg.

In addition Raymond Lorie’s approach provides the opportunity to check today the correctness of programs that will be used in the future. If a UVC program is written in 2000, it can be tested on a UVC interpreter written in 2000 for an M2000 machine. If ten years later, in 2000+10, a new machine architecture comes up, then a new UVC interpreter can easily be written. It can be checked by running the same UVC program through both the 2000 and 2000+10 UVC interpreter. In other words any UVC interpreter can be checked by comparison with the interpreter of the previous generation.

At the time this document was written there were no standards for the preservation of digital material using emulation. The set-up of a test-bed for emulation for digital preservation has however been described in [Ref-4].

The migration strategy may be used next to the emulation strategy (see [Ref-7]). One could argue that some types of publication cannot be electronically preserved (e.g. you can never recreate the look-and-feel of a paper publication), you can however convert these publications to a format that can be stored electronically.

Below some links to preservation information are listed:

- A regularly maintained bibliography of publications on digital preservation can be found at <http://homes.ukoln.ac.uk/~lismd/preservation.html>.
- Preserving Access to Digital Information (PADI): <http://www.nla.gov.au/padi/>

9 Architectural design of a DSEP

9.1 Standards for requirements specification

There are many standards for requirements specification, the only one stated here is the Unified Modelling Language (UML), since this is a standard modelling language for object-oriented development⁸.

UML is a language for specifying, visualising, constructing, and documenting the artefacts of software systems. Using UML, programmers and application architects can make a blueprint of a project, which, in turn, makes the actual software development process easier.

UML was created at Rational Software by methodologists Grady Booch, Ivar Jacobson, and Jim Rumbaugh with input from other leading methodologists, many software vendors, as well as end-users. Its aim is to unify the various existing systems into a best-of-breed modelling language. UML was adopted as a standard by the Object Management Group (<http://www.omg.org/uml>) in November 1997.

There are many commercial UML modelling tools available, for a large list refer to http://www.objectsbydesign.com/tools/umltools_byCompany.html

9.2 Standards for specifying user and system interaction

One part of specifying user and system interaction is the specification of the system behaviour, this can be done by use of use-cases; these use-cases are part of UML (see section 9.1).

Another part of the specification of user interaction is by the design of the User Interface. The system interaction should also be specified by means of Application Programming Interface (API) descriptions, specifications of used communication protocols and/or specifications of used message formats.

9.3 Standards for IT architecture specification

There are no independent standard methods for architecture specification and only little information can be found on company specific methodologies.

Andersen Consulting has its Eagle Architecture Specification Method (http://www.ac.com/services/eagle/eagl_thought3.html), which does however only look at the technology side of the IT architecture.

IBM uses SIMethod for custom application development, package integration, application maintenance outsourcing, and solution consulting and integration. SIMethod does take into account organisation changes during large implementation projects; SIMethod plans and implements business and technology changes in parallel (there is however no public information available on SIMethod).

9.4 All DSEP processes on one machine vs. a networked solution

When specifying the architecture for DSEP, it should be determined whether the DSEP processes should run on a single machine or if they should run on different machines.

⁸ Note that this method is for object oriented programming, if another programming paradigm is used another modelling method could be better fit.

If all processes run on a single machine the processes can use standard facilities offered by the operating system, such as inter-process communication (see section 3.1.1.7 on page 14). The implementation of DSEP will be fairly simple if all processes run on one hardware-box, since there is no need for the amount of security that is needed in networked solutions. There are however some negative aspects to this solution:

- The “one hardware-box” solution could lead to poor system performance during peak-load; e.g. if the Ingest process receives a large number of SIPs the performance of the other processes may decrease, thus decreasing the responsiveness of for instance the Access process.
- The “one hardware-box” solution isn’t very robust, if the machine crashes all of DSEP crashes.
- In the “one hardware-box” solution a single very powerful machine is needed to run the DSEP processes, database server and other software.
- It is harder to allow for growth. Note that multiple instances of the same process could run on the same machine, thus potentially increasing system responsiveness. However if more processes are running on the same machine, more processing power and other system resources are also needed on the same machine.

It could be better to have the DSEP processes running on different machines and have them communicate using a network. In the networked solution:

- Processes run on different machines, therefore if one process experiences peak-load, it does not necessarily affect the other processes.
- The crash of a single machine does not necessarily imply the crash of the whole DSEP. Processes could be duplicated on different machines and even if they are not duplicated a crash could only imply that part of the functionality of DSEP is temporarily unavailable (e.g. if the “Ingest-machine” crashes publications can still be accessed since Access, Archival Storage and Data Management are still available).
- Less powerful and thus cheaper machines can be used. For a possible database server a powerful machine might still be needed; for processes such as Ingest and Access much simpler machines may be sufficient.
- It is easier to grow by increasing the number of processes. Note however that the system design should then allow for multiple instances of DSEP processes running on multiple machines thus making growth easier but design more complicated since you are now designing a distributed computing environment.

It should be noted that the networked solution requires security and message/file buffering. The security is needed since communication may use networks that are open to anyone. A form of message/file buffering is needed to allow for robustness in the communication, i.e. if the receiving side in a communication session is temporarily unavailable this should not create a problem for the sending side.

10 Standards for implementation of a DSEP

10.1 Coding standards

When coding the parts of the DSEP not provided by standard products, there should be some coding standards for the developers to follow. Code can be written by dozens of developers, each with their own style. By adopting coding standards and enforcing them through code inspections, you make sure that all the code has the same feel - making it easier to understand and modify the code in the future.

Coding standards are used to enforce:

- Naming conventions for global and local variables, procedures, classes, files, etc.
- Formatting conventions for the code: how should a function declaration be formatted (placement of braces, indentation, etc), how should a variable declaration be formatted, etc.
- Comment conventions, e.g. each procedure is preceded by comment describing the function of the procedure, the input variables of the procedure, the possible return values of the procedure.
- File organisation, e.g. where should header-files be placed.
- Conventions for writing portable code; needed if programming in C/C++ for multiple platforms; the standard for this is ANSI-C.

There are many standards for different programming languages, some general minimal rules are however stated in “In Defence of Coding Standards - How to Create Coding Standards that Work”, Kirrily Robert, Jan. 12 2000 (<http://www.perl.com/pub/2000/01/CodingStandards.html>):

1. The verbosity of all names should be proportional to the scope of their use.
2. The plurality of a variable name should reflect the plurality of the data it contains; e.g. name is a single name, while names is an array of names.
3. In general, follow the language's conventions in variable naming and other things. If the language uses `variable_names_like_this`, you should too. If it uses `ThisKindOfName`, follow that.
4. Failing that, use `UPPER_CASE` for global variables, `StudyCaps` for classes, and `lower_case` for most other things. Note the distinction between words by using either underscores or `StudyCaps`.
5. Function or subroutine names should be verbs or verb clauses. It is unnecessary to start a function name with `do_`.
6. Filenames should contain underscores between words, except where they are executables in `$PATH`. Filenames should be all lower case, except for class files which maybe in `StudyCaps` if the language's common usage dictates it.

The rules stated above are language independent; for specific languages extra rules can be added to the rules.

A huge amount of coding standards can be found on the World Wide Web, the quality of these standards is however doubtful since none of these are official standards. Below some URLs are given for sites containing information on or examples of coding standards. Note that it is assumed that only Java and C/C++ are used for implementation, other languages should however not be excluded as possible implementation languages.

- *Java coding standards*
 - A Draft Java Coding Standard from the author of Concurrent Programming in Java by Doug Lea: <http://g.oswego.edu/dl/html/javaCodingStd.html>
 - Java coding standards form Sun: <http://java.sun.com/docs/codeconv/html/CodeConvTOC.doc.html>
- *C/C++ coding standards*
 - Multiple links on C/C++ programming style: <http://cplus.about.com/compute/cplus/mstyle.htm>

- Birkbeck College, C++ coding standards for components submitted to a software library of generic components: <http://www.cryst.bbk.ac.uk/~classlib/bioinf/standards.html>
- Programming in C++, Rules and Recommendations, Mats Henricson and Erik Nyquist, 1992: <http://www.cs.umd.edu/users/cml/cstyle/Ellemtel-rules.html>
- An example of coding standards for C/C++ can be found at the GNU-site: <http://www.gnu.org/prep/standards.html>.

10.2 Documentation standards for technical documentation

10.2.1 Types of documents

During implementation multiple documents should be produced, some of these are:

- Module or class design documentation (depending on the programming methodology), which describes the technical detail of the software components that are developed.
- Programming guides and/or API descriptions to aid developers when using specific modules or classes.
- User documentation to help future users with planning and installation tasks, administration tasks, and every day usage⁹.

The IEEE has set standards for a large number of software engineering documents, a list of these documents can be found at http://standards.ieee.org/reading/ieee/std_public/description/se/.

10.2.2 Document format standards

The tools that are used within the Koninklijke Bibliotheek dictate the document format standards; therefore all documents will probably be in MS-Word format.

There are however tools for source code documentation generation, an example of which is JavaDoc (<http://java.sun.com/products/jdk/javadoc/>) that can be used to generate API documentation in HTML using comments in Java source code. It should however be remembered that automated tools almost never lead to user-friendly documentation.

10.3 Standards for software configuration management

When creating a complex system there are a number of factors that can lead to problems:

- Work is done by a team;
- The end-product is built gradually;
- The end-product consists of many components with inter-dependencies;
- These components can be specified, designed, produced and changed independent from each other;
- More than one configuration can be needed; e.g. one for test and one for release.

In such a complex environment some form of Configuration Management is needed. A large list of papers on Configuration Management can be found on <http://www.cmtoday.com/yp/papers.html>. There are also many tools available for Configuration Management: both non-commercial tools such as RVS and CVS (see <http://www.cyclic.com/>), and commercial tools (see <http://www.cmtoday.com/yp/commercial.html>).

⁹ One could argue that the developers are not the right people to write user manuals, they should however supply input to these user manuals.

10.4 Standards for testing

10.4.1 Software test methods

Depending on the system being tested and the method of development used different test strategies may be employed. A test strategy is a general approach to testing rather than a particular type of test. Three common examples of testing strategy are:

- *Top-down testing*, which tests the high levels of a system before looking at the more detailed components;
- *Bottom-up testing*, which is the opposite - starting with the lower levels and working up the hierarchy;
- *Stress testing*, which pushes the system to or beyond its specified limits to see how it handles high system load.

Testing may either be in the form of *white-box* or *black-box*. Black box, or function based, testing is used to test all functions and non-functional attributes of a system. The aim of the test is to see what the output is when data is fed into a program and how well the program functions. It is not concerned with the internals of the software, i.e.: its not of importance how the end result is reached, as long as the end result is reached. Three commonly used black-box methods are:

- *Equivalence partitioning* in which a set of classes of input conditions is identified, where each set covers a large set of other possible values. From these classes a minimal number of test cases is then identified. So, if the input required is a number between 1 and 1000, the test cases would be one valid number (e.g. 397) and 2 invalid numbers (<1 and >1000).
- *Boundary value analysis* is similar to equivalence partitioning but values are selected that cover the edge of each valid range – so for the above example we would choose 0, 1, 1000 and 1001.
- *Error guessing* - this is an ad hoc approach used to identify tests that are likely to expose errors. Examples of this may be using null values or blanks.

White box (structural) testing looks at the internal parts of the system - what is happening inside the program. Whereas black box testing will only exercise 50-70% of the code, white-box testing should exercise the rest of the code to some level. Ideally we would test each entry-to exit path within each module but this is usually prohibitive. A problem with white-box testing is that this is only possible if you have access to the source code or if debug facilities have been added. This type of testing is therefore most likely to occur in the early phases of software development when still in the unit and integration testing phases (see section 10.4.2). When performing system and acceptance tests white-box testing can be very difficult if not impossible.

10.4.2 Software test phases

When testing software a number of phases can be distinguished, in which different types of tests are performed:

1. Unit testing ensures that the individual modules or units operate correctly. This is done independently of other system components.
2. Integration testing tests collections of modules that have been aggregated into sub-systems. Many of the errors that arise here do so because of problems with the interface between these modules.
3. System testing occurs when the sub-systems have been integrated to complete the system. This is when the entire system can be tested against the System Specification and any problems with unanticipated interactions between the sub-system and system components are discovered.
4. Factory, site and user acceptance testing is the final phase before the system is handed over to the client. It is usually part of the contractual agreement that the client will only accept the system for operational use once it has satisfied this stage.

10.4.3 Software test standards

The IEEE has published a number of documents describing standards for test documentation:

- 829-1983 IEEE Standard for Software Test Documentation (<http://www.orguss.demon.co.uk/swtest/ieee/>).
- 1008-1987 IEEE Standard for Software Unit Testing
- 1012-1986 IEEE Standard for Software Verification and Validation Plans
- 1059-1993 IEEE Guide for Software Verification and Validation Plans

The tables of contents for these documents can be found at http://standards.ieee.org/reading/ieee/std_public/description/se/.