How Can We Know the Dancer from The Dance?

This is an Author's original manuscript of an article to be published in *The New Review of Information Networking*, Volume 19, Issue 1, 2014, to be available online at:
http://www.tandfonline.com/loi/rinn20

Sheila M. Morrissey
Senior Research Developer
Portico
sheila.morrissey@ithaka.org

How Can We Know the Dancer from The Dance?

Intention and the Preservation of Digital Objects

O chestnut-tree, great-rooted blossomer,
Are you the leaf, the blossom or the bole?
O body swayed to music, O brightening glance,
How can we know the dancer from the dance?

"Among Schoolchildren" William Butler Yeats

It's not a fallacy, it's a design feature

There is a certain delight in seeing that questions, which over the past several decades vexed practitioners of literary criticism, now clamor for resolution in computer science – or at least with that portion of it concerned with software engineering, and with the long-term preservation of digital objects.

Since the middle of the twentieth century, many of the debates in literary criticism centered on the relative authority, or pertinence to critical activity, to be attributed to one or another leg of the three-legged stool of author, text, reader. So, for example, "New Critics" such as William K. Wimsatt and Monroe Beardsley inveighed against what they termed "the intentional fallacy." [Wimsatt 1946] They insisted that the author's intention – whether that intention is gleaned from some inferred psychological state, or some written statement external to the text itself—has no ultimate control over the meaning of the text. Wimsatt also declaimed against what he termed the "affective fallacy" [Wimsatt 1949] – that is, the affect — the emotional response — created in the reader by reading the text is equally out of bounds as a matter of concern for literary criticism.

You could construct a taxonomy of schools of literary criticism in the ensuing decades by the relative weight practitioners brought to bear on each vertex of the triangle of authorial intent, textual autonomy, and reader response.[1]

How does this relate to the preservation of digital objects? If for the vertices we take content creator, digital object, and content user, we can perhaps see how and why anyone charged with curating and preserving digital objects for the long term has to weigh and balance the intention of the content creator, the perceptions and intentions of the content user, and the "authority" of the digital object itself.

An example might help to illustrate.

[1] See [Garber] for an overview.

David Clipsham, who among many other activities develops file format signatures for the UK National Archive's widely used Pronom file format registry[2], has tweeted[3] a link to a web page[4] that has hugely variant renderings, depending upon what browser you use to view it.

The page seems perfectly ordinary when rendered by the Internet Explorer (IE) Version 9 browser on a Dell personal computer running Windows 7 Professional Operating System:

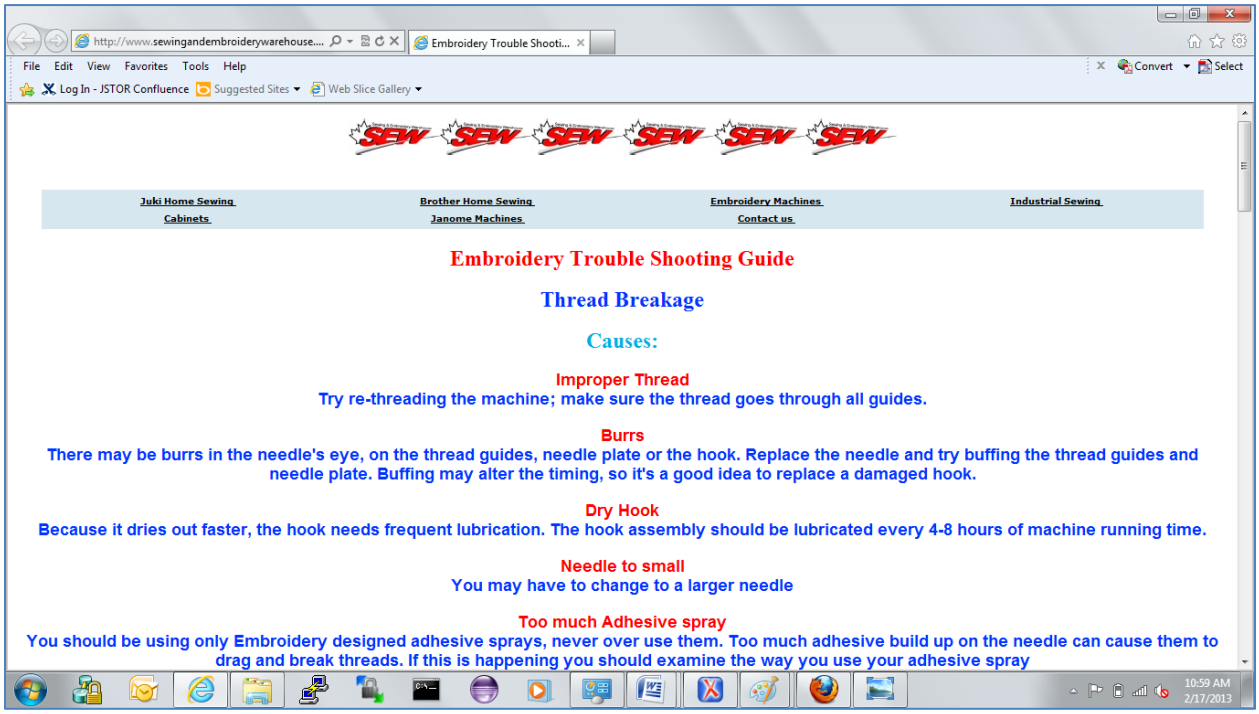

Figure 1

But in the Firefox Version 17 browser on the same computer, something appears to go seriously awry with the font sizes. The text starts fine. But then its gets bigger and bigger and bigger:

[2] http://www.nationalarchives.gov.uk/PRONOM/Default.aspx
[3] https://twitter.com/Britpunk80/status/302536782300463104
[4] http://www.sewingandembroiderywarehouse.com/embtrb.htm

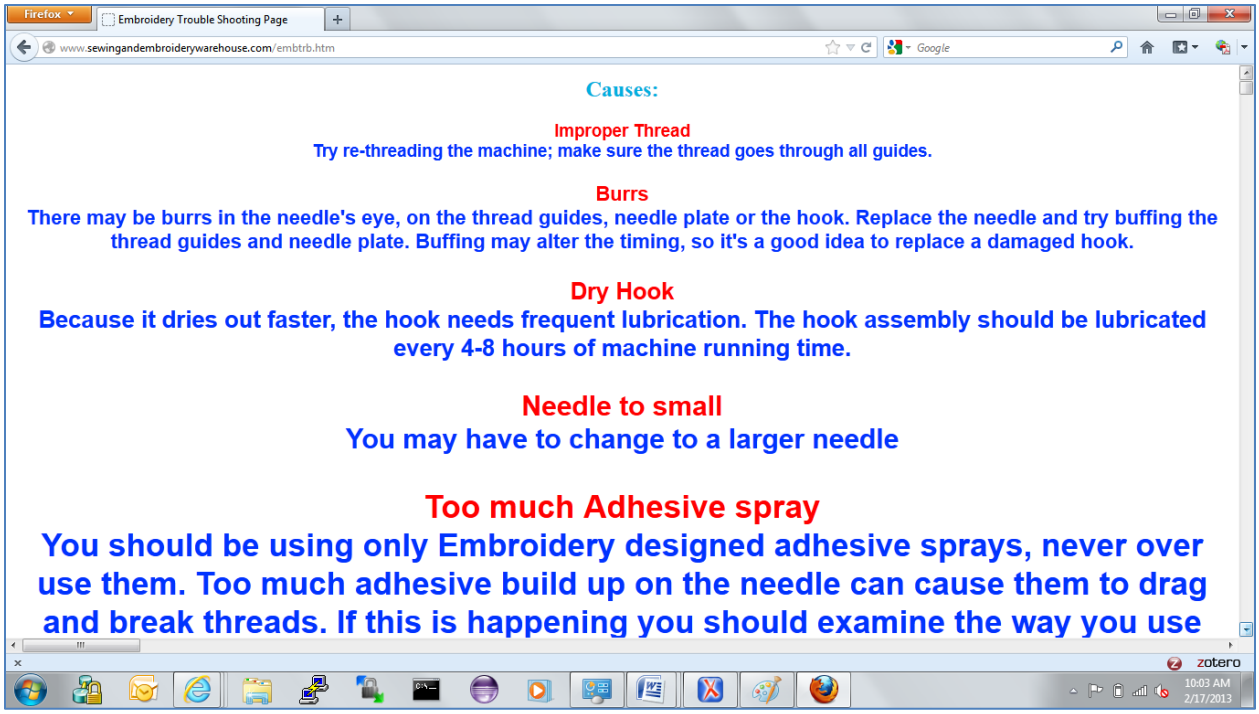

Figure 2

until finally it's off the page:

Figure 3

Once we stop scrolling, we can see that this fairly simple HTML page and its various renditions raise some very interesting questions we have to grapple with when we consider the web page as an object of digital preservation activities.

Leaf, blossom, or bole?[5]

We in the digital preservation community spend a lot of time, ingenuity, and computer cycles on identifying and characterizing the digital objects we are preserving. We try to glean some basic facts about each object in our collections. What format is this file? Is it a valid instance of that format? If not, what "rules" is it breaking?

It looks like the HTML file for this web page is breaking some of the rules of HTML. Feedback[6] from the WC3 online validator tool service highlights the failure to put an end tag on the `<font>` elements used to make the text change color from red to blue and back again. The validator finds other missing end tags, and it finds many other "illegalities" as well.

So we know the file broke the rules. But what about the browsers? Is IE9 breaking the rules? Is Firefox? Is *either* of these two wildly different views of this page a "valid" view? What *is* (or would be) a valid view?

Why does this matter?

For one thing, it challenges some of the heuristics we employ when we try to assess the "riskiness" of a format, or to recommend formats likely to be robust over the long term. So, for example, two of the sustainability factors suggested by the Library of Congress to assess the riskiness of a file format seem to weigh in favor of HTML. [Arms] These are disclosure ("the degree to which complete specifications and tools for validating technical integrity exist and are accessible to those creating and sustaining digital content") and adoption ("the degree to which the format is already used by the primary creators, disseminators, or users of information resources").

For some widely adopted formats with a high degree of disclosure, such as HTML or PDF, it has been possible to create tools like the W3C HTML validator, or JHOVE[7] or JHOVE2[8], to test conformance of an instance of a format to the format's specification. But the results of large scale web and other digital archiving present us with large bodies of content in these same formats that, like the example page above, report many errors in validity, yet still can be rendered -- at least in some fashion or another, by some one or another of a large number of applications from different sources and different vendors.

---

[5] Or, variously, "source, process, performance?"[Heslop]

[6] http://validator.w3.org/check?uri=http%3A%2F%2Fwww.sewingandembroiderywarehouse.com%2Fembtrb.htm&charset=%28detect+automatically%29&doctype=Inline&group=0

[7] http://jhove.sourceforge.net/

[8] http://jhove2.org

What do we miss when we consider characterization of a file as the static analysis of the features detailed in the file's format specification, and the degree to which those features conform to the constraints in that specification? The relationship among a format's specification, any instance of that format, and the software that creates or renders it is a tangled one. This is one reason why simply identifying an instance of a format can be so challenging. It is also a reason why the multiplicity of format creators and renderers is an ambivalent criterion for assessing the long-term viability of a digital object, and is not simply, as noted in the Library of Congress weighting, "valuable in its own right and as evidence that specifications are adequate."

We don't have any tool that tells us whether or not a particular version of a particular brand of browser conforms to the WC3 rules for an "HTML user agent." More importantly, we don't have any "registry" that tells us what an HMTL page will "look like" when it breaks one or more of the HTML rules and is being rendered by a particular version of a particular brand of browser. The same can be said for PDF renderers, [Morr 2012] and for the behavior of various word processing programs when they process inputs both in their own "native" formats, and in the formats of other word processing software. [Cochrane]

"Classic" format characterization can often also obscure the almost recombinant quality of the information we tend of think of as the static content of a format instance. When traversed by rendering software, that content invokes other software and other instances of other formats (and their rendering software, and so on) — all of which must be available as resources to make the original object viewable. [Jackson].

How do we encapsulate this dependency graph effectively? How do we make sure we have all the information that will be needed in the future to make a digital artifact viable, without at the same time, for any one digital object, creating an intractably huge graph of representation information? [McDonough] What information, to give a pragmatic example, do we need to construct an emulation stack for *this* instance of *that* format? What information do we need in order to know what *various* emulation stacks we ought to have on offer for *this* instance of *that* format?

Do what I meant, not what I said

Even supposing the availability of a tool to match file feature characterization information with the capabilities of rendering software with respect to those features, we still have the question to answer: What is it, exactly, that we are trying to preserve? Or, perhaps more precisely, what use or uses are we trying to preserve this digital object *for?*

It is certainly one reasonable conclusion, when looking at the problematic sample page above, to say that Internet Explorer has got it right. That is, we conclude that the creator of the page *seems* to have wanted to say "Stop using this color with this size of this font, and switch to that color with that size of that font" – but seems to have chosen a technically invalid way of indicating that. On that reading, the heuristics employed by the Internet Explorer code to "forgive" the invalid HTML encoding is an instance of an "intentional design feature" rather than an intentional fallacy. The software engineering term for

this design feature, or heuristic, is known as the robustness principle: "be conservative in what you do, be liberal in what you accept from others." [Postel]

How do we rationalize the results of this design feature? As "humans in the loop," we consider the text of the page, abstracted from typographical special effects. We follow links with such labels as "contact us" and find information about a sewing and embroidery warehouse. We conclude that the principal, or at least *a* principal, purpose of the page is the straightforward transmission of facts about embroidery tools and techniques. And we conclude that what we want to preserve is the "signal" transmitted by the unadorned text contents of the HTML elements.

All this is arguing from effects, of course – perceptions by a human viewer. Such information is not effectively available to, or actionable by, the browser code, which operates on syntactic cues provided by the HTML tags, such as detection of missing end tags for elements. This is one reason why the large-scale automated use of emulation for migration (for example, opening an early-version WordStar file in an emulator, and then saving the file in a format that can be rendered in more current software) remains problematic. We have not yet devised a syntactic method for selecting and capturing that ambient contextual information so easily synthesized by a human viewer. Such methods would have to somehow make the leap from syntax to semantics, from information to meaning, to produce as a possible rendition what IE produces by syntactic inference.

IE's "choices" are intended gracefully to handle unexpected or invalid content. They are orthogonal to the semantically enriched choices of whatever automated syntactic capture and decision-making tools we might ultimately develop. Nonetheless, IE's "corrected" view of this web page affords a historically actual, and arguably authentic, experience of the page (as indeed can be said of Firefox's). And that distinctive view — that "affective design feature" — is also arguably a suitable object of preservation.

Do what I said, not what I meant

But what if we – and the rendering tool – have misinterpreted the author's intent? What if the purpose of this web page was not to convey information about embroidery tools and techniques? What if the idea was to create some sort of post-modern self-referential aesthetic experience? What if the whole point of this page is to make a viewer laugh, as the letters get bigger and bigger and run off the screen?

That is precisely the point of "Form Art", created in 1997 by Alexei Shulgin, and maintained in the Rhizome Artbase collection[9]. This work presents HTML pages with standard web page widgets – buttons, checkboxes, and so on — scattered across the page. None of the widgets is associated with conventional page or form navigation. As the creator describes this work:

> The work's appearance relies largely on whichever operating system the viewer is using to access it, resulting in a morphing aesthetic that updates itself over time, in tandem with software's constant evolution.

[9] http://rhizome.org/artbase/artwork/48528/

For this artifact, it makes no difference what a piece of browser software "forgives" or mangles, or, for that matter, renders "by the book." All that matters is that the rendering software creates some sort of effect in response to the HTML encodings, and that the user experiences some affect from that effect. "Normalizing" the view via some one or other version of a browser and operating system combination would in fact comprise a misinterpretation of the artwork, based on a fallacious interpretation of the author's intent. Here, respecting the author's intent means precisely not trying to interpret the author's intent as Internet Explorer effectively does in rendering the embroidery page.

This applies to artifacts other than those self-consciously created to employ variations in the rendition stack for aesthetic effect.

Software engineers are familiar with the notion of "leaky abstractions." Developers use abstractions to "divide and rule" the problem space they have undertaken to model with their software. Typically this takes the form of hierarchic layers, where activity at a lower layer is isolated "under the hood". An upper layer operates like someone turning the key or pushing a starter button to start a car. If, between rides, an engine sub-system is swapped out and replaced, this should make no difference to the driver.

In software, such abstractions are another case of "all very well in theory, but not in practice":

> What is clear then is that there is a basic discrepancy between our existing view of abstraction and the reality of day-to-day programming. We say that we design clean, powerful abstractions that hide their implementation, and then use those abstractions, without thinking about their implementation, to build higher-level functionality. But, the reality is that the implementation cannot always be hidden, its performance characteristics can show through in important ways. In fact, the client programmer is well aware of them, and is limited by them just as they are by the abstraction itself. [Kiczales]

Creators of web pages know this very well. Much of the "cruft" visible when you view the HTML source of a page is code that makes deliberate use of the peculiarities of different browsing software. The intent is typically either to make a page appear similar across different browsers, or, at the very least, to prevent the page from "breaking" across those same browsers.[10] It would be interesting to know how mystifying those "hacks" will appear to anyone inspecting that HTML source a century or so in the future.

When we construct an emulation stack, we are implementing an abstraction — a layered view of hardware, operating system, and application program. As good practitioners, we deliberately try to engineer systems that minimize the leakiness of our abstractions. And in so doing, we intentionally attempt to block access to those lower layers from applications that will be running in the emulator. In so doing, we can miss the "intent" of applications that were quite witting about the leakiness of lower layers in the hardware and software systems of their time, and employed that leakiness as a "design feature" of their code. This is certainly true of game software. [McDonough et al] It is also true in more conventional application software, as Kiczales has described. Running that leaky software on the

[1010] See, for example, [Meyer]

emulation stack might not obviously "break" – by crashing the application, for example. But neither is it guaranteed to render as intended by the author of the application software. That original performance will not have been preserved.

What *did* I mean?

Rhizome is able to respect author intent in "Form Art" by non-intervention – at least at present, where HTML-aware browsers are commonly available. But non-intervention is not always possible even, or especially, when respecting that original intent. Rhizome expresses how their practice attempts to resolve this dilemma:

> Our goal is to leave the original source code of an artwork intact and to always maintain original copies of an artwork's files. We believe that an artwork's source code is inextricably bound with the artist's process and practice, and exemplifies the technological and cultural landscape in which the work was created. As such, source code should always be preserved. That said, there is a long tradition of intervention in the preservation of traditional artworks, and we believe that such practices can also be used to preserve digital art. The need to leave code untouched is outweighed by the need to make these important works viewable in perpetuity. If an artwork needs to be updated to comply with contemporary technologies, we create a separate, specially-named copy of the work that we modify as needed, leaving the original file untouched and available online.[11]

Rhizome's experience tells us that, at least for the foreseeable future, the curation of digital objects is going to involve a considerable amount of introspection into preserved code. If we are lucky, we will have the source code available to us. Source code is preserved as text. And at least one feature of software language design is to create a human-intelligible abstraction layer over lower-level machine instructions. Such code is thus likely to be less opaque to human inspection than the binary image of an executable file. But it is not completely transparent.

As with any digital object we are preserving, we have to ask of source code files: what does it mean to characterize this object? For a start, we need to capture the fundamental properties (language, version, operating system and other dependencies) detailed in the 2008 JISC study of the significant properties of software [Matthews], and articulated by taxonomies such as TOTEM. [Delve]. But beyond that, it will entail description, or at least awareness, of the higher level abstractions employed by developers when constructing their code:

> Young as it is as a discipline, software engineering has a history, and conventions and practices that have evolved over time. A condensed history of those conventions and practices would begin with people flipping binary switches, then coding in assembler languages, then using FORTRAN, COBOL, and other procedural languages, then refining those procedural languages with techniques of structured programming. The taxonomic tree would branch out into object oriented idioms and languages, functional programming, declarative programming, resolution

[11] http://rhizome.org/artbase/about/

> theorem proving logic programming, the use of non-deterministic and genetic algorithms in machine learning, and, cutting across many of these, the use of massively parallel programming idioms. It is not only particular programming languages that have a vogue and then disappear from common use; so do the large-scale conceptual structures of software construction. And the community of users schooled in these various languages and idioms fluctuates – and can diminish to the vanishing point – over time. [Morr 2010]

These idioms include such developer shorthand as the "design patterns" [Vlissides] that describe archetypal solutions to commonly recurring problems in software, such as creating software "adapters" to enable common methods of invocation across code developed at different times, or for different purposes than its current use. But it also means sensitivity to the organizing metaphors that informed design choices, even from the earliest days of computing.

This is true even down to the level of hardware. Thus the designers of the early EDSAC machine spoke of memory and registers as storage "tanks." [Campbell-Kelly] The so-called von Neumann machine, constructed beginning in 1946 at the Institute for Advanced Study, introduces its design in terms of "certain main organs relating to arithmetic, memory storage, control, and connection with the human operator." [Burks] At present, we are so accustomed to the notion of information being organized on our computers as files in a hierarchical directory tree structure, that it can be something of a surprise to those with no experience of earlier operating systems to read the description of the design of the original UNIX operating system and learn that this construct was an innovation. [Ritchie]

The evolution in "how software operates" already presents challenges to researchers attempting to construct emulation stacks for software that is less than thirty years old. [von Suchodoletz] And what changed once, of course, can change again. Certainly not encapsulated in any one instance of preserved source code, but *somewhere*, (and, if we're very lucky in machine-actionable form) we will need to preserved the "folk history" that informs the semantics of the seemingly straightforward text files that are source code. [Morr 2011] [McDonough]

> Effective knowledge of the constraints in the use of such software means maintaining not just the static documentation of … programming libraries, but also a living awareness in the community of use of these artifacts about potential issues with temporal data structures in older architectures and coding implementations. [Morr 2010]

Do what you say, not what I meant

There is more than one possible performance of a digital object. We know the dancer from (by means of) the dance; we know the same dancer in new ways, depending on the dance.

Increasingly few of us, even at the time of writing this (2013), have access to the operating systems in use when "Form Art" was created. This accords with the author's expressed intent, which was to create effects and user experiences that changed over time as software and hardware systems change. This does however make it very challenging to reconstitute what a viewer in 1997 might have seen, if we are

so minded. Such a recreation of an historical view, though perhaps contrary to the author's intent, is arguably a legitimate use of the author's creation.

When we consider which digital objects and which renderings of those objects to target for preservation, we employ the abstraction of "community of use," within which we create another abstraction, "*the* user." And this user's expectation establishes what is considered a normal rendering for a given form of expression. But the extreme plasticity of digital objects — as fundamental an aspect of the preservation challenge as their extreme fragility – complicates the definition of what "user expectation" means.

This is another way of saying that, rather than there being a single "community of use" or user within that community, there are multiple possible, and often originally unenvisioned, communities of use, as well as multiple, potentially unanticipated, uses by those communities. When scholarly journals transitioned from print to electronic form, for example, they made use of PDF to create a digital analogue of the printed page. The intended use of an article in that format was an individual human reader "reading" the electronic "pages" much as that reader would have read paper pages. Not yet two decades into this transition, a new scholarly use of the "intellectual content" of those same PDF-encapsulated articles is not an individual scholar reading an individual article and interpreting its content, but an automated process consuming millions of such articles, and presenting the results of computations upon those contents to a scholar for interpretation.

The complexity of textual representation in PDF means that extracting text, and, by implication, the "intellectual content" of any given article, can be extremely problematic. It is mediated by software that works with varying degrees both of conformance to the PDF specification, and of success in textual extraction, on artifacts produced by many different PDF creators, which themselves implement the PDF specification with varying degrees of correctness. [Morr 2012]

There are a couple of implications in this of interest to preservationists, beyond the above-mentioned need to correlate format instance characterization with format renderer capabilities. We could easily describe text-mining of PDF files as a sort of massive aggregate format migration. The intent of text mining is the surfacing of saliences in large corpora of content. These saliences then inform new scholarly statements about bodies of historical texts or literary materials. How can the experiences gained by preservations in the risks and potential deformations of content in format migrations be made an active part of the intellectual assessments made by scholars employing text mining tools, so that their conclusions might be appropriately qualified and refined?

Leaf, blossom, *and* bole

What breadcrumbs can we glean from such "literary criticism" as this to leave behind for future archaeologists of our present-day digital artifacts?

Understanding the entanglement of author, text, and reader; of software, artifact, and viewer; of the many layers of abstraction and metaphor, and the interactions among them, suggest things we might wish to capture.

We will want to characterize format instances and the software that generates or renders them in terms of each other. We will want to characterize software with respect to author intent (metaphor and abstractions, the leakiness of those abstractions). We will want to contextualize artifacts, but in such a way as to "expose the interface", so to speak, with respect to the many, various, and emerging user intents for those artifacts. We will have to recognize that the meanings of abstractions at the time of creation, and at time of capture, are themselves not stable, and are subject to misinterpretation. [Morr 2011, McDonough 2013] We will think of a scholarly electronic edition of a text, for example, as necessarily a variorum edition. We will need to "divide and rule" – to undertake many such efforts as those of the Archive Team[12] to capture the raw materials of software system interpretation, and then, as we find the resources, find ways to make those materials "actionable" by automated systems.

Necessarily, pragmatically, given the sheer volume of already-existing digital content to be preserved, we as preservationists focus on large-scale, sometimes coarse-grained capture of digital content and contextual information, whether structured as metadata, or otherwise. We construct emulation stacks for the most "typical" hardware and software combinations. And then, after a certain point, we trust to digital archaeologists, critics, interpreters of the future to assemble and reconstitute the living experience of the artifacts of our time.

[12] http://www.archiveteam.org/index.php?title=Main_Page

References

All URLS effective as of 31 August 2013.

[Arms] Arms, Caroline, Fleischhauer, Carl, and Kate Murray. "Sustainability of Digital Formats: Planning for *Library of Congress Collections.*" *Available at* http://www.digitalpreservation.gov/formats/sustain/sustain.shtml

[Burks] Burks, Arthur W., Herman Goldstine, and John von Neumann. "Preliminary discussion of the logical design of an electronic computing instrument." Institute for Advanced Study. 1946. *Available at http://www.ece.cmu.edu/~ece447/s12/lib/exe/fetch.php?media=wiki:neumann-1946.pdf*

[Campbell-Kelly] Campbell-Kelly, Martin. "EdsacPC A Tutorial Guide to the EDSAC Simulator Windows 95, 98 and NT Edition." July 2001. *Available at* http://www.dcs.warwick.ac.uk/~edsac/Software/EdsacTG.pdf

[Cochrane] Cochrane, Euan. "Rendering Matters: Report on the results of research into digital object rendering", Archives New Zealand (2012). Available at http://archives.govt.nz/rendering-matters-report-results-research-digital-object-rendering

[Delve] Delve, Janet and David Anderson. *The Trustworthy Online Technical Environmental Metadata Database – TOTEM.* Verlag Dr. Kovač GmbH. 2012.

[Garber] Garber, Marjorie. *The use and abuse of literature*. Random House LLC. 2011.

[**Heslop**] Heslop, H., Davis, S. and Wilson, A. "An Approach to the Preservation of Digital Records." (2002). Available at http://www.naa.gov.au/Images/An-approach-Green-Paper_tcm16-47161.pdf

[Jackson] Jackson, Andrew N. "Using Automated Dependency Analysis to Generate Representation Information." In *iPres 2011 The Eighth International Conference on Preservation of Digital Objects*., pg. 89 (2011).

[Kiczales] Kiczales, Gregor. "Towards a New Model of Abstraction in Software Engineering." In *Proceedings of the IMSA'92 Workshop on Reflection and Meta-level Architectures, 1992*. Available at http://www2.parc.com/csl/groups/sda/publications/papers/Kiczales-IMSA92/for-web.pdf

[Matthews] Matthews, Brian, Brian McIlwrath, David Giaretta and Esther Conway. "The Significant Properties of Software: A Study." 2008. available at: http://www.jisc.ac.uk/media/documents/programmes/preservation/significantpropertiesofsoftware-final.doc

[McDonough] McDonough, Jerome. "Some Assembly Required: Reflections on XML Semantics, Digital Preservation and the Construction of Knowledge." Presented at Balisage: The Markup Conference 2013, Montréal, Canada, August 6 - 9, 2013. In *Proceedings of Balisage: The Markup Conference 2013*. Balisage Series on Markup Technologies, Volume 10 (2013). doi:10.4242/BalisageVol10.McDonough01.

[McDonough et al] McDonough, J., Olendorf, R., Kirschenbaum, M., Kraus, K., Reside, D., Donahue, R., Phelps, A., Egert, C., Lowood, H., & Rojo, S. Preserving Virtual Worlds Final Report. (2010). Available at http://hdl.handle.net/2142/17097

[Meyer] Meyer, Eric. *On CSS: Mastering the Language of Web Design.* New Riders Publishers. 2002.

[**Morr 2010]** Morrissey, Sheila M. "The economy of free and open source software in the preservation of digital artefacts", *Library Hi Tech*, Vol. 28 Iss: 2, pp.211 – 223 (2013). Available at http://www.portico.org/digital-preservation/wp-content/uploads/2010/11/The-Economy-of-Free-and-Open-Source-Software-in-the-Preservation-of-Digital-Artifacts.pdf

[Morr 2011] Morrissey, Sheila M. "'More What You'd Call "Guidelines" Than Actual Rules': Variation in the Use of Standards." *Journal of Electronic Publishing*. Vol. 14 Iss: 1 (Summer 2011). DOI: http://dx.doi.org/10.3998/3336451.0014.104

[Morr 2012] Morrissey, Sheila M. "The Network is the Format: PDF and the Long-term Use of Digital Content." Presented at IS&T Archiving 2012, Copenhagen, DK, June 2012. In *Proceedings of Archiving 2012*. Vol. 8, pp. 200 – 203. Available at http://www.portico.org/digital-preservation/wp-content/uploads/2012/12/Archiving2012TheNetworkIsTheFormat.pdf

[Postel] Postel, J. (ed.). "Internet Protocol - DARPA Internet Program Protocol Specification." RFC 793.USC/Information Sciences Institute. September 1981. Available at http://tools.ietf.org/pdf/rfc793.pdf

[Ritchie] Ritchie, Dennis M. and Ken Thompson. "The UNIX Time-Sharing Sytem". *Communications of the ACM.* VOl. 17, No. 7. (July 1974).

[Vlissides] Vlissides, John, R. Helm, R. Johnson, and E. Gamma. "Design patterns: Elements of reusable object-oriented software." Reading: Addison-Wesley 49 (1995).

[von Suchodoletz] von Suchodoletz, Dirk, and Jeffrey van der Hoeven. "Emulation: From Digital Artefact to Remotely Rendered Environments." Presented at iPres 2008, London, UK, September 2008. In *Proceedings of the Fifth International Conference on Preservation of Digital Objects*, pp. 93-98. Available at http://www.bl.uk/ipres2008/ipres2008-proceedings.pdf

[Wimsatt 1946] Wimsatt Jr., William K. and Monroe C. Beardsley. "The Intentional Fallacy." *The Sewanee Review,* Vol. 54 (Jul. - Sep., 1946), pp. 468-488. Available at http://www.jstor.org/stable/27537676

[Wimsatt 1949] Wimsatt Jr.,William K. and Monroe C. Beardsley. "The Affective Fallacy." The Sewanee Review , Vol. 57, No. 1 (Winter, 1949), pp. 31-55. Available at http://www.jstor.org/stable/27537883