Bluegrass: Lessons for Deploying Virtual Worlds Within the Enterprise

Steven Rohall, Li-Te Cheng, and John Patterson
Collaborative User Experience Group
IBM T.J. Watson Research Center, Cambridge, Massachusetts
{steven_rohall, li-te_cheng, john_patterson}@us.ibm.com

ABSTRACT

This paper describes the challenges of deploying current, state-of-the-art virtual worlds within the typical enterprise. We start by describing Bluegrass, our prototype desktop virtual world embedded in a collaborative software development environment. Then we discuss some of the issues we encountered while trying to deploy Bluegrass to test its effectiveness in improving the distributed software development process. While some of the issues are specific to Bluegrass, we conclude that a number of the issues are common to any similar sort of virtual world. We call for the development of new virtual world technologies, lightweight and focusing more on people rather than places, to more effectively test their usefulness within the enterprise.

Author Keywords

Collaborative virtual environments, collaborative software development, computer-supported cooperative work.

ACM Classification Keywords

H5.3. [Information interfaces and Presentation]: Group and Organization Interfaces – Computer-supported cooperative work.

INTRODUCTION

Distributed software development teams face numerous collaboration challenges and many tools have been developed to address the problems [2]. For co-located teams, a transparent, shared physical workspace environment fosters awareness, encourages *ad-hoc* meetings, and socialization [15]. Virtual worlds build on this metaphor of arranging people in a spatial, collaborative environment [1]. In this paper, we detail our design thoughts and experiences with Bluegrass, a prototype that explores how a desktop virtual world can support distributed software teams. Bluegrass is embedded within Rational Jazz Team Concert [8], a collaborative software development environment, and complements Jazz's capabilities with facilities for visualization, meeting support, and socializing to support such teams.

There has been a lot of hype concerning virtual worlds over the past several years. One of the key reasons for prototyping Bluegrass was to be able to deploy it within several software development teams and then study its effect. Would virtual world technology enable a distributed team to perform more like a co-located team? Unfortunately, we were not able to study Bluegrass in this manner. In this paper, we first describe Bluegrass and then we proceed to discuss the issues which prevented it from being deployed and studied. We call for the development of new, lightweight virtual world technologies.

THE BLUEGRASS SYSTEM

The World is a Visualization

The Bluegrass world is a visualization that provides awareness and insight. Visualizations inhabited by users can take advantage of spatial and collaborative features [10]. For example, an entire visual landscape can be based on software being developed [3]. Bluegrass is shaped by social information about software teams instead of software data.

As seen in Figure 1a, a developer entering the Bluegrass world appears at a plot of land allocated for the team. A large tree with the team's name at the center of the plot acts as a landmark for the team's space in the world. Around the tree are personal gazebos and meeting areas (Figure 1e).

The placement of teams is automatically computed from team hierarchy information available from the Jazz server. Top-level teams are placed in a circle at the very bottom of a basin. Sub-teams are placed on plateaus ringing the basin. Teams further away from the top-level teams are placed on higher plateaus. In this open landscape, users can easily spot activity emanating from other team areas – for example, in Figure 1b, streams of white dots can be seen at the Work Item team.

Information in Bluegrass appears as bubbles that float and fade away. Avatars in Bluegrass automatically emit a bubble at a regular interval to indicate their presence (Figure 1c). Moving around creates a trail and gatherings can be seen from afar. While these presence trails are very similar to [6], in Bluegrass they can provide detailed status from external sources, and can be directly manipulated (e.g. grabbed and placed on the ground like in Figure 1d). Figure 1b shows that the dots emanating from the Work Item team area are bubbles from a gazebo (left) indicating the owner is out visiting another team, an RSS feed (middle) showing a stream of work updates, and a user (right) saying that he is working on a particular task.

Meetings Instantiate Ideas, Generate Work

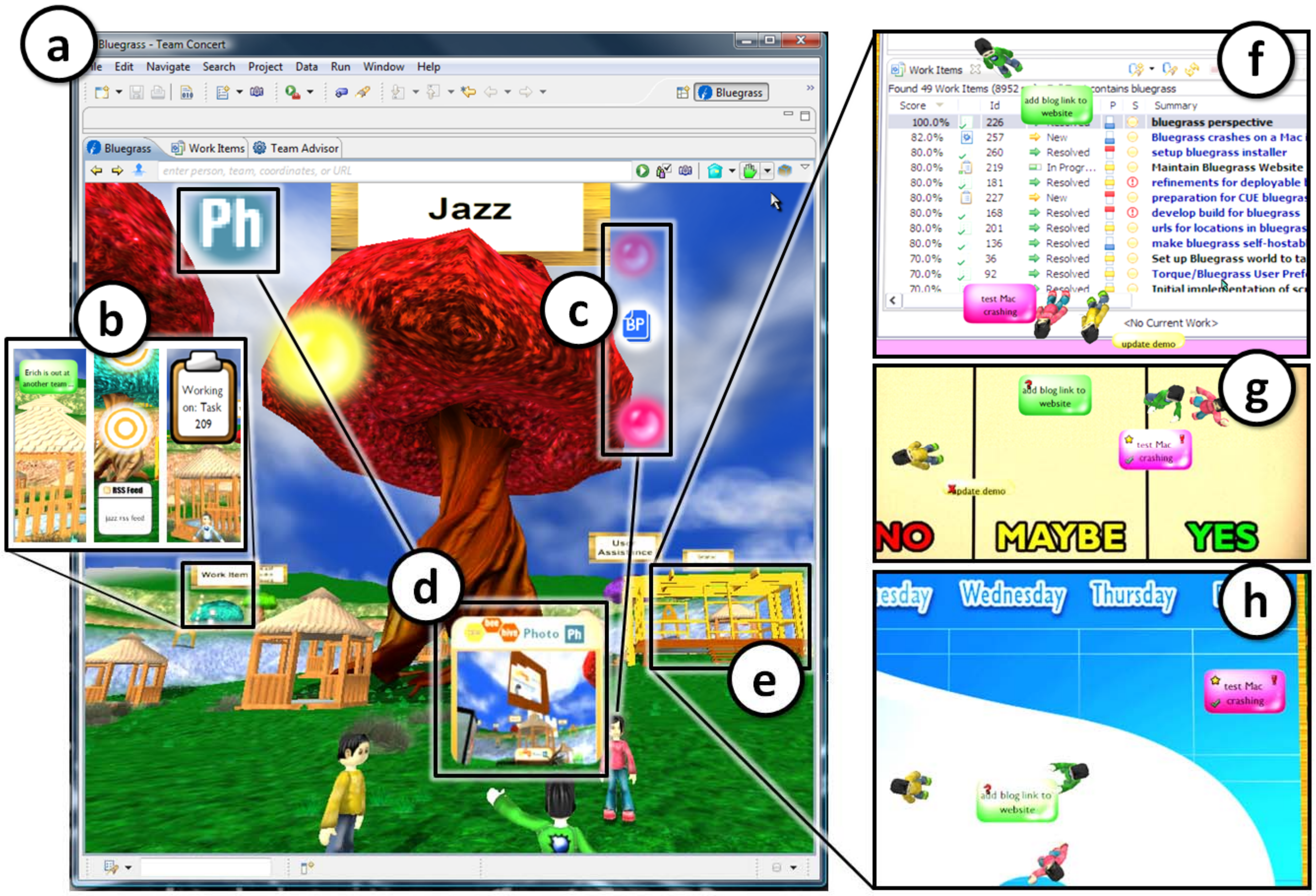
Programming is a primary task to bring users into a software development environment. Some virtual worlds immerse users in a world, and then embed tools to dynamically modify and present code in the world itself [1][12][13]. However, most software teams already have conventional software development environments not integrated with any virtual world. These environments can be augmented with plugins for contextualized collaboration [7]. Bluegrass is such a collaboration plugin for a specific environment. To complement the existing collaborative aspects of its hosting environment, Bluegrass focuses on meetings, not code development, as the instrumental reason to bring software teams together in the virtual world.

The Bluegrass meeting area resembles a patio. Users entering will have their personal "over-the-shoulder" point of view changed to a shared top-down perspective (Figure 1f-h). The floor can project different surfaces (e.g. a calendar, a shared application). Anyone can create and manipulate objects on top of the surface to build layouts similar to a whiteboard. Participants can "vote with their feet" by moving to highlighted places on the projected surface similar to [6]. The meeting space can be saved and loaded for later review. The tasks manipulated in Figures 1f-h resemble sticky notes on a whiteboard. These objects are created by chatting. Chatting in Bluegrass is similar to virtual worlds like IMVU [9] where typing creates a floating bubble. In Bluegrass however, bubbles can be persisted and moved by clicking and dragging. These bubbles can be annotated like sticky notes with different icons.

Anyone can persist and annotate another's chat bubbles, thus some people might generate ideas, while others might pick out and place the interesting ones, and others can place annotations. This can help engage users with different roles in brainstorming and triaging meeting activities. A right-click menu option allows any participant to transform a sticky note in the virtual world into a work item that can be tracked in Jazz. The text is used to prepopulate a work item form supplied by Jazz. This helps move abstract concepts and conversation back to the software development context.

Socialization through Social Networking

By virtually co-locating them in a shared virtual space, distributed team members can mingle and build up a sense of community. For example, home offices and meeting areas can be represented as a shared virtual office space to foster



Work Items

Found 49 Work Items (8952 … contains bluegrass

Score		Id		P	S	Summary
100.0%	✓	226				bluegrass perspective
82.0%		257	New			Bluegrass crashes on a Mac
80.0%	✓	260	Resolved			setup bluegrass installer
80.0%		219	In Progr...			Maintain Bluegrass Website
80.0%	✓	181	Resolved		!	refinements for deployable
80.0%		227	New			preparation for CUE bluegras
80.0%	✓	168	Resolved		!	develop build for bluegrass
80.0%	✓	201	Resolved			urls for locations in bluegras
80.0%	✓	136	Resolved			make bluegrass self-hostab
70.0%	✓	36	Resolved			Set up Bluegrass world to ta
70.0%	✓	92	Resolved			Torque/Bluegrass User Pref
70.0%			Resolved			Initial implementation of sc

Figure 1: Bluegrass – (a) virtual world inside Rational Jazz Team Concert; (b) objects and avatars can emit bubbles viewable from afar; (c) different bubbles can be emitted, (d) such as a screenshot from a social networking profile that can be persisted; (e) team meeting areas provide top-down spaces used for: (f) application sharing and persisting chat bubbles as tasks to discuss, (g) voting and marking up bubbles with annotations like checkmarks, (h) assigning calendar tasks tied back to Jazz's work item system

a campus community [11]. This can be extended by incorporating social networking services. Virtual worlds such as IMVU combine a virtual world for synchronous mingling with a profile page similar to Facebook.com for asynchronous interactions, such as blogging and contacts [9]. Bluegrass taps into Beehive, an internal corporate social networking service similar to Facebook [5], to encourage casual conversation.

In the background, Bluegrass examines the user's profile in Beehive, and portrays small bits of profile information as bubbles. The bubbles are shown using icons that represent contacts, photos, and lists (Figure 1c). Clicking will create a persisted, moveable card with more information (Figure 1d). Thus, these items serve as icebreakers to start conversations among nearby users. These icebreakers can also be turned into group games that can help in team-building [4]. In Bluegrass, persisted information from Beehive can be transformed into jigsaw puzzles via a right-click menu. As users play together, the personalized content helps users get to know one another better.

Observations

From the portrayal of a colorful landscape, to playful cartoony avatars, to the various ways to engage and manipulate objects in the environment, a number of people remarked about how Bluegrass could foster a sense of fun. A corporate customer remarked that virtual worlds like Bluegrass can "bring back the fun to software development". While "fun" does not necessarily translate to greater productivity, it may lead to community-building for a distributed software team of strangers.

The meeting support and social networking aspects of Bluegrass garnered the most attention. A number of people remarked that the top-down meeting space did not truly exploit the 3-D nature of the virtual world. On the other hand, people were uncomfortable with presentations keystoned in 3-D and preferred a flat projection. Integrated audio support like in [11][13] and fine-grained avatar expressiveness as discussed in [14] were desired.

Integrating with social networking profiles felt natural to most. The Bluegrass team's experiments with posting Bluegrass snapshots back to Beehive (Figure 1d) even led to serendipitous conversations via Beehive, and later, meetings and demonstrations with new interested parties.

A virtual world seeks to immerse its users, but if these users are already immersed in work, there may be a clash along many dimensions. This clash is especially evident for users unfamiliar with virtual worlds. One clash was around visual metaphors. Some found mapping a fanciful landscape of trees and bubbles unintuitive, and visually overwhelming. Another clash centered on workflow. Although embedding Bluegrass inside Jazz relieves the need to context-switch between applications, the way users navigated, searched, programmed, etc. in Jazz was too different from doing similar activities in the virtual world. Throwing new concepts into the development environment such as setting up and manipulating an avatar also felt like extra work. A final clash was around work culture and geography. While some liked the look of Bluegrass, others felt it too childish and unprofessional.

And, for those who were familiar with and predisposed to use virtual worlds, the lack of avatar customizability and expressiveness were issues. Bluegrass avatars were statue-like and users did not feel connected to them. Finally, it was pointed out that virtual worlds would not help with teams with significant time zone differences – asynchronous tools like social networking sites would be better suited.

DEPLOYMENT CHALLENGES

The goal in prototyping Bluegrass was to produce a system complete enough and robust enough to deploy with several distributed software development teams so that we could study its usage and effectiveness. Unfortunately, we were not able to deploy Bluegrass in this manner. One issue was the need for subject teams to be using the Rational Jazz system. At the time we were developing Bluegrass, Jazz was still being developed and had not been widely adopted within our company. But, there were deployment issues even among those teams that were using Jazz. These were due to the fact that Bluegrass was built like a conventional thick virtual world application and used a commercial game engine integrated with the Jazz environment.

Heavy Client Requirements

Bluegrass dwarfed the already-high requirements of the Jazz environment. Running Bluegrass consumed one full CPU of a dual-core machine, even when Bluegrass was obscured and not being actively used. This was due to the use of the commercial game engine for prototyping. Response time in a multi-user, first-person shooter game needs to be very fast. As a result, games are continually transmitting events on the network as well as continually re-rendering the environment to give players an accurate portrayal of the game. In an enterprise application like software development, this level of performance is not necessary and, in fact, was seen as a problem when it was difficult to use other applications on the same computer.

Indeed, since games are typically the only application running on a computer while they are being played, Bluegrass had serious problems cooperating with other applications. Just having the screensaver kick in or having the user connect to an external projector when Bluegrass was running would result in the computer crashing. This is unacceptable in a corporate environment where users leaver their computers or hook them up to projectors for meetings all the time.

Bluegrass was also a huge download—over 1Gbyte compressed, larger than the Jazz client installation itself. Users need to have plenty of extra disk space. The hard drives of developers with lots of tools and code on their machines are already quite full. Finally, there were video driver issues—users need to be sure they had the right video driver version (which may not have been the most recent) in order to run Bluegrass.

Network Issues

As mentioned above, multi-user games require a large network pipe to ensure that players have a fair and fast game experience. In a corporate environment, where users are often working remotely from customer sites or from home, this isn't always true. Bluegrass would crash if packets were lost. In addition, remote users would often need to use a VPN client to access the enterprise network. The VPN client encrypted all traffic, adding another 30% CPU overhead when Bluegrass was being used. For some users, this meant that their laptops would overheat and shut down due to thermal sensors. Indeed, this wasn't just a problem with Bluegrass; users reported similar problems with other virtual world applications such as Second Life.

All-in-all, we were placing too many technical demands on our small pool of potential users. Any study of virtual worlds we performed on such users would be so overwhelmed by the fact that we ruined their ability to do their "day jobs" that we decided the only prudent action would be to reconsider how we hoped to test the usefulness of virtual worlds within the corporate enterprise.

FUTURE DIRECTIONS

The biggest challenge we faced was deployment of our prototype in a way in which we could study its usage. Existing multi-user virtual worlds built with commercial game engines or on top of systems like Second Life all suffer from the same problems: heavy requirements on CPUs, disks, video cards, and networks. They just don't work well on corporate computers and networks where people must use other applications such as email, word processors, and web browsers.

The root of these issues is the blending of a heavyweight virtual world experience into the context of a large collaborative software development environment. The virtual world experience carries an enormous set of preconceptions about what a 3-D experience should be like (e.g. a virtual landscape, immersion, avatars, etc) which are new to some users, and clash with the experience of the familiar software development environment. These clashes not only manifest in technology gaps (e.g. video drivers, system resources), but also mental models and visual metaphors. While Bluegrass tried to be a contextual collaboration tool in the Jazz environment, it is not unobtrusive enough. Given that environments such as Jazz already have an abstract notion of a "place" (e.g. workspace, team area, etc), we plan in the future to focus strictly on "people" – namely, an avatar service to elaborate the notion of people, and social interactions like those discussed in [14]. Instead of game engine client technology, we plan to use Flash-based web technology for ease of deployment.

In conclusion, we believe the challenge of blending virtual worlds into a collaborative context is not about creating the highest fidelity immersive experience. Rather, we need to explore what are the minimum, essential elements of a virtual world that can make a difference in collaborative work. Such elements may offer some of the richness of a virtual world, such as an avatar service. These not only can complement software development, but collaboration in general. In future work, we plan to study how much of a difference this light-weight, minimalist approach can make.

REFERENCES

1. Bartle, R. *Designing Virtual Worlds*. New Riders Publishing, Indianapolis, IN, USA, 2004.
2. Booch, G. and Brown, A.W. Collaborative development environments. In *Advances in Computers*, Vol. 59, Academic Press (2003).
3. Dossick, S.E. and Kaiser, G.E. CHIME: A metadata-based distributed software development environment. In *SIGSOFT Softw. Eng. Notes*, ACM Press (Nov. 1999), 464-475.
4. Ellis, J., Kellogg, W., Luther, K., and Bessiere, K. Games for virtual team building. In *Proc. DIS 2008*, ACM Press (2008).
5. Geyer, W., Dugan, C., DiMicco, J., Millen, D.R., Brownholtz, B., Muller, M. Use and resuse of shared lists as a social content type. In *Proc. CHI 2008*, ACM Press (2008), 1545-1554.
6. Harry, D. and Donath, J. Information spaces – building meeting rooms in virtual environments. *Ext. Abstracts CHI 2008*, ACM Press (2008), 3741-3746.
7. Hupfer, S., Cheng, L., Ross, S., Patterson, J. Introducing collaboration into an application development environment. In *Proc. CSCW 2004*, ACM Press (2004), 21-24.
8. IBM Rational Jazz. http://jazz.net
9. IMVU, http://imvu.com
10. Knight, C. and Munro, M. Should users inhabit visualizations? In *Proc. WETICE 2000*, IEEE Computer Society Press (2000), 43-50.
11. MPK20: Sun's Virtual Workplace. http://research.sun.com/projects/mc/mpk20.html
12. Phelps, A.M., Bierre, K.J. and Parks, D.M. MUPPETS: multi-user programming pedagogy for enhancing traditional study. In *Proc. CITC4 2003*, ACM Press (2003), 100-105.
13. Second Life. http://secondlife.com
14. Slater, M. and Steed, A. Meeting people virtually: experiments in shared virtual environments. In *The Social Life of Avatars*, Springer-Verlag (2002), 146-171.
15. Teasley, S., Covi, L., Krishnan, M.S. and Olson, J.S. How does radical collocation help a team succeed? In *Proc. CSCW 2000*, ACM Press (2000), 339-346.