Using Agents as a Currency of Exchange between End-Users

Alexander Repenning, Martin Rausch, Jonathan Phillips, Andri Ioannidou
Center for LifeLong Learning & Design
University of Colorado, Boulder CO 80309-0430
{ralex, mrausch, phillipj, andri}@cs.colorado.edu
http://www.cs.colorado.edu/~l3d/systems/agentsheets/

Abstract: The Behavior Exchange is an AgentSheets-based forum employing the web for the collaborative creation of SimCity™-like interactive simulations. While initially the Behavior Exchange was geared to empower geographically distant users to build and exchange simulation pieces it turned out to be just as useful in facilitating collaboration between kids working in the same class room. Example applications of this technology include the design of sustainable eco systems in life science curricula letting kids create their own animal species and exchanging them via the Behavior Exchange.

Introduction

The web is predominately broadcasting medium of information. However, from the point of view of education it is too passive, as the majority of its users are primarily consumers of information. Becoming an active designer or producer on the web is not trivial. Authoring is usually delegated to HTML literate web-masters that serve as high-tech scribes, in the same way as illiterate people delegated writing to scribes during the Middle Ages. Newer generations of WYSIWYG HTML editors simplify the task of creating static web pages but provide little support for more dynamic content such as simulation and animation. Java allows programmers to include programs into web pages, but to computer end-users this technology remains out of reach.

For end-users to harness the power of the web and be encouraged for more active and productive participation, the image of the web as a broadcast medium should be expanded to include end-user mechanisms in support of collaborative design, construction and learning. This can be done by supporting:

- ***Bi-directional use of the web***: Enable and motivate consumers of information to become producers of resources on the web.
- ***Richness of content***: Make rich and expressive computational artifacts, such as simulation components and behaving agents, utilizing the web as a forum of exchange.

From the educators' perspective, enabling and motivating consumers to become producers of resources on the web and therefore utilizing it as a bi-directional medium for collaborative learning, introduces a more social aspect to the classroom setting, which has traditionally been instructionist [Papert 1993]. With constructionism [Papert 1980], however, an alternative approach to learning has been offered. Constructionism proposes that people best construct new knowledge when they are engaged in personally meaningful tasks. In this spirit, the web needs to become an open medium for active participation in order to become an effective educational medium.

Web content need not be restricted to textual information, pictures and movies, but can include rich and expressive computational artifacts. For instance, the web can become a medium through which users exchange individual components - that we call agents - of SimCity™-like interactive simulations. Such a medium allows users to add their own agents to a community repository. Users need to be able to find relevant agents in the repository and integrate them with their own simulations.

This paper describes the framework that enables and motivates end-users to become producers of computational web artifacts, namely AgentSheets, presents the Behavior Exchange, a web-based forum supporting the social aspects of this process, and describes our experiences with students producing interactive simulations by sharing agents over the web.

End-User Programmable Agents

To empower web users in becoming active producers rather than passive consumers, we have employed the end-user programmable design environment of AgentSheets [Repenning and Sumner 1995], available for Macintoshes. We use the notion of agents as shareable computational units that can be created and exchanged over the web.

AgentSheets is used to create SimCity™-like interactive simulations, domain-oriented visual programming environments, games and infobots. Combined with its novel programming approach, Visual AgenTalk (VAT) [Repenning and Ambach 1996], AgentSheets is a versatile computational medium for a variety of computer end-users ranging from K-12 students to professionals. Over 1000 applications have been created with AgentSheets in areas including education, art, computer science, biology, medicine, and engineering.

Users of AgentSheets create agents for their simulations, by defining both their look and behavior. An agent is given its look by use of simple drawing tools, such as a bitmap editor. The behavior of an agent is defined using VAT, a graphical rule-based language that was specifically designed for computer end-users with no programming experience.

In AgentSheets, agents are autonomous processes able to perceive and act in their environment. An extendible set of sensors allows agents to perceive mouse clicks, sound input, voice commands, keyboard input and even read and parse web pages. Agents can act by moving in a simulation world, changing their appearance, play sounds, speak and open URLs. The mapping between perception and action is determined by the agent's end-user programmable behavior, in the form of IF-THEN rules. A collection of these agents that interact with each other make up an AgentSheets simulation.

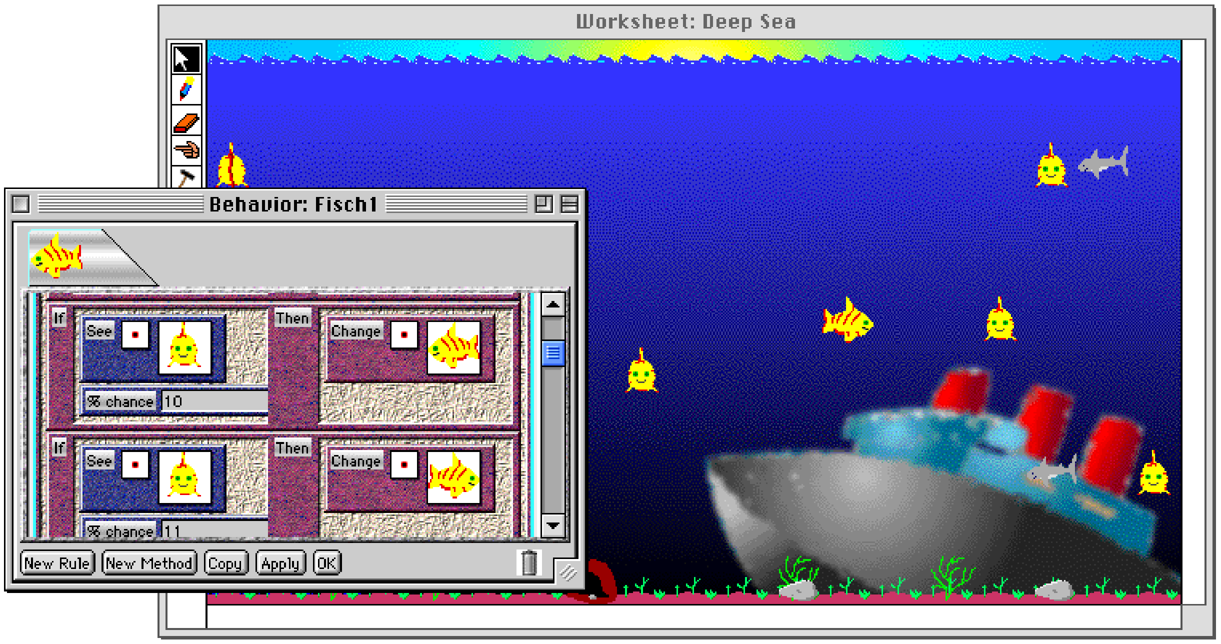


Figure 1: The AgentSheets Environment; Left: Behavior Editor, Right: A fish simulation world..

Figure 1 shows the AgentSheets Fish Tank simulation. The worksheet (window in the back) contains agents representing different types of fish, rocks, divers, air bubbles and water. The Behavior editor (window in the front) contains a Visual AgenTalk program for the yellow fish. The first rule in the window declares that, with a 10% chance, the fish will turn left. Once a simulation is finished, using the Ristretto™ Java generator built into AgentSheets [Repenning and Ioannidou 1997], the complete project can be instantly turned into a Java applet and published on the web.

The goal of this work is to support more directly the requirements of programming in social settings where programming is no longer considered a solitaire activity. In contrast to the original before-and-after graphical rules based language [Bell and Lewis 1993, Furnas 1991, Kirsch 1964] used in AgentSheets [Repenning 1994] and KidSim/Cocoa [Smith, et al. 1994] the new Visual AgenTalk language was designed with web-based collaboration in mind [Repenning and Ambach 1996]. This kind of support requires that computational artifacts can be easily shared [MacLean, et al. 1990], comprehended and modified.

Agents as Social Currency

The Behavior Exchange [http://www.agentsheets.com/behavior-exchange.html], an evolving web-based information space, allows users to efficiently exchange agents that have been created in AgentSheets through the web. It contains two kinds of information. Informal information is not interpreted by the computer. The look of an agent, e.g., a scuba diver, textual descriptions concerning what the agent does, who created it, why and how it is used belong into the informal information category. Formal information is interpreted by the computer. All the rules that determine the behavior of an agent are considered formal information.

The combination of informal and formal information turns these agents into a social currency of exchange. Users produce agents and share them. Other users pick them up and modify them to better fit into their own environment. There can be a variety of reasons to trade agents. Agents can be complete solutions to problems. For instance, somebody may set up an agent to retrieve weather information from web pages and summarize it in useful ways and place this agent in the Behavior Exchange. Other users may find this functionality useful and wish to download it and use it. Still others may find this agent is a partial solution to their problem and wish to not only download it but to modify it as well and tailor it to fit their needs.

Exchanging Agents: A Scenario

For the Behavior Exchange, the web serves as a medium through which the formal and informal information of agents is exchanged. Collaborative activities include building, sharing, locating, acquiring, comprehending, and modifying agent. These activities are explained in the following sections in form of a scenario.

Building

Bob, is working on a fish tank simulation [Fig. 1]. Bob wants to have a scuba diver that can breathe in this fish tank. He creates a scuba diver and a bubble agent for which he defines two icons using the AgentSheets icon editor. Bob wants the scuba diver to create bubbles when he hits the space bar on the keyboard. Using the behavior editor Bob defines this simple behavior in a single rule [Fig. 2].

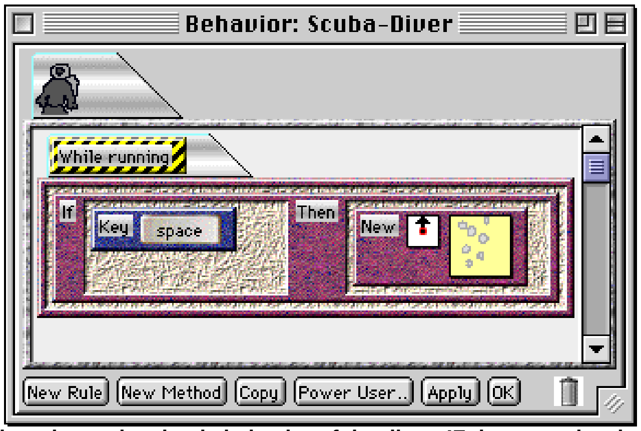


Figure 2. The Behavior Editor shows the simple behavior of the diver: IF the space key is pressed (conditions are on the left) THEN the diver creates a new air bubble (actions are on the right) above itself.

AgentSheets includes a rich set of condition and action commands that can be dragged into IF-THEN rules structures. Considerably more complex behaviors can be built than the one shown including procedural abstractions, numerical operations, colorization [Repenning and Ambach 1996], but a simple example is used to focus on the exchange of agents. Command parameters, e.g., *which key* to test for or *what location* to use to put a new agent, are all defined using direct manipulation. For instance, Bob specifies the key that he wants to use to trigger the bubbles by clicking at the Key condition and pressing the key of his choice on the keyboard.

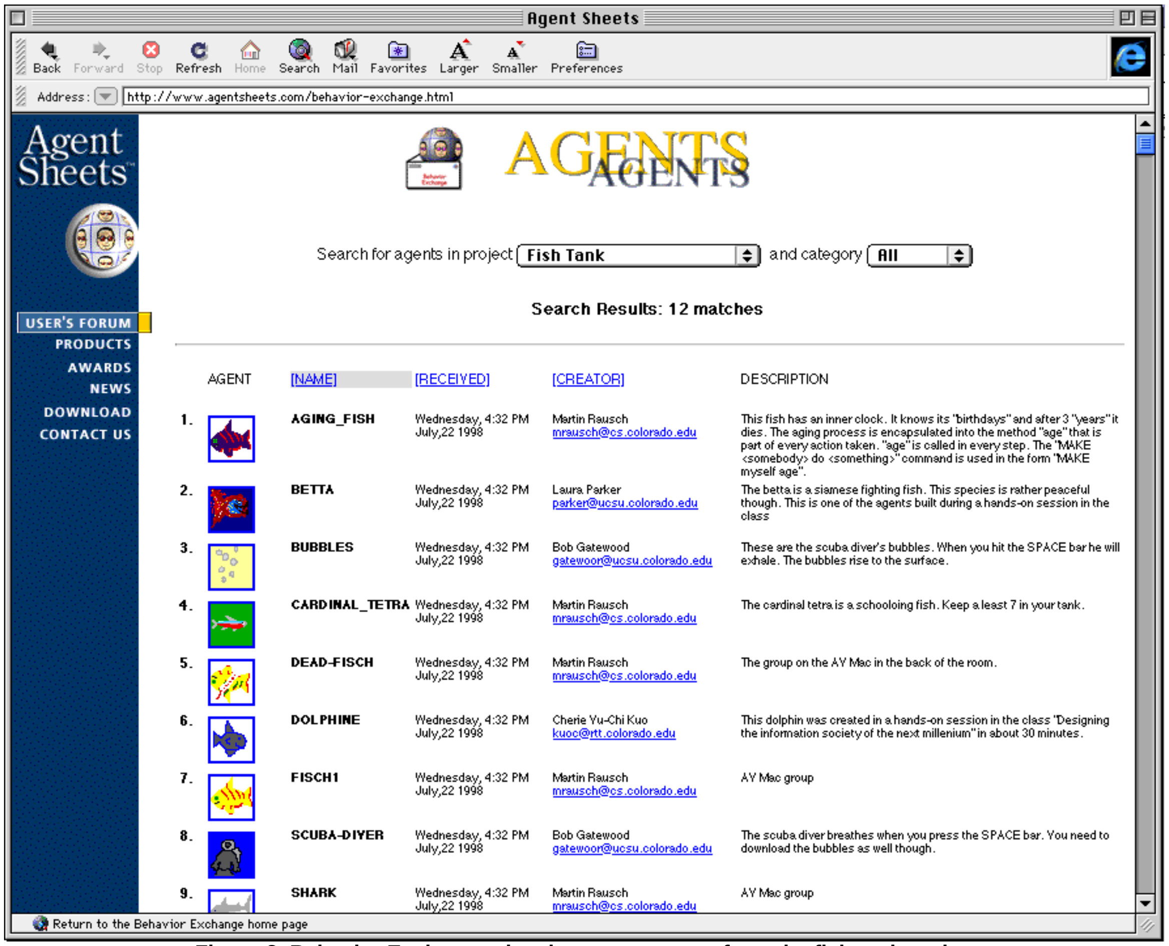
Sharing

Bob wants to share his scuba diver and bubble agents by submitting them to the Behavior Exchange. Bob simply selects the agents to be shared and annotates each agent with some text explaining what the agents do and how they can be used. This text will be visible to other Behavior Exchange users and will help them to find and use relevant agents.

Location

Beth visits the Behavior Exchange with the goal in mind to find some new agents for her fish tank (the fish tank comes with the AgentSheets distribution). AgentSheets takes Beth directly to the Behavior Exchange via the web browser [Fig. 3].

Beth has a number of ways to locate interesting agents. She can browse the Behavior Exchange by selecting projects or by selecting categories. Agents can be sorted according to name, date, and creator. Beth, selects the Fish Tank project and browses the agent thumb nails presented to her. She is interested in the scuba diver. Clicking the scuba diver reveals the comments made by Bob about the scuba diver. In this case the comment are quite important since they point out that Beth should also take the bubble agent if she is interested in the scuba diver.



Search Results: 12 matches

	AGENT	[NAME]	[RECEIVED]	[CREATOR]	DESCRIPTION
1.		AGING_FISH	Wednesday, 4:32 PM July,22 1998	Martin Rausch mrausch@cs.colorado.edu	This fish has an inner clock. It knows its "birthdays" and after 3 "years" it dies. The aging process is encapsulated into the method "age" that is part of every action taken. "age" is called in every step. The "MAKE <somebody> do <something>" command is used in the form "MAKE myself age".
2.		BETTA	Wednesday, 4:32 PM July,22 1998	Laura Parker parker@ucsu.colorado.edu	The betta is a siamese fighting fish. This species is rather peaceful though. This is one of the agents built during a hands-on session in the class
3.		BUBBLES	Wednesday, 4:32 PM July,22 1998	Bob Gatewood gatewoor@ucsu.colorado.edu	These are the scuba diver's bubbles. When you hit the SPACE bar he will exhale. The bubbles rise to the surface.
4.		CARDINAL_TETRA	Wednesday, 4:32 PM July,22 1998	Martin Rausch mrausch@cs.colorado.edu	The cardinal tetra is a schooloing fish. Keep a least 7 in your tank.
5.		DEAD-FISCH	Wednesday, 4:32 PM July,22 1998	Martin Rausch mrausch@cs.colorado.edu	The group on the AY Mac in the back of the room.
6.		DOLPHINE	Wednesday, 4:32 PM July,22 1998	Cherie Yu-Chi Kuo kuoc@rtt.colorado.edu	This dolphin was created in a hands-on session in the class "Designing the information society of the next millenium" in about 30 minutes.
7.		FISCH1	Wednesday, 4:32 PM July,22 1998	Martin Rausch mrausch@cs.colorado.edu	AY Mac group
8.		SCUBA-DIYER	Wednesday, 4:32 PM July,22 1998	Bob Gatewood gatewoor@ucsu.colorado.edu	The scuba diver breathes when you press the SPACE bar. You need to download the bubbles as well though.
9.		SHARK	Wednesday, 4:32 PM July,22 1998	Martin Rausch mrausch@cs.colorado.edu	AY Mac group

Figure 3. Behavior Exchange showing some agents from the fish tank project.

Acquisition

Beth acquires the scuba diver by simply clicking it in the web page. The scuba diver gets downloaded and automatically added to the AgentSheets agent gallery allowing Beth to put any number of scuba divers into her simulation.

Comprehension

Beth gets full access to the newly acquired agents. She inspects the behavior of the scuba diver agent by opening up its behavior editor. She can test any part of the diver's behavior within her own AgentSheets environment. She can test rules or even individual conditions and actions by simply dragging and dropping them onto a diver, or any other agent, in her worksheet. For instance, when dragging the New action,

, out of the diver's behavior [Fig. 2] onto a the sea weed, , in the fish tank, a bubble will be created above the sea weed. If the simulation is running the bubble will float onto the surface of the tank and pop.

We believe that the ability to tinker with language pieces without the need to first combine them into complete programs is crucial to the usability of programming approaches for non-programmers. This approach, that we call Tactile Programming [Repenning and Ambach 1996] due to the graspability of language components, supports exploration and significantly simplifies debugging which can be hard in rule based languages [Gilmore, et al. 1995, Rader, et al. 1997].

Modify Agents

After understanding the behavior of the scuba diver, Beth modifies the current divers behavior. She want to have a more autonomous diver creating bubbles without having to press keys and she wants to count the bubbles. She replaces the Key condition with a OnceEvery-n-Seconds condition and adds a Visual AgentTalk formula incrementing the value of a new attribute called "bubbles" [Fig. 4].

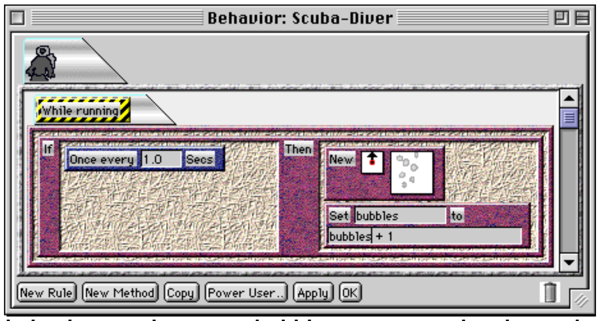


Figure 4. Modified diver behavior creating a new bubble every second and counting the number of bubbles.

The combination of tinkering support with ease of change can overcome some of the fundamental problems encountered in graphical rewrite rule-based systems such as the earliest version of AgentSheets and it's derivative Cocoa. For instance to change a graphical rewrite rule in Cocoa users need to recreate the example situation in which the rule was created. This can be tedious to the point where users resist modification or prefer to create new rules masking old ones instead of modifying the old rule [Rader, et al. 1997].

Experiences and Conclusions

We have gathered experiences in using the Behavior Exchange in elementary, middle and high schools. At a middle school we initially explored design issues and affordances of a web-based exchange forum in a minimally structured school environment. A large group of kids at the Centennial middle school in Boulder meet every Friday at the computer club. The goal of the computer club is to learn about computers in a playful atmosphere. Kids produce drawings, create newspapers and browse the web. One policy of the computer club is that kids are not allowed to play computer games but they are allowed to build their own games. A group of 8 kids designed a game about the Boulder walking mall. The game featured a tourist as the main character. The tourist had some money to buy gifts in a variety of shops or to support artistic presentations such as jugglers. The objective of the game was that the tourist successfully walks from one side of the mall to the other buying things but at the same time avoiding obstacles such a muggers and other game characters. In the process of building the game the Behavior Exchange allowed subgroups to develop their own characters and share them with the entire group.

In an elementary school the Behavior Exchange was used to facilitate collaborative eco system design. Under the direction of Clayton Lewis this project explored the suitability of simulation building as a means to science education. This activity was part of the regular life science curriculum. Instead of just studying food webs the traditional way the kids had to design their own species (animal or plant) using AgentSheets. The species created by individual kids were combined into complete eco systems. Supported by worksheets the kids explored the sustainability of their food webs in action. Quickly they were able to determine if they had a sustainable environment or if their species needed further adjustments.

In a high school the Behavior Exchange was used as reuse mechanisms. In a course titled "Exposing the Human Grotesque" a social science teacher and her students built an interactive city including a character that can be walked through the city to find stories of people and buildings. The Behavior Exchange served the role as a repository of city design components. Using existing agents such as roads, trains, and cars the kids, mostly with psychology and literature backgrounds, got their city off the ground very quickly.

While initially the Behavior Exchange was geared to empower geographically distant users to build and exchange simulation pieces it turned out to be just as useful in facilitating collaboration between kids working in the same class room possibly right next to each other. Often when thinking about web-based collaboration support one tends to think about communication between spatially distant members of a virtual community. We found that networked school computers, while satisfactorily connected to the outside world, provide little if any support for local collaborations. Partly this is because of safety concerns found in many schools. Freely exchanged floppies or application programs downloaded from the web can be the sources of devastating viruses. The Behavior Exchange supports collaboration in local communities without the need to disable virus checking or to add extra infrastructure.

The work presented in this paper is only a first step towards the reconceptualization of the kind of information distributed through the web. Our experiences with students using AgentSheets and the Behavior Exchange suggests that using agents as social currency enables and motivates end-users to become more active participants in the evolution of the web. This shift transforms the web from an information broadcast medium to a medium for collaborative design and learning. Such a medium supports discussions and collaborative learning through the sharing of ideas and artifacts. AgentSheets can be downloaded at http://www.agentsheets.com

Acknowledgments

We thank Scott Dixon, teacher at Centennial Middle School, and Eric Scharff from the Center of LifeLong Learning & Design for their support. This work is funded by NSF DMI-9761360, RED925-3425, and DARPA CDA-940860.

References

[Bell and Lewis 1993] Bell, B. and Lewis, C., 1993. ChemTrains: A Language for Creating Behaving Pictures. In1993 IEEE Workshop on Visual Languages (Bergen, Norway). IEEE Computer Society Press, 188-195.

[Furnas 1991] Furnas, G. W., 1991. New Graphical Reasoning Models for Understanding Graphical Interfaces. InProceedings CHI'91 (New Orleans, LA). ACM Press, 71-78.

[Gilmore 1995] Gilmore, D., Pheasey, K., Underwood, J. and Underwood, G., 1995. Learning graphical programming: An evaluation of KidSim. InProceedings of the Fifth IFIP Conference on Human-Computer Interaction (London).

[Kirsch 1964] Kirsch, R., A. (1964). Computer Interpretation of English and Text and Picture Patterns. IEEE Transactions on Electronic Computers 13, 4, 363-376.

[MacLean et al. 1990] MacLean, A., Carter, K., Lövstrand, L. and Moran, T., 1990. User-Tailorable Systems: Pressing the Issues with Buttons. InProceedings CHI'90 (Seattle, WA.). ACM Press, 175-182.

[Papert 1980] Papert, S. (1980) Mindstorms: Children, Computers and Powerful Ideas. New York: Basic Books.

[Papert 1993] Papert, S. 1993. Instructionism versus Constructism. In The Children's Machine (pp. 137-156). BasicBooks.

[Rader et al. 1997] Rader, C., Brand, C. and Lewis, C., 1997. Degrees of Comprehension: Children's Understanding of a Visual Programming Environment. InProceedings of the 1997 Conference of Human Factors in Computing Systems (Atlanta, GA). ACM Press, 351-358.

[Repenning 1994] Repenning, A. (1994). Programming Substrates to Create Interactive Learning Environments. *Journal of Interactive Learning Environments, Special Issue on End-User Environments 4*, 1, 45-74.

[Repenning and Ambach 1996] Repenning, A. and Ambach, J., 1996. Tactile Programming: A Unified Manipulation Paradigm Supporting Program Comprehension, Composition and Sharing. InProceedings of the 1996 IEEE Symposium of Visual Languages (Boulder, CO). Computer Society, 102-109.

[Repenning and Ioannidou 1997] Repenning, A. and Ioannidou, A., 1997. Behavior Processors: Layers between End-Users and Java Virtual Machines. InProceedings of the 1997 IEEE Symposium of Visual Languages (Capri, Italy). Computer Society, 402-409.

[Repenning and Sumner 1995] Repenning, A. and Sumner, T. (1995). Agentsheets: A Medium for Creating Domain-Oriented Visual Languages. IEEE Computer 28, 3, 17-25.

[Smith et al. 1994] Smith, D. C., Cypher, A. and Spohrer, J. (1994). KidSim: Programming Agents Without a Programming Language. Communications of the ACM 37, 7, 54-68.