Document Name			
Document Title	Significant Properties Testing Report: Electronic Mail		
Work Package			
Author(s) & project role	Gareth Knight, Digital Curation Specialist		
Date	12 February 2010	**Filename**	email-significant-properties-reportv1
Access	☑ Project and JISC internal		☑ General dissemination

Document History		
Version	**Date**	**Comments**
1.1	12 February 2010	Copied text into updated template
1.0	30 March 2009	Final version
0.3	10 March 09	Second draft of testing report
0.2	20 August 08	First draft of Significant properties testing report
0.1	13 February 08	Preliminary list of properties and criteria outlined in 'Framework for the definition of significant Properties'

Contributors

The following people have made direct or indirect contribution to this report: Adrian Brown, Stephen Grace, Lynne Montague, Maureen Pennock and Paul Wheatley.

Intended Audience

This document is written for use by the InSPECT project team, the JISC community and those interested in digital preservation.

Table of Contents

Project Overview

Significant properties are those aspects of a digital record that must be preserved over time in order for the Information Object to remain accessible and meaningful. The InSPECT Project is funded by JISC to investigate methods for maintaining the authenticity of digital resources across digital environments and transformation processes. It has produced a framework for the analysis of significant properties and creating a set of reports that outline its application to four object types - types – audio recordings, raster images, structured text and e-mail – that will contribute and advance strategies for the characterisation and maintenance of significant properties over time.

Purpose of the report

This report examines the notion of significant properties as it applies to electronic mail, a common form of digital communication. It seeks to identify the significant properties of email that must be maintained by examining each of its constituent elements and analyzing their designated function. It goes on to examine strategies that may be utilized to maintain access to e-mail assets in the long-term. Finally, it outlines a set of experiments that were performed by the project team to identify and evaluate tools that may be utilized to convert significant properties from one form to another.

1. Introduction

1.1. Overview of email

Electronic mail, commonly shortened to email, is a method of creating and transmitting primarily text-based messages over an electronic communication system. It is widely used within academia, business and society as a whole as a method for asynchronous communication and interaction with others. An email may be sent to one or more mailboxes, if the sender provides an appropriate address. The distribution method represents the primary characteristic that distinguishes email from other object types or communication methods, rather than any element of its encoding. The majority of emails are transmitted using Simple Mail Transfer Protocol (SMTP), a standard for data transmission across Internet Protocol (IP) networks[1]. The use of the standard also distinguishes it from other text-based communication forms, such as text messaging that use the Short Message Service (SMS) and instant messaging using services provided by Microsoft, Yahoo and others.

The storage and management of email records has been an area of increased scrutiny since the late 1990s. Institutions located in government, academia, as well as the commercial sector increasingly recognize the need and value in maintaining a copy of email records that have passed through their mail servers, although there are interesting variations in the length of time that they are stored and the purpose for which they are used. It is common for institutions to maintain email records to comply with legislative requirements for an allocated period of time (e.g. 10 years) and delete them at a later date. Other organizations, such as the British Library and The National Archives are taking a long-term view of the potential value of email records, interpreting them as cultural artefacts that serve a similar function to letters and other correspondence of earlier time periods. The preservation requirements of email have also received several high profile studies and developments in recent years. Notable work in this area has been performed by the Digital Preservation Testbed at the Dutch National archives and Dutch Ministry of the Interior and Kingdom Relations, as well as later work by the North Carolina State Archives and Smithsonian Institution Archives[2].

1.2. Structure of an email object

As a digital asset, an email may be instantiated as a compound object that may contain a diverse set of structured and semi-structured information. The specification for email messages is defined in several documents, collectively called the Multipurpose Internet Mail Extensions (MIME) that indicate a base set of information and an structure for its organization. An email is likely to consist of two mandatory and a third optional component:

1. *Header:* The header contains structured data specified by the Creator and the software application that indicate the provenance of the message and may provide a brief description of its

[1] http://tools.ietf.org/html/rfc5321
[2] http://www.archives.ncdcr.gov/mail-account

content. Information that may be provided in an email header includes details of the originator (a sender's name and email address), one or more intended recipients (recipient name and email address), the path that was taken to deliver the message (if it is an email that has been transmitted), as well as optional information indicating the subject and keywords.

2. *Body:* The message body frequently contains unstructured text and other data specified by the Creator that represents the primary content of the message. The purpose and method of presentation will vary, dependent on several factors such as the Creator's choices, purpose of the message and the functionality provided by the creation tools.

In addition, an email may contain a third component:

3. *Attachment:* An attachment indicates additional objects that are associated with the message. The attached object(s) may provide useful information that supplements or should be considered alongside the message body, or may simply be an artefact of the software application that was used to write the email of which the author has no knowledge (for example, winmail.dat files are often embedded with emails sent using Microsoft Outlook). The provision of an attachment with an email is optional and will often vary in the content type and method in which it interacts with the message body. An attachment may be simply identified as an associated file or rendered within the message body (e.g. a raster image displayed on a HTML page; a sound recording that may be replayed using a set of controls; or a script that changes an element of the message, such as the date).

1.3. Standards supported by email objects

As a compound object, the constituent components may be drawn from one of several standards that have been created to fulfil different functions and have different capabilities. An examination of the creation and use of emails within a software environment indicates that an email may contain representation information obtained from three inter-related sources:

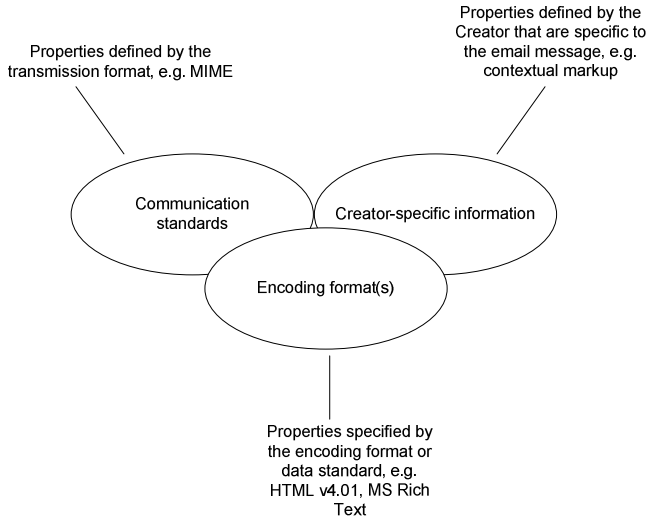

Figure 1: Significant properties of an email may originate from three sources

1.3.1. Communication standards

The de-facto format for the creation of email transmitted over the Internet is defined by the Network Working Group of the Internet Engineering Task Force (IETF) in RFC 5322[3] and a series of other RFC (Request For Comment) memorandums[4] that are collectively referred to as the Multipurpose Internet Mail Extensions (MIME). The format builds upon the earlier Simple Mail Transfer Protocol (SMTP), providing a standard set of elements that must be provided to support the transmission of

[3] http://tools.ietf.org/html/rfc5322
[4] RFC 2045, RFC 2046, RFC 2047, RFC 4288, RFC 4289 and RFC 2049

email messages. The metadata embedded within an email header contains a mix of machine and human-created information produced on the client machine and by one or more mail servers. The metadata provided at the client-side includes: the creator or sender's name which is entered when the email account is created or email application is setup; the user account and mail server from which the message originates (also provided by the user when the email application is setup); the transmission date, which is automatically assigned by the client machine at the point of transmission; and qualitative information provided by the Sender, such as subject and keywords. The metadata provided at the server-side will include: information on the mail servers through which the message has passed and the time stamp.

1.3.2. Encoding formats

The choice of encoding format dictates the form in which information may be represented within the message body. Textual information contained within a message body is commonly encoded as 7-bit ASCII or Unicode. The creation and distribution of messages in 7-bit ASCII plain text was, at one stage the only method to create and store emails. However, contemporary email clients support the MIME standard, which allows the use of other character sets, binary attachments and multi-part message bodies. Emails may be distributed as plain text (i.e. without mark-up), HTML (HyperText Mark-up Language), or Rich Text Format (RTF), which enable the creator to specify attributes of the appearance, such as text font, colour, size, as well as various elements of the page display.

1.3.3. Creator or community-specific information

Information presented in the message body and attachments may be written using bespoke mark-up that is distinct to the creator (a informal mark-up style that a person has developed themselves e.g. pseudo code) or the designated community for which the message is intended. Creator-specific mark-up is difficult to analyse using automated when handling a small number of information objects created by an author. However, digital forensic analysis on a large number of objects may produce accurate results. In addition, the assessor may perform qualitative analysis by interviewing or producing a questionnaire for the author to complete.

1.4. Application of the Performance model

To determine the significant properties of a digital Record, a consistent, formal method of identifying the important aspects is required. The National Archives of Australia (2002) has developed a 'Performance Model', which has been adopted by the InSPECT Project. The principle of the model is that the process of rendering the Information Object in a form that can be understood by a user requires some interaction between the underlying data object and interpretative software. The model is comprised of three components:

1. Source: the encoded data object that contains the text, still images, moving images, or other content for interpretation;
2. Process: the method in which the encoded data is interpreted, e.g. a software tool, an algorithm;
3. Performance: the recreation of the Information Object in a form that can be understood by the user.

A key concept in the Performance model is the recognition that the method in which the Source is processed will vary between members of the Designated Community and are likely to change over time as a result of the evolving technological environment. Although an email object is platform-independent and resistant to the majority of technological changes, the method in which the data object is processed can make a difference in its rendering to the recipient. This is illustrated in Figure 2

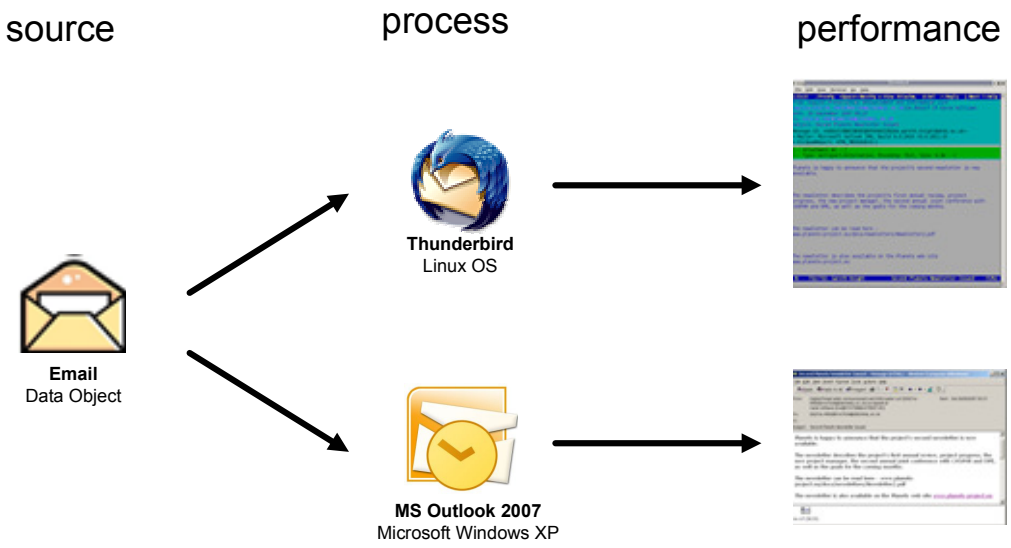

Figure 2 Application of the Performance Model to emails

The visual recreation of the email message differs considerably between the two software applications. This raises the question of what is the essence of the performance that must be retained. What are the requirements of the email object to understand the information contained within or, to reverse the question, what can be removed before it becomes non-intelligible?

2. Testing requirements

2.1. Significant properties that must be maintained

The identification of properties of a digital object that are worthy of preservation is not a simple task that can be analysed based upon a set of universal rules. A set of rules defined for one category of digital object may prove to be too restrictive when applied to unusual variations, or inappropriate for other object types. Instead, the InSPECT Project team has developed a methodology to identify factors that establish the authenticity and integrity of the Information Object through a combined technical and epistemological approach.

During the process of investigating the creation, storage and use of emails in a research environment it was found that the classification of significant properties was influenced by four key elements:

1. The form that the creator has chosen to express an intellectual or artistic idea and the method that they have used to communicate information
2. The function for which the digital object has been created to perform or the aims and objectives that its use will achieve.
3. The method in which information is encoded and stored in a digital environment, influenced by the encoding format and data standards in use.
4. The interpretation of the audience – the intended recipient of the email or an unknown future user – that is accessing the email information to achieve an objective.

The authenticity requirements of an email share many similarities with other types of object, such as unstructured and structured text. The message body contains text or other types of content that convey information. However, the function that an email performs in a digital environment requires the consideration of additional requirements for authenticity. Each email may possess complex inter-relationships with other emails (multiple emails that compose a thread) and intra-relationships between sub-components (e.g. the relationship between an email and attached objects). An email is often used as a form of information transmission between two or more users, which requires provenance information to be understood. An email may contain text that can be understood only in the context of an email that has been sent previously. Alternatively, the context of the message may differ according to the sender and the recipient. For example, an email that contains an invitation for authors to contribute papers to a conference may be interpreted in different contexts – an invitation

that is posted by the event organisers to a mailing list may represent an open invitation for authors to contribute; an forwarded invitation that is sent by an person to a colleague in the same organisation may represent implicit expectation that the recipient should write a paper for the conference. The context information of who created the email, the date it was sent and the intended recipients provide valuable information on the provenance of the message itself and should be maintained.

To demonstrate that the authenticity and integrity of an email object has been maintained, the project team recognised the need to determine two types of information:

1. *Information created by an author that is intended for communication to a designated community;*
    a. What information is intended for communication by an author?
    b. How is it stored within the email? Is it contained within the message body, attachment, or both?
2. Information that establishes the provenance of the email, indicating its purpose and the processes through which it was created and transmitted;
    a. Who created the information intended for communication?
    b. When did they create/transmit the information?
    c. What environment did they create it in?
    d. Who is the intended recipient of the email?
    e. When was the email received by the recipient?
    f. What route did the information take to reach the recipient?
    g. Does the information have any relationship to earlier email messages?

The challenge for the curator is to identify characteristics of the email that enable them to address each question, in order to fulfil the required curatorial function of maintaining the authenticity and integrity. It is possible that the curator will be able to answer some, but not all of the questions. The email that is the target of analysis may not contain the necessary information, as a result of its status within the email lifecycle (e.g. an email in a 'sent items' folder will not contain provenance information capable of addressing 2e or 2f) or may have been manually or automatically altered to remove 'superfluous' details (e.g. Microsoft Outlook often removes provenance necessary to address 2e, 2f, or 2g when writing data to a text file.

To develop a list of the properties that may be significant for establishing the authenticity and integrity of an email, the evaluator reviewed several specifications and standards that are widely used for the storage and transmission of emails and attempted to classify each element by the function it performed in isolation or in conjunction with other elements. The evaluator subsequent cross-matched the function of each element to the outlined questions and used the results to establish a set of properties that are worthy of preservation within the designated context.

2.2.1. Message header

A message header of an email contains a large number of properties that are beneficial for recording the communication process within an internet protocol environment. To ascertain the properties of an email message that may be significant, the evaluator consulted the public documentation provided for the MIME standard, identified a set of 31 unique elements that may be encapsulated in an email message and attempted to classify each by the functions that it may perform. As a result of the analysis, it was found that properties performed one or more of three functions that, in combination may be used to establish the authenticity of a transmitted message[5]. Properties may establish:

1. Message provenance indicating the person or software tool responsible for its creation, one or more recipients who are the intended audience and the date/time when it was transmitted.
2. Structural relationship that uniquely identifies the current message and identified its predecessors within a discussion thread.
3. Contextual information that indicates the purpose of the message.

[5] A similar conclusion was made by the Digital Preservation Testbed which examined the use of email messages within governmental departments.

For presentational purposes, the analysis of the message header is broken into four headings for Agent, Structure, Context and transmission route. Each of these broad headings contains sub-elements that may have different levels of importance in different scenarios.

2.2.1.1. Agent information

The function of email as a form of communication between people, organizations and software requires particular consideration of the agents – the creator and intended audience - associated with the message that is being communicated. The RFC memorandum for email refers to five distinct agents that each has a distinct relationship with the email as a digital asset and one another:

1. *Creator:* The Creator refers to one or more agents - people and/or software - that have created the information content contained within an email message.

2. *Sender:* The Sender refers to the agent that has submitted the email to a mail server for delivery. The Sender may be a person who manually submits a message via an email application stored on their client machine or accessible through a remote user interface, or software that automatically sends a message. The Sender may or may not be the Creator of the message. Scenarios in which the Creator and Sender are distinct include: a Sender forwarding a message that they have received from a mailing list or other mail account; and a software tool that emails an error report created by monitoring software to a system administrator.

3. *Recipient:* The Recipients refers to one or more agents to which an email is intended for delivery. The recipient may be a person who has an account with a mail provider, a mailing list that subsequently distributes the message to its subscribers, a software tracking system that logs the information in a database, or a mail account used for other purposes. A Recipient may be classified into one of three categories that may indicate intrinsic information on the Sender's interpretation of the recipient's relationship with the information content:

    a. *Primary recipient:* The 'To' field may be used by the Sender to indicate the primary recipient of the message, e.g. one or more persons to which the message is directed, has particular relevance, or are expected to act upon the information.
    b. *Secondary recipient:* The Sender may indicate a secondary recipient through the use of the Carbon Copy field, e.g. one or more persons may need to be informed of the message, but are not intended to be the primary recipient who is expected to act upon it.
    c. *Tertiary recipient:* The Sender may indicate a tertiary category of recipient through the use of the Blind Carbon Copy (BCC) field. Similar to the secondary recipient, it may indicate that the recipient should be informed of the message, but are not intended to act upon it. The inclusion of a user in BCC rather than CC may indicate that the sender wishes the recipient to be informed of the information, but does not wish other recipients to know that it has been sent to their mail account.

The categories in which the recipient is placed may also provide useful information regarding its intended audience[6]. For example, the primary recipient may be the person who is expected to perform activities specified in the email, while secondary and tertiary recipients are included for the purpose of monitoring the work.

The specification of a designated community – one or more intended recipients – is a characteristic that is unique within the object types being examined within the InSPECT project - raster images, sound recordings and presentation mark-up typically identify the Creator and Publisher, but do not indicate the intended audience. However, it shares similarity with other purpose-specific object types, such as Learning Objects that are created for use by a specific audience[7].

Each Agent may possess five metadata elements that identify the person and their relationship to the email.

---

[6] Resnick, P. (ed.) (2001). RFC 2822 (RFC2822). Retrieved on January 13, 2008 from: http://www.faqs.org/rfcs/rfc2822.html

[7] Ashley, K, Davis, R & Pinsent, E. (2008) Significant Properties of e-Learning Objects (SPeLOs). Version 1.0. Retrieved on March 30, 2009 from http://www.jisc.ac.uk/whatwedo/programmes/preservation/2008sigprops.aspx

Name	Definition	Function Classification	Function description	Examples
local-part	The user account of the Agent assigned by a mail provider. The local-part is identified by alphanumeric characters prior to the @ symbol of an email address.	Context: provenance	Establishes the provenance (and as a result support or contradict its authenticity) of message by identifying the user account that was used to transmit the message.	
domain-part	The host or domain name used by a DNS to indicate the mail provider that handles the email message.	Context: provenance	Establishes the provenance (and as a result support or contradict its authenticity) of message by identifying the domain from which the message originated.	
domain-literal	The IP address of the source or destination domain.	Context: provenance	Establishes the provenance (and as a result support or contradict its authenticity) of message by identifying the machine address from which the message originated	
display-name	A plain text indication of the Agent's name	Context: provenance	Establishes the provenance (and as a result support or contradict its authenticity) of message by identifying the name of the Agent specified for the mail account.	
relationship-type	The relationship that the Agent has with the email message, e.g. creator, sender, recipient (primary, CC, BCC)	Structure: relationship	Establishes the provenance (and as a result support or contradict its authenticity) of message by identifying how each agent relates to the email	

Table 1: Agent information

The storage of Agent details contributes to the establishment of the emails authenticity. However, it is well recognized that details of the Sender and intended recipient can be inaccurate, faked, or hidden. This may be the result of intentional or unintentional mis-configuration on the client machine or mail server (e.g. the correct display-name and date may not be correctly configured on the client machine, the user account and domain may be 'spoofed', sender information may be removed by a mailing list server).

The value of properties for establishing authenticity will vary – an email address (consisting of local and domain part) or domain-literal enable a curator to identify a user account from which the email is purported to originate. However, there is the potential for ambiguity if the email contains the Sender's display-name only (or some variation of the display-name and local or domain-part). E.g. the email may have originated from a Sender with the same name in the same or different institution.

2.2.1.2. Structural information

The use of email as a form of communication between two or more people raises the possibility that some aspects of the information may require consideration of previously received messages to be understood in context. An email is assigned one or more identifiers that may be used to organize several emails with a common purpose into a discussion thread. The identifier indicates a unique value of the current email and references one or more earlier emails to which the current email is a response. The format for the message is defined in RFC-822[8], which specifies it must consist of two parts:

1. A unique identifier assigned by the host, such as an integer number indicating the number of messages submitted to the network, a string derived from the date and time that the email was submitted, or some other generated value. For example, 4D54BC5A400D244F60AF523146BE5F3.
2. The full name of the host and domain from which the email was sent, e.g. KCL-MAIL04.kclad.ds.kcl.ac.uk.

The two parts must be separated by a @ (at sign) and contained with left and right angle brackets.

In addition, an email may contain intrinsic structure that indicates the email possesses one or more attached files. The attachment may be one or more data objects that may be interpreted in isolation (e.g. a report stored as a Microsoft Word document) or that are intrinsic to the interpretation of the message body (e.g. an embedded image).

Table 2 indicates four types of relationship that an email may possess.

Name	Definition	Function Classification	Function description	Examples
message-id	A unique identifier created by the domain from which the email originated that is embedded within the email header. The message-id is found in received emails and is not present in local emails.	Structure: external	The message-id should be used when attempting to understand the relationship between two or more emails that constitute a thread. It is beneficial when the subject line has changed. However, it has only limited use when handling a single email.	<4D54BC5A400D24499EBF985F6AF60AF523146BE5F3@KCL-MAIL04.kclad.ds.kcl.ac.uk>
References	One or more identifiers that may be expressed as a series of message-IDs separated by a space or a	Structure: external	The References should be used to interpret the relationship between the current email and earlier emails. However, it has little value when the object	Thread-Index: Aclfop3/9YSrsX2SRLaUH9KmAJ/waQAgqZAQBCiFsqAADhbg9wAug/bAAAEbNfkAADTqYAAADIRO

[8] Networking Working Group (2001). RFC2822 - Internet Message Format. Retrieved on March 30, 2009 from: http://www.faqs.org/rfcs/rfc2822.html

	tokenized set of words and message identifiers.		to be preserved is a standalone email that does not relate to a previous email, or handling an email where the referenced email has been deleted.	
In-Reply-To	A series of words and message identifiers that indicate the email to which the message is a reply.	Structure: external	In-Reply-To may be beneficial to indicate the relationship between the current email and earlier email within a message body. However, it has been noted that the complicated syntax and syntactically incorrect values cannot be interpreted by many message readers.[9]	In-Reply-To: Your message of 10 Jan 1998 20:22:41 -0000
Attachment	An identifier that indicates one or more attachments associated with the email	Structure: external	The attachment may identify objects that must be identified to understand the message (e.g. a message body containing the text 'please comment on attached document' cannot be understood if the attachment is not referenced. However, it is unnecessary if an email does not contain an attachment.	

Table 2: Structural information that may contribute to the correct interpretation of an email

In combination, 'Message-id' and 'References' or 'In-Reply-To' address the requirements of question 2g. The 'Attachment' element may indirectly address 1a and 1b.

2.2.1.3. Context information

The message header may contain a short description of the topic and purpose of the email, which can be used to address question 1a and 1b. Contextual information may be manually entered by the Sender or automatically generated by their mail application.

Name	Definition	Function Classification	Function description	Examples
subject	A short string that may identify the topic of the message. The subject line may	Context: description	The Subject may provide qualitative information that indicates the message purpose.	Re: InSPECT Email report

[9] Bernstein, D.J. (n.d.). Threading: Message-ID, References, In-Reply-To. http://cr.yp.to/immhf/thread.html

	be blank, indicate the content of the email to which the Sender is replying, or contain other information.		Additionally, it may provide a simple method to sort several emails into a thread when used in conjunction with the received date.	
keywords	Words and phrases that may summarise the content of the message. Keywords may be created by the person or software application that creates, receives the message, or archives the message[10].	Context: description	The Subject may provide qualitative information that indicates the message purpose.	Significant properties, representation information, email, mark-up

Table 3: Contextual information that may contribute to the correct interpretation of an email

2.2.1.4. Transmission Information

Transmission information indicates the time period in which an email was created and received and the route it took to reach a recipient. It may be utilized to address the requirements of question 2b and 2e.

Name	Definition	Function Classification	Function description	Examples
sent-date	The date and time that an email was completed by a Creator and/or transmitted by the Sender, or received by a Recipient	Context: Provenance	The sent-date is obtained from the system settings of the sender's machine. It may indicate the datetime in which an idea was expressed. However, there is the potential that the datetime has been accidentally or deliberately altered, which may result in the value being untrustworthy.	
Received-date	The date and time that an email was received by the recipient's host.	Context: Provenance	Indicates the datetime that an email was received. However, it does not confirm that the email was downloaded or read by a recipient.	
Trace-field	Indicates the route that the email took to travel from the	Context: Provenance	The trace fields are external to the control of the sender and recipient and,	Received: from jeremiah.qub.ac.uk ([143.117.14.19] helo=

[10] The Digital Preservation Testbed Project (2003) released a plug-in for email archiving that enables archivists to create keyword terms when exporting email records from Microsoft Outlook.

	sender from the recipient and when it occurred. A repeatable value consisting of an optional "Return-Path" field and one or more "Received" fields.		therefore may be thought more trustworthy than the Sent and Received date for validation.	mailhub2.qub.ac.uk) by elder mx with esmtp id 1Lgdgl-0007Uc-Km for gareth.knight@ahds.ac.uk; Mon, 09 Mar 2009 11:30:32 +0000

Table 4: Transmission information that may contribute to the correct interpretation of an email

2.2.1.5. Message Body

The message body contains information that has been inserted by a Creator – a person or software – that is intended to be communicated to one or more recipients. It is often the primary type of information content that must be preserved (an exception are emails used to transfer attached data that do not contain any text in the message body). It may contain semi-structured or unstructured information of various types – Its function as a correspondence may impose specific conventions, e.g. the message may begin with a salutation and conclude with a valediction and a message signature[11]. It may contain text written in one or more languages that are organised into one or more paragraphs, tables, lists and other categorisation forms. Information in the message body may be encoded as plain text, HTML, or Rich Text Format. The latter two enable a Creator to specify additional visual attributes, such as layout, colour, size, and so on.

The designation of the significant properties of the message body is potentially ambiguous, which can be argued from two perspectives:

1. *Minimal:* A minimal position argues that the significant properties of the message body should be limited to the visible characters of the message and, to a lesser extent the hidden formatting of the text, e.g. line breaks. Additional mark-up embedded within the message text, such as font size and colour are unnecessary and may be removed. The minimal position may be supported through reference to the intended user of the email message, who may be using a text-based mail client, such as Pine to read an email and may be unable to view the additional mark-up.
2. *Maximum:* The opposing, 'maximum' position is that each and every one of the mark-up elements contained within a message body is significant and should be maintained for accuracy. The mark-up of text may contain intrinsic information for communication. For example, an email may contain the statement, *"Please comment on the text highlighted in red in the following paragraphs*". Although the message text remains understandable, the statement indicates that text colour is used to communicate information which has been lost.

Each perspective has supporters that are able to highlight specific examples to support both approaches. In the circumstance, it may be pragmatic to develop a rules-based system that implements the first or second approach based upon the type of information contained within the message body.

To develop a proposed list of elements that may contribute to the understanding of an HTML or Rich Text-encoded email message, the assessor consulted the element list specified for the Significant Properties of Structured Text report[12] (Montague, 2009) and considered how each element might be applied to the creation of emails messages, based upon its use within the sample data set. The analysis resulted in the identification of a subset of 50 elements which might, in some circumstances be used to communicate intrinsic information on the message.

For brevity, the element list has been classified into eight categories based upon their function and descriptive accuracy (i.e. Can it be explicitly stated that the element provides a semantic meaning, or

[11] Software tools exist that are capable of distinguishing between unique information content and standardized. For example, several mail applications automatically remove a standardized signature.
[12] The Significant Properties of Structured Text report will be published at http://www.significantproperties.org.uk/outputs.html

is a presentational element that may have ambiguous meaning?). It is advised that the reader consult the Significant Properties of Structured Text report for a full description of each element.

Category	Element name	Significant for preservation
Page display	Body background	Yes, in certain circumstances It is considered unlikely that the background display will have a direct contribution to the intellectual content of the message. However, it may contribute to the artistic interpretation of the creator in a limited number of scenarios.
Layout tags mark-up	Paragraph Line break Horizontal rule Div Centre	Yes, in certain circumstances. The page layout may indicate intrinsic information on the relationship between groups of content, or may be used for presentational purposes.
Table layout mark-up	Table width Table caption alignment Table cell alignment Table cell height Table cell width Table cell wrapping	Yes, in certain circumstances. The indicated elements are likely to be used for presentational purposes only. However, there is the potential that the layout may communicate artistic choices for a limited number of objects.
Presentational text mark-up tags	Preformatted text Headings 1-6 Emphasis Strong emphasis Bold Italics Underline Strikethrough Body text colour	Yes, in some circumstances The significance of presentational mark-up is relative – it may be used for informational purposes, indicating key terms (emphasis, underline), the internal structure of a message (Headings), or simply used for presentational purposes. The value of mark-up is relative to the content of an email message. For example, an author may use colour to convey meaning (e.g. for emphasis). Forensic analysis might be performed upon an email object, interpreting the colours relationship to specific word use (e.g. to provide a simple example, the inclusion of the word ‘red’ and #FF0000 in an HTML email may indicate that specific meaning is implied.
Semantic text mark-up	Language Inserted text Deleted text Samp Cite Dfn Code Abbreviation Acronym Quotations Subscript Superscript Address Unordered list Ordered list List item Definition list	Yes Semantic mark-up communicates unambiguous information on the purpose of the enclosed text.
Table semantic mark-up	table caption Table summary Table headers Table footer	Yes, if table is present. Semantic mark-up communicates unambiguous information on the purpose of the enclosed text.

	Table cell scope Table cell abbreviation Table cell axis Table: column group	
Table structural mark-up	Table ID Table column span Table row span	Yes, if navigation is required. Structure mark-up may be used to indicate relationships between key information.
Related objects	Image Link Applet URLs (a href)	Yes, if outside relationships are present.

Table 5: A list of mark-up properties that might be considered significant for the message body

Table 5 provides a list of HTML mark-up that may be useful, however it does not necessarily follow that it is necessary to maintain the entire element list for every email. A rule-based system might be developed that is able to assess the function performed by each element based upon a Curatorial profile (the requirements established by a Curator) and a text analysis of the interpreted relationship between mark-up and text for each person. Table 6 outlines several curation scenarios, based upon message content and curatorial choices and specifies the element list that it is advisable to maintain for each situation.

Message content	**Curatorial profile**	**Properties that should be maintained**
Message body contains tables, URLs and images	The email has artistic value – the curator wishes to maintain an exact recreation of the visual appearance of an email	All 8 categories
Message body contains tables, URLs and images	The curator wishes to maintain information content, but does not require an exact recreation of the visual appearance of the email	Layout tags mark-up; Semantic text mark-up; Table semantic mark-up; Table structural mark-up; Related objects
Message body contains tables, URLs and images	The curator wishes to maintain message text and mark-up that they can confirm contains semantic meaning. Presentational mark-up, which may have ambiguous interpretation is not required.	Semantic text mark-up; Table semantic mark-up; Table structural mark-up; Related objects;
Message body contains tables, URLs and images	The curator wishes to maintain message text and internal mark-up that may provide semantic meaning. However, visual accuracy is not required and external content should be removed.	Layout tags mark-up; Presentational text mark-up tags; Semantic text mark-up; Table semantic mark-up; Table structural mark-up;

Table 6: Scenarios in which mark-up elements in the message body may be significant

Summary

During the analysis it was found that 14 properties of the message header and 50 properties of the message body contributed information that established the authenticity and integrity of the email, but that their value was relative in different scenarios. To consider the variable requirements for different scenarios, the project has identified properties to fulfil three scenarios: a core set of properties that indicate the minimum amount of information necessary to establish authenticity and integrity; a scenario-based approach for understanding the role of an email within a discussion thread; and a maximum 'ideal' set of information.

Core property set
The core property set indicates the minimum amount of information that is considered necessary to establish the authenticity and integrity of the email message

1. Local-part

2. Domain-part
3. Relationship
4. Subject
5. Trace-field
6. Message body with no mark-up
7. Attachments

If the above information is provided, the curator is able to establish the email account from which the message originated, the transmission route of the message (as noted previously, this would not apply to sent emails), the intended recipient(s), the qualitative information to be communicated in the subject and message body and an indication that an attachment has been provided.

Message thread scenario
Email is frequently used as a communication method between two or more people. To understand the context in which a message was created it may be necessary to refer to earlier messages. To identify the thread of a discussion, the following fields should be provided, in addition to the core property set:

1. Local-part
2. Domain-part
3. Relationship
4. Subject
5. Trace-field
6. Message body with no mark-up
7. Attachments
8. Message-ID
9. References

If the above information is provided, the curator will be able to identify the role of the message within a discussion thread and the previous emails to which it is a possible response.

Recommended property set
The recommended property set indicates additional information that should be provided in an ideal scenario, if it is present within the email. The list

1. Local-part
2. Domain-part
3. Domain-literal (if present)
4. Relationship
5. Subject
6. Trace-field
7. Attachments
8. Message-ID
9. References
10. Sent-date
11. Received date
12. Display name
13. In-reply-to
14. Keywords
15. Message body & associated mark-up (see table 6 for scenarios)

3. Methodology

3.1. Representation Formats

Representation format is a general term that describes the method in which information is stored. In its abstract form, a representation format may be applied to many types of information. Restrictions on the type and extent of information are imposed when handling representation formats intended for a specific purpose. To provide a simple example, a representation format for image data is unlikely to be able to contain audio. Limitations may be imposed, even if information is stored in a representation format of the correct type. Specific properties of the information content may be degraded or removed when it is stored in a representation format.

Email messages may be described as a type of compound object that contain a combination of structured and unstructured information. Unlike the Internet protocols used for information exchange, the format used for email storage has never been formally defined through the RFC standardization process. It is therefore common to encounter differences in the representation formats used by each email client.

Representation formats are interpreted by the type of information that they contain, as opposed to any characteristic of the format specification itself. An email may be stored in any format that allows the storage of text-based information, as text (ASCII, Unicode) and binary encoded data (Microsoft Personal Folders). Variation of each encoding type is identified by the organisational structure and mark-up contained. For example, mail may be stored individually using maildir[13] or EML, or as a combination of one or more emails in a single file using mboxrd, mboxcl, or other variations.

Due to the use of common formats to encode email messages, it is feasible to store all of the significant properties identified by the InSPECT Project. However, the ability to maintain the significant properties across several format conversions may present difficulties, as a result of the capabilities of the software tool(s) in use to decode and export the significant properties. For example, some software tools do not save the headers with the message body, do not export the attachment, or remove other important information.

3.2. Common representation formats

Several Representation formats are widely used for the storage of email messages:

- **Microsoft Outlook Message (.msg):** The Outlook message (.msg) format refers to an email that has been exported from Microsoft Outlook as a distinct file. An Outlook message consists of an email header and message text, which may be accompanied by additional 'attachments'. It conforms to 'COM Structured OLE2 Compound Document' (otherwise known as 'DocFile', a container format developed and used by Microsoft, Inc. to encapsulate information created in Microsoft Office applications.. The format specification is proprietary and the company has not publicly released the format specification. However, aspects of the format have been reverse-engineered for the purpose of creating an import/export module for the OpenOffice suite

- **Microsoft Outlook Personal Folder (.pst):** Outlook Personal Folders is a compound format created and maintained by Microsoft, Inc. An Outlook .pst file is used to store one or more email messages and attachments, calendar events and other items. Email messages are grouped into a hierarchical structure by folder/sub-folder. The format has progressed through several versions that offer different functionality. Messages stored in the Outlook Personal Folders 2007 format are encoded in Unicode. The format allows a maximum file size up to 33TB

- **mbox:** The mbox family refers to four related, but only semi-compatible formats for the storage of one or more email messages and attachments. The four formats - mboxo, mboxrd, mboxcl, and mboxcl2 – originate from different versions of Unix. Each mbox file represents a

[13] http://www.qmail.org/qmail-manual-html/man5/maildir.html

set of email messages that are ordered sequentially and grouped into a 'folder'. Email messages are stored in their source format, e.g. plain text may be stored as ASCII or Unicode, binary data is stored as Base64-encoded text. The format is well supported by a number of email applications and, thanks to its text-based composition can be processed, rendered and converted by a wide range of text processing software.

- **Maildir:** Maildir[14] is an organizational structure for the storage of one or more emails on a file system. Each email is stored as a distinct file in one of three sub-directories: the '*tmp*' sub-directory temporarily stores emails during processing; '*new*' contains newly delivered emails; and *'cur'* contains emails that have been processed by the client's mail-reader software. The storage of each email as a distinct file in the file structure is cited as workaround to file locking issues that affect compound formats, such as mbox that update the mail data file that the user is accessing[15]. However, the filename convention used for the storage of emails may cause incompatibility in implementations of Maildir for Unix-compatible and Microsoft Windows operating systems. The colon character is an illegal character in Microsoft Windows. However, there is no standard on the alternative character that may be used in the environment[16].

- **Email Account XML schema**: The Email Account XML schema[17] is a new format for the storage of information found in a single email account. It was co-developed by the North Carolina State Archives and Smithsonian Institution Archives to preserve historical email messages.

3.3. Software tools

3.3.1. Requirements

The criteria for identification and selection of software tools are intended to be inclusive, considering a range of software available on many different software platforms and published under different types of licence. General criteria for the selection of software tools

1. *Task:* Able to identify some or all properties of an Information Object that are considered to be significant;
2. *Task:* Able to extract significant properties of source format and store them in an open, well documented destination format;
3. *Environment:* Can be compiled or operated on a number of computing operating systems;
4. *Environment:* Can be implemented in a processing workflow;
5. Distribution: Are publicly available as a full product or in demo form for testing;
6. *Legal:* Provide clear guidance on the licence for use of the software in a production environment. Particular preference given to open source licence models;
7. *Documentation:* Are well documented.

3.3.2. Software tools available

The ability to identify, extract and convert the significant properties of an email object require a combination of mainstream software tools that are able to analyse representation formats and bespoke development to combine software into an integrated workflow. Although initial preference was given to software tools that exhibited each one of the above criteria, it was found that the number of tools available to process proprietary email formats are limited in their number and functionality. For example, some software tools do not save the headers with the message body, do not export the attachment, or remove other important information. The project team identified several software tools that were able to process email objects and selected a subset for testing.

- **Aperture:** Aperture is a Java framework for the analysis extraction and querying of full text and metadata from various types of information system (e.g. file systems, web sites, mail boxes) and file formats (e.g. documents, images). The framework uses a Windows dynamic linked library

[14] http://www.qmail.org/man/man5/maildir.html
[15] http://www.qmail.org/man/man5/maildir.html
[16] http://docs.python.org/3.0/library/mailbox.html
[17] http://www.archives.ncdcr.gov/mail-account

(jacob.dll) to decode Outlook PST files, analysing those referenced in the Windows registry by default.

- **ReadPST:** ReadPST is a command-line utility for converting Microsoft Outlook Personal folders (PST) to MBox or KMail format, as used by several Unix, Linux and Windows based email clients. As a conversion tool, ReadPST does not convert the information content to XML. However, the MBox format is well-documented and can be used as the basis for subsequent reformatting.

- **XENA:** A Java-based tool developed by the National Archives of Australia (NAA) to convert a selection of file formats to XML representations, for the purpose of long-term preservation. XENA uses ReadPST to decode Microsoft Outlook Personal folders. It subsequently repackages the content into an email-based namespace developed by the National Archives of Australia.

4. Experiment

4.1. Sample data to be analysed

To demonstrate the identification, extraction and conversion of properties in a production environment the project team obtained data samples from several sources which were used as the basis for analysis. Prior to data selection, it was established that the data should represent real-world examples, i.e. emails created in a production environment, as opposed to emails created in a controlled environment for analysis purposes. Specifically, emails were selected that were:

1) Created and stored in different software applications (e.g. Microsoft Outlook, Outlook Express, Mozilla Thunderbird) and file formats (msg, pst, txt, html and an mbox variant).
2) Contained information stored at different stages in its lifecycle, e.g. emails sent by the creator stored in 'sent items' that do not contain route information in the header, emails received by the recipient.

An Initial arrangement was made with The British Library to obtain email data for testing. However, delays in the rights clearance process resulted in the need to obtain data from alternative sources. To rectify the issue, the project team approached AHDS History, which agreed to provide a representative sample of emails that they had created during the previous year. This resulted in the provision of 50 emails that were provided as a Microsoft Outlook PST and a set of individually stored Outlook MSG files.

Classification	**Number**
No. of received emails without attachments:	19
No. of received emails with attachments:	11
No. of sent emails without attachments	17
No. of sent emails with attachments:	3
Total no. of emails	50

Table 7: Breakdown of emails provided by AHDS History

An examination of the 14 emails with attachments confirmed that they were word processing documents created in Microsoft Word 2003.

The project team also sent a request l to the JISC Repositories mailing list on 11 June 2008 (see appendix) requesting subscribers to provide sample emails for analysis (see Appendix 3: Email). In response, a JISCMail administrator suggested the use of the jisc-repositories mailing list archives for analysis. The project team subsequently obtained a sample of emails distributed to the Jisc Repositories mailing list during the 2006 – 2007 period, which were stored in the mbox and Outlook PST format. This consisted of 1,585 emails[18]., nine of which contained digital signatures and 67 emails with attachments stored in various file formats. Table 7 provides a breakdown of the email attachments by file format.

Format	*Number*
ASCII text	31
iCalendar data	1
JPEG image	1
Microsoft Excel spreadsheet	1
Microsoft Word document	7
Portable Document Format document	9
TIFF image	2
vCard	26
Total	*78*

Table 8: Breakdown of email attachments by file format

[18] Email messages are stored as Unicode or ASCII in a Microsoft Outlook data file. It is feasible to write a script that counts the number of occurrences of a specific value (e.g. the delivery-date). However, the script would need to be written to ignore forwarded emails attached to an email message.

4.2. Testing Environment

All software testing was performed on a Dell GX260 fitted with a 32-bit Pentium 4 2GHz CPU, 1GB RAM and installed with Microsoft Windows XP Professional (version 2002) Service Pack 3.
Experiments that required a live Microsoft Outlook instance were performed within a VirtualBox virtual machine installed with Windows XP Professional (version 2002) Service Pack 3 and configured to use 368MB RAM. A virtual machine was used to avoid corruption of the host machine's Outlook setup and to avoid 'contamination' caused by interaction between the current experiment and previous testing.

4.3. Experiment testing

The following experiments consisted of four distinct stages, outlined in Figure 3.

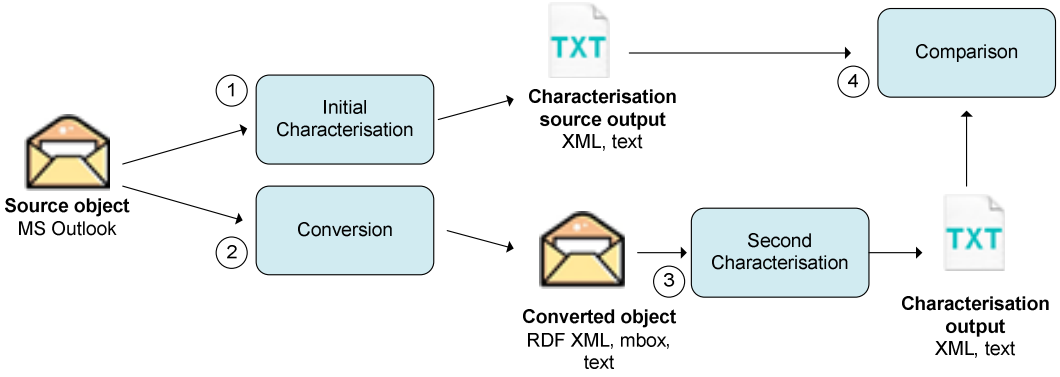

Figure 3: Illustration of the automated experiment procedure

1. **Initial characterisation:** The first characterisation stage examines the source object and extracts appropriate representation information. The number of emails contained in the source object was counted through a combination of manual and automated verification. The information is utilized as a base line against which later characterisation activities are compared.

2. **Conversion:** The source objects are converted into several different file formats, including RDF-based XML, mbox and plain text.

3. **Second characterisation:** The second characterisation stage examines the converted objects and extracts appropriate representation information.

4. **Comparison:** The result of the format conversion is evaluated through a combination of automated and manual comparison. A comparison is made between information extracted from the source and converted object and a visual assessment is made of a sample of emails to identify noticeable differences.

4.4. Experiment

4.4.1. PST Experiment 1: Convert Outlook PST to XML-based RDF using Aperture

For the first experiment we tested the feasibility of converting e-mail messages contained within an Outlook PST file to an XML-based format conforming to the Resource Description Framework (RDF). The RDF data model is widely used as a method for modelling complex resources and associated values, which are expressed as set of subject-predicate-object triples. By storing e-mail messages within an XML-based RDF structure, it is possible to record the complex inter-relationship between an e-mail and any attachments or other e-mails.

The software applications available to convert Outlook PST to another format are limited to a handful of software tools, the majority of which are unable to export to an XML-based format. An investigation uncovered only one software tool – Aperture - that was able to export an Outlook PST file to an XML-based format without extensive customisation. Aperture is a generic Java framework that can analyse, extract and query text from various types of object and system. The framework operates by making a connection with the MS Outlook instance – the installed version of Microsoft Outlook that is recorded in the registry – and crawls the registered e-mails, appointments and contacts. Therefore, it is necessary for the user to record the location of the PST file that is the target of preservation action within the Windows registry, by mounting it in Microsoft Outlook prior to processing.

For the experiment we created a Windows XP virtual machine, installed with Microsoft Outlook 2003 (configured with a non-working mail account and the default mail folder), Java v6 and Aperture 1.1.0 beta within VirtualBox. The Aperture outlookcrawler batch script was executed through the Windows command line, which detected three Outlook data files (the default Outlook 'inbox', calendar and other items, the ahds-history.pst and the Jisc-Repositories.pst) and exported the e-mails and contact details to RDF-compliant XML written as a file on the disk. The process of crawling an Outlook Personal Folder and converting it to RDF was memory intensive and the application repeatedly gave an out of memory error. The memory allocated to the virtual machine was increased to 1.5GB and the memory allocated to Java was also raised, but this did not resolve the issue.

The PST-to-RDF conversion makes extensive changes to the organisational structure of the e-mails. The RDF output conforms to the NEPOMUK Message Ontology[19], containing triples necessary to define the entities within the email archive. Specifically, it makes a logical distinction between Agent information (the sender and recipients of e-mail messages) classified as RDF informationObjects and the e-mail which is classed as a dataObject. The informationObject related to each contact is stored once without the RDF-XML and assigned a unique identifier (UUID URN), to reduce duplication. The informationObject contains a forenames (nameGiven), surname (nameFamily), full name (a concatenation of nameGiven and nameFamily), as well as contact details (address, telephone number) that can be extracted from a vCard. An informationObject ID may be associated with one or more dataObjects, which represent the email itself. The dataObject contains a subject, sent and received date, a message body, as well as any inter-relationship between mails.
The conversion strategy adopted for the experiment was validated through a combination of manual and automated comparison. Although the use of automated comparison tools was preferred, the project team was unable to locate software capable of performing the required level of analysis on a native Outlook PST data file. The verification functionality offered by many software tools require a third-party plug-in (e.g. ReadPST) to export the PST to a text format for analysis. This introduced the potential risk that errors in the PST-to-text conversion would produce erroneous results.

1. *Count of the number of email messages*
   The number of emails contained in the source and destination formats were counted and compared through a combination of manual and automated verification. The number of emails contained in the destination format was confirmed to be 1,020 – 500 less than the number of emails in the jisc-repositories Outlook PST, by counting the number of instances of the <plainTextMessage Content> tag in the RDF-XML output. The discrepancy may be caused by the memory allocated errors outlined above.

2. *Count of the number of email attachments*
   The number of email attachments contained in the source and destination formats were counted and compared through automated verification methods. The number of attachments contained in the RDF-based XML was counted through an automated search and count of the number of instances of base64 text, which were unable to locate any attachments within the RDF-XML. The results were verified through a visual inspection, which confirmed that Aperture had not processed any of the email attachments.

3. *Comparison of the significant properties contained in the message header of the source and destination formats*
   The significant properties of the message header were compared through visual inspection of the source and destination formats. The email messages analysed for the test bed contained

[19] http://www.semanticdesktop.org/ontologies/2007/03/22/nmo/

various properties identified on page**Error! Bookmark not defined.**. However, only a limited number of the properties were exported to RDF-XML. Table X indicates the elements that were correctly identified and converted.

Property	Object Type	Expression method	
Message identifier	informationObject	UUID URN	A unique identifier assigned by Aperture that associates the sender and recipient (InformationObjects) with an email message (dataObject). These are equivalent, but are not the same as the message-ID.
Forenames	informationObject	nameGiven	
Surname	informationObject	nameFamily	
Full name	informationObject	nameGiven & nameFamily	
Email address	informationObject		
Address	informationObject		Details extracted from vCard
Telephone	informationObject		Details extracted from vCard
Subject	dataObject	<messageSubject>	
Received date	dataObject	<receivedDate>	Conforms to http://www.w3.org/2001/XMLSchema#dateTime
Sent date	dataObject	<sentDate>	Conforms to http://www.w3.org/2001/XMLSchema#dateTime
Subject	dataObject	<messageSubject>	

Table 9: Message header properties recognised and exported by Aperture

4. *Comparison of the significant properties of the message body contained in the source and destination format*
   The message body of the source and destination format was compared through a combination of automated and manual analysis. For the former, an Outlook batch export plug-in was used to write each message contained in the source format to a set of plain text files. This was subsequently compared to text contained in the <plainTextMessageContent> of the RDF-XML[20]. The analysis revealed some additional processing performed by Aperture on the message, converting carriage-returns to '
' (without quotes), as required by the Canonical XML standard[21]. A visual comparison of selected emails confirmed initial concerns that presentational mark-up found in the source format were removed from the destination format.

The normalization of email objects stored in Outlook PST to an RDF-based XML format represents an effective strategy for maintaining access to the information objects in an open format and using it as the basis for further analysis. It is evident that the Aperture developers have placed development emphasis upon the latter purpose rather than the former, as demonstrated by the type of properties that are maintained. However, further development could be performed on the tool by the preservation community to extract additional properties. The approach taken by Aperture to analyse and extract the properties of digital objects may also be useful for tool development within this field.

4.4.2. Experiment 2: Convert Outlook PST to mbox using ReadPST

For the second experiment we tested the feasibility of converting e-mail messages contained within an Outlook PST file to the MBox format. MBox is a common format for the storage of one or more email messages and associated attachments. Each message is stored sequentially within the file. A message encoded in the mbox format begins with a 'From' line, followed by a set of header elements, a message body and attached files encoded in Base64. The final line of the email remains blank.

[20] Aperture converts the message body to plain text and stores it within a <plainTextMessageContent> element.
[21] http://www.w3.org/TR/xml-c14n

Through the use of the MBox format it is possible to store one or more emails, possessing appropriate header information, message body and attachments within a single file which may be easily decoded and rendered in a single relational structure.

The software applications available to convert Outlook PST to another format are, as noted limited to a handful of software tools. ReadPST is a fork of the libPST library that can process Microsoft Outlook 'Personal Folders' and convert them to the MBox or KMail format. The source code is freely available and compiled binaries are available for the Unix, MS Windows and Apple Macintosh operating systems. It is used by several software applications, most notably XENA by The National Archives of Australia to decode Outlook PST files for subsequent processing.

For the experiment, we tested ReadPST 0.5.2-1[22] in a Microsoft Windows XP environment. ReadPST was executed through the Windows command line, specifying the source PST file and the output directory[23]. Each Outlook 'folder' was output to an MBox file that took the folder name. In total, 3 MBox files were created: 1) 'Personal Folders', equivalent to the top-level MS Outlook folder that contains the Jisc-repositories folder; 2) 'jisc-repositories', which contains emails from the JISC repositories mailing list; and 3) 'Deleted Items', equivalent to the MS Outlook folder that is used to store emails that have been allocated for deletion by the user. The first and third files are zero bytes in size and the second file is 24,537 kilobytes in size.

The results of the conversion strategy adopted for the experiment was evaluated through the implementation of four validation methods. These utilise a combination of manual and automated processes.

1. *Count of the number of email messages*
   The number of emails contained in the source and destination formats were counted and compared through a combination of manual and automated verification. The number of e-mails contained in the Outlook PST was confirmed to be 1,585 emails, by manually opening the PST file in Microsoft Outlook and using it to count the number of emails in the email folder[24]. The number of emails contained in the destination format was confirmed to be the same number, by counting the number of instances of the 'delivery-date' term in the MBox record.

2. *Count of the number of email attachments*
   The number of email attachments contained in the source and destination formats were counted and compared through automated verification methods. The number of attachments contained in the mbox file was counted through the creation of a simple script that counted the number of instances of 'content-disposition' in the file and wrote the filename to a list. The comparison revealed a considerable difference between the number of attachments present in the source Outlook PST (78 attachments) and those present in the mbox (472). The discrepancy was resolved through a manual examination, which established that 335 of the base64 encoded 'attachments' contained message body text from emails that had been transmitted as Rich Text or HTML. A number of base64-encoded images were also found that were displayed in the message (through HTML 'Img src' tag), but located on external servers. It appears that ReadPST downloaded the referenced on third-party servers and re-encoded them as Base64 within the mbox file.

3. *Comparison of the significant properties contained in the message header*
   The significant properties of the message header were compared through automated and visual inspection of the source and destination formats. The email messages contained within the destination format contained all of the significant properties contained within the source format.

[22] ReadPST 0.5.2-1 was obtained from http://www-uxsup.csx.cam.ac.uk/pub/windows/cygwin/release/readpst/?C=S;O=A
[23] An explanation of the configurable options available can be found at http://www.five-ten-sg.com/libpst/rn01re01.html
[24] Email messages are stored as Unicode or ASCII in an Microsoft Outlook data file. It is feasible to write a script that counts the number of occurrences of a specific value (e.g. the delivery-date). However, the script would need to be written to ignore forwarded emails attached to an email message.

4. *Comparison of the significant properties contained in the message body*
   The message bodies of the source and destination format were compared through a combination of automated and manual analysis. For the source PST, an Outlook batch export plug-in was used to write each message contained in the source format to a set of email (.eml) messages. For the mbox, each email was written to a separate message file containing the header and body. Base64-encoded Rich Text was decoded and converted to plain text for comparison. Both outputs were cross-matched via the message-id and the message text compared. The message body of the variants was broadly similar, although differences in non-english, blank spaces and some punctuation were noted.

The significant properties of an email object and its constituent components were correctly maintained by ReadPST when migrating the Outlook PST format to mbox. However, the diverse types of message content contained in an email made it problematic to validate the conversion using entirely automated methods.

4.4.3. PST Experiment 3: Convert Outlook PST to text using Outport

For the third experiment we tested the feasibility of converting email messages contained within an Outlook PST file to an ASCII text format. To perform the experiment, we tested Outport, an open source application capable of exporting Microsoft Outlook calendar, contacts, mail messages, journal notes and tasks to the Evolution, Outlook item, text and several other formats. The source code is freely available; however it has a number of dependencies that limit its use to the MS Windows platform. Most notably, it makes reference to the registry and calls the Microsoft Outlook installation to perform the initial export.

For the experiment, we installed Outport 1.1.25 within a Windows XP virtual machine, installed with Microsoft Outlook 2003. Outport identified the default Outlook Personal Folder configured by MS Outlook when it was installed. However, it did not recognise jisc-Repositories.pst or ahds-history.pst which was configured as a personal folder within MS Outlook. To workaround the issue, the jisc-repositories folder was copied to the default mail folder. Outport was executed through the Windows graphical interface, specifying the source mail folder and the output directory. A sub-directory was created with the Outlook folder name and each email message was written in date order. Plain text emails were output as ASCII text and mark-up text (HTML and Rich Text) were exported as HTML 4.0 Transitional (318 objects in total). The earliest chronological email was output as message1.txt and subsequent emails received an incremental number. Email attachments were written in their native format to a subdirectory using the same naming convention (e.g. email attachments for message123.msg were written to /message123/). Email messages written in plain text were exported as ASCII text and

The results of the conversion strategy adopted for the experiment was evaluated through the implementation of four validation methods. These utilise a combination of manual and automated processes.

1. *Count of the number of email messages*
   The number of emails contained in the source and destination formats were counted and compared through a combination of manual and automated verification. The number of e-mails contained in the Outlook PST and those in the output folder were confirmed to be the same number.

2. *Count of the number of email attachments*
   The number of email attachments contained in the source and destination were counted and compared through automated verification methods. The number of output attachments was validated by counting the number of files contained within each sub-directory. The comparison confirmed that all 85 of the attachments had been extracted correctly.

3. *Comparison of the significant properties contained in the message header*
   The significant properties of the message header were compared through automated and visual inspection of the source and destination formats. The email messages contained within the ASCII text output contained seven of the requisite properties: Creator email address and display name; Sender email and display name; Sent-date; recipient email address and; email subject. However,

it did not contain additional client (e.g. mail application), provenance, or relationship information (e.g. threaded messages, email attachments). The omission of additional metadata from the text output appears to have been a conscious decision by the software developers - Outport provides an exact copy of the message header when exporting to the MSG format.

4. *Comparison of the significant properties contained in the message body*
   The message bodies of the source and destination format were compared through a combination of automated and manual analysis. For the source PST, an Outlook batch export plug-in was used to write each message contained in the source format to a set of email (.eml) messages and compared to the Outport-generated text-based messages by cross-matching the sent-date and subject. The message text of the Outport-text output was 99% accurate, although differences in non-english, blank spaces and some punctuation were noted. The message text contained in the Outport-HTML output was also accurate. A visual comparison of 10% of the HTML documents confirmed that the presentational markup was broadly equivalent to the formatting (e.g. font size and colour) of the source objects. However, the font type was incorrect. Additionally, it was noted that the message text did not wrap correctly, which would require further processing.

The significant properties of an email object and its constituent components were correctly maintained by Outport when migrating the Outlook PST format to individual email messages. However, the tool imposes limitations that limit its ability to be integrated into a processing workflow. Specifically, it is limited to export of the default mail folder only and must be controlled through the Windows GUI.

4.5. Tool comparison

The following table provides a cross-analysis of the information that is identified and extracted by each software tool.

Property value	Aperture	ReadPST	XENA	Outport
creator.local - part	N	Y	Y	Y
Creator.domain - part	N	Y	Y	Y
Creator.domain - literal	N	Y	Y	Y
Creator.display - name	N	Y	Y	Y
sender.local - part	Y	Y	Y	Y
sender.domain - part	Y	Y	Y	Y
sender.domain - literal	U[25]	U	U	U
sender . display - name	Y	Y	Y	Y
reply-to.local - part	N	Y	Y	Y
reply-to.domain - part	N	Y	Y	Y
reply-to.domain - literal	N	Y	Y	Y
reply-to.display - name	N	Y	Y	Y
recipients - primary(No.).local - part	Y[26]	Y	Y	Y
recipients - primary(No.).domain – part	Y	Y	Y	Y
recipients - primary(No.).domain – literal	Y	Y	Y	Y
recipients - primary(No.).display – name	Y	Y	Y	Y
recipients - secondary(No.).local - part	Y	Y	Y	Y
recipients - secondary(No.).domain - part	Y	Y	Y	Y
recipients - secondary(No.).domain - literal	U	U	U	U
recipients - secondary(No.).display - name	Y	Y	Y	Y
recipients - other(No.).local - part	Y	Y	Y	Y
recipients - other(No.).domain - part	Y	Y	Y	Y
recipients - other(No.).domain – literal	U	U	U	U
recipients - other(No.).display - name	Y	Y	Y	Y
creation - date	Y	Y	Y	Y
send - date	Y	Y	Y	Y

[25] Literal domain not found in any email analysed.
[26] Aperture identifies multiple recipients as recipient1, recipient2, recipient3, etc. It does not indicate if they were the primary recipients, secondary recipients (CC), or third recipients (BCC).

received - date	Y	Y	Y	Y
message - id	P[27]	Y	Y	Y
id - domain	N[28]	Y	Y	Y
subject	Y	Y	Y	Y
keywords	N	Y	Y	Y
Attachments	N	Y	Y	Y
hyperlink	P[29]	Y	Y	Y
message - body	Y[30]	Y	Y	Y

Table 10: Comparison of conversion tools

3. Conclusion

As an object type that consists predominantly of semi-structured text, a presumption might be made that the audit and conversion of email objects would be a simple process. Email formats are, in essence text-based formats that could encapsulate any type of information. However, the experiments outlined in this report demonstrate that email management continues to cause problems. Due to the constraints imposed by existing software tools, the ability to convert between formats and assess the success of format conversion is limited.

The characterisation stage is necessary to audit that emails have been exported correctly, in terms of the number of emails and the significant properties contained by each. Although email formats are based upon published standards, the encoding formats in which emails are stored on client machines are difficult to audit. Many of the software tools examined required an active installation of Outlook to perform an audit or required some form of initial conversion before they could analyse the data. This made it difficult to audit the source objects using automated tools, requiring manual checks to be performed. This is likely to be impractical for an institution responsible for managing a large number of emails.

The conversion tools tested by the project were extensive in the information that they could export and diverse in the formats that they supported. However, no single application provided extensive functionality. The tools performed format conversion based upon a standard setting, rather than taking into consideration the specific requirements (e.g. mark-up) of each email object. They were also unable to record the relationship between the message body and attachments, for instance.

Recommendations:

- Recommend that work is commissioned to develop audit tools capable of analysing the significant properties of email objects and compare results across different formats.

- Recommend that further experimentation is performed using other email management tools, such as those listed in Appendix B.

- Recommend that the JISC investigate preservation practices for emails objects within institutions, with specific consideration of scenarios in which different types of properties are required.

- Recommend that tools are developed that implement a granular approach to format conversion of email objects, rather than the existing approach of converting all emails to a single output format. A rule-based system may be defined that examines qualitative features of an email (e.g. specific mark-up elements, the combination of different colours) that may indicate intrinsic information within the message body and implement a specific conversion activity that preserves these features.

[27] Message IDs converted to UUIDs
[28] Reliant on unlikely possibility that two duplicate UUIDS will be generated.
[29] Hyperlinks are maintained as part of message body. However, they are not stored in separate element.
[30] Converted to plain text and stored in plainTextMessageContent element. A carriage return is preserved in the plain text. However, no distinction is made between sub-elements of the message body, such as paragraphs, signatures, or other information.

- Recommend that a format archive is created that contains emails stored in a variety of file formats to assist with tool development and testing.

Appendix 1: Representation Formats

The majority of the representation formats examined by the project team are widely used for the storage of email objects. However, they are not currently recognized by the PRONOM Technical Registry (http://www.nationalarchives.gov.uk/pronom/). The following descriptions are provided for addition to the PRONOM registry.

Microsoft Outlook Message (MSG)

Name	Microsoft Outlook Message
Version	
Other names	
Family	Text (email)
Classification	Binary
Orientation	
Byte Order	
Related Formats(1)	Has priority over OLE2 Compound Document Format
Related Formats(2)	Is subtype of OLE2 Compound Document Format
Related Formats(3)	
Developed By	Microsoft Corporation
Supported By	
Format Extension	.msg
Description	Outlook .msg is used to refer to email messages stored in Microsoft Outlook that have been saved as distinct files. An .msg file consists of an email header and message text, which may be accompanied by additional 'attachments'. The Outlook MSG format conforms to 'COM Structured OLE2 Compound Document' (otherwise known as 'DocFile', a container format developed and used by Microsoft, Inc. to encapsulate information created in Microsoft Office applications. The format specification is proprietary and the company has not publicly released the format specification. However, aspects of the format have been reverse-engineered for the purpose of creating an import/export module for the OpenOffice suite
Identification	An OLE2 Compound Document may be identified by the file header which begin with D0 CF 11 E0 A1 B1 1A E1. The document structure is comprised of a series of pointers that indicate the type and location of information containers. For example, the text string "r e c i p" (hex: 72 00 65 00 63 00 69 00 70) indicates the recipient of a message and "a.t.t.a.c.h (hex: 61 00 74 00 74 00 61 00 63 00 68) identifies an attachment[31].
DROID support	DROID v3.0 installed with signature file version 13 provides a Positive (specific format) identification of .msg files as being encoded in the OLE2 Compound Document Format (PUID: fmt/111). However, it notes there may be a mismatch between the format encoding and the file extension.
Notes	

Microsoft Personal Folders (PST) 2007

Name	Microsoft Outlook Personal Folders
Version	2007
Other names	Personal Storage Table
Identifiers	
Family	Text (email)
Classification	Binary
Orientation	
Byte Order	

[31] The type of OLE2 Compound Document may be clarified by analysing the containers stored in a file.

Related Formats(1)	Microsoft Outlook Personal Folders 2003
Related Formats(2)	Microsoft Outlook Personal Folders 1997
Related Formats(3)	
Developed By	Microsoft Corporation
Supported By	Microsoft Exchange Client, Microsoft Windows Messaging, Microsoft Outlook and Microsoft Office Outlook
Format Extension	.pst
Description	Outlook Personal Folders is a compound format created and maintained by Microsoft, Inc. An Outlook .pst file is used to store one or more email messages and attachments, calendar events and other items. Email messages are grouped into a hierarchical structure by folder/sub-folder. Messages stored in the Outlook Personal Folders 2007 format are encoded in Unicode. The format allows a maximum file size up to 33TB
Identification	
DROID support	The Outlook Personal Folder format is not current recognized by the DROID software.
Notes	

Microsoft Personal Folders (PST) 2003

Name	Microsoft Outlook Personal Folders
Version	2003
Other names	Personal Storage Table
Identifiers	
Family	Text (email)
Classification	Binary
Orientation	
Byte Order	
Related Formats(1)	Microsoft Outlook Personal Folders 2007
Related Formats(2)	Microsoft Outlook Personal Folders 1997
Related Formats(3)	
Developed By	Microsoft Corporation
Supported By	Microsoft Exchange Client, Microsoft Windows Messaging, Microsoft Outlook and Microsoft Office Outlook
Format Extension	.pst
Description	Outlook Personal Folders is a compound format created and maintained by Microsoft, Inc. An Outlook .pst file is used to store one or more email messages and attachments, calendar events and other items. Email messages are grouped into a hierarchical structure by folder/sub-folder. Messages stored in the Outlook Personal Folders 2007 format are encoded in Unicode. The format allows a maximum file size up to 33TB
Identification	
PRONOM support	The Outlook Personal Folder format is not current recognized by DROID.
Notes	

Microsoft Personal Folders (PST) 1997

Name	Microsoft Outlook Personal Folders
Version	1997
Other names	Personal Storage Table
Identifiers	
Family	Text (email)
Classification	Binary
Orientation	
Byte Order	

Related Formats(1)	Microsoft Outlook Personal Folders 2007
Related Formats(2)	Microsoft Outlook Personal Folders 2003
Related Formats(3)	
Developed By	Microsoft Corporation
Supported By	Microsoft Exchange Client, Microsoft Windows Messaging, Microsoft Outlook and Microsoft Office Outlook
Format Extension	.pst
Description	Outlook Personal Folders is a compound format created and maintained by Microsoft, Inc. An Outlook .pst file is used to store one or more email messages and attachments, calendar events and other items. Email messages are grouped into a hierarchical structure by folder/sub-folder. Messages stored in the Outlook Personal Folders 2007 format are encoded in Unicode. The format allows a maximum file size up to 33TB An Outlook Personal Folder 1997 is encoded in the ANSI format. The maximum file size is 2GB. Data corruption may occur if an Outlook Personal Folder file is allowed to expand over the 2GB limit.
Identification	
PRONOM support	
Notes	

Appendix 2: Software Tools

The project examined a number of software tools capable of analyzing representation formats used for the storage of emails. To document the process it adopted the format adopted by the CAIRO project for its tool survey[32].

Aperture

Tool Name	Aperture
Source URL	http://aperture.sourceforge.net/
Formats supported	Plain text, HTML, XHTML, XML, PDF, RTF, Word, Excel, Powerpoint, Visio, Publisher, Microsoft Works, OpenOffice 1.x: Writer, Calc, Impress, Draw, StarOffice 6.x - 7.x+: Writer, Calc, Impress, Draw OpenDocument (OpenOffice 2.x, StarOffice 8.x), Corel WordPerfect, Quattro, Presentations Emails (.eml files), ical files
Technology Base	
Operating system	Cross-platform
Dependencies	Java 1.4.
Licence	Academic Free License v3.0 (http://aperture.sourceforge.net/license_AFL3.0.html)
Category	Description, conversion
Description	Aperture is a Java framework for extracting and querying full-text content and metadata from various information systems (e.g. file systems, web sites, mail boxes) and the file formats (e.g. documents, images) occurring in these systems.
Output methods	RDF
Notes	Aperture is a Java framework for the analysis extraction and querying of full text and metadata from various types of information system (e.g. file systems, web sites, mail boxes) and file formats (e.g. documents, images). The framework uses a Windows dynamic linked library (jacob.dll) to decode Outlook PST files, analysing those referenced in the Windows registry by default. Information content contained within the PST file, including the email messages and contacts list converted to RDF

[32] Further details of the format can be found on p11 of the Cairo Tools Survey, located at http://cairo.paradigm.ac.uk/projectdocs/index.html

	statements of different granularity. Each contact (sender or recipient) is stored as an Information Object, assigned a UUID URN and pertinent information stored – forenames (nameGiven), surname (nameFamily), full name (a concatentation of nameGiven and nameFamily, as well as contact details (address, telephone no.) that can be extracted from a vCard. Emails are identified as DataObjects and some information stored on each email – sender and recipient, subject, sent and received date, as well as message body, as well as the inter-relationship between mails. A notable omission from the analysis process are email attachments and provenance information, which are not transferred in the current iteration of the analysis tool.

ReadPST

Tool Name	ReadPST
Source URL	http://alioth.debian.org/projects/libpst/
Formats supported	PST
Technology Base	Command-line
Operating system	Win32, Linux compatibles
Dependencies	cygwin1.dll (on Windows)
License	General Public License (GPL)
Category	conversion
Description	readpst is a utility for converting Microsoft Outlook mail files (personal folders) to standard UNIX mbox format. This is a fork of the libpst project at sourceforge.
Output methods	MBOX, KMail
Notes	ReadPST is a command-line utility for converting Microsoft Outlook Personal folders (PST) to the MBox or KMail format, as used by several Unix, Linux and Windows based email clients. When performing PST-to-mbox conversion, each email 'folder' configured in Microsoft Outlook is exported as an mbox file, containing all emails and attachments. The format for the header and message text remains the same (plain text and possibly HTML), accompanied by message attachments encoded as base64. When performing PST-to-KMail conversion, each email and attachment is stored as separate files on the file system. Each email is assigned an incrementally increasing 9 digit filename (000000000). Email attachments are renamed to the 9-digit file identifier + the original filename (e.g. 000000020-proposal.doc) and stored in the same directory, to provide simple file association.

XENA (XML Electronic Normalising of Archives)

Tool Name	XENA (XML Electronic Normalising of Archives)
Source URL	http://xena.sourceforge.net/
Formats supported	PST, MBox, Trim[33]
Technology Base	Java
Operating system	Cross-platform
Dependencies	
License	GNU General Public License
Category	Conversion, description
Description	XENA is a Java-based tool developed by the National Archives of Australia (NAA) to convert a selection of file formats to XML representations, for the purpose of long-term preservation.
Output methods	XML
Notes	XENA uses ReadPST to decode Microsoft Outlook Personal folders and, by extension suffers from any faults found in that utility. For example, the input_source_uri indicates the email source as file:/[number]. It

[33] See http://xena.sourceforge.net/help.php?page=normformats.html for a full list of supported formats.

	subsequently repackages the content into an email-based namespace developed by the National Archives of Australia. Practical experimentation revealed several potential issues with the analysis process: a schema for the XML document could be not located at the URL provided; the tool produced error messages when analysing a specific PST file and was unable to continue (the same version of ReadPST correctly processed the PST file when used as a standalone application) and, in the sample tested, the tool misidentified the first line of message body in the header.

Appendix 3: Email

The following email was posted to the JISC Repositories mailing list on 11 June 2008 requesting the submission of sample emails for analysis.

Dear colleagues,

The JISC-funded INSPECT Project is developing a generalised methodology to determine the significant properties of digital object types. Significant properties are those aspects of the digital object which must be preserved over time in order for it to remain accessible and meaningful. To assist and demonstrate the methodology we are analysing a range of digital objects - e-mails, structured text, raster images and digital audio.

The InSPECT Project invites interested parties to submit sample data for analysis by the project. We invite contribution of emails stored in one or more of the following formats:

- 1-10 emails stored in Outlook MSG, some with attachment and others without.

- A Outlook PST file containing 1-10 emails with one or more with small attachments.

- 1-10 Thunderbird EML files, some with attachment and others without.

- 1-10 emails in a variation of the MBox format, some with attachment and others without.

- 1-10 HTML and text files

Contributors should provide a brief summary of each email (attachment status and other relevant information) and indicate the software application that was used to create and store the emails.

It is planned that some of the contributions will be made available or referenced on the project web site. Please ensure that emails do not contain sensitive information and that you have appropriate permission to provide them (spam messages and scripts will not be accepted).

Please send sample data as a compressed attachment to gareth.knight@kcl.ac.uk.

Appendix 4: Email Property measurement

The table below indicates measurements required to audit the significant properties of the message header and record appropriate values.

property value	component	property definition	function class	function description	constraint type [1]	constraint reason [1]	constraint unit [1]	constraint type [2]	constraint reason [2]	constraint unit [2]	datatype	comments
local - part	creator	The username or other identifier in use by the creator, prior to the @ symbol	context	creator	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - part	creator	A host name or domain name that is used by a DNS to indicate the origin of the message	context	creator	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - literal	creator	An indicator of the source domain of the message specified by its IP (numeric) address.	context	creator	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				[ ] . 0 - 9	The use of domain literals is discouraged in RFC 822.
display - name	creator	A plain text indication of the agent's name	context	creator	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				Alphanumeric	
local - part	sender	The username or other identifier in use by the creator, prior to the @ symbol	context	sender	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - part	sender	A host name or domain name that is used by a DNS to indicate the origin of the message	context	sender	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - literal	sender	An indicator of the source	context	sender	equality	Indicates the presence /	Boolean (present /				[ ] . 0 - 9	The use of domain literals is

property value	component	property definition	function class	function description	constraint type [1]	constraint reason [1]	constraint unit [1]	constraint type [2]	constraint reason [2]	constraint unit [2]	datatype	comments
		domain of the message specified by its IP (numeric) address.				absence of the value in the Record	absent)					discouraged in RFC 822.
display - name	sender	A plain text indication of the agent's name	context	sender	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				Alphanumeric	
local - part	reply - to	The username or other identifier in use by the creator, prior to the @ symbol	context	reply - to	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - part	reply - to	A host name or domain name that is used by a DNS to indicate the origin of the message	context	reply - to	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - literal	reply - to	An indicator of the source domain of the message specified by its IP (numeric) address.	context	reply - to	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				[ ] . 0 - 9	The use of domain literals is discouraged in RFC 822.
display - name	reply - to	A plain text indication of the agent's name	context	reply - to	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				Alphanumeric	
local - part	recipients - primary(No.)	The username or other identifier in use by the creator, prior to the @ symbol	context	primary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - part	recipients - primary(No.)	A host name or domain name that is used by a	context	primary Recipient	equality	Indicates the presence / absence of	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64	

property value	component	property definition	function class	function description	constraint type [1]	constraint reason [1]	constraint unit [1]	constraint type [2]	constraint reason [2]	constraint unit [2]	datatype	comments
		DNS to indicate the origin of the message				the value in the Record					characters (RFC 2821), case sensitive	
domain - literal	recipients - primary(No.)	An indicator of the source domain of the message specified by its IP (numeric) address.	context	primary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				[ ] . 0 - 9	The use of domain literals is discouraged in RFC 822.
display - name	recipients - primary(No.)	A plain text indication of the agent's name	context	primary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				Alphanumeric	
local - part	recipients - secondary(No .)	The username or other identifier in use by the creator, prior to the @ symbol	context	secondary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - part	recipients - secondary(No .)	A host name or domain name that is used by a DNS to indicate the origin of the message	context	secondary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - literal	recipients - secondary(No .)	An indicator of the source domain of the message specified by its IP (numeric) address.	context	secondary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				[ ] . 0 - 9	The use of domain literals is discouraged in RFC 822.
display - name	recipients - secondary(No .)	A plain text indication of the agent's name	context	secondary Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				Alphanumeric	
local - part	recipients - other(No.)	The username or other identifier in use by the creator,	context	other Recipient	equality	Indicates the presence / absence of the value in	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC	

property value	component	property definition	function class	function description	constraint type [1]	constraint reason [1]	constraint unit [1]	constraint type [2]	constraint reason [2]	constraint unit [2]	datatype	comments
		prior to the @ symbol				the Record					2821), case sensitive	
domain - part	recipients - other(No.)	A host name or domain name that is used by a DNS to indicate the origin of the message	context	other Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				US - ASCII (RFC 2822) only, maximum. 64 characters (RFC 2821), case sensitive	
domain - literal	recipients - other(No.)	An indicator of the source domain of the message specified by its IP (numeric) address.	context	other Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				[ ] . 0 - 9	The use of domain literals is discouraged in RFC 822.
display - name	recipients - other(No.)	A plain text indication of the agent's name	context	other Recipient	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				Alphanumeric	
creation - date		The date and time that an e - mail was completed by a Creator	context	date	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				ISO 8601 (datetime)	
send - date		The date and time that an e - mail was completed by a Creator	context	date	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				ISO 8601 (datetime)	
received - date		The date and time that an e - mail was received by a recipient	context	date	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)				ISO 8601 (datetime)	
message - id		A unique, machine - processable identifier	structure	identifier	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)					
id - domain		An indicator of the domain in which the	structure	identifier	equality							

property value	component	property definition	function class	function description	constraint type [1]	constraint reason [1]	constraint unit [1]	constraint type [2]	constraint reason [2]	constraint unit [2]	datatype	comments
		message - id is unique.										
message - id		A unique, machine - processable identifier	structure	reply - to - id	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)					
id - domain		An indicator of the domain in which the message - id is unique.	structure	reply - to - id	equality							
message - id		A unique, machine - processable identifier	structure	references	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)					
id - domain		An indicator of the domain in which the message - id is unique.	structure	references	equality							
subject		A short string that identifies the topic of the message.	content	subject	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)	equality	Indicates the no. of characters	characterLength	ASCII. Maximum of 255 characters	
keywords		A list of important words and phrases that might be useful for the recipient.	context	keywords	equality	Indicates the presence / absence of the value in the Record	Boolean (present / absent)	equality	Indicates the no. of keywords	characterLength		
associatedComponents		An indicator that the message contained attachments or other associated components, in addition to the message body.	structure	relation	equality	Indicates the number of components that are associated with the Record	Integer					
hyperlink		An indicator that the message contains	structure	hyperlink	equality	Indicates the presence / absence of	Boolean (present / absent)					

property value	component	property definition	function class	function description	constraint type [1]	constraint reason [1]	constraint unit [1]	constraint type [2]	constraint reason [2]	constraint unit [2]	datatype	comments
		hyperlinks that must be maintained.				hyperlinks in the Record that must be maintained						
message - body		The message body of the Record that represents the primary content	content	message - body	equality	Indicates the number of characters contained in the message body of the Record	Integer					

References

Montague, L. (2009). The Significant Properties of Structured Text. Retrieved on March 29, 2009 from: http://www.significantproperties.org.uk/outputs.html

Anon (n.d). Headers, Tables, and Macros. Retrieved on March 29, 2009 from: http://www.qmail.org/qmail-manual-html/man5/maildir.html

Anon (n.d). maildir - directory for incoming mail messages. Retrieved on March 29, 2009 from: http://www.qmail.org/man/man5/maildir.html

Ashley, K, Davis, R & Pinsent, E. (2008) Significant Properties of e-Learning Objects (SPeLOs). Version 1.0. Retrieved on March 30, 2009 from http://www.jisc.ac.uk/whatwedo/programmes/preservation/2008sigprops.aspx

Boyer, J. (2001). Canonical XML. Version 1.0. Retrieved on March 29, 2009 from: http://www.w3.org/TR/2001/REC-xml-c14n-20010315

Digital Preservation Testbed (2003). From digital volatility to digital permanence. Preserving email. Retrieved on March 30, 2009 from http://www.digitaleduurzaamheid.nl/index.cfm?paginakeuze=185

Klensin, J. 2008. Simple Mail Transfer Protocol. Retrieved on March 29, 2009 from: http://tools.ietf.org/html/rfc5321

Knight, G. (2008). Framework for the definition of significant propertiest. Retrieved on March 29, 2009 from: http://www.significantproperties.org.uk/outputs.html

Mylka, A. et al (2007). NEPOMUK Message Ontology. Retrieved on March 29, 2009 from: http://www.semanticdesktop.org/ontologies/2007/03/22/nmo/

National Archives of Australia (n.d.). Xena – Digital preservation software. Retrieved on March 29, 2009 from: http://xena.sourceforge.net/

Network Working Group. (ed.) (2001). RFC2045 - Multipurpose Internet Mail Extensions (MIME) Part One. Retrieved on March 30, 2009 from: http://www.faqs.org/rfcs/rfc2045.html

Network Working Group. (ed.) (2001). RFC2046 - Multipurpose Internet Mail Extensions (MIME) Part Two. Retrieved on March 30, 2009 from: http://www.faqs.org/rfcs/rfc2046.html

Network Working Group. (ed) (2005). RFC4289. Multipurpose Internet Mail Extensions (MIME) Part Four: Registration Procedures. Retrieved on March 30, 2009 from: http://www.ietf.org/rfc/rfc4289.txt

Network Working Group. (ed.) (1996). RFC2046 MIME (Multipurpose Internet Mail Extensions) Part Three: Message Header Extensions for Non-ASCII Text. Retrieved on March 30, 2009 from: http://tools.ietf.org/html/rfc2047

Network Working Group. (ed.) (1996). RFC 2049 - Multipurpose Internet Mail Extensions (MIME) Part Five: Conformance Criteria and Examples. Retrieved on March 30, 2009 from: http://www.ietf.org/rfc/rfc2049.txt

Network Working Group. (ed.) (2001). RFC2822 - Internet Message Format. Retrieved on March 30, 2009 from: http://www.faqs.org/rfcs/rfc2822.html

Network Working Group. (ed) (2005). RFC 4288 - Media Type Specifications and Registration Procedures. Retrieved on March 30, 2009 from: http://www.ietf.org/rfc/rfc4288.txt

Network Working Group. (ed.) (2008). RFC 5322 Internet Message Format. Draft standard. Retrieved on March 29, 2009 from: http://tools.ietf.org/html/rfc5322

North Carolina State Archives (2008). XML Schema for a Single E-Mail Account. Retrieved on March 29, 2009 from: http://www.archives.ncdcr.gov/mail-account

Python Software Foundation (n.d). mailbox — Manipulate mailboxes in various formats. Retrieved on March 29, 2009 from: http://docs.python.org/3.0/library/mailbox.html