Simplified Distributed Authoring Via Component-based Object Construction and Deconstruction in Collaborative Croquet Spaces

Howard Stearns, Joshua Gargus, Martin Schuetze
Division of Information Technology
University of Wisconsin-Madison
[hstearns, gargus]@wisc.edu, martin.schuetze@gmail.com
and
Julian Lombardi
Office of Information Technology
Duke University
julian@duke.edu

Abstract

We introduce a component architecture in support of a simplified approached to distributed authorship of functional content and user interface elements within Croquet-based 3D spaces. Known as "Brie," the approach is designed to take advantage of social opportunities afforded by Croquet's collaborative architecture by greatly simplifying content creation through an intuitive and highly flexible user interface approach. By reducing the technical demands of authoring functional content in Croquet-based multiuser environments, we hope to facilitate expansion of the potential community of content developers to include graphic designers and technically naïve users. The Brie component-based approach addresses this in two ways: 1) it establishes a fundamentally new way for users to construct new objects and behaviors de novo, and 2) it establishes a way for users to easily deconstruct and recombine objects and behaviors received from other users, discovered through a search mechanism, or encountered as objects within existing Croquet spaces.

1. Introduction

The Croquet platform [1][2] enables new dimensions of online collaboration by allowing users to easily create their own 3D environments, and to encounter other users within those environments. An obstacle to fully realizing this vision is the prohibitive cost of developing 3D content, as video game budgets approach those of Hollywood blockbusters.

This is a particularly troubling problem for Croquet. Content creation threatens to become a bottleneck that stands in the way of fully realizing the benefits promised by Reed's Law [3]. It is therefore imperative to circumvent this obstacle; this is what Brie is designed to do.

Brie's solution to the problem of content creation is technical as well as social: our approach is to empower any user (not only computer artists and programmers) to develop and re-deploy compelling content within 3D worlds of their own individual or collective creation by reducing the costs of content creation. Firstly, the Brie component architecture provides tools that are easy to use and that use direct-manipulation of live objects to leverage a user's familiarity with object handling and manipulation in the real world. Secondly, the Brie user interface encourages a culture of object and component sharing between users of massively multi-user worlds. Users of Brie-enabled Croquet spaces can copy and modify objects and behaviors that others have already created. Such objects and components may be encountered in Croquet spaces, shared among users, or searched for in global content [4].

The Brie component-based architecture also seeks to provide a highly extensible framework that allows multiple user interfaces (henceforth "UIs") to function simultaneously and independently within a single shared space [5].

2. What is Brie?

Brie is a language architecture and framework that supports direct manipulation and creation of 3D content and user interfaces. The component-based

architecture of both the language and the UI framework is described in a companion paper [6].

Brie's language is an extension of Squeak Smalltalk [7] that facilitates the creation of interoperable components within shared Croquet-based 3D spaces. Such components can interact with (and be composed of) independently created components. Interaction between components will be possible even if each component is designed and implemented without knowledge of the others.

The Brie framework also delivers a proof-of-concept UI that forms the basis for the discussion in the second half of this paper.

3. Definition of Users

For the purposes of this paper, we classify potential Croquet users into four groups (based on analogous groups for the current World Wide Web):

- "Consumers"
- "Authors"
- "Programmers"
- "Wizards"

The *consumer* group is the potentially largest of all of these groups. It consists of people who browse content that has been authored by others. *Authors* are the creators of worlds and the content contained therein. In Web terms they range in skill from wiki authors and bloggers to expert multimedia artists. *Programmers* extend existing content development and deployment technologies (i.e., they might write Photoshop plugins or Javascript for a web page to communicate with a server). Wizards are those programmers that develop sophisticated software frameworks and architectures.

Brie seeks to transform large numbers of consumers into authors by reducing the barriers to creation of compelling and functional content. It also allows unaided authors to implement sophisticated functional 3D objects that would otherwise require the aid of a Programmer.

Although Brie's design includes features that make life easier for Programmers and Wizards, the focus of this effort is to increase the number and effectiveness of Authors. Features of interest to Programmers and Wizards are discussed in a companion paper [6].

4. Utilizing the Edge of the Network

Reed's Law states that the value of a Group-Forming Network (or GFN: a network that enables groups of more than 2 people to interact) grows exponentially with the number of participants [3]. The mathematics of Reed's law provides the fundamental motivation for Croquet's existence. This broad framework encompasses many dimensions that affect the value provided by GFNs. In order to maximize this value, Brie's design seeks to empower users at the edge of the network so that they may create their own content, and their own ways of interacting with that content.

4.1. Decentralized Content Creation

A simple explanation for the scalability of peer-to-peer network architectures is that they distribute capability and responsibility amongst all peers, and thereby avoid bottlenecks. Looking at the current economics of virtual worlds, it is clear that the technical expertise required for creating such worlds reduces the potential for such worlds to be deep and functionally complex. Limiting factors such as the availability of technical expertise or the cost of bringing such expertise to bear on the creation of 3D spaces act to preclude generation of very large and content rich worlds. Brie eliminates such bottlenecks by making it relatively simple for anyone to create new content. Furthermore, content created by users draws directly on their expertise within their field, specialization, or interest. This stands in stark contrast to the all too common situations in which content must be created by technology experts who do not grasp the subtleties of the application domain.

4.2. Behavior as Content

In Croquet, code is simply another medium [1]. The Brie component architecture takes advantage of this fact by supporting end-user authoring of behaviors, not merely of objects and their appearances. Brie allows units of behavior to be reified as first-class objects. In this way, such objects can be made visible in the 3D space, and may be copied, inspected, and modified collaboratively directly within the space.

4.3. User Interface as Content

Brie makes no distinction between the objects in a world and the objects that comprise the UI for interacting with that world. That is, elements such as menus, information panels, interaction managers, highlights and affordances are all composed of Brie objects.

4.4. Feedback Loops

Software engineering approaches such as Agile Development [8] seek to meet the end-user's needs by reducing the time expended for each development iteration. Brie takes the next logical step: it shortens the cycle by removing the programmer in many cases.

Brie cuts out the programmer by allowing users to reuse existing content directly in much the same way that Software Mass Customization allows programmers to compose pre-existing software modules that have been made available for this purpose.

Subject to the assignment of appropriate permissions, the Brie component architecture allows anything encountered in a Croquet-based 3D world to be copied, inspected, or modified at any time. Copies can be used in the same space, or saved for later use in a completely different context.

The ability to directly modify content (at runtime!) allows users to immediately rectify faults and make improvements, without the interruption of switching to a separate development environment. Crucially, Croquet's collaborative dialogue between users remains unbroken.

As with other content, UI elements can be collaboratively modified as they are being used. We hope that this will stimulate authors to develop a great diversity of domain-specific UIs that would not otherwise exist due to the cost factors. We also anticipate that a very important feedback loop will ensue once designers begin to build UIs to help them build UIs, without the help of a programmer. As with all content, these UIs can be specific to a single user or utilized by any subgroup of users within a collaborative space.

Croquet already provides mechanisms for deliberatively publishing content so that it is available for search and reuse without visiting the original world in which it was created. Combined with the ease of customization, we hope that the tendency for authors to share what they have created will lead to the rapid proliferation of content that can be reused and repurposed.

4.5. Distributed Cognition

Distributed cognition is a cognitive theory which states that human knowledge and cognition are not confined to the individual, but are also embodied in human-created artifacts in our environment [9]. For example, consider the meaning that can be communicated by diagrams on a public whiteboard, even when the creator of those diagrams is no longer present. In a persistent online environment, the creator of an artifact can therefore participate in a GFN without being present within the environment. Stated differently, such environments become mechanisms for the publication, persistence, and even distribution of works.

UIs are particularly interesting artifacts to consider in the light of this notion of distributed cognition. Due to the large amount of design required to create good UIs, we consider them to embody a concentrated amount of cognition. Since they are the mediators of all user interactions with Croquet spaces (and their contents) they have the effect of exerting a great influence on how users perceive, conceptualize, and interact with the space, its contents, and with other users.

Together, Reed's Law and the theory of Distributed Cognition suggest that even a modest increase in the proportion of users who are also authors can result in a substantial increase in Croquet's overall utility.

5. Construction and Deconstruction

Our design of Brie involves two complementary approaches to facilitating the authoring of 3D Croquet spaces. Construction is the first approach and involves building artifacts from scratch. Deconstruction is the second and involves taking existing artifacts apart.

Figure 1. A meta-medium application consisting of a 3D molecular model and 2D Flash and a Web page, all annotated with text voice, video and 3D portals.

5.1. Construction

User interfaces can be varied, and everything is subject to permission. However, in general, every Brie object can be positioned anywhere within its container or within the enclosing virtual space. The default user

interface allows this to be done by direct manipulation (e.g., by dragging with a mouse, pen, or other pointing device).

One obvious use of this is that applications may be developed simply by assembling pre-existing elements into a desirable arrangement. Unlike the Web, in which pages must generally be coded in an HTML editor locally and then explicitly published to a separately maintained server, Croquet worlds may be instantly created by any user (or group of users) at any time. The ability to collaboratively author such spaces happens automatically and transparently (although it is subject to creator-specified control). Likewise, a Brie-enabled Croquet space can be created by anyone and then populated with content simply by arranging and modifying copies of pre-existing content. Such content can be presentations of text, time-based linear media, and even entire applications. Content may include not just disembodied data that needs a separate player for viewing, but the whole application and user interface. Any application available on a networked computer can be presented this way. Therefore, the Croquet spaces may be considered as a meta-medium for the use of other media.

Additionally, Brie objects are designed to be small components capable of being assembled by simply dragging them to a *container object*. In this way, complex shapes and architectures may be assembled from primitive geometry. However, since such components can also possess functionality in the form of behaviors, Brie objects make it possible for authors to develop working applications through the recombination of found objects.

5.2. Deconstruction

As a way of promoting the use of existing code, Brie objects may be used directly or also deconstructed into their constituent objects. This is done through direct manipulation of a compound Brie object. All behaviors associated with that compound object or its constituent components are accessible through an information panel that is available for every Brie object. Once identified, components can be further inspected, copied, modified, and repurposed for use elsewhere.

5.3. Persistence and Copying

Even consumers (casual users of content) can easily copy Brie objects (subject to permission). Such copy behavior is typically available through any object's authoring menu. Since all objects can be decomposed into constituent geometry, materials, sounds, and behaviors, users can easily copy whole applications or any interesting parts of them.

In the next version of Croquet (the Hedgehog release), objects are capable of persisting in the state in which they are left. Users can come and go from a Croquet-based 3D world, and always get the most recent replication of that world. For Brie, this means that objects constructed interactively are immediately useable by themselves and by others, and will remain directly useable upon revisiting the world they reside in. These characteristics allow an incremental, iterative, collaborative, and very "live" application development process.

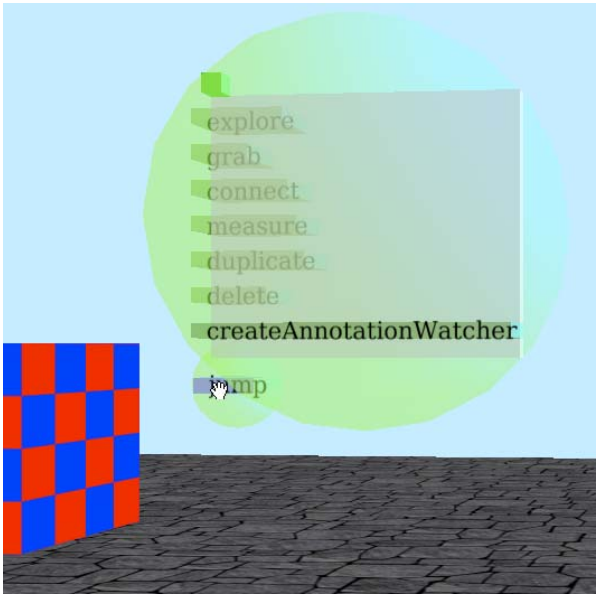

Figure 2. Pulling the "jump" behavior out of a menu.

5.4. Consequences: A Culture of Sharing

With the use of Brie, programming and creating applications becomes a treasure hunt. When you encounter an application (or functionality) that does something you like, you can copy (with permission) any combination of the look or feel or content. Easy sharing combined with deconstruction/reconstruction makes it easy to put things together that may not have been envisioned by the authors of the original subcomponent/behavior. This capitalizes on existing trends of a creative "sampling" culture [10]. By contrast, the Web makes it easy to sample text – with or without permission. Existing Peer-to-Peer systems make it easy to sample other media. The Internet at large makes it easy to trade in application "skins" that are especially created for this purpose. By allowing the permission-controlled deconstruction of geometry, user

interfaces, media, and application behavior, Brie allows sampling and adaptation at many more levels.

6. Brie's Default User Interface

Brie is a language architecture and framework designed to support any user interface that an author can envision. As part of our research and development efforts, we have created a very basic default user interface (henceforth referred to as the Brie UI) as a proof-of-concept, and as a way to motivate design decisions during Brie's development. As we describe some interesting aspects of the default Brie UI, we ask the reader to bear in mind that it is only one very basic example of a UI that can be built in Brie.

The default Brie UI assumes a 3-button mouse. These are assigned as follows:

- left: standard object interaction
- middle: movement and navigation of the user's avatar
- right: special "authoring" interaction

Currently, the default Brie UI uses the standard Croquet interface for movement and navigation. We describe in more detail what we mean by "standard interaction" and "authoring", and then reflect on how we arrived at this division of responsibilities.

6.1. Left Mouse Button: Standard Interaction

The basic UI model familiar to most Consumers is that of the desktop and icon-based representations of files and applications. The default Brie UI leverages this familiarity by providing a 3D analogue to the 2D desktop in most modern operating systems. That is, all Brie objects -- whether 3D or not, and whether "content" or UI -- act similarly to 3D desktop icons in terms of highlighting and selection. Everything can be accomplished with a one-button mouse, or the left button of a multi-button mouse. Users can select by clicking, and deselect by clicking on "nothing." The usual menu and button selection is also done with a simple click. Users can toggle selections of a multiple selection by holding down a modifier key while selecting. As in some desktop UIs, one can drill down through containers with successive left-clicks. There are visual and aural responses to each gesture (such as mouse-over and selection). Everything can be interactively modified to act, look, and sound as desired.

To avoid distracting other users in the same space, the highlight is visible only to the user whose mouse is hovering over the object. The object can define any highlight behavior, including additional mouse-sensitive affordances, descriptive text (e.g., in a heads-up display area), and sound.

When an object does not implement some behavior on clicking (such as a button does), it can (by default) be moved by dragging it around. Dropping it onto another object embeds it within that object, so that it moves with the container.

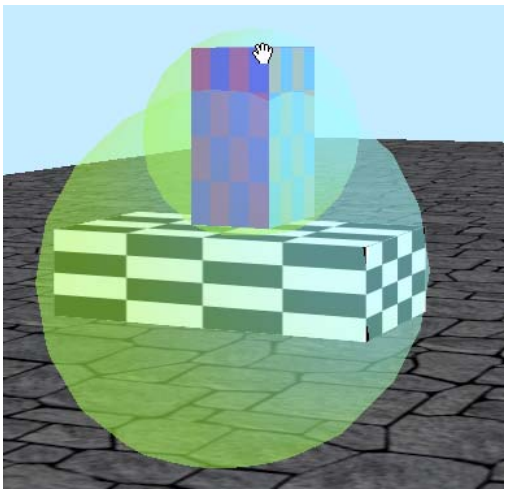

Figure 3. Creating a compound object by dragging one object over another to become a child of the static object. The spherical highlight indicates that the contained object is a valid drop target.

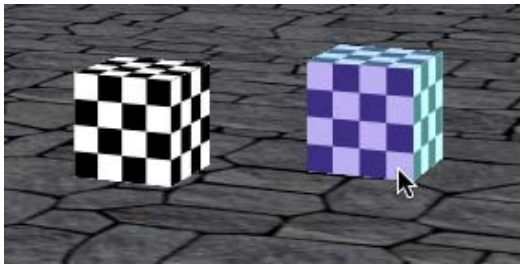

Figure 4. An object and an object with mouseover.

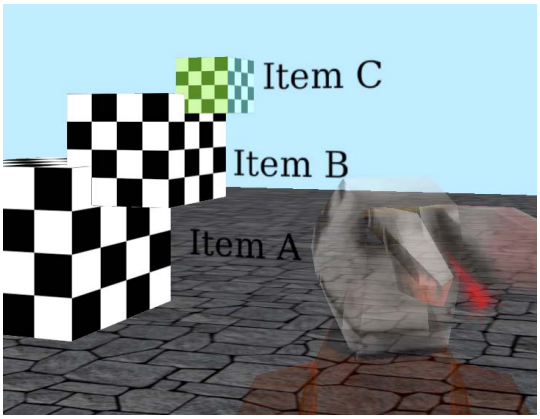

Figure 5. Drilling down through a compound object. The user has selected node "C."

6.2. Right Mouse Button: Authoring

When an object such as a button or menu item does exhibit some behavior when clicked on, we need an "author mode" that still allows the object to be selected, moved, and otherwise manipulated, else we could never again change the object! The user interface can be put into "author mode" by selecting an object with the right mouse button. While in this mode, any normal left-click behavior is suppressed, so that, for example, the object can be moved by dragging it with the left mouse button pressed. Furthermore, a second right click will bring up a context menu. One of the options on the menu will bring up an information panel from which all the behaviors of the object can be examined and manipulated.

Figure 6. An object with menu after two right clicks (or a double right click).

6.3. Why we did it this way

We believe that the subtle distinction between using and authoring a space is quite universal, and therefore chose our allocation of mouse buttons to reflect it. The user quickly internalizes the association between the interaction mode and the corresponding button.

The author/use distinction is also a natural boundary for granting permission. For instance, in an educational space, students might only be authorized to act as users (interacting with a space designed to meet pedagogical objectives), whereas the teacher has complete liberty to change anything in the space.

For these reasons, and for compatibility with other Brie UIs, we advocate that others who write their own Brie UIs to follow this convention.

6.4. Examples

6.4.1. Manipulating a Menu As An Ordinary Object

Menus are usually displayed for the initiating user only, and are oriented and positioned relative to the camera. (See 7.1. "Filters and Interactors," below.) However, they also have a button on them that "pushes" them out into the normal collaborative 3D space, where they can be seen, manipulated, and shared by everyone. In this case, "2D Fixed To Camera" implies "private to me", while "3D in Scene" implies "shared by everyone" [11]. After manipulation as an ordinary 3D object, a user can press the button again to "pull" the menu back into their own user interface, so that the menu will appear (privately, in the fixed-to-camera overlay space) when the user next calls for a menu for the original object.

The following examples all do their manipulation while the menu is "pushed out". The customization can then be shared by any user in the space who then "pulls" the modified menu back in to their own UI.

6.4.2. Customizing a Menu

The objects in a menu can be manipulated just like any other. However, since the labels act as active buttons when selected with the left mouse button, we must use the right mouse button to select the menu, as described in Section 6.2. We then drill down to the menu item we want by clicking on it. Since the menu item is now selected (for authoring), we can then drag it to the desired location.

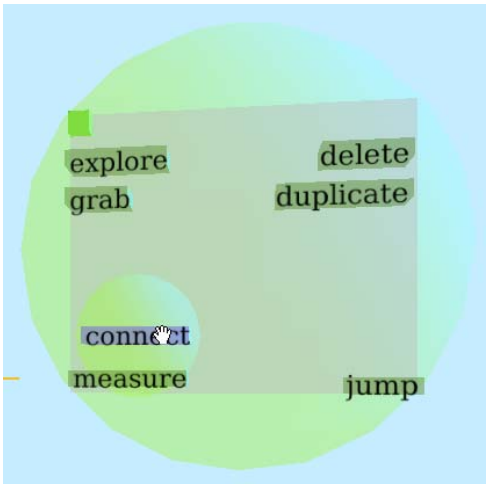

Figure 7. Rearranging the items in a menu.

6.4.3. Moving the "Jump" Menu Item From One Object's Menu to Another.

We can pull a menu item off the menu entirely. Until it is attached to some other object, it will still apply to the original object. For example, the separated "jump" menu item, when clicked, will still cause the object to jump.

The standard technique of dropping an object onto another can be used to add the "jump" item to a menu for another object; clicking on the menu item now causes the new menu's object to jump. It is easy to see how this could be used to create a custom control panel made of menu items and other controls for various objects; this is the "deconstruction" described in Section 5.2).

Of course, if we did not want to break the original menus, we could duplicate their menu items rather than tearing them out of the menu.

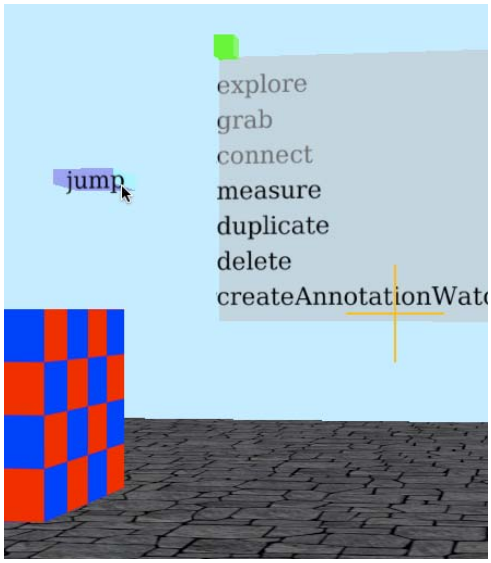

Figure 8. The "jump" menu item pulled off the menu for an object. The stand-alone jump item is not merely text, but still acts as a button which, when pressed, makes the original object jump.

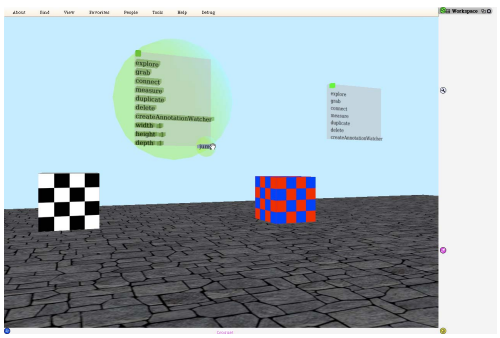

Figure 9. The "jump" menu item being added to another object's menu.

7. Related Work

The direct manipulation principles were pioneered by Self [12]. These ideas are also present with construction/deconstruction of user interfaces in Morphic [13]. Brie applies these ideas more pervasively, and combines them with Croquet's P2P collaboration.

In game design, Sims creator Will Wright is developing a new game based on the sharing of content created by other users on the Internet [14]. But Spore content creation is merely highly parameterized. Brie allows deconstruction followed by reuse in arbitrary reconstruction.

7.1. Filters and Interactors

A general mechanism is used to filter all content, including user interface elements, so as not to clutter the scene for all users. This mechanism is also available for per-user interfaces. However, the state of the user interface is reified in a shareable object (a view screen) so that users can "take off their viewing glasses" and share them with others. An example of the use of these is given in [15].

7.2. Ownership and Control

Although we are inspired by the vision of a medium where "authoring is always on" [16], we recognize that there are valid reasons to support the notion of ownership of worlds and objects within them, and to allow owners to exercise various forms of control. For example, a teacher might not want students to rearrange a carefully designed learning space.

This issue is being approached from two directions. At the infrastructure level, the Croquet architects have recognized the inability of standard access control techniques to handle an unstructured global network of interconnected Croquet spaces, and are incorporating results from the field of capability-based security [17]. At the social level, we will explore expectations of users and faculty as Brie applications are built at the University of Wisconsin.

8. Conclusion

The opportunities to enhance human performance with computers are too diverse to be entirely developed by a small community of specialists. Our approach is to empower end users to actively participate in the

creation, assembly, and architecture of the applications they use. Embracing the largest possible user group is the key to developing the rich, dynamic, and planetary scale community we hope to build. We are therefore evolving the technologies to support not only application use but also application creation and evolution by non-technical users. Rather than making incremental improvements in how existing applications are used or even developed, we are seeking a sea-change in which it is possible for ordinary users to do things that they could not previously do at all. We have looked at UI refinement by end-users as one example. We believe this work will also stimulate much thinking, problem solving, and code refinement across the larger development community.

9. Acknowledgement

This work was undertaken under the direction of the Division of Information Technology of the University of Wisconsin-Madison. It was funded in part under a contract with the National Institute of Information and Communications Technology (http://www.nict.go.jp)

10. References

[1] D. A. Smith, A. Kay, A. Raab, and D. P. Reed, "Croquet – A Collaboration System Architecture." *Proceedings of the First Conference on Creating, Connecting, and Collaborating through Computing (C5 '03)*, IEEE Computer Society Press, 2003.

[2] J. Lombardi and M. McCahill, "Enabling Social Dimensions of Learning Through a Persistent, Unified, Massively Multi-User, and Self-Organizing Virtual Environment", *Proceedings of the Second Conference on Creating, Connecting, and Collaborating through Computing (C5 '04)*, IEEE Computer Society Press, 2004.

[3] D. P. Reed, "That Sneaky Exponential – Beyond Metcalfe's Law to the Power of Community Building", http://www.reed.com/Papers/GFN/reedslaw.html

[4] M. McCahill and J. Lombardi, "Design for an Extensible Croquet-Based Framework to Deliver a Persistent, Unified, Massively Multi-User, and Self-Organizing Virtual Environment", *Proceedings of the Second Conference on Creating, Connecting, and Collaborating through Computing (C5 '04)*, IEEE Computer Society Press, 2004.

[5] D. A. Smith et. al., "Filters and Tasks in Croquet", *Proceedings of the Third Conference on Creating, Connecting, and Collaborating through Computing (C5 '05)*, IEEE Computer Society Press, 2005.

[6] H. Stearns, J. Gargus, M. Schuetze, and J. Lombardi, "Simplified Distributed Authoring Via Component-based Object Construction and Deconstruction in Collaborative Croquet Spaces", *Proceedings of the Fourth Conference on Creating, Connecting, and Collaborating through Computing (C5 '06)*, IEEE Computer Society Press, 2006.

[7] Squeak, http://squeak.org

[8] Agile Alliance, http://www.agilealliance.org

[9] E. Hutchins, *Cognition in the Wild*, MIT Press, Cambridge, MA. 1995.

[10] L. Lessig. "The People Own Ideas!" *Technology Review, June.* Cambridge, MA, 2005.

[11] J. Lombardi and M. McCahill, "User Interfaces for Self and Others in Croquet Learning Spaces", *Proceedings of the Third Conference on Creating, Connecting, and Collaborating through Computing (C5 '05)*, IEEE Computer Society Press, 2005.

[12] R. B. Smith and D. Ungar. "Programming as an Experience, The Inspiration for Self" in J. Noble, A Taivalsaari & I. Moore, eds, *Prototype-Based Programming: Concepts, Languages, Applications*. Springer-Verlag, 1997.

[13] "An Introduction to Morphic: The Squeak User Interface Framework." in M. Guzdial, K. Rose, *Squeak: Open Personal Computing and Multimedia*. Prentice-Hall, Englewood Cliffs, NJ, 2001.

[14] Spore, http://spore.ea.com

[15] R. Kadobayashi, J. Lombardi, M. McCahill, H. Stearns, K. Tanaka, and A. Kay. "3D Model Annotation from Multiple Viewpoints for Croquet", *Proceedings of the Fourth Conference on Creating, Connecting, and Collaborating through Computing (C5 '06)*, IEEE Computer Society Press, 2006.

[16] A. Kay, "Background on How Children Learn", http://squeakland.org/school/HTML/essays/how_child_learn.html

[17] M. S. Miller and J. Shapiro, "Paradigm Regained: Abstraction Mechanisms for Access Control", *8th Asian Computing Science Conference (ASIAN03)*, 2003.