A Conversation with Steve Hagan

PHOTOGRAPHY BY RICHARD CLARK

Oracle Corporation, which bills itself as the world's largest enterprise software company, with \$10 billion in revenues, some 40,000 employees, and operations in 60 countries, has ample opportunity to put distributed development to the test. Among those on the front lines of Oracle's distributed effort is Steve Hagan, the engineering vice president of the Server Technologies division, based at Oracle's New England Development Center in Nashua, New Hampshire, located clear across the country from Oracle's Redwood Shores, California, headquarters.

At Oracle,

DISTRIBUTED DEVELOPMENT

IS A WAY OF LIFE.

Hagan is in charge of several portions of the Oracle 9i database server development, including disaster recovery, utilities, migration, multimedia, and extensible databases, as well as portions of the iAS mid-tier application server. There are more than 400 people at the Nashua location.

A 30-year veteran of the software industry, Hagan has been at Oracle for nine years. Before joining Oracle, he

spent seven years at Digital Equipment Corporation as the senior engineering manager for database technologies. Prior to that he spent eight years in the design and development of databases for engineering applications. He received an M.A. in computer science from the University of Southern California.

Hagan is interviewed by Anthony I. (Tony) Wasserman, principal of San Francisco-based Software Methods and Tools, where he helps software startups with technical and business issues. Until recently, Wasserman was director of Mobile Middleware Labs for Hewlett-Packard, where he managed a distributed development team working on software infrastructure for mobile Web services. Prior to that he was vice president of Bluestone Software, responsible for its West Coast Labs.

Wasserman was founder and CEO of Interactive Development Environments Inc. (IDE), developer of the innovative Software through Pictures multi-user modeling environment. Prior to starting IDE, Wasserman was a University of California professor. He earned a Ph.D. in computer science from the University of Wisconsin, Madison, and an A.B. in mathematics and physics from U.C., Berkeley. Wasserman is a Fellow of both the ACM and IEEE and was co-founder of ACM's SIGSOFT.

TONY WASSERMAN I go back a long way with issues surrounding the various approaches to software development and have seen everything from highly structured processes to completely informal styles. Perhaps we can start with you describing something about your team's process.

STEVE HAGAN We're a remote site here in the New Hampshire area with Oracle headquarters in the San Francisco area. So part of our job has been working on the development environment and building software in the distributed environment.

One of the areas that we've focused on in my group at Oracle is the software development process itself. It's a maxim that the higher quality of the software you put out, the lower your maintenance costs—and it's easier then to keep more of your developers working on building new functionality as opposed to maintenance work. You also decrease your time to market. Your new functionality per year—per calendar year and per person year—is maximized by doing a good job in the software process.

TW Can you tell us just a little bit about the kind of product that you develop?

SH We're part of the Server Technologies division at Oracle. Within Server Technologies, we build the database itself, which is what Oracle is best known for; the mid-tier software, Internet Application Server; and the management software, Oracle Enterprise Manager; as well as the new Collaboration Suite software.

TW What is the size of your team and what roles are covered by the various members of the team—the people you oversee directly?

SH At this site, in Nashua, New Hampshire, there are about 400 people. Depending on how you define us, the team is responsible from the inception of the software—coming up with a concept for a new idea, through the functional specifications, design, code, implementation, tests, beta, documentation, and post-shipment support.

TW Since you're not based at Oracle corporate headquarters in Redwood Shores, in what ways does that help or hinder your work?

SH As in any large company, there's hallway talk and gossip that's really not relevant to getting your job done. Because we're remote, we're actually shielded from a lot of that, and we can do more heads-down work or blue-sky new project ideas and development. We don't get into some of the background noise that you get in a larger location.

One of the benefits of being here in New Hampshire is that our average turnover rounds to less than 1 percent, compared with the 15 to 20 percent that was common in Silicon Valley during the bubble economy. Voluntary turnover rounds to zero—there's always some housecleaning you have to do. We are able to keep an extremely stable technical talent base here, and that's one of the reasons why the company has these remote sites. We are far less subject to the siren songs of startups than they are in Silicon Valley.

TW Well, the sirens have definitely quieted [in the San Francisco Bay area].

SH These days, the sirens are at the large companies—many very senior developers have returned to Oracle. Also, Oracle hires through a college program involving 14 specific universities. Here in New Hampshire, we're within 400 miles of 10 of them.

One hindrance of being a remote site is that you're not sitting down the hall from your boss and peers. Sudden

meetings come up that actually could affect the direction of your projects. That's obviously the downside whenever you're in a remote site.

■
TW Which raises the question of how you do your team meetings with your boss, three time zones and 2,500 miles away.

SH We have a videoconference connection to headquar-

ters. Every two weeks, there's a team meeting—one with my boss's staff, and then an extended staff meeting. Once a week there is a developers' meeting, with a broader audience that is tracking the detailed status of the projects under way: the database, followed by the application server, followed by the Collaboration Suite, followed by the Enterprise Manager. There's at least one meeting of that size a week, and by this I mean probably 40 people, because these are such large projects.

Using the videoconference is much better than being on the phone because you can see who is speaking, and then, since you know a lot of these people, you have an idea of where they're coming from in the conversation. Plus we do have teleconferences. It's fair to say that managers are in these telecons every two days or so, as well as in contact by individual personal phone calls and e-mail.

TW How often do you find yourself in Redwood Shores?

SH In general, it's at least once a month, and usually I'll average between 10 and 18 trips a year; it just varies with the year.

But that's at the management level. I've got engineers who simply don't like to travel, and so they rarely go to California. It's a function of being willing to travel and the extent to which you do a cross-group project such that you want to meet face to face.

Part of the question on a distributed team is how you bring new ideas forward. As people come up with new ideas here at the site—which is a good thing—at some point, you must bring them forward to people on the West Coast, depending on the amount of funding that's required. Then you need to go out there.

We were an acquisition, sold to Oracle by Digital Equipment Corporation nine years ago. The executive decision made at that time is that you must have the communication and movement of people back and forth to keep projects going smoothly. So we factor the travel expenses into all the budgets.

TW Do you use instant messaging, or do you do live demos with products like [Glance Networks'] Glance or [SightSpeed Inc.'s] SightSpeed?

SH We have a product called iMeeting that's part of Oracle's Collaboration Suite, which we use internally for most of our meetings these days when a speaker needs to do a presentation. One person will be talking and then you can watch the presentation material live on your desktop, or your laptop, anywhere on the planet. The speaker controls the ordering of the slides/material and can edit them live, while you are watching and listening. We don't really use IM. Perhaps the sales force does, but within development, we haven't found a lot of value to it, over what we already do.

TW I was looking at the new technology that people use in some startups; they use instant messaging, and they

use blogs to record design decisions or progress reports, much in the way that we used to use notebooks. Has that caught on with your team?

SH Again, not so much, because of the iMeeting and e-mail trails that were already successfully in place here. We have all of our specs and design material online in what we call Oracle Files Online, again part of the Collaboration Suite, and that gives you shared access to specs and to comments within them.

TW So that subsumes things like Groove?

SH Right.

TW In terms of sharing files, obviously for developers and to a lesser extent for marketing types, there's a need to do version control and configuration management. Do you do anything special because it's distributed, or do you just attach file systems wherever they are in the world and use those for your backups?

SH We should distinguish between the source-code management system and a document management system. We put quite a bit of work on the source-code management system. The reason is you have to have developers potentially sharing the same code module for possibly the same project, possibly overlapping projects that intersect within that code. So you need a significant amount of version control, and yet sharing capability, so that you can work on the code, know you have the right version, do your builds, your tests, etc.

We used to be a [IBM Rational] ClearCase shop, and it didn't scale to the thousands of people working on the same code and the number of different sites that we needed. So we have a homegrown distributed replicated source-management system that combines essential repository, which is based on the Oracle database, but then also implements a multi-version file system, replicated out to all the sites, using a much lighter-weight communication protocol and also lighter-weight storage. We actually have done quite a bit of evolving as we learned what communication bandwidth we need to support the level of development we're doing with this number of developers and locations.

TW Are you helped by the nine-hour time difference between you and India?

SH It does affect us quite a bit from a technical perspective on when you perform integration and builds, for example. They will be available at one time of day in one part of the world and another time of day in the other.

You have the time lag between, say, headquarters and the remote sites. That has to be built into our processes.

TW There are any number of companies that are sending technical work offshore, and there certainly are any number of reports about communication difficulties and even variances on the English language, as well as some of the more personal kinds of issues that arise. Now that you've got people working in many different time zones—to the extent that at any time of day, there's somebody out there doing something on one of your projects—how many hours a day do you check your e-mail?

SH A fair number. I'll probably do my last check around 10 at night, East Coast time. A lot of people tend to check very early in the morning. Unfortunately e-mail has become too much of an umbilical cord for all of us these days.

TW A lot of organizations that do distributed development try to have major subsystems or components or functions associated with a location. In your case, it sounds like there's less of that and that you put a gang of people together to work on a particular subsystem or function—and they are where they are.

SH We actually have a remote person in the U.K. running a group of about half a dozen people here in New Hampshire. I think she probably spends four hours a day on the phone, but the results have been superb. She has built an entirely new test system from scratch and gotten very good results on it.

I'll give you examples of some domains that we've consciously chosen to build up in this location. For example, the spatial database work is all done at this location, as is the multimedia work.

I have a dozen Ph.D.s specifically in the spatial database domain because it's so specialized, and you want to have a clustered talent base. Spatial database is actual

geometry—XYZ coordinates. The U.S. Census Bureau, for example, uses this to manage a detailed map of the U.S., and all the associated demographics, down to very fine levels of granularity. Also, think of a road network, which is simply the management of a graph within the database, as are utility lines, power lines, etc. We now model this within a relational database.

The area of high availability and disaster recovery is also done here, as are the utilities. We're picking sections of the product and project that are being done here, such that we have a core competency. So we have a core competency here of high availability, disaster recovery, spatial, InterMedia, object database capabilities, and extensible databases.

InterMedia in standard terms is image, audio, video. What's getting hot today is medical imaging—replacing film with digital images, which are easier to save for auditing purposes. Also a law just passed that allows a digital image of a check to serve as a legal document, as opposed to shipping a paper check around. Putting all the images in the database, along with abilities to search it, etc., are done in this group.

So we built these core competencies at the site,

which then makes it easier to attract people, because when they come in to interview, it doesn't sound so generic. They know which area they are interested in.

TW When we were chatting before we started the interview, you were describing some of the differences in methodologies of development for commercial or government release, and some of the work you've done there. Do you think there's anything from that experience that's worth sharing with readers?

SH When you are doing software in a commercial environment, where there's going to be a next release—and actually if you're successful there will be many, many next releases—it actually is more tenable, I think, for

distributed development. You have to get very clear interfaces and clear responsibilities, but you can do it because you'll never finish all the functionality—it keeps growing. So you can stage it from this release to the next release, and which site is doing what, and it gradually evolves.

I worked on government software for probably 10 years, and the prime issue there is that it's (mostly) a one-time shot. Say you build a giant system for the Air Force or the Navy or the Army. Basically they've gone out on contract and written this several-thousand-page spec, and you build it and you ship one version and it's done. A few minor upgrades. And then everybody disappears. Your top people have already moved onto a different project, so their knowledge isn't around.

In commercial software, the knowledge in people's heads is vital, and the knowledge that's in different locations becomes important. That's where the tenure of the people matters, going back to what I was saying about turnover. You want to have the critical mass of knowledge in people's heads at each site, and have them around for the next version. That's why the distributed development approach works well on the commercial side.

On the government side, it's as if you've contracted out to a distributed site and then it disappears. Should you have a critical problem, they are gone, and you'll never track them down again.

I'm not faulting government, since some good systems have come from this approach. It's just the way its business is run. It's a onetime development deal.

TW Are there methods of architecting software or ways to develop software that work particularly well or don't work particularly well in a distributed environment—for instance, componentizing or permissions or interface assignments?

SH Because you do get the extra cost imposed by communication in a distributed environment, it really exposes areas that aren't clear or that are excessively intertwined. So you have the accepted best practices of development—

abstraction and clean interfaces and hierarchy and all that—and if you get them wrong, you'll know it because you are going to end up spending a lot of time on the phone trying to figure it out and clarify things.

We've been doing distributed development here for about nine years, and it took us quite a while to learn the right way to do it. I can give you a concrete example. When we first started, we had a guy in headquarters whose job was to handle communication between us and headquarters—kind of an ombudsman for us. I think that position lasted about six months before we figured out that what we needed were direct ties of people to people

instead of having a person to filter it all through. No one person has that amount of bandwidth.

A cluster of distributed software development sites in Asia.

TW Right, because that creates a bottleneck in a narrow pipe?

SH Exactly. It sounded like a great idea to have one clean single interface for communication, but what we realized was that there was a better way to do it. The same is true of our source-management

system where we thought we

had it right, back with ClearCase and MultiSite, and then we had to do a lot of work to really fit it into the way we needed to develop in our culture.

Also, one of the things that is much more important when you're doing distributed development is good-quality written specifications, because people at multiple sites are reading it and relying on it.

Another thing that is really important and that has helped us in development is to get to know the counterparts for us at headquarters at all levels, all the way down into the individual engineer level. Originally, the engineers would send mail to someone at headquarters and they might not get a response. But once the two knew each other, they always got a response. Developers would go out to California and even would live out there for a couple of weeks or months in the beginning to get to know the code base and people.

I think anybody who tries to start up a distributed development effort will go through a learning process. There's a startup cost, and that's again why it makes sense

in a commercial environment—maybe not so much in a single-version, single-customer environment—because you incur that startup cost of getting a distributed effort going, and you then get to amortize that cost over the benefits that accrue with multiple versions.

TW If you look back nine years, we can see that our collaboration tools are much better. The Web is ubiquitous, e-mail is ubiquitous, instant messaging is there, and even the bandwidth for videoconferencing is so much better than it was just a few years ago. Companies that are starting distributed development now can learn from the lessons of the pioneers who got arrows in their backs. So for somebody trying to do this now, do you think, apart from the cultural differences that remain, that the technological differences make life easier?

SH I think they certainly can if they're used properly. In the right corporate culture I'm sure IM and blogs are great tools. They certainly didn't exist several years ago, and it's going to be a matter of figuring out which of those technologies work and how to use them appropriately, so that it doesn't become all about checking your blogs every day.

TW Exactly. You've mentioned some of the benefits of distributed development projects, but maybe you could sum up the benefits and the costs of doing distributed development.

SH You need to pay attention to why a remote location got started. Most tend to be from buyouts where the purchase results in a shorter time to market for a product in a new domain for the company. What you've also done is immediately increased the domain skills within the company as a whole, and with newfound synergies that occur, you can go into even more new markets. Then with the

site having done that—as long as things are going successfully and if you're in a good geographical location for recruiting new people, as we happen to be—then you can grow the group.

Some locations, like New Hampshire, are much cheaper places to live than Silicon Valley, as far as housing and other costs of living. We find it easy to attract people with families looking for a stable company and a location where they can afford a house; they can join, be here for a long time, and be part of a leading technology company. That's a major advantage of Nashua.

Again, as long as your location is stable and has little

turnover, it works out quite well because people become skilled in the technology. They know the domain, the code, etc. If we had turnover of 20 or 25 percent a year, it would not make sense to have us as a site because you would constantly be retraining people.

There's a bit of a cost when you're remote, just because of the extra communication. That's adding perhaps 5 percent to the cost on an entire project. However, because of your stability, your inherent knowledge of the domain, the development discipline, and the code, you far more than make up for that cost.

Finally, there is a very positive people situation for the company. There are a reasonable number of high-talent recruits who for various reasons simply don't want to work in Silicon Valley. If they grew up or went to school in New England or the Northeast, they often have family and/or friend connections and want to be in this geographic area. We are now able to include and welcome these people into the company. Q

We would like to acknowledge Susan Hillson, senior director of engineering at Oracle, and Roy Swonger, director of software methodologies, who contributed to the information in this interview.

Magnetized photos keep track of distributed development groups.