The case study we'll discuss in this and the next Lab class is reproduced from the book:

- *An EAI Solution using WebSphere Business Integration (V4.1)*
  *Publish Date July 2003*
  *Lead Author: Lee Gavin*
  *Other Authors: Gerd Diederichs, Piotr Golec, Hendrik Greyvenstein, Ken Palmer, Sreekumar Rajagopalan, Arvind Viswanathan*
  *ISBN 0738426547*
  *http://www.redbooks.ibm.com/redbooks/pdfs/sg246849.pdf*

## *The existing IT infrastructure at ITSO Redboats*

When ITSO Redboats was started a number of years ago, the company did not attach a great deal of importance to automating its operations. It had the usual range of legacy mainframe applications used for manufacturing systems and reporting purposes. But the business model was very successful and it soon had to acknowledge that some automation was going to be needed, although management felt that the company was far too small for a "real" IT solution.

Then a "friend of a friend" recommended a small local programming shop that offered a custom-written solution for the company's call center, which, it was said, could grow with ITSO Redboats' requirements. Technical jargon was thrown around, such as Java, J2EE, and, DB2 and Application Server. The people at ITSO Redboats did not really understand any of this, but neither did they care. They were into sailboats, not computers.

When the new application was delivered, they actually quite liked what they saw. There were just a few easy-to-understand screens for the call center operators, which covered the most important functions of their customer-facing work. They could work with customer details and capture orders. Some more functions, such as marketing-related reports, were planned for the future. Looking at the implementation in more detail than the customer was prepared to, we find a J2EE-compliant Java application that uses DB2/UDB as its database, with customer-related and order-related tables as the main components. Once an order was captured, the processing - fulfillment and invoicing - was strictly paper based.

But ITSO Redboats kept growing faster than the owners could ever have dreamed, and it soon became clear to everyone that they were rapidly reaching the stage where a "real" IT solution would be warranted. Soon, an SAP R/3 implementation project was under way to provide support for comprehensive order management, materials management, and eventually, the takeover of the functionality of the existing system, which was now dubbed the "CRM" system, since it was seen as primarily concerned with the customer relationships. But very early in the implementation project, it became clear to the users that they would much prefer the user interface that they already knew and that the CRM application should not be replaced. Instead, they asked, could the two systems be integrated?

In addition, there were some issues being investigated with the "back end", the supply side of the business. This reflected to a degree the fact that ITSO Redboats management wanted to keep their options open as to which components of their sailboats they wanted to manufacture themselves and which they would rather outsource. As a result, they did not feel they could commit to a definitive preference regarding manufacturing or trading systems. They had also learned by now that integration of their business systems with those of their suppliers could be

achieved and would allow much faster, more reliable, and more cost-effective execution of the respective business processes.

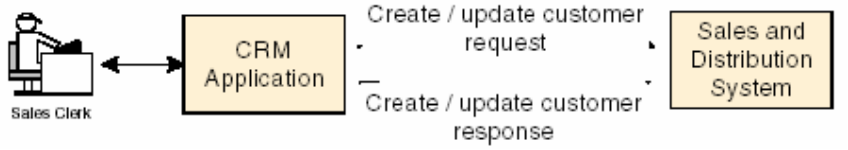
To sum it up, what the ITSO Redboats management wanted was the existing CRM application to remain the front end that would go on driving their business from a customer focus as it had done since it was first introduced. They acknowledged that it would have to be modified to add support for new functions and they were quite happy with that. Any core functionality related to their back-end processes should, however, be implemented according to a "best fit" policy, which preferred existing implementations over packaged solutions and packages over custom implementations. Among packages, a "best of breed" approach would be taken. Integration middleware was seen as the facilitator, the "glue" that would put all the components together.

## Requirement 1: Synchronous application connectivity

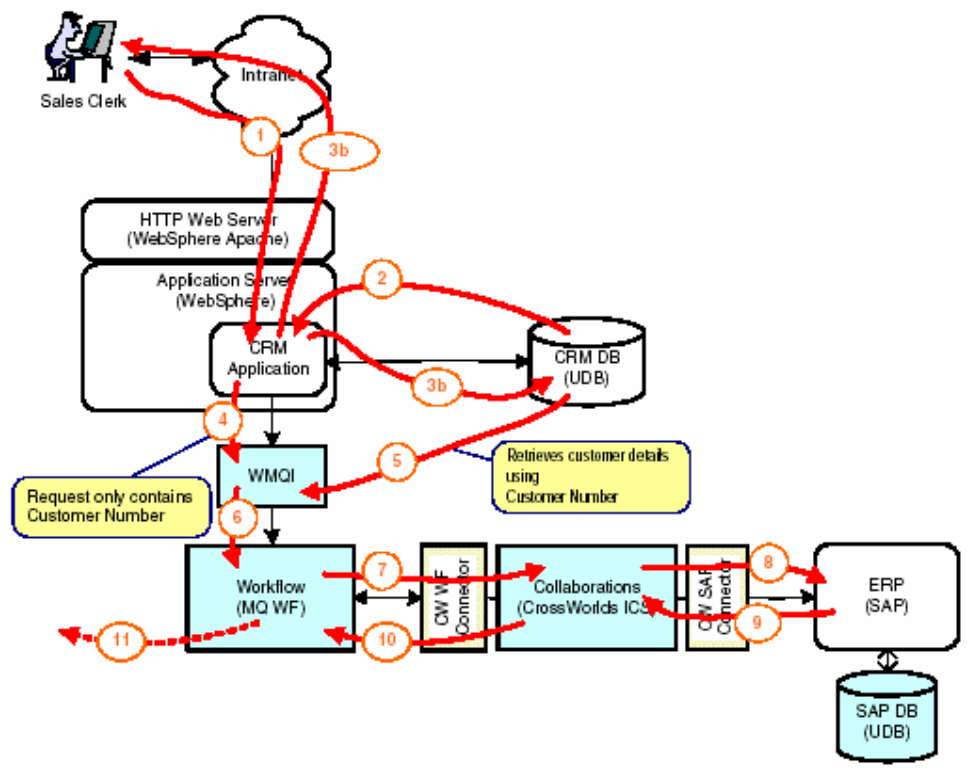
The first phase of the integration project, a much-needed function, a stock level inquiry, internally dubbed "stock check", is implemented. Stock-on-hand figures are not held in the current CRM system. They are, however, held in the new SAP system. The new stock check function will allow the call center staff to query stock levels without having to obtain this information from the SAP system manually. Because this function is crucial for the ever-growing field sales force of ITSO Redboats, it will also be made available as an extranet application, so that sales partners and agents can access it from their offices or mobile computers.


## Requirement 2: Asynchronous application connectivity

The second phase introduces synchronization of customer details between the CRM application and the order management system in SAP. Whenever a call center operator has added a new customer to the CRM database or updated an existing record, this will automatically initiate a "customer synchronization" function, which will carry the new information to the SAP system. This is to ensure that all relevant customer details are always at the latest level in both systems.



## Solution for the use case 'Update customer'




1) The clerk enters the customer number.
2) The system retrieves the customer details.
3) The clerk edits the details on the screen and the system:
   a) Updates the details in the CRM database.
   b) Displays the confirmation to the clerk.
4) The CRM application sends the update request containing the customer number value to WebSphere MQ Integrator.
5) WebSphere MQ Integrator combines that request with the full customer details retrieved from the CRM database.
6) It transforms the message into the UPES XML format before sending it to MQWF
7) MQWF starts a new CustomerCreateUpdate workflow. The workflow sends a request to ICS to perform the customer sync collaboration and starts waiting for the response from the collaboration.
8) The customer sync collaboration sends a service call to update the customer details in the SAP system using the SAP connector.
9) The service call returns from SAP to the collaboration.
10) The collaboration completes by sending a reply to the workflow containing the completion status. The workflow resumes and examines the completion status code returned by from ICS. If the status indicates the successful update, the workflow completes here.
11) If the status is not successful, the workflow can initiate an exception handling process that may involve human intervention.

**Task 1: Describe at the same level of detail as shown for the previous use case the processing and message exchange to be performed for the use case 'Stock check' (don't forget the scenario where sales agents connect from outside the company, using their PDAs, Personal Digital Assistants)**

**Task 2: Based on the whitepaper:**

- *Test-Driven Development in Integration Projects*
  *http://www.enterpriseintegrationpatterns.com/docs/TestDrivenEAI.pdf*

**Develop and explain a testing strategy for the use case 'Update customer'**