Conserving Digital Art for the Ages

Paper presented at Media in Transition 7: *Unstable Platforms: The Promise and Peril of Transition*. Boston, MIT, May 13-15, 2011.

FRANCIS T. MARCHESE

Dept. of Computer Science
Pace University
New York, NY 10038
http://csis.pace.edu/~marchese
email: fmarchese@pace.edu

Will digital art created in the late twentieth and early twenty-first century be displayable 500 years in the future? The ability to exhibit such artwork will depend upon new thinking and practices developed today by artists, conservators, and curators. This paper discusses how the use of software engineering methodologies can provide a means for transforming conservation practices used for traditional art into methods more appropriate for digital-based media. It will show as well how software engineering processes will aid digital art scholarship by augmenting and organizing an artwork's components in such a way as to enhance accessibility by art historians. Finally, it will discuss how digital artists who choose to adapt software engineering practices to their artistic process will be able to naturally extend the lifespan of their artwork.

INTRODUCTION

Over the past decade artists have capitalized on innovations in computing and the nearly limitless access to streams of information from the World Wide Web to construct artworks that have ranged from ephemeral performance to immersive installation. The same relentless advances in the evolution of computer technology that have stimulated and driven the widespread growth in digital art creation could quickly lead as well to its certain demise. Museum conservators of digital media whose job it is to preserve these works are faced with the daunting if not impossible task of managing this art so as to make it displayable at any time in the future. Just consider the following issues. Digital artists employ a diversity of computer languages that currently include C/C++, Java, JavaScript, C#, Flash, HTML, XML, Python, Processing, Perl, and Max/MSP. They exploit software development libraries and environments for such things as mobile app development, sound composition, virtual worlds, and computer games. Underlying system software substrates that support their artwork are in continuous flux with operating systems and network protocols evolving, database formats changing, and globally accessible resources either disappearing or becoming redistributed. And computer hardware underlying digital art is guaranteed to become obsolete. Thus, it is certain that a change in the simplest digital protocol could make the museum display of an artwork difficult even twenty years in the future, much less five hundred years hence.

Solutions to these problems will evolve from the combined efforts of artists, conservators, and curators to create a set of best practices to support a museum's long-term management of digital artwork. These practices can be adapted from the field of software engineering. Software engineering focuses on both the software as product and the processes utilized to create and maintain it [1]. Software engineering already plays an essential role in software development within the aerospace, high technology, and financial service industries. Because its processes are extensions of the standard business life-cycle [2], its tools and techniques may be integrated into a museum's conservation practice. Software engineering as a process may engage all stakeholders who comprise an art museum's business practice including artists, curators, conservators, installers, maintainers, museum directors, art historians, and viewers; and can reflect and integrate this process into a museum's current best practices.

DIGITAL ART CONSERVATION

Preservation is an essential part of museum practice. Once an institution has decided to acquire an artwork, its evaluation and care is entrusted to the museum's conservators. The practice of art conservation is a formal scientific activity defined in the following way by the International Council of Museums Committee for Conservation [3]:

The activity of the conservator-restorer (conservation) consists of technical examination, preservation, and conservation-restoration of cultural property: Examination is the preliminary procedure taken to determine the documentary significance of an artifact; original structure and materials; the extent of its deterioration, alteration, and loss; and the documentation of these findings. Preservation is action taken to retard or prevent deterioration of or damage to cultural properties by control of their environment and/or treatment of their structure in order to maintain them as nearly as possible in an unchanging state. Restoration is action taken to make a deteriorated or damaged artifact understandable, with minimal sacrifice of aesthetic and historic integrity.

Traditional conservation practice thus focuses on an artwork as an integrated physical whole, the integrity of which must be preserved. Change is defined as a process that will deleteriously affect the stability of an artwork, moving it away from its original reference state and altering its identity. Conservators have adapted this approach to preserving computer supported artwork by explicitly targeting both physical and digital artifacts. In the former case, computer technology is stockpiled to support the artwork in the inevitable event that a component fails (e.g. CPU, hard drive, power supply, etc.). In the latter, digital artifacts such as computer programs or digital videos must be archived to durable media. Because any digital storage medium (e.g. tape, CD-ROM, DVD) ultimately either decays or becomes obsolete, all digital artifacts must be routinely refreshed to a new storage medium. In general, museums' approaches to digital preservation have focused primarily at the institutional level, considering organizational goals, priorities, available resources, and management policies [4]. A digital object (e.g. document, image, and video) within this context becomes a discrete entity with well defined attributes that can be managed throughout its lifetime. The goal of the digital preservation community as a whole has been to create standards and develop best practices for the conversion of digital material into "archival" formats that can be manipulated and shared.

Because change is a fundamental part of its nature, time-based digital artwork, that is artwork whose aesthetic experience evolves over time, does not fit this definition. Museum conservators in charge of maintaining time-based media realize this, and are attempting to expand the conservation paradigm to accommodate digital art. Pip Laurenson, Head of Time-based Media Conservation at the Tate Modern museum, has proposed a redefinition of conservation practice to accommodate time-based media, where conservation becomes the means by which an artwork's essential properties are documented, understood, and maintained. Its aim is the preservation of an artwork's identity, so that it may be displayed in the future as different possible authentic installations [5]. For Laurenson, the identity of a digital work should be considered as a collection of properties which include: the artist's instructions, approved installations intended to act as models, an understanding of the context in which the art was made, and the degree to which the artist specifications reflect his or her practice at the time the art was created. Hence, an extended set of documentation is assembled to help define and contextualize the artwork with the express purpose of making the artwork displayable at some future date. Such things as artist interviews, questionnaires [6], artist-conservator-curator collaborative discussions, conservation workshops [7], and documentation of a program's source code are all recent experiments that remain to be integrated into the formal scientific activity of art conservation [8]. Yet, this expanded notion of an artwork's identity as a collection of concepts and artifacts may be used as a starting point for a formal identification and documentation of an artwork employing the principles and practices from the field of software engineering.

DOCUMENTATION AND CONSERVATION

When a museum acquires a digital artwork it is usually assumed that the work will be exhibited as is, with no further enhancements expected from the artist creator. This work along with its conceptual and technological underpinnings is now frozen in time. In computing this is known as a legacy system [9]. Assessing the state of a legacy artwork at some point in time may be difficult for many reasons. The artwork's existing documentation may be incomplete. Or the curatorial context which supported the original acquisition and installation of the artwork may have been reoriented, thus making maintenance of some legacy artworks more difficult. The exact nature of the digital artwork itself may be difficult to assess. It may have been assembled from many diverse components without a consistent design or programming style. Or it may be in either executable form or written in an arcane programming language.

From a business perspective, there are four approaches for dealing with traditional legacy systems. The first is to scrap the system outright, as business practices have changed and the system is no longer needed. Continued maintenance of the software is possible if the system continues to work well. If the system's usefulness continues to degrade over time, then the system or its parts must be transformed to improve maintainability. Finally, a system must be replaced, if either obsolete hardware or software precludes further operation, or the new system can be built at reasonable cost. For conservators, each one of these approaches creates its own set of issues, with degrees of intervention from minimal adaptation to full restoration, many of which are related to the importance of the artwork – an importance that should be expected to change over time. This is why it is most important that the long term process of conserving a digital artwork must be based on a thorough understanding of its structure or architecture, as represented by a breadth of documentation.

Documentation is an intrinsic component of any system. Software engineering provides a systematic methodology for creating and maintaining documentation to support communication, preservation of system and institutional memory, and processes such as system auditing. Within this context a computer system's documentation should supply comprehensive information about its capabilities, architecture, design details, features, and limitations. It should encompass the following five components [10]:

1. *Requirements* - Statements that identify the capabilities and characteristics of a digital artwork. This is the conceptual foundation for what has been created.
2. *Architecture/Design* – An overview of software that includes the software's relationship to its environment and construction principles used in design of the software components. Typically a system's architecture is documented as a collection of diagrams or charts that show its parts and their interconnections.
3. *Technical* - Source code, algorithms, and interfaces are documented. Comments may be embedded within the system's source code and/or parts of external documentation.
4. *End User* – Manuals are created (e.g. static documents, hypermedia, training videos, etc.) for the end-user, system administrators, and support staff.
5. *Supplementary Materials* – Anything else related to the system. This includes: legal documents, design histories, interviews, scholarly books, installation plans, drawings, models, documentary videos, websites, etc.

Each component is important to the representation of a digital system. Each may operate at a different level of abstraction or within a particular context. Requirements documentation presents the conceptual view of what the system is expected to do. It is written to be understood by all the stakeholders who comprise an art museum's business practice. Architecture/Design documentation functions very much like an architect's sketch of a building, showing all its components and how they fit together. Technical documentation represents the bricks-and-mortar of the artwork, conveying information about how the artwork is constructed.

Besides facilitating an artwork's conservation this documentation could also support scholarship, enabling art historians to understand an artist's working process and evolution of practice. A computer system's structure is a reflection of the conceptual space in which the artist had been working at the time the art was created. The number of software components, their hierarchy, and the interconnections among them, should give an idea of how the artist viewed a representation problem, and how it was transformed into a computer system. An analysis of documentation should yield answers to questions about authorship, educational context, craftsmanship, aesthetics, development process, technical context, and conceptual foundations. Was software written in the hand of the artist, or was it built by others? Who influenced the artist conceptually or technologically? How well is the program written and system built? How well conceived and designed is the system? Does it possess an elegance and refinement comparable to any other beautifully created object? What were the design strategies used by the artist? What were the development tools available at the time the artwork was created? And what theories of computing did the artist use?

DOCUMENTATION AND DIGITAL ARTISTS

Documentation is an intrinsic part of a digital artwork. A program's source code and associated data are its *de facto* documentation. The question remains - how much more documentation is required to provide a sufficient representation of a digital artwork? This is an open question for any software engineering project, and depends on factors such as project size, complexity, and expected system lifespan. It may be argued that it is not the artist's responsibility to sufficiently document an artwork; but if contemporary curatorial practice is an indication of what the distant future will hold, the following scenario is most likely to occur. Artwork selection will not only be based on its importance to the canon but also the availability of resources, such as staffing, time, and funds required for its installation. An artwork that may be the best example of a theme or idiom may need to be replaced in an exhibition by a lesser work, because its own documentation proves insufficient for its recreation.

It is posited here that a digital artist shares a certain responsibility for the long-term preservation of an artwork. Traditional artists who employ archival media and follow well established best practices for creating a stable physical artwork produce works with a high probability of standing the test of time. For those artists who consciously choose to work in a non-archival way, the long term preservation and ultimate exhibition of their works is a problem with an indeterminate solution. It should be remembered that major museums possess culturally significant artworks that cannot be exhibited because of their fragility or degree of deterioration!

Digital artists have choices analogous to their traditional counterparts for maintaining their artwork's longevity. Best practices exist for the development of computer software in different ways. For example, the use of standard programming styles, data structures, algorithms, and the selective insertion of comments into source code represent programming best practices. Software engineering best practices work at a different level. The software engineering conceptual process of analyze-design-build is consistent with artistic practice. And the tools used by software engineers during the analysis and design phases of software development, allow software designers literally to sketch out a system's architectural design. Including these design representations with an artwork's source code expands it representational details to encompass the Requirements, Analysis/Design, and Technical documentation categories discussed above. It should provide as well a sufficient description of the artwork's identity to allow it to be recreated at some future time if all other conservation strategies fail.

FUTURE INSTALLATIONS

Two examples of digital art will be discussed from the perspective of an installation that would be part of an exhibition five hundred years in the future. This is not an unreasonable expectation given that five century old paintings by Leonardo, Raphael, Titian, and Bellini grace the walls of major art museums throughout the world. And through a combination of the original artist's craftsmanship and conservation science these works will most likely have an equal or better chance of being displayed at the same future time as this digital work.

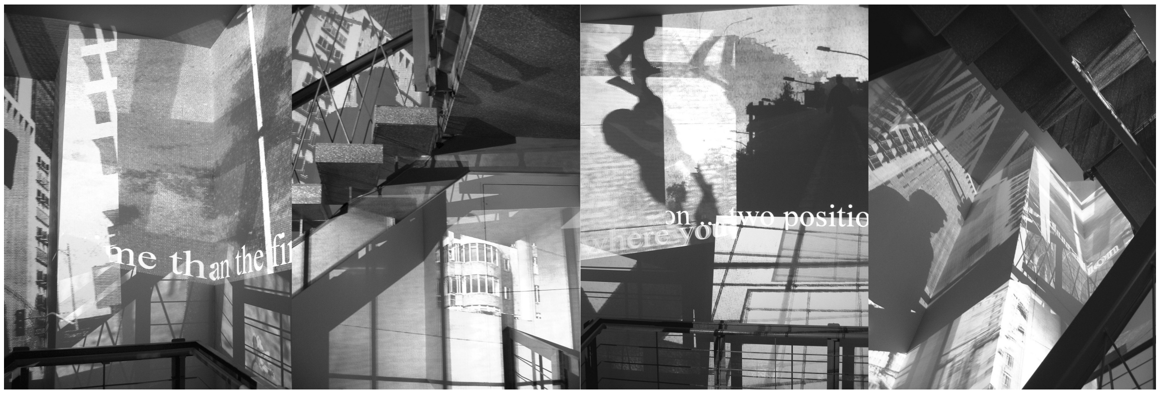

Figure 1. Images of *Trigger* installation at Pace Digital Gallery.

Trigger

Trigger was a site-specific, sensor-activated, immersively projected, interactive art installation by the California artist Jody Zellen that debuted at the Pace Digital Gallery in fall 2005 [11]. Zellen created *Trigger* to explore the transient stories that emerge from our relationships with urban spaces. *Trigger* filled the gallery with overlapping videos from seven projectors, infusing its space with transient sounds. When visitors passed motion sensors, sounds and videos changed, evoking the ephemeral nature of urban space and the fleeting and shifting perceptions of it (Figure 1).

Trigger's documentation included: the artwork's original Flash videos, a video walkthrough of the artwork, images of the installation, a catalog, and a short technical paper about the artwork [12]. Neither executable software nor source code survived after *Trigger* installation. The technical paper explained the collaborative process through which *Trigger* was created and put forward its functional requirements, architecture, and design – key information for its possible recreation. The artwork's functional requirements that identified its capabilities and characteristics were communicated in the technical paper as follows:

- The artwork would support a large number of projectors, sensors, and speakers.
- The artwork's sensors would capture viewer motion.
- Captured motions would trigger events.
- Events would be communicated to the artist's multimedia programs.
- The multimedia programs would change the content of the displays and alter sounds.

Trigger's architectural design conveyed the high level interrelationships among its components without specifying the processing details. It possessed a simple architecture of three loosely coupled components: a microcontroller-sensor system, application software, and interface software linking the sensor system to the application. Each of these components worked independently, communicating by simple message passing. Such a relationship exhibited the attribute of *low coupling*, a fundamental software design paradigm. Finally, the technical paper briefly described each component's responsibilities. In all, Trigger's technical discussion comprised no more than a page, but combined with its supplemental materials, such as video walkthrough and installation images, sufficient information for a distant future recreation of the artwork that would maintain both its identity and reflect Zellen's artistic practice.

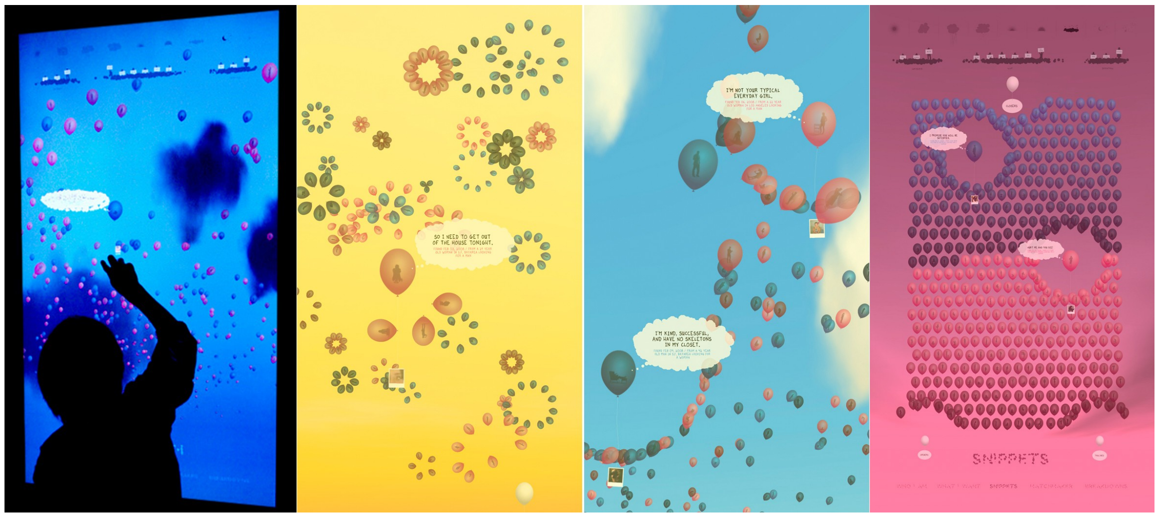

Figure 2. Images of *I Want You to Want Me*. Interaction and selected screen captures. See [13] for details.

I Want You to Want Me

I Want You to Want Me is an interactive installation about online dating designed and built by Jonathan Harris & Sep Kamvar [13], commissioned by the Museum of Modern Art (MoMA) in New York and installed on Valentine's Day 2008 as part of its *Design and the Elastic Mind* show [14]. Displayed on a vertically mounted 56" touch screen monitor, the artwork portrays a sky filled with hundreds of pink (female) and blue (male) balloons, each representing an individual's online dating profile, harvested and coalesced from several dozen Internet dating websites (Figure 2). Viewers can touch individual balloons to reveal personal information about the dater found inside, and can rearrange the balloons in various ways to highlight different aspects of the world of online dating, including the most popular first dates, top desires, self-descriptions, and interests.

I Want You to Want Me is based on a client-server architecture. A client front-end locally controls the 3D graphics display while the application backend is housed in a server farm in California. A URL server provides addresses to a web crawler which sends dating site information to an information extractor the responsibility of which is to fill a database with information about individuals, approximately one million elements in size. Data from this database is accessed and passed on to the front-end application using an API that sets up search strategies and queries. The programming languages and components used to build the artwork include C++, Java, PHP, OpenGL, and SQL. The servers run the UNIX operating system and the client runs Windows XP. Dcumentation available for this artwork included an installation manual, screen shots, images of installation, and design sketches.

Discussion

Trigger's clearly defined functional requirements and simple architecture make its future recreation possible. One approach would be to instantiate its architecture in a data flow programming language. Data flow programming languages have had a long history, evolving from workstation-based scientific visualization systems such as AVS [15] in the late 1980s. Recognized for their ease and versatility in defining parallel event-based systems, these visual programming languages would be expected to continue to evolve over time. A contemporary example is Cycling 74's Max/MSP that is used by multimedia artists, composers and musicians, and in the theater to control lighting and staging. Indeed, the original developers of *Trigger* had considered using Max/MSP in 2005, but were constrained from doing so.

In contrast *I Want You to Want Me* would present greater challenges for its future installation. It is far more complex than *Trigger*, with its processing and database distributed over multiple computing platforms and locations. It requires several programming languages for its construction and execution. The structure of its database and the information mining algorithms employed to extract data are unknown, although one of the artwork's creators (Kamvar) has developed such algorithms as part of his research, and his publications could provide some enlightenment. It is unclear how tightly the computer display system is coupled to the computer graphics software, but the assured obsolescence of the entire hardware system would mean that 26th century conservators will need to understand how the system is built in order to adapt it to the computer systems of their time. Exacerbating this problem is that no detailed documentation exists for *I Want You to Want Me* describing its static and dynamic architectural designs. At best MoMA owns this work's source code, at worst only it executable code. If MoMA owns the source code, the museum conservators can work their way through each program component to construct the architectural design documents that will allow them to reinstantiate the artwork in a future computing system. However, if MoMA only owns the executable code, then in light of the paucity of its other documentation, the display of *I Want You to Want Me* in the distant future will most certainly be doomed because of insufficient technical documentation. Finally, even if the future conservators have access to the artwork's source code, its size and complexity may precluded recreation because of time, staffing, and financial constraints.

The future display of *Trigger, I Want You to Want Me,* and other digital artwork depends not only on the quantity of documentation but also the quality. Although *Trigger* possesses very little documentation and no source code, its quality stemms from its unambiguous coverage of the two highest level conceptual representations of a system – its requirements (what the system is supposed to do) and architecture/design (how to organize its parts to do it). Conversely, the technical documentation for *I Want You to Want Me* may not be able to support its recreation because it insufficiently records these same high level representations; engendering their recapture through a significant expenditure of effort. Hence, the judicious creation of a digital artwork's documentation will improve its chances of installation in the distant future. How this task will be accomplished will depend on what documentation will become important to conservation practice and how artists integrate software engineering methods into their creative process.

CONCLUSIONS

The display of contemporary digital art at some time in the distant future remains an open problem, the solution to which will come from efforts by artists, conservators, and curators. It has been proposed here that the use of software engineering practices will provide a means for transitioning from the conservation practices used for traditional art to methods more appropriate for computer-based media. This process will aid digital art scholarship as well, by organizing an artwork's components in such a way as to enhance accessibility by art historians. Finally, digital artists who choose to adapt software engineering practice to their artistic process will be able to extend the lifespan of their artwork.

REFERENCES

1. R.S. Pressman, Software Engineering: A Practitioner's Approach, 6th Edition, (New York: McGraw-Hill, 2005).
2. R.K. Ko, "A Computer Scientist's Introductory Guide to Business Process Management (BPM)," Crossroads Vol. 15, No. 4,

11-18 (2009).

3. International Council of Museums Committee for Conservation, “The Conservator-Restorer: a Definition of the Profession, Section 2.1,” Retrieved January 10, 2011 from http://www.icom-cc.org/47/about-icom-cc/definition-of-profession/.

4. NINCH Working Group on Best Practices, “The NINCH Guide to Good Practice in the Digital Representation and Management of Cultural Heritage Materials Online: The National Initiative for a Networked Cultural Heritage,” (2002). Retrieved January 10, 2011 from http://www.nyu.edu/its/humanities/ninchguide/.

5. P. Laurenson, “Authenticity, Change and Loss in the Conservation of Time-Based Media Installations,” Tate Papers, Autumn, (2006). Retrieved March 15, 2011 from http://www.tate.org.uk/research/tateresearch/tatepapers/06autumn/laurenson.htm

6. J. Ippolito, “Accommodating the Unpredictable: The Variable Media Questionnaire,” Permanence through Change: The Variable Media Approach Guggenheim Museum Publications and The Daniel Langlois Foundation for Art, Science, and Technology, (2003).

7. ERPANET, The Archiving and Preservation of Born-Digital Art Workshop, Briefing Paper for the ERPANET Workshop on Preservation of Digital Art, (2004). Retrieved January 10, 2011 from http://www.erpanet.org/events/2004/glasgowart/briefingpaper.pdf.

8. For a concise review of the current state of the problem see T.A. Yeung, S. Carpendale and S. Greenberg, “Preservation of Art in the Digital Realm,” The Proceedings of iPRES2008: The Fifth International Conference on Digital Preservation, (London: British Library, 2008).

9. J. Ransom, I. Sommerville, and I. Warren, “A Method for Assessing Legacy Systems for Evolution,” Proceedings of the 2nd Euromicro Conference on Software Maintenance and Reengineering (Csmr'98) (March 08 - 11, 1998), (Washington: IEEE Computer Society, 1998) p. 128.

10. For a complete discussion see C. Larman, Applying UML and Patterns: An Introduction to Object-Oriented Analysis and Design and Iterative Development. 3rd Edition (New Jersey, Prentice-Hall, 2005).

11. J. Zellen, “Trigger,” Pace Digital Gallery, New York, NY, (October 18 - November 8, 2005). Retrieved March 15, 2011 from http://www.jodyzellen.com/pace2.html.

12. F.T. Marchese, “The Making of Trigger and the Agile Engineering of Artist-Scientist Collaboration,” Proceedings of the Tenth International Conference on Information Visualization: IV'06 (London, July), (Washington: IEEE Press, 2006), pp. 839-844.

13. J. Harris and S. Kamvar, “I Want You to Want Me,” (2008). Retrieved April 15, 2011 from http://iwantyoutowantme.org/

14. “Design and the Elastic Mind,” Museum of Modern Art (MoMA), (2008). Retrieved April 15, 2011 from http://www.moma.org/interactives/exhibitions/2008/elasticmind/

15. C. Upson, T. Faulhaber Jr., D. Kamins, D. Laidlaw, D. Schlegel, J. Vroom, R. Gurwitz, and A. van Dam, "The Application Visualization System: A Computational Environment for Scientific Visualization,” IEEE Computer Graphics and Applications , Vol. 9, No. 4, 30-42 (1989).