IBM

Moderator: Angelique Matheny
November 29, 2007
12:00 pm CT

Operator: Good afternoon, my name is (Jessica) and I will be your conference operator today. At this time I would like to welcome everyone to the Grady Booch Architecture conference call.

All lines have been placed on mute to prevent any background noise. After the speaker's remarks there will be a question and answer session. If you would like to ask a question during this time, simply press star then the number 1 on your telephone keypad. If you would like to withdraw your question, press the pound key.

Thank you, Ms. Matheny you may begin your conference.

Angelique Matheny: Hello and welcome everyone to this rational talk view telecom series Grady Booch on architecture. I'm Angelique Matheny with IBM Rational. And I'll be your host for today's call.

Our speaker Grady Booch is recognized internationally for his innovative work on software architecture, software engineering and modeling. He has been with IBM Rational as its two scientists since Rational's founding in 1981

and was named an IBM fellow in March 2003. Grady is one of the original developers of the unified modeling language and was also one of the original developers of several Rational's products. Grady has served as architect and architectural mentor for numerous complex software intensive projects around the world.

Today's call will focus on the importance of architecture within modern applications as well as the importance of reuse through patterns or any intellectual property within those architectures, especially within modern architectures such as SOA.

You will (unintelligible) towards this teleconference, we want to keep it very interactive. At the end of Grady's Booch presentation we'll open up the call for an interactive Q&A session. These calls are really for you to get your questions answered by our expert panelist, so write those questions down as you think of them and jump in.

So let's get started. I'll turn the call over to Grady. Grady?

Grady Booch: Thanks very much and thank you for your patience with us as we began this a little bit late. It seems to take a while to get people through the log in process or dial in process, so here we are.

I'm going to spend about ten minutes just chatting for a moment with regards to what I'm up to and what the current state of the practice is in the space of architecture and then we'll open it up for about 40-50 minutes of Q&A.

You know having had the opportunity to be in this discipline of software intensive systems for really my entire professional life, I have seen methods

come and go, I've seen tools come and go, I've seen languages come and go, but I'm utterly convinced of a couple of things that I'd like to declare here.

The first is that simply software development has been and it is and will remain fundamentally hard. As (Fred Brooks) has observed many times, there is a essential complexity to the things we create and that complexity is truly inescapable. It is part of the nature of the things that we build and it's part of the nature of the tools and underpinnings of what we build with.

So insofar we as software professionals try to deal with that complexity, we can never get rid of it, we can never eliminate it, the best we can hope to do is master it over time. Today, and having had the opportunity to work with lots of different companies around the world, I observed that there are some things in common among those organizations that are most successful and certainly, notably absent among those are less successful.

And there are three basic principles that I find at play. The fourth one, which very few companies get to but the more successful ones tend to do so, and I'll innumerate those, 3 plus 1 and then summarize what I see in terms of the best process practices that those organizations tend to follow.

Oh those 3 plus 1 principles include a focus upon building crisp understandable abstractions, a focus upon having a good separations of concerns and a focus upon having a clear distribution of responsibility among those abstractions within the system.

Now recognize that at any given point of time, you can't always follow those principles, but it is the case if you have a process in place that lets you always get back to them, then indeed you will be in a place where you'll be building an enduring business.

The fourth element is one, as I alluded to, that very few organizations get to but it’s the one whose - ones who are hyper productive seem to manage to get around. And that is the notion of building systems that are simple. Given that we build things that have this essential complexity, it actually requires energy to make this simple. And that involves, not just using re-factoring as a means of designing towards something but also towards designing towards simpler kinds of systems.

It's another reason why I am so obsessed these days about architecture and its value in terms of controlling the plexity. There a couple of maxims that I'd like to leave with you. One of which is the notion that every system has an architecture although, as the second maxim will observe, most architectures are accidental not intentional. What I mean by that is that, and this is the third maxim, all architecture is designed but not all design is architecture. Therefore architecture represents the set of significant design decisions that we make to accumulate over the time of our project.

And those design decisions are often ones that accumulate from the tens of thousands of small decisions that are made on a daily basis by various stake holders engaged with the system. Once we're done, we can step back and say, "Oh I have an architectural style X with these particular kinds of patterns." But it's fairly rare for most organizations to, as they build their systems, to be cognoscente of intentional about the architecture they will produce.

Most will tend to fall within a couple of doomsday scenarios. One is that they'll tend to identify architecture as simply selecting technology. If I use products X, Y, and Z, and this legacy A, B, and C, and I integrate them in a certain way ergo I'm done. And that's certainly an approach, but it's not

necessarily one that's enduring and it's also one that tends to force you to be whipped back and forth based upon the churn that goes on that technology.

The other observation I make - or another downside of how people approach this is they'll tend to vastly over engineer their architectures. They'll build things for the worst possible circumstances, the greatest unknowns, but that's never very satisfying because then you tend to end up with systems that are far too large. I'm a great believe in the agile principle as one perceives as building systems that you should do the least thing possible to advance but with an eye upon where the mountain is so that I'm not just doing a hill climbing locally and forget that I'm going to the wrong mountain.

So those four principles I find among organizations who are most successful and frankly, they can be wrapped up in the following kind of process that I see most organizations follow that are successful. And that is they tend to grow a system through the incremental and (unintelligible) release the executable and testable architectures.

There are three important elements to that observation. The first is architecture is a central artifact and that architecture is an executable one. That's the second element of it. So by architecture I mean a focus upon the risks of a system, a focus upon the major design decisions and a process that mitigates that risk over time. So that's why architecture is central to me and the governess of the system.

Executable is important because ultimately the most important artifact we produce is not models, is not documentation and it's not other things like that but it’s a raw running naked code. Now that being said, these other things are absolutely essential because they want to build the right code at the right time with the least amount of effort. It will allow me to endure over time.

The third element of this process that it's very much incremental and iterative. We know through practice that requirements are really not done until after I deliver the system. We know that we cannot even really ask op priory the right questions about the system until we build pieces. And thus we may have a vision, we may have a target, but we need to have a process in place that allows us to over time get closer and closer to the real target and make in course corrections along the way.

So these are the principles that I find organizations tend to follow. Now, that being said, let me offer a few other specifics then we'll throw it open for questions here.

You know building a system is indeed hard work, so we have to use all the things at our disposal to help us along the way. Many of those things that make it difficult for us to bridge - vision the execution, are - and many of them are social as well too. On a technical side, this is why things like UML has gotten such traction and frankly very please with the penetration of the UML in the industry. I think, you know, there are lots more people that could benefit from it, but I'm certainly please with how it's been used.

And UML in my experience has been quite adequate in helping organizations and individuals reason about and deliver interesting systems. Earlier also I mentioned the notion of patterns and, well as one example services rented architecture are just one kind of implementation mechanism for a message based pattern.

And so all the successful organization that I tend to find -- tend to encounter, do have this culture of patterns within them and they are useful because they represent a higher of level of abstraction of reuse. Reuse is more than

whacking out lines of code that I have others use, but rather one has to reach a threshold where the cost of reuse and the cost of understandability is less than the benefit one gets from using that thing. And rarely does one cross that threshold of (unintelligible) code. Once I start reusing patters, once I start reusing entire subsystems and systems, then there is economic viability therein.

And those are all kind of technical elements. On the social element, reality is software development is a team sport, and is one build the system and focuses upon executable architectures; you have to deal with collaboration from a variety of stakeholders. Some of those stakeholders are code warriors, some of those stakeholders are local, some of them are distributed and that distribution is what really causes a lot of problems because it's not just geographical distribution, its also temporal distribution.

Geographic distribution is obvious but the temporal distribution means, for me, that I'll have people build something, they'll walk away from it and there's a loss of information from design implementation and there's noise and a low bandwidth of communication if I don't capture those design decisions from the people who were temporally displaced.

Again the reason why I go back to architecture, architecture and its focus allows me to capture the essential things that transcend the temporal and geographic distribution.

So, you know, there we have it. I'd like to open it up for some questions and answers here. And I'm willing to field anything that would be on your mind.

So operator lets open it up shall we? Operators are standing by.

Operator: At this time I would like to remind everyone in order to ask a question, press star then a number 1 on your telephone keypad. We'll pause for just a moment to compile the Q&A roster.

Grady Booch: If there are no questions I'll start asking you all questions.

Operator: Again, if you would like to ask a question, press star then a number 1 on your telephone keypad.

Your first question comes from Steve Weaver.

Grady Booch: Hello Steve.

Steve Weaver: Hi Grady, and thanks for the introduction. You mentioned to collaboration on your opening discussion there, how do you see that the Jazz project from rational fitting into this collaborative enterprise that is developing software?

Grady Booch: You bet. You know, I'm not here to talk about specific products, so, you know, talking some of the rational local field folks and that's probably the right thing. But I will observe that, you know, what we're trying to do with Jazz is provide a platform for the temporal and geographic distribution.

Jazz had its inception from actually two interesting places, the team that built Eclipse are the primary architects of Jazz so you can see already that we've got some real horsepower behind the Jazz folks. Also few years ago, Rational had funded some research with the IBM Watson folks with regards to collaborative technology. We funded actually a half dozen or so projects, one of which included things like presence in the context of IM and also collaboration among buddies on actually building work products themselves.

So I'm very pleased that it's those two forces that led to the beginnings of Jazz and so we're on a very interesting track.

I would mention that we're also trying to look even beyond Jazz. There's a research project that we're funding out of Cambridge in the United States, looking at virtual worlds for the purpose of collaboration. Things like Second Life, not Second Life in particular, but Second Life like technology. I mean, personally I've given a couple of dozen lectures in virtual worlds already and IBM at large has about 50 islands in second life which we use actively for training in other kinds of things.

Predominately it's been used, not just for the U.S. but also in South America and in Asia as well. So we know there's some real value there. So a short answer is, yeah Jazz is very much in the center of where we think we're going and we're kind of watching your back with regards to whether the technology you'll might be able to leverage in the coming years.

Steve Weaver: What do you think is the - is really the biggest challenge in a collaborative environment and do you have any kind of tips to maybe help mitigate that?

Grady Booch: Sure, happy to do so. Let me point you guys to a paper that exist on my website. I've had some problems with my site because they've been switching over lines here. I need to replace my (unintelligible) admin which happens to be me.

But go to the following URL, it's www.booch.com/architecture/blog.jsp. And on the left hand side you'll see papers and if you click there, you'll see some papers and presentation about CD; Collaborative Development environments.

In particular there's a presentation I gave for Eclipse con, the very first Eclipse con about the history about developing environments. And it kind of sets where Eclipse is in that context and where we might go from beyond there. And there's also a paper that (Alan Brown) and I wrote some years ago for a journal, actually a book, series about what we think a CD could be and what it should be. So I would - so those are the reference that you might want to take a look at.

I would answer your question in the short by saying, you know it's not so much a technical problem, although there are technical elements, it's more of a social problem. It's a social problem of, "how do I build trust among people who are geographically and temporally dispersed and what can I do to connect them in those trustworthy ways." T

His is part of the so called "water cooler" problem. If I got a team in the U.S. and a team in India and China and these people never really met one another yet they're supposedly going to build something together. Well I could do all sorts of technical things to make sure that they're working on the right bits and they know about what's changed and the like, that's useful and interesting. But until so far those people have a trust and understanding of each other as a human then it's much more difficult to proceed.

That's one of things that jazzes me -- sorry for the use of the word -- about where Jazz is headed with regards to some of its collaboration facilities and also what jazzes me about the whole virtual world stuff because it begins to address the serendipitous opportunistic kind of contacts one can have that build that level of trust.

So that's the social element. The technical element is one of the problem of how do I preserve intellectual property over time? There was a customer I was

dealing with at a -- lets just say politely their - the CIO's office is a state government back on the East Coast and they're not New York but they're really close to them so that kind of narrows it down. They lamented the fact that if you look at their development staff, they got lots of people who are fairly young and lots of people who are -- let's politely say -- not so young and virtually nobody in the middle and their really really scared about these folks at the older end retiring really soon and a lot of intellectual property, you know walking out the door literally.

And this is a manifestation of the problem of the fact that most interesting architectural decisions are kept in tribal memory. So this is one of the hard problems of collaboration in collaborative environments. How do you extract that knowledge that intense IP that tribal memory so that others can use it? This again brings me back to architecture because a systems architecture tends to be a manifestation of those key design decisions that are often in people's heads. And insofar as we can make these things manifest, visible, approachable, then we've reduced the noise, reduced the cost of communication.

And so that's where a lot of the CDE stuff tends to focus in my mind in terms of its value.

Great question.

Steve Weaver: Hey thanks and that was a great answer. I appreciate your time Grady.

Grady Booch: I'll pay you off because that was a good question and it let me do a good answer so.

Anyone else?

Operator: Your next question comes from Gary Cernosekjr.

Gary Cernosekjr: Hi Grady, Gary's here. So I wanted to ask you about the history of Rational has focused primarily on software architecture but a lot of other types of architecture out there; business architecture, data, deployment, do you see these architectures converging over time and is that related maybe to the movement behind enterprise architecture?

Grady Booch: Sure, in fact it's curious because I just a call with the (Tow Gas) folks before I joined us here. You know architecture is such an emotionally laden word, it's kind of like the phrase, "object oriented," it doesn't really mean a lot until you dive into it but - and the phrase, "architecture and (unintelligible)," like many others, are so laden with baggage it's hard to know what it is.

You are quite correct Gary that my focus and Rational's focus has been primarily on the technical software architecture bit but there are many other kinds of architectures. As a parallel, if you look at just the problems of civil engineering, there's building architecture, there's development architecture, there's urban architecture, city planning, there's, you know, even such grandiose schemes as whole community planning, you know, talk to Disney and their planned community in Southern California - or Southern Florida for example.

So yes, there are many many kinds of architecture and there aren't unrelated to one another, as Herbert Simon would say, "They are nearly independent with one another." We start dealing with Enterprise architecture, that too is an emotionally laden term, but most people tend to mean it by saying it's the architecture, the business has to do with business. And that being said, there are both technical elements; IE their software bits and hardware bits, but

there's lots of non technical elements. It's the people structure, the organization, it's the business processes.

And you are correct in observing that there is increasing interest in the business community to really get a handle upon its quote, "architecture." This is why we see folks like the (Tow Gas) in particular, focusing upon the processes and others that help us. Get at least a vehicle to talk to one another about it. This is also why on the Rational side, we've got the (RUP) folks working with the global service method folks, trying to bash those ideas together so that we can begin to have a common vocabulary, a common process, common language with the people who do the true Enterprise architecture.

A lot of where I spend my time is working with architects out on the field, out in the wild so to speak. So I'll talk with CIOs and talk with the Senior Architects and often times the discussions are rarely about the technical architecture but hey deal with the very cool interface between the technical, and the social on the business.

So yes, that's where the enterprise architecture stuff lands. And yes that's when Rationals and IBM at large is spending some of it's time as well.

Gary Cernosekjr: All right, great. Thanks.

Operator: Again if you would like to ask a question, press star then a number 1 on your telephone keypad. We'll pause for a just a moment to compile the Q&A roster.

Your next questions comes from Leshek Fiedorowicz.

Leshek Fiedorowicz: Hi. Gary you mentioned that architecture here can be explicit or implicit, or can happen by an accident or can get out of control. And the first step is to extract that architecture and present it somehow. I'd like to ask you for your thoughts on the extraction process and then presentation process that would allow others to understand what was there.

Reservation of the tribal memory is another question that I wanted to ask, what do you think about including the web to all like technologies into the process, into that process of extraction. It is - I mean by that storing human behavior in the process of discovery of artifacture, looking at artifact. Especially humans that are - that have the knowledge about this architecture.

Right.

Grady Booch: So I heard three questions there, one is with regards to the software archeology process, the second is with regards to representation and the third is with regards to preservations of information.

I really like where you're headed with that question because architecture in isolation is generally doomed to failure because rarely can we architect in a green field fashion.

There's another IBM fellow by the name of (Chris Winters) out of the UK that spends a lot of time in this space in which he called brown field development. And what he means by that it's you know, kind of an evocative phrase there, but it's a reality that most organizations have a whole lot of legacy they have to deal with and therefore before I can even begin thinking about architecture transformation, its really important to know what it is I have in the first place.

And therefore software archeology is an important element because it's part of the process of understanding the portfolio that I have and reasoning about it. In the last two years I see more and more people that have software architect on their business cards. And I'm really looking forward to the day where somebody has the title of software archeologist because this is a skill that is poorly understood.

Quick aside before I actually get to the larger problem. This is the problem that spreads across so many pieces of the industry. Among other things I've been working with the computer history museum out of the bay area, with regards to the preservation of classic software. So this team, there's actually a software curator and a software collections committee, is actively collecting old classic software and a source code for them. We think it would be a travesty if future generations did not have the raw artifacts to the things that have transformed our world.

We have the hardware but we need to keep the bits as well too. Now, we're collecting these things but it also raises the problem, well what are these things mean and how do we present them, how do we show their essence, how do we show their beauty? That's the same problem every organization has, the brown field case, which is a problem of software archeology.

Now, I have been doing a lot of software archeological digs with the project I have to build a handbook of software architecture. And I'm not saying, you know, this is the absolute way to do it, or this is the only way to do it, but there aren't a lot of people that I've found that are doing this kind of archeological stuff in a formal fashion.

So here's my goal, for each of the systems I'm looking at, I want to be able to capture their architecture, how do I do that if I have modest access to the

architects and modest access to the source code. Which is exactly the situation that every project is in.

My experience has been that it requires an approach on multiple fronts. One of those fronts is I'll look at the artifacts that they have, what documentation exists, what can they show me? That generally is rarely satisfying because most of the stuff is out of date for a re-organization.

The best it will do for me is to offer at least a vocabulary of the problem space and the solution space so that I can start asking intelligent questions.

The next step then - oh quick aside, I'll also do pageant searches, these often times patent filings will tell me a great deal about the systems as built.

The next thing I'll do is sit down with the architects themselves and this part of extracting the Tribal memory because it's these people who have the design decision and patterns in their hands. And it's really fast and in doing this, in a couple of cases, most recently I've sat down with the two co-architects of various projects and it's really funny and sad in a way to see them fight with one another about, oh this is the way it is, and not this is the way it is, and because even within the architect, there's not necessarily) clarity to what a system is. But that process of interview and that process of investigation often leads with a number of artifacts that say, "Hey this is what the as built system is." The third element is I think there are opportunities for a code assisted recovery of stuff. We - Rational and IBM research - who are funding a number of projects with regards to retrieval of design information as a code blocking project, we Rational acquired a small worlds company some years ago and some of those facilities are within our modeling tools now. So I think there's some cools stuff to be done there and it's on those three fronts that we can a reasonable amount of archeological digging.

So that's your first question. Second question addresses how do I represent this stuff? Well there's a (Zackman framework) that talks about X number of views, (Tow Gap) talks about Y number of views if you're in a special place like the DOD, there's things like the Modaf and the Dodaf that have their own particular approach to the world.

From a technical architecture perspective, I'm a great fan of (Felipe Crushion's) 4 plus 1 model view. And I found it. It's those spy views that appear to be necessary and sufficient, to capture the technical architectural decisions that I would like to preserve. Those five views very quickly being the UK's view, the logical view, the process view, the implementation view and the deployment view.

And for all those systems that I've investigated thus far, the UNL and the four plus one model view have been quite sufficient and necessary for my purposes.

Your third question addresses how - and I got so deep into it, I'm trying to remember what it is. I think it's a matter of how do I communicate that tribal memory onto folks and the short answer that I'd offer you is, from my experience of doing this archeological digs, generally if I can offer up a architecture document that focused around those four plus one views that focuses upon the key design decisions that capture some of the many information of a, like, tell me historically what happened first, what happened before this? Because those historical positions also often impact the as built. And now if I can whack that together in a document that people can use online, and if I add to that, the capturing of the design patterns and architectural patterns have been used.

Man, that appears to be an 80% solution for many of their projects I encounter. So hopefully I heard your questions properly and those are the three bits of it, but that's my answer to you. Thanks for the question.

Leshek Fiedorowicz: Great answer. Thank you.

Operator: Your next question comes from Ken Vodicka.

Ken Vodicka: Hello. Quick question, you're talking a little bit about, in the bay area, the history museum and stuff, but go in the other direction. Can you tell us a little bit about some of the cool things you're working on at maybe five years out or so.

Grady Booch: Sure. Oh that's a very sweet question for you to ask. Cool things I'm working on, five years out. So there are two things that are consuming me and the short term. And the short term being three to five years and there are a couple of things that are in my head for the five year and beyond place. To treat a five year place is in the area of architecture and collaboration.

The work I'm doing on the handbook is going to be another three to five years for me. I've done thirty archeological digs so far, I'm awash in information right now that I'm trying put it in a presentable fashion and then I've got probably another three plus years to do the remaining 70 archeological digs. I may quit at 50, I may quite at 75, I don't know, when I get bored on or when I have enough but that would keep me busy there.

And I'm learning a heap of a lot about archeological digs and representations and the like and I hope we can do some tool pull through as well.

(Charlesa Gonzalez) is helping me out on the pattern side in particular and we've uncovered just already about 2500 design patterns that people have used and so there's a lot of cool stuff that 's coming out of it. So that's one thing.

The other place where I'm spending effort in the next three to five years is in the space of collaboration. And I'm really particularly jazzed by the whole virtual worlds bit, this is why I'm intentionally spending time in virtual spaces. I got my first customer meeting in Second Life of all things, I'm really excited about that. Where the customer wanted to meet me there and so we're going to do that. And you'll see me doing more of that kind of stuff. That's three to five years.

In the five year out kind of timeframe, I'm really interested in two problems where I'm spending some degree of energy. One is in the multi core, quantum computing space. Reality is, as Sam Adams has put it, the days of uniprocessor frequency scaling are over, IE (unintelligible) is more (unintelligible), it's dying. And thus we see the shift from Intel IMD and IBM and others, to multi core systems.

Now we know sort of what we need to do in terms of a path on the hardware side. But man this is wickedly hard on the software side. And you see some canaries in the mind right now in an industry such as the gaming community, who are using the cell and the asymmetric processor that are really struggling with how to use those processors. And you see the writing on the wall that multi core systems are going to be the norm on out. So the five year time frame for me is "what the heck do we need to do know?" to get ourselves ready for the place where multi core systems absolutely dominate.

I have a little bit of further out of the whole problems of quantic computing which say, you know, gosh given that (unintelligible) dying there is some

alternative models to computing that look really interesting. And we've seen some breakthroughs in the quantum repeating space, it looked (unintelligible).

The difficulty is we know how to do it on the theoretical and hardware side that how does one program and manage these blasted things. We don't even probably have the right programming mind we just do it here. So I'm spending a few cycles to worry about those kinds of things.

The other space where I worry about are the sort of five to - five beyond time frame, is the following reality. In that time frame, we can reasonably predict that at a reasonable level of fidelity, bandwidth is unlimited and free, computation is unlimited free and connectivity is unlimited and free, so the question we ask ourselves, what is computing look like in that space? What is the world look like if I had digital paper that I can plaster all sorts of devices with.

And that has a couple of elements to the problem, once of which is vary interesting and social, how does our civilization change because of it. And so I spent a little time thinking about that. And the second is a very pragmatic one for IBM, how does IBM make any money in that space. So we have to keep reinventing ourselves, not just IBM but God, how about Microsofts and the Googles in the world make money in that space.

So those are the areas where I spend a few of my cycles as well.

Ken Vodicka: Interesting. Thank you very much.

Gary Booch: You bet.

I'm curious, I'm going to turn the question around. What do you think we should be worried about five years beyond?

Ken Vodicka: Again, the power of the process and stuff and seeing how software will match up to that is interesting. I mean, just looking at some of the things, you know, the games and stuff, that they - my kids are playing with today, it's fascinating to see where - to kind of better understand where we're going to be five ten years from now.

Grady Booch: Yes. Are there Halo three freaks by any chance? Do they live there?

Ken Vodicka: I - sorry?

Grady Booch: They play Halo 3 or what kind of games are they into? The reason for the question is I'm looking for death match partner so…

Ken Vodicka: They're still young, they're still into a lot of the Wii games that…

Grady Booch: Okay, cool, cool. Yes it's amazing to see, as I say, that community is the canary in the mind because, man the amount of computing power that's out there. The other canary in the mind is the whole entertainment world. If you look at where CGI is headed, (John Cameron) has been green lighted recently a year or so ago for a 200 million dollar plus movie that's entirely CGI and so we're' going to see - expected some breakthroughs with regards to photo, realistic rendering of human phases in life, which is sort of the last holy grail for that community. You tie that together with real time, you know, body sensors and the like, and man, the ability for true tele-immersion is there its, kind of like hollow decks here we come.

Ken Vodicka: Kind of scary.

Grady Booch: It is. Well it's scary but exciting. Exciting in a sense for me that here I am sitting in my office in Denver. Well not my office, but my home in Denver. Where could I be five years from now in terms of the tele-presence. This is why Second Life is so cool for me. What would happen if I take Second Life and I add to that sensors on my body so that it actually tracks my movement and allows me to see where I'm actually looking and vice versa so that I become immersive.

You know this kind of stuff is what snow crash and (Neil Stevenson) speaks about. The weird not far way from the technical reality of making that happen, what's our world like if that can be so. And if you add to that the notions to get a little bit of further off field, you know, growing cost of energy in the light it makes sense for us as a community, then they can come rally to that to say "gosh maybe I don't need to move atoms as much, maybe I can move bits."

Ken Vodicka: Thank you.

Grady Booch: You bet.

Operator: Again, if you would like to ask a question, press star then a number 1 on your telephone keypad.

At this time there are no further questions.

Grady Booch: Then we might want to wrap it up then I guess.

Angelique Matheny: Yes, I think so. I think we'll wrap it up. And if, just on the side, if anyone knows a good Halo 3 partner for Grady let us know. Grady thank you so much, that was a very valuable session today. We appreciate you taking the

time to share your knowledge and experience on today's topic. So thank you for taking the time out of your day again.

Grady Booch: You bet. Take…

Angelique Matheny: Go ahead.

Grady Booch: No I just wanted to say take care everyone. Thanks for coming.

Angelique Matheny: Great. If you'd like to listen to this conference again, or share it with your colleagues, this will be made available for replay in MP3 format in about 48 hours. Check out the Rational talks to you page at IBM dot com slash software slash rational slash talk. This link also includes the other titles in the series so mark your calendar and make sure to register for our next talk on Tuesday, December 4, Manage ROP base CMMI initiatives at 1:00 p.m. with Judy Murphy.

And again I'd like to thank our speaker Grady Booch for being with us today to talk about architecture. We'd also like to thank you our audience for your interest in IBM, we hope to see you back for one of our events in the near future. Thank you very much. This concludes this session.

Operator: This concludes today's conference call, you may now disconnect.

END