Enterprise Application Integration

Christoph Bussler
Digital Enterprise Research Institute (DERI), Ireland

ENTERPRISE APPLICATION INTEGRATION (EAI) TECHNOLOGY

As long as businesses only have one enterprise application or back-end application system, there is no need to share data with any other system in the company. All data that has to be managed is contained within one back-end application system. However, as businesses grow, more back-end application systems find their way into their information technology infrastructure, managing different business data. These back-end application systems are not independent of each other; in general they contain similar business data or are part of the same business processes. This requires their integration for exchanging data between them. The technology that allows this is called enterprise application integration (EAI) technology. EAI technology is able to connect to back-end application systems in order to retrieve and to insert data. Once connected, EAI technology supports the definition of how extracted data is propagated to back-end application systems, solving the integration problem.

BACKGROUND

Typical examples of back-end application systems that are deployed as part of a company’s IT infrastructure are an enterprise resource planning (ERP) system and a manufacturing resource planning (MRP) system. In the general case, different back-end application systems store potentially different data about the same objects, like customers or machine parts. For example, a part might be described in an ERP as well as an MRP system. The reason for the part being described in two different back-end application systems is that different aspects of the same part are managed. In fact, this means that the not necessarily equal representation of the object exists twice, once in every system. If there are more than two systems, then it might be very well the case that the same object is represented several times. Any changes to the object have to be applied in every system that contains the object. And, since this cannot happen simultaneously in the general case, during the period of changing, the same object will be represented differently until the changes have been applied to all representations in all back-end application systems. Furthermore, in most cases there is no record of how many systems represent the same object. It might very well be the case and actually often it is the case that a change is not applied to all objects because it is not known which back-end application system has a representation of the object in the first place.

In summary, the same object can be represented in different back-end application systems, the updates to an object can cause delays and inconsistencies, and locations of object representations can be unknown.

A second use case is that precisely the same object is replicated in different back-end application systems. In this case the update of the object in one system has to be applied to all the other systems that store the same object. The objects are replicas of each other since all have to be updated in the same way so their content is exactly the same. Only when all the objects are updated are they consistent again and the overall status across the back-end application systems is consistent again.

A third use case is that applications participate in common business processes. For example, first a part is being purchased through the ERP system, and upon delivery it is entered and marked as available in the MRP system. The business process behind this is consisting of several steps, namely, purchase a part, receive the part, make the part available, and so on. In this case the back-end application systems do not share common data, but their state depends on the progress of a business process, and it has to update the back-end application systems accordingly. In this sense they share a common business process, each managing the data involved in it.

All three use cases, while looking quite different from each other, have to be implemented by companies in order to keep their business data consistent. EAI technology (Bussler, 2003; Hohpe & Woolf, 2003) allows companies to accomplish this as it provides the necessary functionality as described next.

Enterprise Application Integration Technology

Enterprise application integration technology addresses the various scenarios that have been introduced above by providing the required functionality. In the following, the different functionalities will be introduced step by step. First, EAI technology provides a reliable communication