

The need for Application Intercommunication

Epionet are e-pioneers in the world of rapid business service and enterprise portal¹ development. Unlike traditional consumer portal sites, the *enterprise* portal offers businesses more than just a good start. It provides each user with access to the exact set of **business services** relevant to that user's role.

These services represent an evolution in technology. Traditional e-business solutions allow the existence of logic **islands** of data / business services which hinder process design. Invariably, the inherent bureaucracy between these islands is an obstacle.

What is Application Intercommunication?

Application intercommunication refers to the exchange of information between separate applications, which may be running on separate systems. Providing such intercommunication removes the islands from business process, conserving scarce skills and resources.

Communication between independent business services may take place explicitly (by request or as part of the business logic) or on a more subtle level (communicating system events etc.). The **Epiowave** handles this communication transparently and effectively.

Why is it important for applications to communicate?

The transparent exchange of information between applications allows focus on **business process**. This is a key goal of true e-business portals. It is also just one of Epionet's product features. Hiding the technology from the businessperson / developer means speedy logic design while requiring minimal skills/resources. Epionet's intercommunication facilities can be used in many instances of application design.

Common reasons for application intercommunication:

1. **Triggers for the flagging of business data**
2. Triggers for other internal processing
3. Event handling based on a user request
4. Event handling based not on request, but on system design
5. Cascading events / communicating multiple actions within the system
6. Passing security information
7. Exchanging session and state information ...and many more.

Multi-tier portal applications need to be **integrated, not isolated**. However, these 'plug and play' services must remain independent. For efficient capture of business logic, an independence / isolation compromise must be maintained.

¹A browser-based single entry point for accessing the entire scope of a particular company's business environment that is based on the user's business role.

Where in my system should intercommunication happen?

Intercommunication is necessary on two levels:

1. System use level - finished applications requesting data / functionality
2. Development level – aiding application creation by users within the Epiowave

At the system use level, users may request information that ‘belongs’ to another application. By ‘belong’ we mean that a service may use functionality that exists as a component within another service. A request like this may also be made **by the application itself** as part of the business process, rather than the user.

At the development level the challenge is greater. In user-created applications of the past, there have been many problems with more traditional **business event models**. In particular, dealing with the communication of system events uncovers serious caching and file handling issues. This is because events are handled **within code** that implements a completely different piece of business logic.

The Epionet Solution

The EpioBuilder is a platform for the development of powerful ebusiness web services. Built into the environment (and any EpioBuilder-developed application) is a solid intercommunication mechanism. For example, the EpioBuilder’s **event management model** implements business service intercommunication effectively.

Every application is a stand-alone entity making use of functional **components**. Components ‘talk’ to one another across the Epiowave environment with unprecedented transparency.

In order to simplify the underlying technology for the developer, an **abstraction layer** is provided (see *fig 1.*). The **BSS** (Business Services Server) abstraction layer deals with event data and many varieties of business service intercommunication.

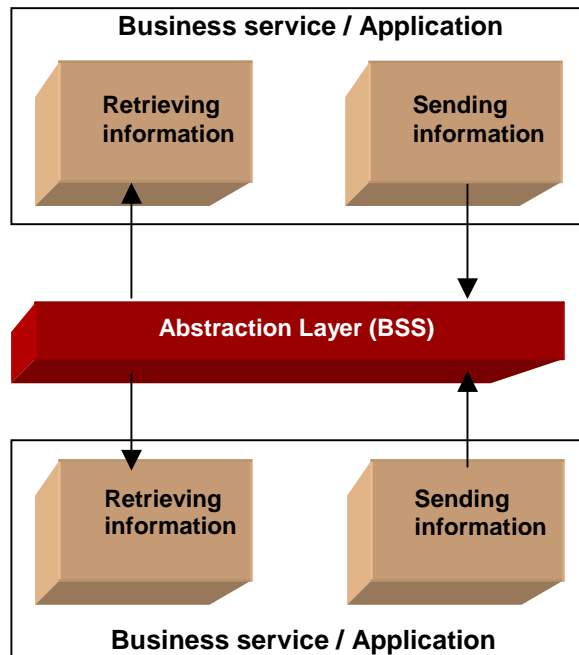


Fig 1. The need for a layer of abstraction between those services which retrieve and transmit information.

The Benefits of Application Intercommunication

Business process', which are enabled with full intercommunication, increase productivity and save on scarce resources. More than application intercommunication, the BSS provides a single exchange interface for different technologies.

Some Epiowave Core services benefiting from application intercommunication

- Metrics – recording and communicating session flow across all applications
- User feedback and session playback
- Managing user roles and privileges - security
- User environment - personalisation, browser detection etc.
- Logging – easy capture of all system activities
- Validation

How is intercommunication incorporated into my system?

Technology

XML/XSL technology provides a mechanism for application intercommunication. XML is the language used to transfer data so that it may be understood by more than one system. XSL is a method of interpreting that language for any data format. There are also standard protocols used to transmit and receive information as XML.

However, unlike the EpioBuilder, these aids do not provide **transparent intercommunication** for the developer. The issue is how to cut through this technological red tape.

How the EpioBuilder contends with the issue:

- Requests and responses are packaged by the BSS
- Metadata for services, parameters etc. is stored and available to the business developer
- The BSS provides components and their methods for simple use. These components may be distributed anywhere on the web
- Acts as a request broker and thus handles business events efficiently
- A suite of events allows total separation of the event triggering from the contextual business logic
- Interface with all components is very simple

EpioBuilder offers more transparency and less developing effort, allowing complete focus on business logic. In the near future, such **transparently captured** business process will not be built into code but chosen from simple menus within the development environment.