Using SAP NetWeaver Process Integration (PI) to ensure the safe transfer of data across security domain boundaries

by Eduard Neuwirt and Manfred Reinart

Eduard Neuwirt
Senior Developer,
SAP AG

Manfred Reinart
Development Architect,
SAP AG

(Full bios appear on page 74.)

In many business areas, including defense, finance, healthcare, and aerospace, an external authority defines security constraints, which may even be statutory. So, fulfilling security requirements becomes crucial for IT projects. Companies safeguard their IT resources behind firewalls to protect their precious data against threats from both inside and outside the organization. To fulfill strict security requirements, different security levels separate systems and networks into domains, depending on the sensitivity of the information that needs to be managed.

Some situations might restrict the channels, protocols, and special authorizations through which information flows. Typically, information flows upward from lower-level domains to more restricted areas without limitation. Transferring information downward from a higher level to less restricted areas, on the other hand, is prohibited by default from a security point of view.

Although permitting data flow from a higher security layer into a lower security layer is necessary in specific situations, it can make "business as usual" impossible from a security point of view. Information from the higher security layer is often essential for processes running in the lower security area. For example, in the banking environment, customer information is one of the most important and secure types of data. It is necessary to locate the data for customer accounts, transactions, and contract details in a high security area. Access to this information from an unauthorized person or organization could lead to serious consequences for the customer and for the bank itself. On the other hand, a bank cannot do its work without a certain number of banking employees having access to portions of this essential, yet confidential data. The solution is to restrict access to the high security data in a well-defined way. The standard answer is to maintain the data in the high security area, but to enable data-handling in the low security area.

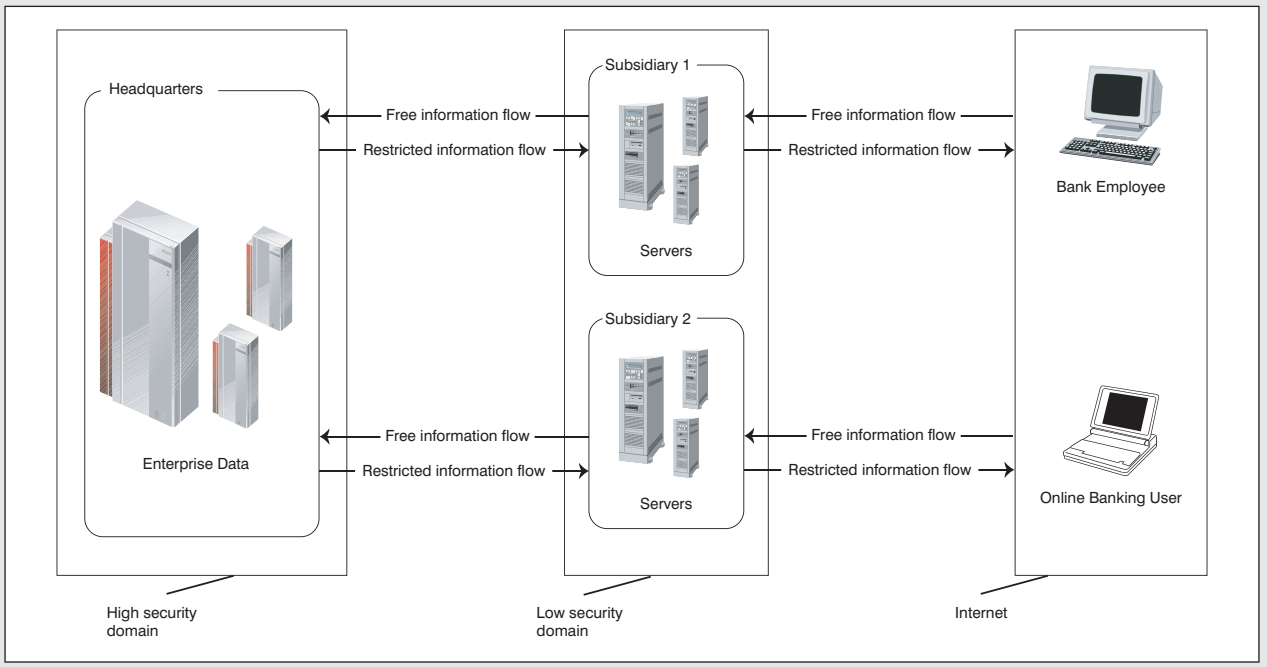

Figure 1 An example of different security domains

Figure 1 illustrates a typical situation for transferring the data between high and low security domains. On the left-hand side, you find a higher security domain. The systems in this domain comprise the heart of the business. Only a few users who work directly on these systems have access to this domain. In the military world, the high security domain would be the area of strategic or tactical decisions. In the financial world, it might be the trading system. On the right-hand side is a low security domain. Most users work in these security domains and perform most of their business activity there. You may access this domain either from a subsidiary, the Internet, or both.

The transfer of information between domains with different security levels is a challenge. Different players will see the same issue from different points of view. The target audience for this article includes security and integration architects, security and SAP NetWeaver administrators, SAP NetWeaver Process Integration (PI)[1] technology consultants, project leaders, and even chief security officers (CSOs).

Note!

The high security domain is often called the "red area"; the low security domain, the "black area"; and the very high security domain, the "purple area." Information flow from the high security domain to the low security domain has strict limitations. You can only send stringently validated (automatically or sometimes manually) data content to the low security domain. Fewer restrictions apply when flowing information in the opposite direction, from a low security domain to a high security domain. In the end, the system in the high security domain will simply ignore the data.

[1] Formerly, SAP NetWeaver Exchange Infrastructure (XI).

The primary goals for each of these players are completely different.

For example, the main objective of a security specialist is to guarantee that the system meets its security requirements. The network security administrator must be able to guarantee network security in planning and production; his or her aim is to provide usable software systems and fulfill security requirements. The goal of the software development project leader may be to implement user-friendly software. The success of the overall project depends on how well the different departments work with one another.

This article shows you how to integrate an SAP ERP system into a distributed environment with different, layered security zones using PI 7.0 to guarantee secure data transfer. To help explain this integration, we use a real-world military data-exchange project we worked on. Obviously, the integration challenge is not military-specific, but any distributed environment with different security layers will require you to master the same challenges.

We begin by explaining a the role of the security gateway, the component that allows data to transfer between high and low security areas. To work efficiently, a security gateway needs rules of authorization, message filtering, and other functions. Next we explore the PI concept of transferring information from an external system to an SAP ERP system. The SAP ERP system runs in the low security domain and the data moves into the high security domain. PI acts as a cross-company or business-to-business (B2B) server and serves as the integration infrastructure. Finally, we touch upon the design (Integration Repository) and configuration (Integration Directory)[2] of PI. We don't cover every aspect of these areas, but we do provide the information and guidance to help you address your security needs.

Prerequisites

PI is the central integration engine in this scenario so it's important to have a basic knowledge of this component. You also need to know about information security. A fundamental knowledge of the security products available for your system may help you to understand some of the technical details of the approach that this article introduces. A basic comprehension of Simple Object Access Protocol (SOAP) — the data-exchange protocol — will also help you to understand the concepts.

Note!

Security is a complex and expansive topic. Our intention is to give you a solid foundation about the intricate part that PI plays in moving data securely from one security domain level to another. To help you gain as much as possible from this article, we recommend that you read the *SAP Professional Journal* articles "A Beginner's Guide to Implementing SAP Exchange Infrastructure (SAP XI) — Paving the Way to Seamless Integration" (March/April 2005) and "A Beginner's Guide to Implementing SAP Exchange Infrastructure (SAP XI) — Designing and Configuring an SAP XI Integration" (May/June 2005), both written by Manish Agarwal. These articles will give you a solid foundation and working knowledge of PI.

The example we use in the article follows the process of sending a damage report to an SAP ERP system from an external system. The external system is a Command and Control Information System (C2IS) and is in the high security area. The SAP ERP

[2] These tools are part of the Integration Builder, which is a Java-based toolset for designing and configuring integration objects in a PI integration scenario. For more information about this development toolset, see the article series listed in the Note on this page.

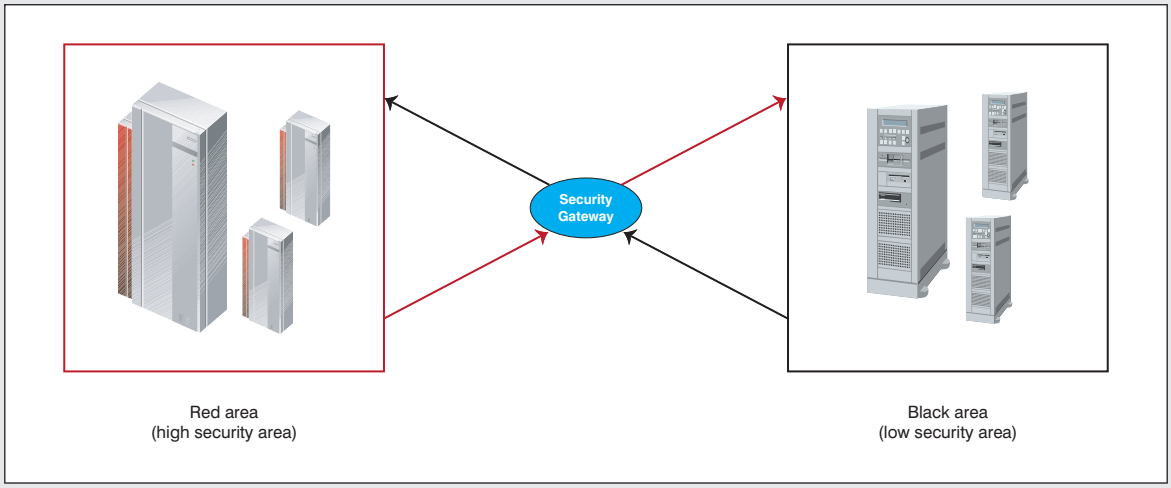

Figure 2 Security gateway as a bridge between different security domains

system is a logistics management system and is always in the low security domain. PI acts as an integration module and transfers the damage report from the external format into SAP ERP.[3] The challenge is to enable data exchange between the two domains without impacting security and functionality issues. To comply with legal requirements, a security gateway serves as a kind of firewall between the two domains. This gateway is a third-party product and its functionality is transparent to us.

Note!

Sometimes, the behavior in which you can send data to the high security area but you can't receive data *from* the high security area is called the "black hole" approach.

[3] For a description of an integration scenario without a security aspect, see our article "Mastering SAP NetWeaver Exchange Infrastructure 7.0: Successfully integrate SAP applications into a non-SAP, non-XML world" (*SAP Professional Journal*, March/April 2007).

The role of the security gateway

Between the two domains lies a "mechanism" that helps to move the data from one domain to another. This mechanism, called a gateway, is pivotal to the communication that takes place between the domains (see **Figure 2**). There are three kinds of gateways: a software solution, a hardware solution, and a combination of the two.

The term "gateway" is one of the most commonly used words in computer science. Almost every component that routes data, controls the flow of data, or acts as a protocol translator is called a gateway, which is misleading. If security isn't a factor in your data flow, the gateway you use will have fewer requirements. A "security gateway," on the other hand, needs the functionality to check message content to determine whether it can securely transfer the content. This description simplifies the function of a security gateway, but this article shows you what makes a security gateway different from another mechanism that just transfers data from one area to another.

The security gateway acts as the decision point for transferring information from a higher security domain into a lower security domain. Deciding upon the preferred solution often depends on the existing security rules for the business. Sometimes, even the vendor and product versions that you use are mandatory (for example, a government contract might require using the security gateways of certified vendors only).

Some security gateways allow a limited set of protocols and endpoints (i.e., the information has to flow via specific gateways). The information can only pass a gateway if the structure and content of the information conform to certain restrictions.

For example, our military project had requirements that we had to meet as we developed a secure environment for transferring non-confidential data from a restricted domain. These requirements included the following:

- The environment had to enable the exchange of information initially sent via HTTP or email with Simple Mail Transfer Protocol (SMTP) as the technical protocol. Only the dedicated sender and the dedicated receiver (authorizations and endpoints) could exchange information. The email format needed to conform to a dedicated military standard (which we described in our March/April 2007 article). It also had to allow the exchange of data for military reports (filtering rules); you would have to filter other data out of the report.
- As with any information traveling from a higher security domain to a lower security domain, you could base the rule for transfer authorization on different aspects of the messages you want to exchange, including the sender, the receiver, the content, or any combination of these.
- There was no direct connectivity between the networks on the Internet protocol (IP) level, so the messages were passed to the security gateway on higher protocol levels (HTTP or SMTP) that would relay them if they passed all authorization checks.

On the application side, proxy settings can enable the functionality of routing messages to the gateway if the application or its runtime environment supports this. For example, with Java-based applications sending SOAP requests via HTTP, it is sufficient to set the runtime properties for HTTP proxy settings. For message transfer via email, things can be a little harder on the client side, because some of the security products involved expect to receive mail messages on nonstandard SMTP ports.

On the content side, messages must conform to a set of criteria — positive or negative — to pass:

- Positive criteria include:
  - Authorized message types
  - Information elements in authorized ranges
  - Catalogs of authorized values (e.g., for names, identifiers, etc.)
  - Ranges of authorized values (e.g., for numeric values, etc.)
  - Valid combinations of filled values
  - Messages or parts thereof must be signed
- Negative criteria include:
  - Encrypted content is refused if the decryption key is not available on security server
  - Information elements not on negative lists (unauthorized content)

A security gateway might only require that a known and accepted authority digitally sign the messages. This authority can be an instance (e.g., a human operator) that checks the message content using a set of criteria and only signs approved messages. In a more automated product, the authorization instance might be another known server that applies checks to the messages automatically. For the military project, we needed a “pure” security gateway that would fulfill the following criteria:

- The gateway must offer the necessary technical functionality to fulfill the requirements.
- The gateway must fulfill the technical requirements, and the German armed forces administration must approve the authorization process. This requirement doesn’t address

anything beyond the security gateway's functionality. Only a few vendors hold the necessary certificates for approval.

- The government must authorize the data-exchange process. Obviously, this point doesn't concern all application scenarios.

Several kinds of products enforce these criteria. They range from security gateways with evaluation and authorization components for a suite of standard communications protocols to pure HTTP traffic-oriented application gateways.

Note!

To avoid any misunderstanding, let's differentiate between a "pure" security gateway and a message broker. The main task of a security gateway is to audit information (several products offer extensive hooks into the Sarbanes-Oxley audit rules). A "message broker" is a program that simply translates a message from the sender's protocol to the receiver's protocol. PI is a typical message broker.

Don't be confused: Today, several synonyms apply to the same functionality. If someone talks about a message broker, B2B server, XML message handler, etc., they are talking about the same functionality.

PI can also act as a simple message-exchange mechanism that just moves information from the sender to the receiver without validating or interacting with the message content in any way. The primary task of PI is to integrate different applications; therefore, it provides the capability for mapping messages (structure and value) and an entire set of adapters for technical data transport, such as remote function call (RFC), file, HTTP, and email.

A wide range of mechanisms that verify whether a message is authorized for transfer across the domain boundary also exist. These mechanisms differ in their grades of automation:

- **Manual check:** A human operator receives messages for transfer in a kind of inbox (e.g., by email). The operator reads the content and eliminates unauthorized parts if necessary, and then decides whether to forward or reject the messages.
- **Semi-automated evaluation:** Verification functionalities, based on criteria that specify allowed or forbidden content, check message content. The human operator, who decides whether to forward or reject the message, assesses the result of this verification.
- **Fully automated verification:** Verification mechanisms, based on the digital signatures expected in the message content for the lower security-level domain, check message content, as well as the sender and receiver. The verification mechanisms may take into account the sender, the receiver, or a combination of both.

One such product, GeNUA's RSGate, implicitly defines the set of authorized message types by using XML schemas in the validation server repository of XML schemas for message types. An administrative procedure that necessitates a digital signature "checks in" the messages. This signature avoids the possible threat of someone tampering with the reference schemas in the validation server.

Now let's look at a situation in which PI serves as a message broker.

Integration scenario

In our defense project, we needed to exchange data between different security areas (the concepts of this successful integration project are described in our March/April 2007 article). The C2IS acted as the data's sender; the SAP ERP system acted as its receiver. Neither back-end application (C2IS or SAP ERP) could provide the correct data format

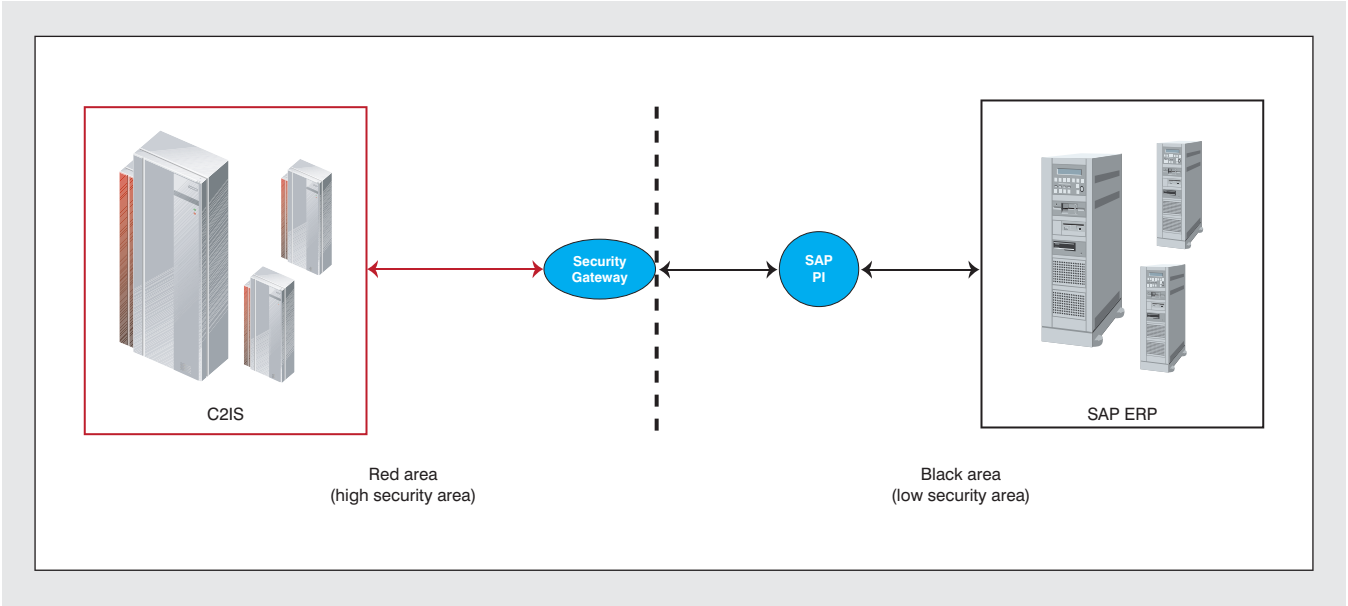

Figure 3 Overview of an integration scenario with SAP NetWeaver PI in the low security area

and content, or "speak" the communications protocol the gateway configuration required. We used PI to convert the data to the protocol and content required to enable its transfer to other security-level domains. We used PI to implement the entire integration logic. PI transformed the original data (a damage report) into SAP ERP format.

Note!

PI doesn't recognize any security issues; it acts only as a process-integration engine. You should consider it to be part of SAP ERP's business logic in this case.

The SAP ERP system received the data and updated its database or started a process, for example, for a maintenance request. The security gateway supported the security part and acted as a checkpoint between the two worlds. This approach enabled us to split the complex scenario into different parts. Each part was independent of one another, so we had control over the entire process (see **Figure 3**).

It's important to understand that you can locate PI in a high security domain, as shown in **Figure 4** (on the next page); in that case no functional difference exists between these two approaches.

Note!

Although you can locate PI in both domains, it doesn't act as a security gateway (see sidebar on page 67). The decision on where to put PI is constraint-free. From a functional point of view, positioning PI in either the low or high security domain doesn't make any difference. We recommend using two PI instances, one in the high security domain and the other in the low security domain, for flexibility; however, performance issues may lead you to use only one instance of PI. In this case, the location of the PI instance doesn't impact the scenario. You can make any necessary adjustments to the configuration in the Integration Directory.

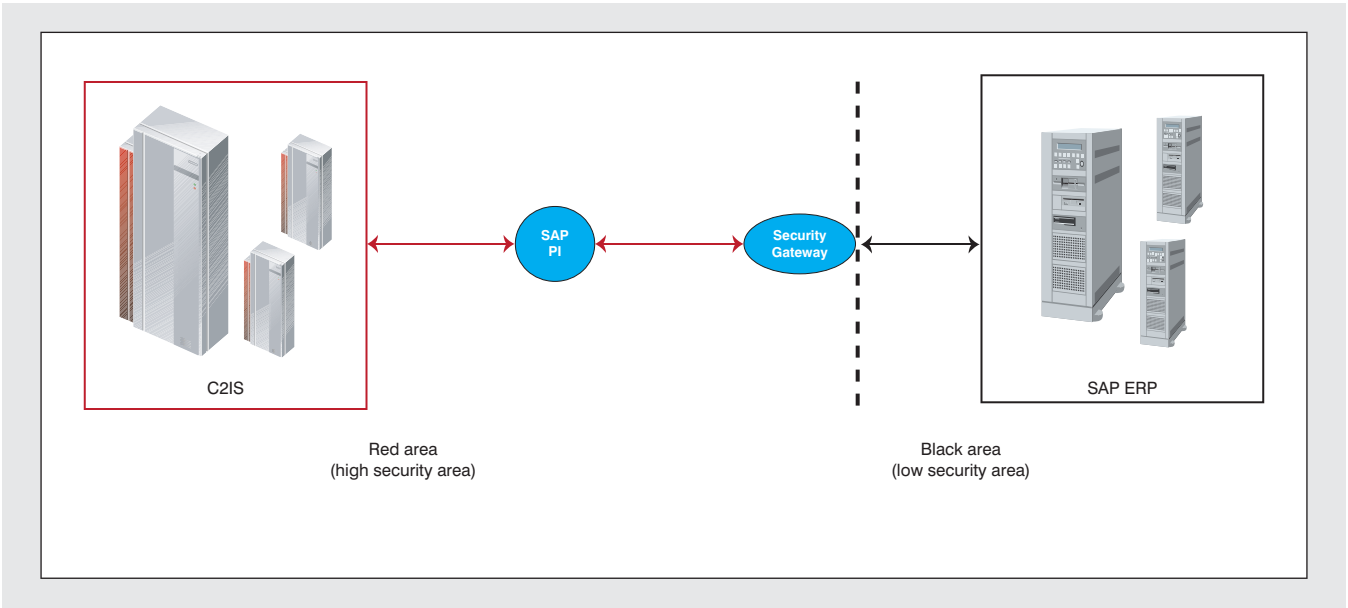

Figure 4 An alternative integration scenario with SAP NetWeaver PI in the high security area

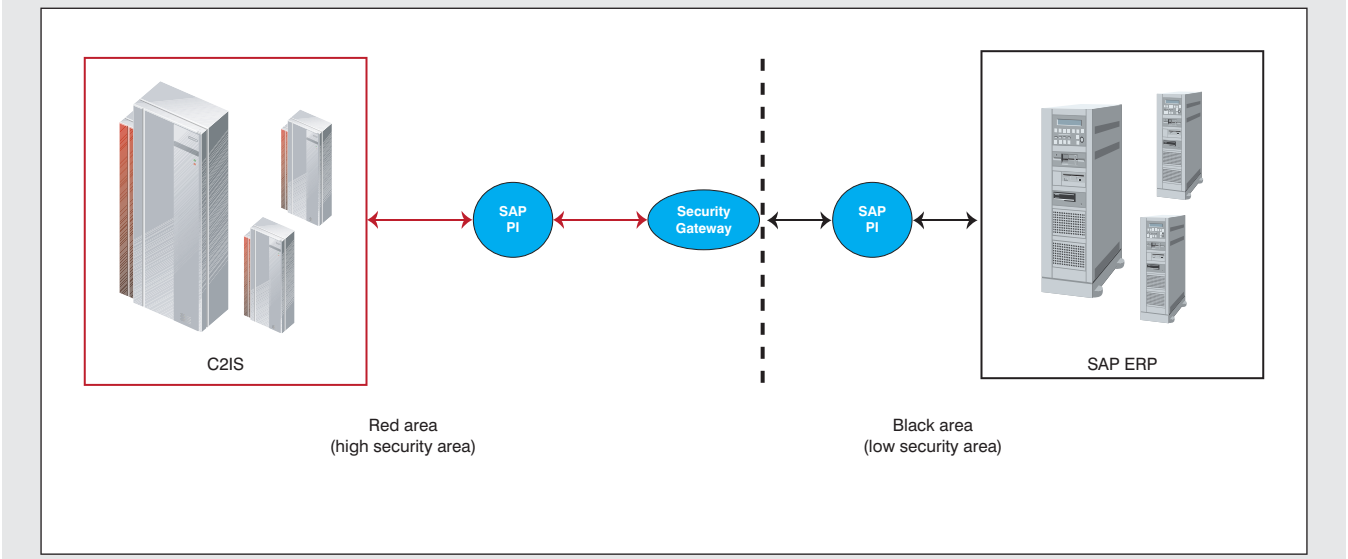

Figure 5 SAP NetWeaver PI in both security areas

As a third alternative, you can use PI in both high and low security domains simultaneously (see **Figure 5**). This solution is the most flexible and enables you to use almost every combination of protocols, products, and communications models.

As an example, let's use a situation that sends a damage report to the SAP ERP system from an external system. The external system is the C2IS and is in the high security area. The SAP ERP system is a logistics management system and is always in the

Could SAP NetWeaver PI act as a security gateway?

PI can produce syntactically correct XML messages (PI is an XML message runtime). PI handles XML messages in the correct protocol; it acts as a simple protocol translator. Linking the external libraries — Java or Extensible Stylesheet Language Transformations (XSLT) — is not a problem. You can also use PI to filter tasks using well-defined message mapping that you can create with Java, XSLT, or PI's message-mapping technology.

But could PI act as a security gateway? Technically, PI offers most of the functionality needed to cover this requirement. However, if you want to use PI as a security gateway, you must make sure that it fulfills all essential legal stipulations before you deploy it in a production environment. Be aware that SAP doesn't endorse PI as a security gateway, so using it will lead to employing highly specialized products in production environments.

Note!

Security gateways form a highly specialized market with very few products (e.g., SNA, RSGate) on the market. This market is not a focus of SAP. SAP offers business applications and an application platform, not security technology. Meanwhile, you can consider PI to be a business integration engine, not a "pure" XML message broker.

low security domain. PI serves as a simple relay on the transmission protocol between the two domains. You can use PI in this same type of scenario for more comprehensive assignments, for example, filtering the incoming and outgoing messages, but for the purposes of this article, let's keep things simple.

The challenge is to enable a data exchange between the two domains without impacting security and functionality issues. To comply with legal requirements, you use a security gateway as a firewall between the two domains. This gateway is a third-party product, and its functionality is transparent to us.

In the ideal case, the sender sends messages with permissible content in all fields according to the enhanced XML schema configured in the validation server. If a message has forbidden content, the server won't forward it across the security domain boundary. From the process side, this is equivalent to the loss of a message in the flow of information because of interruptions in communications. Every remote integration technology has to cope with this even if it involves different security levels.

XML messages must meet the following requirements to be passed from the sender (C2IS in the high security domain) to the receiver (SAP ERP in the low security domain) through the security gateway:

- The security gateway must have an XML schema available for whatever document type the XML message is.
- Beyond the message's purely technical structure, which the application sees when it uses the corresponding message type on service interfaces, further restrictions on the data values in the XML schema are specified. The security gateway's

responsible authorities can freely decide which data to pass and which to reject.

- You must have successful validation of the message document against the security gateway's XML Schema Definition (XSD):
  - The message structure must conform to the schema.
  - The data values in the message structure's fields must fall within the allowed catalogs or the schema's value ranges.

Now, let's look briefly at the design of the PI integration scenario.

PI design: the Integration Repository

To design an integration scenario using PI, you use the Integration Repository tool. We don't describe the entire design process here, but rather demonstrate the process through our example. You can find detailed information on PI design in the two-part *SAP Professional Journal* two-article series by Manish Agarwal mentioned in the Note on page 61. First, let's look at what PI can provide.

Basically, PI can enable service communications across the security gateway — without violating the security rules — in the following situations:

- A participant in the service communication can't "speak" the communications protocol required (e.g., SMTP). You may limit service traffic over the gateway to SMTP as a store-and-forward protocol that doesn't require end-to-end online sessions. In this case, PI can act as a proxy service provider in the higher-level security network (e.g., with a "standard" HTTP/SOAP endpoint) and relay service requests to the lower-level security domain via email. Depending on the protocol capabilities and the endpoint configuration of the service provider in the lower-level security, you can handle the mail message containing the service request directly or use another PI instance in this security domain to retranslate the message to HTTP.
- A participant in the service communication cannot provide correct and acceptable message content that respects all the limitations that the gateway settings impose. PI, as a relaying instance, can adjust the formal aspects of messages, such as the XML schema reference mentioned previously, without changing the message's content. A filter in PI could strip away any unwanted content from service requests or responses while relaying the message, making sure that the gateway receives only valid messages.
- Both cases could occur at the same time.

Continuing with our example, the damage report request goes to PI as an outbound message. After its transformation in PI, the resulting inbound message goes to the SAP ERP system for further processing. To enable the transfer of messages of type `DamageReportRequest`, you need to provide the XML schema for this message type from the service interface (see **Figure 6**).

For the security gateway to accept it, a message schema must contain restrictions for the data values allowed for each data field. Therefore, you need to enhance the original schema that the application provides with field-value catalogs or range definitions on the field level. In **Figure 7**, we show a possible schema extension for the damage report request, which contains a positive list[4] of values.

The security authorities define these restrictions and as a whole the restrictions describe the set of allowed messages. You need to validate the profile of the specific practical restrictions that you can expect in an operational environment against the requirements of the process that the message types support. You should define and set up the enhanced or restricted message schemas in cooperation with the organizations interested in the process and the security authorities.

[4] Positive list means a list of allowed values. The opposite is the negative list, which is a list of forbidden values.


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<xs:schema xmlns:tns="http://defense.sap.com"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://defense.sap.com/" version="1.0">
       <xs:element name="reportDamage"
                    type="tns:reportDamage"/>
       <xs:element name="reportDamageRequest"
                    type="tns:reportDamageRequest"/>
       <xs:complexType name="reportDamage">
              <xs:sequence>
                     <xs:element name="arg0"
                            type="xs:string" minOccurs="0"/>
                     <xs:element name="arg1"
                            type="xs:string" minOccurs="0"/>
                     <xs:element name="arg2"
                            type="xs:string" minOccurs="0"/>
              </xs:sequence>
       </xs:complexType>
       <xs:complexType name="reportDamageRequest"/>
</xs:schema>
```

Figure 6 XML schema for messages of type DamageReportRequest

```
...       <xs:element name="arg0" type="xs:string" minOccurs="0">
             <xs:restriction>
                    <xs:enumeration value="AllowedValue1"/>
                    <xs:enumeration value="AllowedValue2"/>
                    ...
                    <xs:enumeration value="AllowedValueN"/>
             </xs:restriction>
       </xs:element>

...
```

Figure 7 Schema extension for positive values of the element “arg0” for the damage report request

Figure 8 (on the next page) shows the declaration for the request part of the SAP ERP damage report request message. You must provide the values for the following four fields as they apply to the request:

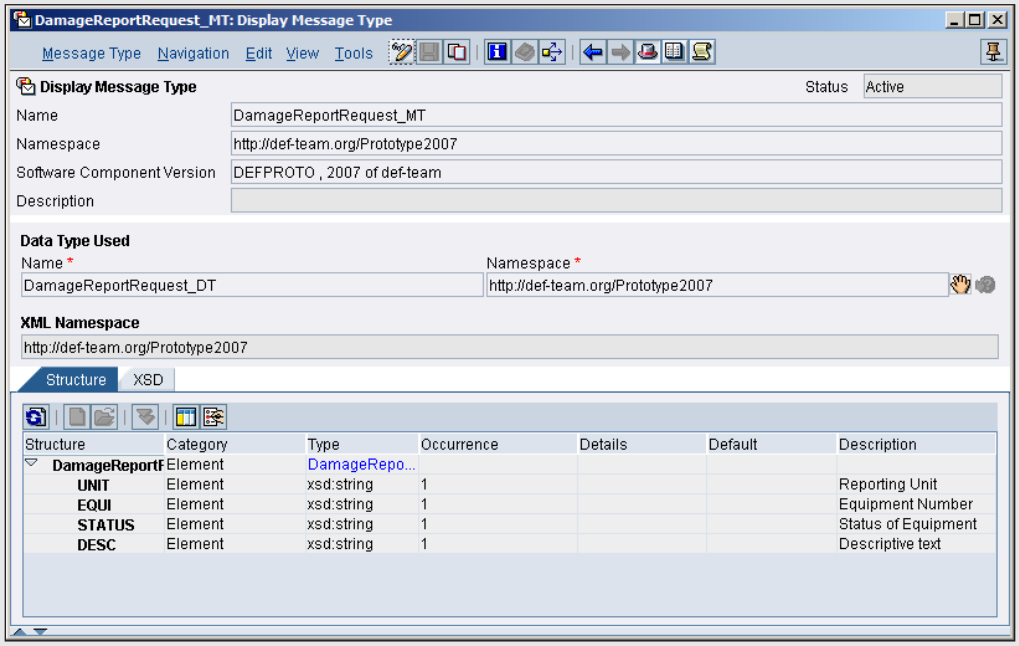

Figure 8 Request part of damage report request message

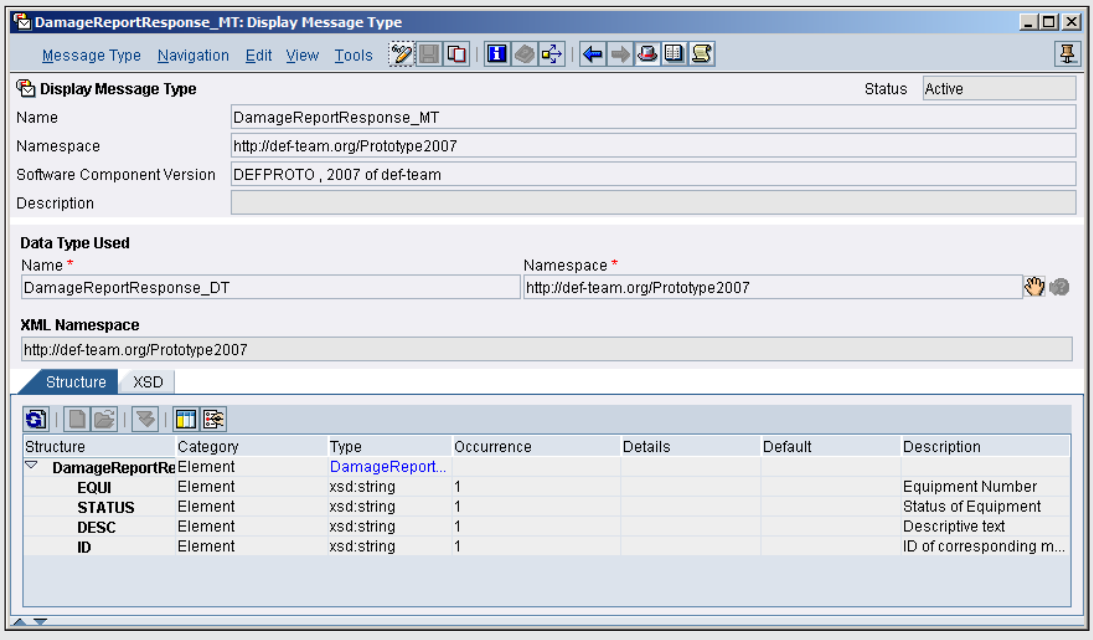

Figure 9 Response part of damage report request message

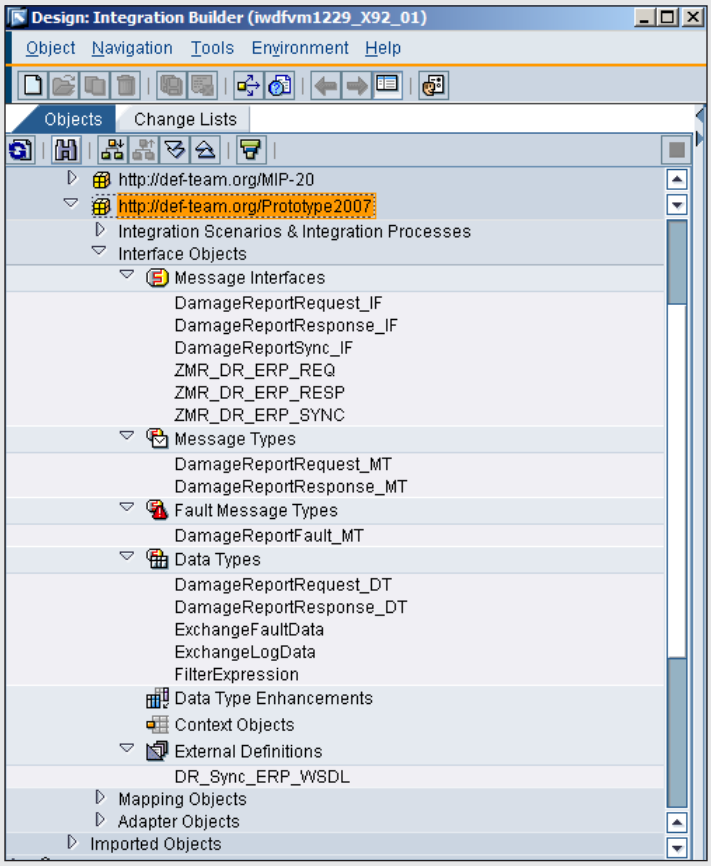

Figure 10 Overview of SAP NetWeaver PI interface objects

- **UNIT:** Identifies the owner of the equipment or reporting unit
- **EQUI:** Represents the damaged equipment
- **STATUS:** Indicates status, for example, operational
- **DESC:** Describes the damage

Figure 9 shows the response part of the external damage report message. The declarations of the request and response parts differ only in the directions. The fields remain the identical.

The necessary declarations of the external damage report request[5] message are available as external Web Service Description Language (WSDL) and XML Schema Definition (XSD) documents. The external authority has to provide these declarations. Therefore, we don't create the WSDL declarations manually, but import them into the Interface Objects folder (see **Figure 10**) in PI.

[5] In keeping with our simple example, we assume that the damage report request has the same format in the SAP ERP system as in the external system. If the formats were different, then PI would transform the external format into SAP ERP format and vice versa.

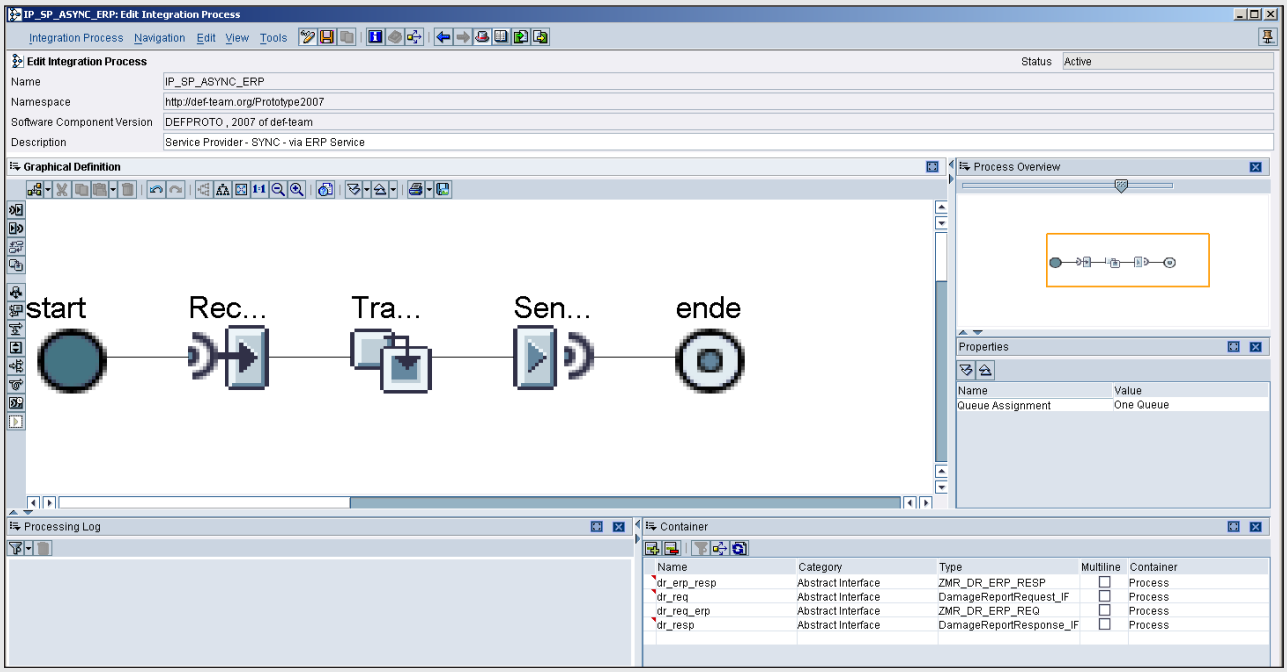

Figure 11 Integration process for an asynchronous receive from the external system and an asynchronous send to the ERP damage report request handling

> ***Note!***
>
> The structure of the damage report may differ from the SAP ERP declaration. If so, the mapping between the two representations may be done directly in PI. For simplicity reasons, we use the same message structure for data exchange.

These declarations enable us to handle all kinds of messages — asynchronous and synchronous.

According to the imported external declarations, we need to create six messages representing the relay service for both sides — the SAP ERP system and the external system (remember the messages have the same format for the sake of simplicity):

- **DamageReportRequest_IF:** This message interface, declared as asynchronous, represents the request part of the external damage report.
- **DamageReportResponse_IF:** This message interface, declared as asynchronous abstract, represents the response part of the external side for the damage report.
- **DamageReportSync_IF:** This message interface, declared as synchronous abstract, represents the entire synchronous damage report with the request and response part.
- **ZMR_DR_ERP_REQ:** This represents the request for the SAP ERP damage report.
- **ZMR_DR_ERP_RESP:** This message interface is the response for the ERP damage report.

- **ZMR_DR_ERP_SYNC:** This represents the synchronous interface on the SAP ERP side with the corresponding request and response types.

To simultaneously handle asynchronous and synchronous messages, we use the integration process technology of PI (sometimes called cross-component Business Process Management, or ccBPM). **Figure 11** shows the integration process that receives an asynchronous request from the external side and sends the data to the SAP ERP system via an asynchronous interface. (We described the handling of synchronous receives and asynchronous sends via Sync-Async-Bridges in our March/April 2007 article.)

Basically, the integration process consists of the following steps:

1. Receive the external outbound message.
2. Transform the external message format into SAP ERP format.
3 Send the message to the SAP ERP system asynchronously.

Our final discussion is on PI configuration, in other words, the Integration Directory that you use to configure the attributes of your integration scenarios.

PI configuration: the Integration Directory

At this stage you need to create the necessary configuration steps in the PI Integration Directory. The PI configuration manifests as a simple SOAP-to-File scenario. The transport layer for the SOAP request is either SMTP (email) or HTTP. We don't describe all the configuration steps here, but you'll find an overview of all the necessary steps in the *SAP Professional Journal* article series by Manish Agarwal mentioned in the Note on page 61.

First, establish the entire system landscape in the form of business services, and then import the integration processes from the Integration Repository and the necessary communications channels.

Furthermore, you need to declare the receiver agreement, sender agreement, and interface determination for the integration scenario. Several *SAP Professional Journal* articles in the past have covered this topic, so we won't bore you with the details. You will find, however, in addition to the other articles mentioned in this article, valuable information in the following *SAP Professional Journal* articles: "Extend the Internal and External Reach of Your Applications with ABAP-Based Web Services" by Arthur Wirthensohn (July/August 2005) and "Easily Integrate Unstructured and Semi-Structured Data into SAP NetWeaver Process Integration (Formerly XI) Using the Conversion Agent" by Prasad Illapani (July/August 2007).

Conclusion

Products for crossing security domain boundaries with security gateway services exist on the market and are becoming accredited for productive use. Technical restrictions (e.g., on the communications protocol level that these kinds of gateways impose), as well as the rules defined for operation, would normally exclude standard service consumers or providers from practical scenarios.

PI can easily act as a flexible and transparent integration platform that helps to overcome these restrictions. It can provide additional means of security monitoring or act as a message-filtering engine. Even though PI fulfills the technical requirements for a security gateway, it can't act as a security gateway because of the legal criteria and missing accreditations necessary for this role.

The combination of PI used for integration approaches and a security gateway as a bridge between high security and low security domains enables a seamless integration of SAP systems, such as SAP ERP, in cross-security domains without restriction. So, not only can you send data from low-level security areas to high-level security areas, the reverse is also possible, enabling you to keep your business processes running smoothly without compromising the security of your data.

Eduard Neuwirt joined SAP in 1999 and became a member of the ABAP Connectivity group, where he worked on the design, tools, and rollout of the ABAP communications infrastructure. Eduard was also responsible for the development of the remote function call (RFC) tools on the external side, including the RFC library and JRFC. Since September 2005, Eduard has worked for the SAP Defense and Public Security Department. He is responsible for the interfaces to external military non-SAP systems. You may reach him at eduard.neuwirt@sap.com.

Manfred Reinart studied computer science as part of his career in the German Armed Forces. He worked as an IT teacher and information management officer at the headquarters of a multinational organization. He was also involved in the development of a Command and Control Information System and worked for a CORBA producer. After he joined SAP in 2003, he participated in the design of a distributed system landscape architecture for the support of armed forces in deployed operations. Since 2004 Manfred has been responsible for the development of interoperability functionalities to non-SAP defense information systems. You may reach him at manfred.reinart@sap.com.

