OPERATOR: Welcome to the Gain a Complete View of Customers Increased CRM Adoption with Rapid Integration Webinar. I would now like to introduce the speakers.

Jon Lal is a senior applications engineer in the Enterprise Business Solutions Group at Blackboard. His main responsibilities include development and deployment of enterprise systems, various Internet applications and the integration architecture between them.

Jon has been responsible for engineering integration solutions for over 10 years and has worked with applications such as salesforce.com, PeopleSoft, Oracle, SAP, [Data Tell], Remedy and Sales Logics. Currently Jon is the principal developer for the CRM systems integration for Blackboard.

Jaime D'Anna is a senior product marketing manager at Cast Iron Systems, driving product marketing activities ranging from corporate messaging, channel enablement and product positioning to leading strategic programs and collateral creation.

With over 15 years of professional experience, Jaime has held roles in product marketing, product strategy and presales engineering for various ERP/CRM and Internet applications in companies such as Oracle, OpenText and

Documentum. He holds a Bachelor of Science degree from Santa Clara University as well as a number of industry and regulatory certifications.

I would now like to introduce today's event moderator, Jaime D'Anna, senior product marketing manager at Cast Iron Systems, now an IBM company. Jaime.

D'ANNA: Thank you so much, Judy. And thank you all for joining us today on our Webinar entitled, Gain a Complete View of Customers and Increased CRM Adoption with Rapid Integration. As mentioned, my name is Jaime D'Anna, senior product marketing manager at Cast Iron Systems -- number one in cloud integration -- now an IBM company.

I'm very pleased to be joined today by Jon Lal, Senior Application Engineer at Blackboard, one of our most valued customers. Many of you in attendance may be considering adopting a cloud strategy and others of you may have already deployed a cloud strategy and are seeking to maximize your investment by integrating the cloud with your on-premise application. In both scenarios, we will show you how partnering with Cast Iron can enable you to accomplish your goals and integrate them to the cloud.

So let's take a moment to set expectations by going over today's agenda. I'll start off with a brief overview of

Cast Iron as a company as well as our technology. Then follow that with some background on the challenges our customers have been facing as they try to connect their enterprise to the cloud. In that context, we'll talk about WebSphere Cast Iron cloud integration and how it was designed specifically to meet the challenges our customers are facing.

We'll follow this with a case study from Jon Lal. And we'll give you a real-life scenario of how he identified critical business challenges that he was able to resolve through integration, specifically achieving a streamlined opportunity to order process by deploying an integrated salesforce.com instance...

...which something many of you in attendance might be finding very relevant, all this with a live demonstration of WebSphere Cast Iron Cloud Integration. And then, open it up to some questions and answers as many of the challenges described in this Webinar may be relevant to what you're currently facing in your own environment.

So to start off, I'd like to briefly highlight some relevant facts about Cast Iron as a company. Early on we identified ourselves as number one in cloud integration. We can make this assertion as we've been around for nearly 10 years and have an established presence as thought leaders in the cloud

integration space, pioneering strategies and technology for SaaS and cloud integrations with specific emphasis and focus on speed and simplicity.

We often uses a tagline, "integration in days" because we have many customer case studies and proof points where we've done just that. In total, we have thousands of customer integrations successfully connecting cloud and on-premise applications.

Our success in solving these integration issues have led to consistent growth as a company in no small part due to our satisfied customer base as evidenced in our retention rate of 96 percent. So the speed, simplicity and effectiveness of our solution is recognized as best-in-class by a number of awards, all of which we proudly invite you to view on our Web site in detail.

So Cast Iron as a company as well as its solution has resonated within the analyst community for years. In fact, Cast Iron's best of breed technology, successful cloud integration strategy and satisfied customer base were among the top drivers which prompted IBM to acquire Cast Iron as a company.

It's no surprise that the analysts have continued to buzz about the IBM acquisition of Cast Iron, which has been

overwhelmingly favorable. Once again, we invite you to our Web site -- and that's www.castiron.com -- to see specific quotes by industry analysts, partners, bloggers and what the market in general are saying.

So now, let's set the stage. As mentioned earlier, many organizations like yourselves are either considering or have adopted a cloud application such as salesforce.com due to the many benefits and considerations such as ease of deployment, reduced maintenance costs and overall cost effectiveness. In fact, with a compound annual growth rate of over 97 percent, it is the fastest-growing sector in the software industry.

The reality is that while many customers are adopting public cloud applications such as salesforce.com or building their own private applications, they still use and maintain their enterprise on-premise applications such as SAP, other packaged applications or even homegrown applications.

This has created a hybrid environment. As such, there's a need to connect the cloud applications with these existing on-premise applications to optimize performance, increase sales productivity and maximize resources and investments.

And once the need for integrating two key systems such as CRM and ERP applications has been assessed, you may realize

that there are a number of other applications in your enterprise which have dependencies and affects on your data such as billing, inventory or even a database containing all customer master data. This growing complexity is why customers like you have been seeking solutions to integrate data and applications.

So previously what our customers have noticed is that while there's an application integration need -- and this is actually growing more and more complex -- the available solutions have been limited. So take, for example, custom code.

If an organization has enough IT resources and programmers to create a one-off custom integration, this can often be a tempting solution. However, this presents a number of resource-intensive hidden costs and maintenance support and any future changes should the need arise to grow the solution to integrate more applications.

The second option is to go with an on demand solution, specializing in simple cloud-to-cloud connectivity. While this may be a tempting low-cost alternative for a simple cloud integration project, it does not really offer the scalability and functionality to address on-premise or hybrid scenarios. In short, pure on demand point solutions are not equipped to handle complex processes and back office

applications.

The third option would be to consider a traditional on-premise solution. These solutions are based on a more classic ETO -- or, Extract Transform and Load -- architecture designed for extracting, processing and storing large quantities of data.

This older architecture also equates to a longer installation and implementation time as well as a much larger IT footprint. Most importantly, SaaS applications are still pretty much an afterthought to these vendors -- meaning, they may have a cloud solution as well but it is a completely separate product to the original on-premise solution.

So you'll end up purchasing and maintaining two or more complex systems to solve one problem. And it's due to this complexity of the hybrid world and the shortcomings of the offerings we just described that integration is among the top concerns to IT executives when it comes to adopting SaaS or cloud applications.

This is reflected in the recent survey by Saugatuck, arguably the leading analyst for cloud computing. When Saugatuck asked a group of IT executives, what were their top concerns regarding SaaS adoption and deployment,

integration was second only to security on their list.

So this concern for integration was not only relevant to enterprise applications, the same integration concerns come up for flat files, data structures and other SaaS applications.

So these concerns, coupled with the fact that previous product offerings described failed to provide adequate reassurances due to their limitations and shortcomings have created an environment where many IT organizations have felt hesitation, if not outright rejection of SaaS applications as reflected by a survey by Gartner.

So we can see the complexity of integrating applications has obviously been a driving factor for adoption and implementation of SAS or cloud solutions. The complexity of integrating the cloud applications to the enterprise and the limitations of current solutions available are precisely why Cast Iron developed WebSphere Cast Iron Cloud Integration for organizations just like yours.

WebSphere Cast Iron Cloud Integration was designed to meet the specific needs of connecting your cloud applications, on-premise applications and any hybrid environment between the two. It can connect one-to-one or scale to connect one-to-many application endpoints providing a platform to

effectively and rapidly consolidate and manage the application functionality and your enterprise regardless of how it was deployed. What makes WebSphere Cast Iron Cloud Integration stand out from the three previous integration options mentioned is that it's complete, proven and trusted.

So let's go into a little bit more detail on why being proven and trusted makes a platform unique as well as the preferred option for organizations seeking to integrate their applications.

The first and arguably the most important point about WebSphere Cast Iron Cloud Integration provides a complete functionality to connect all applications in your enterprise. We're the only platform available that provides complete flexibility and deployment options -- in other words, you can build, run and manage an integration between applications such as salesforce.com and an ERP system and deploy it using a physical on-premise appliance, a virtual appliance or completely multi-tenant cloud service.

It's the only solution that allows for this choice of deployment using the same product and code base interchangeably. Because of this, it is future-proof -- meaning, you can start off with one form factor today and easily move to another to conform to your IT strategies in the future.

The second advantage WebSphere Cast Iron Cloud Integration offers is total connectivity. This means not just native connectivity to industry-leading SaaS, cloud and enterprise applications, but every type of database, custom application, Web services and connectivity protocol.

Another important feature is reusability. WebSphere Cast Iron Cloud Integration provides a user-friendly template-based approach via the Template Integration Process -- or, TIP. These are templates of common integration scenarios which can be used as a starting point for your integration project.

You're not alone in your integration. You should not have to reinvent the wheel. Cast Iron gives you the benefit of previous success and best practices available through our community of users.

Finally, this is one platform for all types of integration projects. You can use WebSphere Cast Iron Cloud Integration for data migration, process integration or even for UI mashups for taking relevant data from a back office application such as SAP and displaying it within a commonly used front office application such as salesforce.com.

So to quickly recap, the value of the complete solution

translates to lower risk, saving time and costs associated with building your own deployment connectors for each project and having one platform for all types of integration projects, meaning an increase and return on investment and a lower total cost of ownership.

That's the high-level overview of what makes WebSphere Cast Iron Cloud Integration complete as an integration platform. Cast Iron took these functional requirements coming from organizations like yours and saw the deficit in the existing product offerings available in the market. They architected this platform with the goal of providing complete functionality to meet the integration needs and requirements that IT and business were demanding.

The reliability of this complete functional set is why application vendors themselves trust Cast Iron to form strategic partnerships. In fact, the world's leading cloud or SaaS mega brands are partnering with Cast Iron, working closely with us to solve the issues of integration for their customers.

We're the only integration provider today that can provide the benefit of all these partnerships. The trust in WebSphere Cast Iron Cloud Integration is definitely evidenced in the close relationship with one of the world's leading cloud application vendors, namely salesforce.com.

The third factor that sets WebSphere Cast Iron Cloud Integration apart from competing solutions is that it is proven with thousands of customer integrations and a wide portfolio of customer success stories. So we can prove by previous success how we can solve your particular integration scenario.

A few quick examples of our proven success in integrating applications which you may find relevant is that we see a number of organizations which have the need to connect their CRM systems -- in this case, salesforce.com -- to various back-end ERP systems.

I'd like to highlight the duration of these projects. As we can see, these projects in very large organizations in many cases spanning multiple geographies, were able to complete and accomplish enterprise application integration in just days.

We should note that once we were able to successfully integrate their initial projects -- meaning, their initial cloud to their on-premise systems -- they were able to utilize Cast Iron as a viable platform for all their subsequent integration projects.

So once the initial project was complete, they were able to

scale upward as needed. So you can see the reusability and the template-based architecture we spoke about facilitates an ease and a velocity, a rapid integration.

So now that we've taken a look at the background of Cast Iron as a company and our best of breed technology which enables customers just like you to quickly connect your enterprise applications to the cloud, I'd like to turn it over right now to Jon Lal, who will be speaking about challenges he was able to resolve through WebSphere Cast Iron Cloud Integration. Jon.

LAL: Thanks, Jaime. Today I'm going to be talking about how we deployed our salesforce.com instance with Cast Iron. We have a need to implement salesforce in addition to the on-premise applications that we have here.

First, a little bit about Blackboard. We were a company that was founded back in 1997 at the time when we noticed that higher ed campuses were investing a lot of money in their network infrastructure and we wanted to take advantage of that. Currently we're a leader in the educational industry with our course management system.

Our mission statement is, simply put, to increase the impact of education by transforming the experience of education. And right now we have a little over 1,500 employees. We

attend to 5,000 clients, and each day millions of users such as teachers and learners use our product.

At Blackboard, we offer four different types of services that we refer to as platforms. Our main platform is our Blackboard Learn product, and that is our course management system which allows institutions to deliver course materials to their students and to assess them online.

We also have a Transact line of business which deals with student cards allowing access to certain parts of campus, vending machines. And we also have agreements with merchants near campuses where students can use money that's deposited onto their card to purchase things like pizzas and soda pops.

We have a Blackboard Connect product, which is a mass notification service to allow administrators of the school to quickly send messages to their communities. And our new line of business is the mobile product, and that just allows the people who use Blackboard to use them from the smartphones that they are already carrying.

So in dealing with this type of business, we have a lot of changing models. And one of the things that we recognize in a growing business is that we had a lot of different manual processes and those were coupled with different systems and

each department seemed to have their own processing system that we needed.

In addition to that, we wanted to bring on salesforce.com, because we saw that it was a leading technology in the industry and having that in place would benefit our business. But at the same time we wanted to keep PeopleSoft in place, and this would allow us to still have all the data that we have in place and allow the teams that we have in place to continue to work.

One of the goals of the project was for convenience. We didn't want one employee to have to log into multiple systems, so we had it segregated where an employee would only log into the system that they were going to be working with. And because of this, we needed to have an integration between the systems that people were working on.

And one of the key goals was that we wanted to ensure that salesforce.com was deployed with an integration at the same time. We wouldn't want to have salesforce deployed independent of everything else and come back and try to integrate it after.

So an overview of the finished product, what we envisioned to see once this was complete, we had some teams using PeopleSoft. We had a couple of other teams using

salesforce. And we had approvals that were going to take place in both systems, and we wanted that to be tracked back and forth.

And then finally, orders and opportunities would also need to go back and forth so that we could visualize and report on what was happening in each system, as well as give a high-level view for forecasting.

To make this all happen, we penciled out what integrations we needed to happen. Like I mentioned, we had PeopleSoft existing; we were going to deploy salesforce.com. And in order to have it functioning properly, we needed integrations between certain things and we needed certain things to reside in only one system.

Accounts and contacts, memberships would reside in both systems, and we wanted the ability for somebody in salesforce to change an address and somebody in PeopleSoft to see that change. So we penciled in bidirectional integration for those.

Our product catalog resides in PeopleSoft financials, and we needed that in salesforce so that we could have the product available for salespeople on the opportunities that they were working. Our campaigns and leads would solely reside in salesforce, so we didn't see a need for integration

there.

Opportunities would be worked in salesforce, but we needed to have them in PeopleSoft as well so that we could have a launch pad for the other objects that were in PeopleSoft that needed the opportunity header information.

And once in PeopleSoft, the opportunity would be transferred to a quote. And as it passed through the sales process, [only an order] and a contract. And those needed to go back to salesforce so that we could ensure that the sales team who was logging only into salesforce was able to keep up to date to where the finance people were at with respect to their opportunities that they had just one.

And lastly, our support team handles all of their help desk tickets in PeopleSoft. And we wanted to have a high level visibility available to salespersons so they were not blindsided by an unhappy customer when they were trying to make an up sell.

So we had a bunch of options in front of us, and one was to make something ourselves; and the second one was to implement something that was already available. The option of building a custom integration piece would allow us to have some control, and we looked at whether we were going to build something or buy it.

We could easily write something with the expertise that we had in house; at the same time we would save time by buying or outsourcing the development of that and we could focus on other things.

The maintenance of code also became a factor in this. We would have to maintain and support it as it grew. And we had to be sure that it was scalable and make sure that we could grow it with our company without having it become a resource hog.

At the same time, because we were deploying salesforce for the first time, we had limited expertise in that, and that was already going to cause a risk for the aggressive timelines that we had in place. And with the custom developed solution, we were going to have issues with making it consistent across different platforms.

The other option that we considered was the Cast Iron solution. We first had that on our radar at Open World 2007. We assessed and looked at the solution, and it came recommended by salesforce. We did our due diligence and reviewed what the users were saying about the community.

We were certain that it could integrate what we needed to do very quickly. And we also noticed that it will be able to

accomplish all the goals that we had set out to accomplish with this project. And the learning curve allowed for a very, very quick development schedule. And so with all those things ahead of us we decided to go with the Cast Iron solution.

We were able to deliver. We had strong time constraints, and the majority of the time was spent defining the process with development for the integrations taking a very short amount of time. We met the goal of deploying salesforce and the integration at the same time. That was very well embraced by the departments that were working on it.

One of the things that we liked to note were the numbers that were present. In the five years that the sales team was using PeopleSoft, they entered 40,000 notes. And in the first quarter of using salesforce, they entered 50,000 notes, which was a huge way of looking at how well the platform was adopted and how well the integration was performing.

We also allowed the sales contract process to become quite significantly streamlined, removing a lot of the manual process. We took out Excel files and eliminated a lot of the e-mail correspondence between the finance teams and the sales teams.

And now we're happy to say that Cast Iron is definitely our central solution for integration. Currently every day our appliance that we have here processes about 10 million rows of data to keep all of our systems in synch.

So looking back at what we've completed, every year we sit down with the teams and look at their feedback and see where we can enhance stuff to make it better for them. So every year since we've deployed, we've had three releases where we introduce new functionality and push more things to salesforce. We've also expanded the footprint of Cast Iron and we're not just using it just to integrate salesforce and PeopleSoft now.

Going forward, our current roadmap includes projects that are outside of sales and finance. We have our client care program that wants to be introduced to the Cast Iron integration facility so that we can push data to them. And we're always investigating other solutions that will help us better our business through efficiencies.

Again, looking back at the whole project from a high level, there's a couple of things that we've picked up along the way. One of them is to definitely use simplified technologies to accomplish what you need to do. We don't see any reason to go ahead and build something when there's something available already.

When diving into a project that is related to integration, you want to have representatives from a lot of departments. Even though the integration doesn't directly affect them, they might bring stuff to the table that you may have missed.

As with any project, a well-defined business process is key before you start so that you won't have to redevelop anything. We've learned that we should segregate our Cast Iron projects accordingly so that we could stop one part of an integration and keep others running. We also noticed that if we have a good amount of tasks in each one of the integration projects, there's a benefit in having that.

And lastly, to understand what endpoints Cast Iron has to offer. There's several that they have, and using them effectively will definitely increase its value on site. So with that, I'd like to hand it back to Jaime.

D'ANNA: Thank you so much, Jon. So I really appreciate this real-life scenario of what you face in your environment. There's a lot of salient points that hopefully in my next portion right here, what I'd like to do is go over on a demo and highlight some of the points that you made here.

So what we're going to do is we're going to essentially take a look at the WebSphere Cast Iron Cloud Integration solution. And the goal here is to reiterate the value of real-time bidirectional integration accomplished through a simplified user-friendly interface with the ability to reuse this information for future projects.

We'd also like to highlight some key functionality along the way which may be relevant to your particular integration project. So let me just go ahead and share my screen right now. All right. There we go. Okay.

So as Jon was mentioning, there is a drastic need in the salesforce to ensure the fact that we don't want to be having any upset salespeople because they were, as Jon said, blindsided by an unhappy customer.

In order to prevent that, we have to essentially streamline a lot of those processes that would give the salesperson a red flag well in advance so they don't make a subsequent phone call to potentially existing customer and find out a) the product has shipped late; b) they have a return; c) they have a support request or any issues of that nature, which if identified well in advance can arm the salesperson with that knowledge and that ability to act accordingly.

So let's take a quick kind of view through the life of a

salesperson, and then we'll transfer that over to the life of an IT person and how they can make that salesperson's job a lot better. I'll just move this over here. Okay. There we go.

So, let's start off by going through this interface, which is salesforce.com, a cloud CRM application that many of you will already be familiar with. We go to accounts, and let's hit accounts modified today -- and we have none. Okay. So we've seen salesforce.com.

We also have a custom ERP solution. In Jon's example, it was PeopleSoft, but for our purposes this was an ERP solution that was built on SQL Server. What is relevant is the fact that this is on premise, and this contains our customer master data as well as a number of other potentially useful pieces of information such as all the previous order history, products shipped, et cetera.

So now let's go into Cast Iron, the Cast Iron cloud. So I'm just going to log in really quickly right here. So I as an IT person understand the value of streamlining the processes for that salesperson as well as making their lives easier by giving them access to a number of different information points that are located in disparate applications.

So what I can do is I can create, modify or run an

integration project at this point to connect any and all applications which facilitate that knowledge transfer. If I didn't have the wherewithal to do this project by myself, once again, going back to our example of reusability, I can search on any number of endpoints.

And in this case, I'm going to choose salesforce.com, click search, and this will give me, once again, a list of templates of the best practice with regard to projects that have taken place previously. So I'm automatically getting a template that can leverage successful integration and apply that to my project.

But moving on, as an IT person, I can say, well, I've actually already done that. So I'm going to choose a project that has already deployed, and I'm going to open this in the Designer from the Cast Iron cloud.

Now, what this is going to allow me to do is leverage, once again, that best practices and project success from a previous successful project and very simply and rapidly walk me as a junior level IT admin through a step-by-step wizard which will enable me to complete a very complex project in mere days.

So as we're opening up the Designer right now, I want to point out that right here we have a very simplified wizard

environment that is going to perform an end-to-end connectivity based on a specific business logic.

Now, this business logic right here, we already mentioned we have salesforce.com as our cloud operation, which we're going to connect to from our home ERP system based on SQL Server. What this piece of logic is saying, that if an object is created in the ERP system but it does not conform for a specific business rule -- in this case, namely, it does not have a telephone number in the telephone field -- we're going to abort the operation. The operation will be terminated.

And if not -- if all the fields are complete -- it will extract the information from our ERP system into salesforce.com. So that's the summary of what we're trying to accomplish in this specific integration scenario.

I'm going to go ahead and click next here. The next step in our workflow is to log into our homegrown ERP system. I've already prepopulated and validated this. I'm just going to skip to next, which is validate the connection to salesforce.com.

Now, once again, I'd just like to say I have already populated this information, but if there were any doubts I could go ahead and test the connection and we know we're

successfully connected. So this provides our connectivity from one application to the next.

Now what I'm going to do is take that information and extract from our ERP system all of these specific objects located in these specific tables. And this tells me what table structure I'm extracting from. We're talking to this database at the database level at this point.

Now, the next step would be to talk to salesforce.com. And here, we have the benefit of talking to salesforce at the API level. Why is that beneficial? Because of the fact that the API already has a set of business rules and logic built in, which facilitates the ease of integration as well as the rocket approach we're trying to take.

So as we see right here, in the object type of account, I could very easily in salesforce.com choose a specific other object type, whether that's opportunity, et cetera. But for our intents and purposes we're going to stick with account.

And I'd also like to point out that if anybody on the salesforce side has created a custom object field, they can do that through underscore C, and those are also manifest and reflected in our structure.

So we're always looking at the latest and greatest, the

greatest API and objects that have been created in salesforce.com, and we have the chance to update along the way. That's the point I want to make, we're always looking at the latest structure and the latest API at salesforce.com to ensure that our integration is dynamic.

Now, going back to our example of connecting the ERP to salesforce.com, out of the box, just by using this template, I'm probably about 60 to 70 percent done.

However, because of the fact that there is no universal naming convention when it comes to the applications, we do realize that at some point there might be an issue with mapping specific fields from one application to another. Namely, in our custom ERP application that we have on site, we see there are many instances of address.

So we have essentially four address fields. That, as being the junior IT admin, that maps to my knowledge to billing street in salesforce.com. So I understand that any of these address fields map to one field in salesforce.com.

Once again, being a junior admin, I can take advantage of very complex functionality all within the simplified user interface. And as such, I have a very complex function called concatenate, which I'm going to add to our mapping structure. What that is going to allow me to do is map

these specific address fields in our homegrown SQL-based application to salesforce.com's billing street.

So without writing a single line of code, I was able to conduct a very complex functionality through this wizard-based interface. So I've done my mapping. Go ahead and click next. Now I'm going to continue on to the business logic that I've already set up where the telephone field equals blank, it's going to cancel or terminate the operation. I can also append to that or add any other business logic I'd like to.

So what I'm going to do is given the fact that we're talking about U.S. customers in this example, I know as an IT admin that U.S. telephone numbers have to have at least seven digits. So what I'm going to do is I'm going to add where the phone number is less than seven, go and add that field there.

Now, I can add as much...more logic, [boolean] statements, et cetera, as I would like. But at this point I think we're done. We'll just click next right here. Now, as a final step, I have the option to publish this once again to the Cast Iron multi-tenant cloud, or to an on-premise appliance or virtual appliance.

Once again, this is the completeness, part of the

completeness of the Cast Iron WebSphere Cast Iron Integration that we have to date because of the fact that it's one product across three different form factors interchangeably.

So let's go see this in action right now. We've already seen how simple it is to create this integration. But in a real life scenario, I as the IT admin have completed the integration and now I'm going to transition roles back to that salesperson who's logged into salesforce.com as well as the ERP system.

So what I'm going to do is I'm going to take the role of that salesperson again, and assuming that I had access to the ERP system, let's take a quick example and fill out a name. So call this Mountain View 3. We're going to put in an address really quickly. State or province, California. Cost code. Okay.

Now, I'm going to click add right here, which should give us a new customer ID, which is 129. This is a little bit relevant, because what I'm going to do right now is I'm going to go into salesforce. Now, we've already shown that today there have been no accounts that have modified. But if I click refresh, let's see what happens. Nothing has happened.

And as an SE, at this point I would be tap dancing around and saying, why this is actual functionality, et cetera. Well, I will say that this is a functionality, and many of those of you on the line probably have already caught it.

What we were missing, according to our workflow rules or business logic, was a telephone number. So let's go ahead and add that telephone number to our record. Click change. Do inquiry to make sure that it took. So we do have a telephone number right now in the telephone field. Let's go back to salesforce, do a refresh once again, and there we have our record.

So what we've seen is we've created a record in our on-premise ERP system and once done, adhering to the specific business logic that the IT admin has put into this integration, that information or data, customer data, very valuable to me as a salesperson, has been extracted and instantly put into salesforce.com. And just to ensure that we see that ERP ID is 129; that maps back to our ERP in our on-premise system. So we see that this is a real record that has been created real time.

Now, as such, once again, this is a bidirectional connectivity. So I'm going to go ahead and set this to active. And if I were a salesperson with privileges to amend records in the ERP system, I could do that from within

salesforce.com. So I can go ahead and change this to save right there.

So what does this mean? This means that as Jon was saying, in order to not be once again blindsided by an unhappy customer, I don't have misinformation -- meaning, I have inputted information here at salesforce and that information would now be made pervasive as far as an update throughout my entire enterprise and all the systems that I've integrated to to ensure that the customer information is up to date; and for instance, an opportunity to order process would be streamlined because I wouldn't have to log in to multiple systems.

Now, back to the IT example. If I were an IT admin and I wanted to turn off some of my ERP licenses, in this case it's not relevant because it's a homegrown application, but if it were something like PeopleSoft or Siebel...rather, JD Edwards, what I would be able to do is I would be able to use salesforce at this point as a single point of entry or a portal into my applications that have relevant data connecting to the opportunity, the account, et cetera.

And what that would mean is that I wouldn't need to log into multiple systems. And I'm just going to do an inquire real quick here. I wouldn't have to log into multiple systems in order to look at or transform any data. So that's a very

valuable tool that we can use in order to increase worker productivity, in order to minimize or in effect bring down to zero the level of mismatched information, multiple data entry, et cetera. So once again, not having to log into multiple systems in order to gain access to very valuable and up to date customer information.

So with that we've taken a look at the current environment of today's enterprise landscape, the business need to connect on-premise applications to the cloud. You've seen the limitations and challenges using custom coding on demand solutions or on-premise solutions in order to solve one point of integration.

So after a view into how our customers such as Blackboard have partnered with us to successfully achieve their integration goals and then taken an indepth real-time view of the WebSphere Cast Iron Cloud Integration, we hope to have demonstrated how this is a truly proven, complete and trusted platform.

So at this point I'd like to open it up to any questions that you might have. And I see we have a number of questions in queue, and let me just start off right here. So, how often are your transactional integrations running? And I'm assuming this is to Jon Lal.

LAL: Thanks, Jaime, yes. We have a bunch of orchestrations, about close to 50 or 60. Some of them are running at 30-second intervals; some are nightly. And how we've determined what frequency stuff needs to happen is based on what the data is.

So, for example, account changes like you were just demonstrating happen at 30 seconds. Whenever somebody changes something in salesforce, we want it to be saved in PeopleSoft almost immediately. However, when the price book changes in our PeopleSoft financial system, it's not really necessary that that happens immediately in salesforce. So we batch those and send those over at night.

D'ANNA: Okay. Next question. Apart from PeopleSoft salesforce.com integration, what else have you been able to do with Cast Iron? And once again, I'm assuming that's to you, Jon.

LAL: Sure. So, right now we have a bunch of other things in play. One that I was just doing recently was a data manipulation project. We have a reference team here at Blackboard who works with our referenceable clients, and they used to keep everything in Excel spreadsheets.

And they had a list of these things, a list of Excel spreadsheets with a bunch of different columns. And they

were moving into salesforce, they were one of the teams that were jumping on to to the salesforce bandwagon. So, they gave me the files and said, can you get this into salesforce?

And there was no quick way, without having to Excel-mine the data and load it into salesforce. What I could do is I could load the Excel files into a database and then with Cast Iron iterate through all of the rows in the database and put stuff in different places. So that worked out well.

Another thing that we just completed was a Web service for our annual conference registration portal. When a potential attendee is trying to register on our site, they can type in the first couple of letters of the company that they belong to and the registration portal will call a Web service that Cast Iron provides and return a list of possible matches so that we don't have typos and we can get other data for that customer at registration, like what products they own and what their address is.

The alternative to Cast Iron would be directly building something in salesforce, creating a custom Web service in salesforce. But going through Cast Iron, we would easily do that with enterprise Web service that they already offer. Those are the two big ones.

D'ANNA: Nice. Very good. So, the next question, where can we find a list of platforms that Cast Iron supports? So by platforms, I'm assuming by this question that this is not the case that the person can retype, by platforms, I'm assuming you mean locations.

And with regard to application endpoints, the answer is there's no list because we can pretty much connect to anything. And as I mentioned, we have very tight relationships with the leading application vendors.

If we don't have an application with a published API we can connect to the database or object level. We can connect using Web services or any given protocol. We can even do round-about extraction and just take flat files from an application and import them into the other.

So there's really no application that we've come across that we can't connect to. If you're talking about platforms like for instance operating systems, in that sense, it's really a non-issue just because of the fact that we are an engine for transforming and basically transferring data from one application to the other.

So we are pretty much agnostic as far as operating systems. We're in an appliance behind your firewall, we're in the cloud, or we're in a virtual instance anywhere you'd like.

So that's the answer there.

The next question, what level of expertise was required? So I'm assuming this is back to Jon Lal's example. I will say that in the demo that I did, we were giving the example of a junior IT exec because what we're trying to convey is the fact that you don't need to be an integration, quote-unquote, expert.

And in so doing, if you'd like to take a quick visit to salary.com just to understand the value of what I'm trying to convey here, take a look at IT admin or IT staff and then type in "integration expert." You'll see the difference or the delta in those salaries can be quite substantive.

So we're trying to say that you don't need a large skill set to be brought with you in order to do these very complex integrations because of the fact that we've simplified the user interface to the extent that pretty much anybody with knowledge of integration or their IT space would be able to do this. And I'll let Jon quantify that as far as what was needed in his situation.

LAL: Yes, there wasn't as far as developing a Cast Iron integration, there wasn't a lot of IT skill set that you acquired. As you saw, everything was drag and drop. The harder part of the integration was the PeopleSoft side.

And I'm not sure if any of you are familiar with the Bohr model, but PeopleSoft doesn't address one address for one account; they're scattered all over the database. And reverse engineering that to get out a simple address was the harder part of it. But the Cast Iron initiative to move stuff over was pretty easy.

D'ANNA: Excellent. So, the next question, what is the typical Cast Iron solution pricing structure, and does it differ across the different solution delivery methods -- i.e., appliance, virtual appliance, cloud. Okay. So the WebSphere Cast Iron Cloud Integration that we are conveying today is one product across three different deployment options. As such, as I said, one product, one code base across the three form factors. There's no difference in pricing with regard to which option you decide to deploy.

Now, that having been said, we do give you a number of pricing options. You can buy the product outright -- meaning, you own it, you put it behind your firewall, you do whatever you want with it. And we'll maintain it, you're still a part of our IBM support system, et cetera.

Or you can basically do term pricing, which is per month pricing so that a lot of people are feeling that, you know, we already purchased a cloud solution. We're used to that

kind of term pricing, we'd like to continue along that route. We can provide that as well.

In all options what I will say is that our solution is priced per endpoint. So if you have two specific applications such as the ones I mentioned CRM to an ERP system, those would be considered two endpoints and we would price accordingly.

I think we have time for one more question: which endpoints are you currently taking advantage of? And once again, I'm thinking this is to Jon.

LAL: Sure. The main one we're using is a salesforce.com connector, which Jaime mentioned goes through the application layer. As with everything that you can touch with salesforce, everything goes to the application layer -- so validation rules that are implemented by the business are still followed.

We have, what we do with PeopleSoft is we have the database endpoint which is an SQL server. We use the HTTP receive and post endpoints, which we mainly use...well, the receive one is used by Cast Iron itself.

The way I've built out some of the orchestrations is I have one orchestration that runs out and sees what needs to be

changed, and it calls another orchestration, sort of like a subroutine or a different kind of a method, and it calls it through an HTTP receive.

And for the post, we don't use the PeopleSoft connecter to talk to PeopleSoft; we actually construct our own SOAP message and send it through PeopleSoft so we have a little bit more control. e-mail, we use FTP to move files from different places. And the Web services endpoint, like I mentioned with the portal registration thing that we just built.

D'ANNA: A lot of people tend to use Web services when it comes to PeopleSoft for just the reasons that you've just mentioned. I really appreciate this insight, Jon. I want to thank you once again on behalf of the IBM Cast Iron, and thank you to you and to Blackboard for participating in this Webinar.

And everybody who is attending will receive an e-mail with the link to this for a subsequent replay. And once again, thank you so much, Jon.

[END OF SEGMENT]