

Speech 2—Part 1 Transcript: The role of DB2 in Web 2.0 and the IOD World

Slide 1: Cover

Welcome to the speech, The role of DB2 in Web 2.0 and in the Information on Demand World. This is the second speech in the DB2 on Campus lecture series. My name is Raul Chong and I'm the DB2 on Campus Program Manager.

Slide 2: Agenda

The agenda for today is divided into two parts: the first part talks about IBM's direction, the role of DB2 in the Information on Demand world, or IOD, and the role of DB2 in Web 2.0. Part 2 will cover the rest of the topics shown in this slide.

Slide 3: Agenda (IBM's direction)

Let's start with IBM's direction and for that we'll talk about some key concepts for IBM's direction, starting with innovation.

Slide 4: IBM's direction – Key concepts

IBM is an innovative company and that is shown by the number of patents that are submitted every year. IBM is normally the first company or the company with the most number of patents per year.

Slide 5: IBM innovation in data servers

Now in terms of data servers, we have that in 1968 we were the first company to deliver a hierarchical database, the first one, which was IMS. In 1981, we were the first company to deliver a relational database, which was DB2, and it was DB2 on the mainframe. We were also the creators of the SQL language. In 2006, we were the first company to deliver what we call a hybrid, or multi-structured database, and that is DB2 starting with version 9. A hybrid database, as we will see later, is a database that can support both XML structures and also relational structures.

Back to Slide 4: IBM's direction – Key concepts

Moving on to the next concept, which is on-demand business.

Slide 6: Today's reality: Complexity

I like to show this chart to talk about the on-demand business. It reflects today's reality, which is complexity. We live in a very complex world with very complex systems and with more and more demanding customers. Systems today need to be up and running 24 x 7. Now, this is the architecture of a system in the industry today; it's a real system. Let's assume that I have to make a change to this system. After I make a change, which maybe took me 5 minutes to do, if I need to wait three weeks or maybe one month to promote the change I did to production, then I would not consider this an on-demand business.

Slide 7: On-demand business

Because an on-demand business is basically a business that is integrated internally and externally and that allows the business to be flexible, and this flexibility allows the business to react quickly to market changes. So that's basically the definition of what an on-demand business is.

Back to Slide 4: IBM's direction – Key concepts

Moving on to the next concept, which is Information on Demand. So, as the name says, it's basically whenever I demand information, I can get it.

Slide 8: IBM's direction: Information as a Service

And Information on Demand is based on providing information as a service. So let's take a look at this slide and let's assume I am the CEO of a supermarket and let's say one of my suppliers at the bottom are using SAP and DB2, and another supplier is using PeopleSoft and Oracle. My internal systems are using an internally developed system, and so on and so on. So what I have are heterogeneous applications and information, and if I want to make money for my company, for my supermarket, probably what I need to do is integrate all of this information, right.

Now, how can I integrate all of this information? Well, I could provide or wrap all of this information as a service. We could use some standards like Web Services, XQuery, GSR170, etc, etc, and once I have wrapped all these applications and information as a service, then I can integrate all of this information. Then I can maybe massage part of the information with other software like master data, Intel, etc, etc, and then pass the massaged data to some other tools and applications processes or people. So, having the information integrated will allow me to achieve what I wanted to do at the beginning, which is to make money, and that can be done by making good decisions and that can be done by obtaining good information.

For example, in the case of a supermarket, this integration will allow me to maybe make a decision as to how I want to put the items in the supermarket in the different aisles. So, for example, I can put beer and diapers in the same aisle because based on this information that has been integrated, that is what the information is telling me. If I put both of these items together, I will be able to sell both products in a larger quantity than if I had them in separate quantities. Intuitively maybe that wouldn't make sense to you, but thanks to this integration of the data, I can make those types of decisions.

Back to Slide 4: IBM's direction – Key concepts

The next concept is Service-Oriented Architecture, or SOA. Also, as you can tell from this name, this is an architecture based on services. So it's not a product, it's a methodology. How to do things based on services.

Slide 9: Why is SOA different?

Now SOA is different because there is loose coupling, which is the opposite of tightness. It is platform and language independent. You can reuse existing technology, for example, many banks use today, still use today, mainframes and COBOL programs and they don't want to convert all of these programs to another platform like Linux, UNIX, or Windows because their programs are very stable, they are very robust, they are working well, so why should a bank spend money to convert them to other platforms?

On the other hand, if they don't convert them, sometimes they feel maybe they are constrained in terms of what they can do with these programs. But now thanks to Web Services and SOA, they can maybe wrap these applications into a Web Service and then make them available this way to other systems that can be Linux, UNIX, and Windows. So that's why we say that SOA allows for reusing existing technology. It's easy and inexpensive. The same idea here; we are reusing existing technology so it will be cheaper. And there is industry momentum, so it's not only IBM saying that SOA is important. There are other companies that are also pushing for this concept or promoting this concept. And everything is based on Web Services, which is based on XML and other industry standards.

Back to Slide 4: IBM's direction – Key concepts

The next concept is Web Services. So this is a service on the web.

Slide 10: What is a Web Service?

Now a simple definition of a Web Service is basically a standard way for an application to call a function over the network, and there's no need to know the location, the platform, or the programming language in which this function was written. And they are fairly easy to create.

Slide 11: DB2 as a Web services consumer: Developing new sales channel

So, a quick example, let's go back to the example of the supermarket and let's say I want to make money very close to Christmas. So I say, "let's buy some cookies." These are Christmas cookies and let's sell them and, let's say, I buy a 100 thousand cookies. Now let's say after Christmas goes by I was only able to sell 60 thousand cookies. So I'm left with 40 thousand. Now, because these are special cookies they will get spoiled very quickly if I don't sell them in a short period of time. So what I can do is, one morning, call different smaller supermarkets and offer them the cookies. So let's say 10 thousand to supermarket A, 5 thousand to supermarket B, etc etc.

But if I take another approach, which is maybe more technical, and assuming that these other supermarkets, and maybe internet companies have Web Services, I could create an application that could be written in JDBC, ODBC, CLI, etc. etc. And in this application, I just insert into a table called "overstock" the 40 thousand cookies, the item that I have, 40 thousand cookies, into this table, and let's say in this table I have created a trigger. So with this trigger, as soon as I insert these 40 thousand cookies I can invoke the trigger. The trigger will be invoking Web Services from e-Bay, and Amazon, and supermarket A, supermarket B, etc. etc. And this trigger is basically putting or making, putting into these companies' systems the fact that, you know, there are 40 thousand cookies that can be sold. So so if I'm a person that is browsing the web, I immediately can see if I'm doing some type of search that these cookies are available and that maybe they are at a cheaper price, they are on sale, let's say I'm looking at amazon.com, and then I can buy them right away from the amazon.com system.

So basically this shows you the power of Web Services, which is basically exchanging information between businesses, exchanging information between my business, which is a supermarket and e-Bay, amazon.com, supermarket A, supermarket B, etc. etc., assuming that these companies have Web Services. So I just developed a new sales channel in a very easy way.

Back to Slide 4: IBM's direction – Key concepts

The next concept is XML.

Slide 12: What is XML?

What is XML? Well, XML stand for extensible market language and the easiest way to explain XML is to compare it with HTML given that many people know HTML. So in HTML we have tags, for example, we have `Raul` and what that tag is telling me is that I want to display Raul in bold. So the tag is describing how to display the data. Now in XML, I will also have some tags but for example here I have a tag called `<name > Raul </name>`. So in this case the tag is actually describing what Raul is, which is a name. So XML is describing the data, while HTML is more describing how to display the data. So that's one of the main differences. Now I can also create another tag that I can call telephone, and another tag that I can call salary, etc. etc. So in a sense I could be creating my own language based on XML, and that's why it's called extensible, because I can extend this language to my own needs. Of course there will some other document which can provide some rules as to what I can put in these tags. This document is called an XML schema and there was also another method called DTD.

Slide 13: XML vs Relational: A schema change made easier...

Now, why is XML important or why should I store information in an XML document and not in the relational model, in tables? Well, there are two main reasons. I can explain why XML may be better than a relational model the first. One is if you are making changes constantly to your schema, to your structure of your system. So let's say I have two tables, department and employee and let's say I have one million rows for the table employee and as you can see there is one column that is storing the telephone number of the employee. Now nowadays, people can have work number, home number, cell phone number, etc. etc. So, my manager, let's say he comes to me he drops by and says, "Raul, you know, we need to store at least also the cell phone number of these employees."

So I could take one approach and say so maybe I can create another column to the employee table between phone and salary and I can call it "cell phone" and I can store the cell phone number there, but that probably is not good because I will be repeating the same information in two different columns, and basically I am not really following the normalization rules of relational databases. So probably what I should do is create a new table, which I can call phone and probably what is missing in the chart is maybe another column called type of phone where I can put that it's a cell phone, work phone number, or home number, and what I will also need to do is move the data from the employee table to the phone table. Now the problem here is it's costly to move the data on one hand, but it's even costlier to to modify all my applications that were using an SQL statement that was pointing just to the employee table. So all the applications that were maybe getting the phone, and using the employee table, need to be modified because now I have the phone information in a different table, which is the phone table; so all of these processes are costly.

Now what happens with XML? In XML , I can have on the left side an XML document where I store the information on different employees and now if I have for Christine another phone number, I can simply add it as another tag in the XML document. For employee Michael, let's say I don't have a home phone number, it's only a work phone number, or the existing home

phone number that he had, and I don't need to add another tag and so on, and so on. And so it's easier to make changes in an XML document. The second reason why it would be better to use XML than relational is for the case where you are storing unstructured or semi-structured type of information.

Slide 14: XPath

The typical example that I give is maybe, let's say is storing DNA information. So let's say in my case if I were to store this in a relational table and I want to store my DNA well I'm a very simple person so maybe I just need 5 columns to store my DNA in a table. Now there may be a very smart person out there and he maybe needs one million columns to store his DNA. So if I want to capture the DNA of all the people in the world or maybe all the people in a given country or city, then maybe I have to create a table with let's say ten million rows, or maybe ten million columns. And you know what happens to my case, well I only had or I only needed 5 columns? So the rest would be null values. So basically if I store this DNA information in a table of one million columns, or ten million columns, I will have a lot of information that is null values and all of this information is sparsed.

So probably a better method or a better model to store this information would be XML. So those were the two cases where XML would be a better choice. The first one as I said is when you are changing your schema constantly, and the second one is when you are storing semi-structured or unstructured type of information. Now this is XPath or this slide shows XPath. XPath is the language that you can use to navigate an XML document. In a relational model we have SQL; in the hierarchical model or XML, we have XPath, and XPath is a subset of XQuery. We'll talk later about XQuery, but this is just a standard language. So it's not that it's just applicable to DB2. It's a standard language and DB2 will use this language to perform some queries that can access XML information. So in this slide you can see on the left side an XML document and on the bottom right side you can see a tree, which is the same thing as the XML document that you can see on the left. So basically these are the same thing but just represented in a different way. On the left side, this is what is called a serialized presentation. On the bottom right side, this is what is called a parsed-hierarchical representation. or representing the XML document as a tree.

Now in XPath, XPath is fairly easy to learn and probably you already know how to work with XPath because you probably know about the **cd** command, about the change directory command. So on Linux, UNIX, and Windows you probably have used **cd** to go from one directory, so you go slash directory, slash sub-directory, slash sub-directory, slash sub-directory, etc etc, right? So when you use the **cd** command what you're doing is basically navigating a tree structure, which is the tree of the directory structure. So in a similar way you can use XPath to navigate an XML document, which is hierarchical and which can be represented as a tree because a tree is hierarchical in nature.

Slide 15: XQuery: The FLWOR Expression

Now as I mentioned before, XQuery is a superset of XPath, or XPath is a subset of XQuery and in XQuery we also support the FLWOR expression. The FLWOR expression stands for: FOR, LET, WHERE, ORDER, RETURN. So this FLWOR expression allows me to perform or to do more manipulation on my XML document. For example, I may have stored my XML document in my XML database in a given format and using a FLWOR expression, and maybe I can

reformat this XML document to a given standard, let's say as an RSS or Atom document.

Slide 16: Why XML?

Now, why is XML so important? Because it is flexible, it is easy to extend, it describes itself. You can transform it to other formats like HTML. It's even platform independent. It doesn't matter if it's in a mainframe in Linux, UNIX, Windows, or Mac. And it's easy to share, and because it's easy to share it is considered the base of Web Services. Web Services, as I mentioned before..., the power of Web Services is that you can exchange information between businesses and because Web Services are based on XML through the WSDL language, the Web Services Description Language, then basically this XML language allows or facilitates the sharing of information.

Slide 17: Who uses XML? Everybody!

Who is using XML today? Everybody! So we have on the left side many languages like for example, FIXML, which stands for Financial Information Exchange Protocol, or Financial Information XML, which is XML where the tags are applicable to financial information, like buying stock, selling stock, etc, etc. Then below, we also have something BSML, which is Bioinformatics Sequence Markup Language. And it's the same idea, it's basically using XML where the tags apply to biological information.

Slide 18: Agenda

So we have basically talked about IBM's direction and now we're going to explain how or where DB2 fits into this IBM's direction. So we're going to move on to the next section, which is the role of DB2 in the Information on Demand world.

Slide 19: IBM's direction

And as I was mentioning, we explained all of these concepts in the previous section for IBM's direction. And probably you're wondering where does DB2 fit in this picture?

Slide 20: The role of DB2 in the Information on Demand world

So all of these concepts depend on one another; on-demand business depends on Information on Demand, which depends on service oriented architecture, which depends on Web Services, which depends on XML, right? So that also shows the importance of XML. Now the question is where do we store XML (documents)? We can store them in files, we can store them in an XML repository, so that means we can develop our own software to just store XML or we recommend you to store them in databases. Now in the case of IBM, it would be DB2, and specifically, we would say DB2 version 9, or starting with version 9, because starting with version 9 DB2 is providing pureXML technology, which is basically storing XML in a native way.

Slide 21: Agenda

So that's the role of DB2 in the Information on Demand world, which is basically to be the repository of XML documents. Now what is the role of DB2 in Web 2.0?

Slide 22: Web 2.0 Tools and technologies: Based on XML

Now what is Web 2.0? A simplest explanation, Web 2.0 is the next generation of the web and Web 2.0 consists on using some tools and techniques that are mentioned here. For example, Ajax, which is, let's say you are using your browser and you're scrolling down and then when you move up again, something at the top may have changed and you didn't have to click on the Refresh button to see the change. So asynchronously, automatically, you see the change happening. The typical example is if you go to financial.yahoo.com and then you put the ticket, the symbol of let's say IBM, and when you click OK, you will see that the price is changing constantly without you clicking the refresh button.

Then you have another technology which is wikis. Typical example is wikipedia, where the community is contributing with different documents. Or, AdSense from Google, where you may get different ads depending on the content of the page. Blog, I'm sure many of you are using blogs to write a diary or your opinion about different things. RSS or Atom: forcing indication of data, this allows for an easier way to distribute information. Mashups: where you can combine different technologies, to develop an application, etc. etc. So all these are Web 2.0 tools and techniques, and they are based mainly on XML. So these technologies are related somehow or another with XML. So again that's showing the importance of XML and again we would say that DB2 is the repository of XML, and that's where DB2 or that would be the role of DB2 in Web 2.0.

Slide 23: In summary

So in summary, the role of DB2 in the Information on Demand world would be to be the repository of XML, which is based on Web Services and other concepts, and under the role of DB2 in Web 2.0 is to be the repository of XML using many Web 2.0 tools and techniques.

Slide 24: DB2 9 is a HYBRID data server: Relational & XML

So, because DB2 supports relational and XML in a native way, we don't call DB2 a relational database any more, or a relational data server anymore, but we call it a hybrid data server.

Slide 25: Agenda

So that was the end of the role of DB2 in Web 2.0 section and we will move on now to the next section, which is "What is pureXML", but that will be covered in the second part of this speech.

Slide 26: To be continued...

Thank you for now and I hope you can move on and start listening to the next part. Thank you.