Interface Strategy for Custom Development

Applies to:
SAP Custom Development and Interface Developments.

Summary
The Client application portfolio will consist of a set of system components configured to operate together in support of standard business processes. Wherever possible, these components will be elements of the mySAP.com product suite. In specific cases, we will incorporate non-SAP applications, either commercial software products or Client custom applications, to provide different or additional functionality.

These application components must be integrated at the data or transaction level. The heterogeneous, distributed application landscape necessitates the configuration or construction of interfaces to allow the required exchange of data.

The Interface Technical Strategy covers the technology aspects of interfaces, mainly the integration of SAP and non-SAP applications. The focus is on our approach to different interface types, connectivity, routing and data transformation. A set of basic principles is set forth to guide the selection of interfacing tools as well as the design and construction of interfaces.

The document covers at a high level our technical direction with regard to all forms of data exchange – application-to-application interfaces, database conversion and loading, EDI, Web-based interfaces, etc. More detailed strategy documents will be delivered in some of these areas. Among the key assumptions are that we will deploy the minimum number of different tools and technologies, scalable, flexible and preferably familiar to the Client.

The boundary of the Client interface architecture is establishing a set of well-defined points of integration. Beyond this boundary, the Markets will be responsible for connecting their own systems to the Client solution through these standard integration points.

Author(s): Aveek Ghose
Company: IBM India
Created on: 30 August 2007

Author Bio
Aveek Ghose has 12 years of IT experience and has worked across the globe in SAP Implementations. Aveek has an MS in Information Systems from George Mason University in USA and a MS in Economics from Virginia Tech in USA and is currently working for IBM India.
Table of Contents

Executive Summary .......................................................................................................................... 3
Purpose ............................................................................................................................................... 3
Scope .................................................................................................................................................. 4
Assumptions ....................................................................................................................................... 5
Interface Definition and Guiding Principles ...................................................................................... 6
  Interface Definition .......................................................................................................................... 6
  Guiding Principles: ............................................................................................................................ 6
Interface Communication Types ........................................................................................................ 6
  Synchronous .................................................................................................................................... 6
  Asynchronous ................................................................................................................................ 7
Interface Usage Types ........................................................................................................................ 7
  SAP to SAP ..................................................................................................................................... 7
  SAP to Client complementary applications ...................................................................................... 7
  SAP to Market applications .............................................................................................................. 7
  Temporary Interfaces ....................................................................................................................... 8
  EDI ................................................................................................................................................... 8
  Web .................................................................................................................................................. 8
  Data conversion ............................................................................................................................... 8
Basic Design ....................................................................................................................................... 9
  Plug-in Point .................................................................................................................................... 9
  Protocols ........................................................................................................................................ 9
  Connectivity ..................................................................................................................................... 9
    Message-based................................................................................................................................. 10
    File Transfer .................................................................................................................................. 10
    Web Services ................................................................................................................................. 10
    VAN (Value Added Network) ........................................................................................................ 10
Technical Standards .......................................................................................................................... 10
Analysis of specific topics .................................................................................................................. 11
  Code Page ...................................................................................................................................... 11
  Flow Control .................................................................................................................................. 11
    Process Logic & Dependencies ..................................................................................................... 11
    Sequencing .................................................................................................................................... 11
    Logical Unit of Work ...................................................................................................................... 11
    Ack/Nack ...................................................................................................................................... 12
    Routing .......................................................................................................................................... 12
  Status Control .................................................................................................................................. 13
    Error Handling ............................................................................................................................... 13
    Tracking ......................................................................................................................................... 13
    Status Reporting ........................................................................................................................... 13
    Auditing .......................................................................................................................................... 13
    Archiving ....................................................................................................................................... 13
    Reprocessing................................................................................................................................. 14
    Data Transformation ..................................................................................................................... 14
    Security ......................................................................................................................................... 15
Development Responsibilities ............................................................................................................ 16
Glossary .............................................................................................................................................. 17
References .......................................................................................................................................... 19
Disclaimer and Liability Notice ......................................................................................................... 20
Executive Summary

The CLIENT application portfolio will consist of a set of system components configured to operate together in support of standard business processes. Wherever possible, these components will be elements of the mySAP.com product suite. In specific cases, we will incorporate non-SAP applications, either commercial software products or Client custom applications, to provide different or additional functionality.

These application components must be integrated at the data or transaction level. The heterogeneous, distributed application landscape necessitates the configuration or construction of interfaces to allow the required exchange of data.

The Interface Technical Strategy covers the technology aspects of interfaces, mainly the integration of SAP and non-SAP applications. The focus is on our approach to different interface types, connectivity, routing and data transformation. A set of basic principles is set forth to guide the selection of interfacing tools as well as the design and construction of interfaces.

The document covers at a high level our technical direction with regard to all forms of data exchange – application-to-application interfaces, database conversion and loading, EDI, Web-based interfaces, etc. More detailed strategy documents will be delivered in some of these areas. Among the key assumptions are that we will deploy the minimum number of different tools and technologies, scalable, flexible and preferably familiar to Client.

The boundary of the CLIENT interface architecture is establishing a set of well-defined points of integration. Beyond this boundary, the Markets will be responsible for connecting their own systems to the CLIENT solution through these standard integration points.

Purpose

The Interface Technical Strategy will provide a framework for making decisions with regard to the design of CLIENT interfaces and the selection of technologies to support them. It provides some general background on interfacing options and discusses the direction that we expect to take for CLIENT interfaces. The interfaces discussed in this are specific to SAP. There is no intention to consider interfaces between two non-SAP applications.

This strategy covers all data movement in and out of mySAP components, but much of the focus is on interfacing that will require the use of additional tools. The document provides a set of basic principles at a relatively high level. Related technical strategy documents will be delivered that expand these principals in specific areas (middleware, EDI, database conversion, etc.)

It is important that the design and construction of CLIENT interfaces be performed in a common manner using standard protocols and tools. This is necessary for achieving the CLIENT vision of common systems and also for meeting the aggressive CLIENT timeline. Deviation from the principles defined here represents exception to the standard strategy. Such exceptions will require compelling justification and the approval of the IS/IT Workstream Lead.
Scope

The scope of the Interface Strategy is driven by the amount of functionality provided by the SAP environment. The Interface Strategy covers the technology aspects of SAP interfaces, mainly the integration of SAP and non-SAP applications.

These areas are IN SCOPE:

- Establish standard terminology.
- Describe protocols and techniques available for interfacing with mySAP.com.
- Define the basic categories of the expected CLIENT interfaces.
- Outline the use of external middleware for supporting interfaces.
- Define location for process logic, i.e. where logic will reside in a technical sense.
- Define location for data mapping and transformation.
- Describe connectivity between applications.
- Establish approach to transaction sequencing enforcement.
- Establish approach to interface dependencies
- Establish approach to receiving application acknowledgement.
- Describe support for tracking, error notification and re-introduction of data messages.
- Define support for data routing.
- All data movement in, out and between SAP components is in scope.

The following areas are OUT OF SCOPE for this technical strategy document:

- Data flow in the context of business processes and functionality (GT, DS/DM)
- Physical landscape for SAP (IS Application Architecture)
- Analysis of Market applications requiring CLIENT integration (Implementation)
- Gap analysis and approval process.
- Custom Development guidelines.
- Organization roles and responsibilities.
- Workflow.
- Non-interface development, including ABAP, Workplace, mini-apps, reports.
- Master Data cleansing.
- Language strategy.
- Capacity planning and sizing.
Assumptions
This technical strategy is based on the following key assumptions. Should these assumptions change or prove to be invalid, the strategy will have to be adjusted accordingly.

- Essentially all application integration will involve at least one of the mySAP.com components. There will be little, if any, direct integration between two non-SAP applications.
- It is preferable to deploy the minimum number of tools and technologies.
- One of SAP's strengths, and one reason for its adoption, is its highly integrated nature. Incorporating application logic in the middleware would dilute this benefit.
- The universe of Client applications will be well known and relatively static. It will be possible to easily identify individual data/transaction requirements (no requirement for publish/subscribe or other dynamic data distribution).
- It is preferable to adopt tools and methods familiar to Client wherever practical.
- The localization of the CLIENT Template will include custom development in SAP and in Market applications to support local interfaces.
- The software technology that we select for our technical architecture will need to be scalable and flexible.
- A common toolset will be able to support interface requirements for all CLIENT implementations.
- Networks are reliable and provide sufficient throughput.
- Resources will be available to support the chosen tools and technology.

This is the high level view of the application interface environment.
Interface Definition and Guiding Principles

Interface Definition

The term “interface” has different meanings according to the context. In the general business systems sense, an interface typically refers to the sharing of related business data between different applications. Examples are order information, inventory information or finance information. The interface might include both original data and updates, and might also cover responses or acknowledgements.

For the purposes of this technical strategy document, we will consider “interface” to have the following more restrictive meaning.

Definition: An interface is the transmission of data from a source application to one or more receiving applications. The transmission is in one direction and the structure/type/format of the source data is always the same for a given interface.

Note – This definition does not include delivery of reports or data to individual desktops.

Guiding Principles:

- Common tools, data and design will be used wherever possible.
- For long-term interfaces, converting applications to use SAP data values will minimize the need for cross-reference data (and its maintenance).
- IDoc structure is the preferred format for exchanging data with SAP.
- Data transformation should be done in middleware whenever feasible.
- Business logic will reside within the business applications. Middleware will be used only for transformation, routing, tracking and delivery of data.
- Point-to-point interfaces should be avoided.
- Wherever possible, the mySAP.com solution is preferred over other technical options.
- Data should be actively pushed from sender to receiver, on a schedule controlled by the sender. Pulling data from the receiver side should be avoided.
- The boundary of the CLIENT interface architecture will be to provide a well-defined point of integration with corresponding standards. Beyond this boundary, the Markets are free to connect their own systems to these interfaces as they see fit.

Interface Communication Types

Communication types can be broadly categorized into synchronous and asynchronous. Within the categories there are some further useful subdivisions.

Synchronous

In a technical sense, a synchronous interface requires that two applications establish and maintain an open connection while corresponding data changes are made on both sides.

Real-Time

The common perspective of a real-time interface is that changing data in one application results in corresponding changes in another application with no visible lag time. In the context of this document, a real-time interface is essentially a synchronous interface.

SAP supports real-time interfaces, for instance through the use of BAPI or tRFC (transactional Remote Function Call) connections. At this point, we do not anticipate any true real-time interfaces as part of the CLIENT solution, with the exception of the standard SAP-to-SAP data flow and possibly some bolt-ons. If requirements for real-time interfaces are raised at a later point, they will be treated as special cases since they imply direct application-to-application connections and cannot be solved in a generic way.

Bolt-on

A bolt-on is non-SAP code that is completely integrated within a SAP transaction via tRFC. In other words, during the execution of the transaction some additional logic is applied that is provided by custom coding or a third-party commercial application. This is not precisely an interface according to our narrow definition, but it is included for completeness.
Asynchronous

In an asynchronous interface, the sending and receiving of data are two distinct operations. The applications do not maintain a connection while the data is transmitted between them. There is always some degree of lag time although it might be very short. Batch and near-real-time interfaces represent the two extremes of asynchronous interfaces. The following descriptions indicate typical characteristics rather than rigorous definitions.

Near-Real-Time

For near-real-time interfaces, the lag between generating, transmitting and receiving the data is kept as short as practical. For individual interfaces, there is generally an upper limit on the acceptable lag time that is dictated by business process requirements. We assume that a typical limit for CLIENT interfaces of this type might be several minutes. Specific interfaces may have more or less stringent requirements.

Near-real-time interfaces will be supported through common middleware and messaging software. On the SAP side, they will typically use the IDoc format and the ALE subsystem. It is expected that the majority of CLIENT interfaces will fall into this category.

Batch

A classic batch interface involves the extraction of data from one application into a file, transfer of the file and loading of the data into the receiving application. It is commonly executed on a predefined schedule, but can be event-driven. Sufficient time must elapse between executions of a batch interface to avoid overwriting files or overlapping processing.

SAP inbound batch interfaces will be supported by ABAP programs using the techniques of batch input or call transaction in order to update SAP tables. For some business processes, standard programs using direct input (direct update on SAP tables vs. data processing simulating a dialogue transaction as in batch input / call transaction) are provided.

Outbound batch interfaces require an ABAP program extracting the file.

Batch interfaces are generally only to be implemented where there is no standard IDoc type available since batch interfaces have no standardized monitoring. Nonetheless, it is expected that the CLIENT implementation will involve a significant number of batch interfaces. Batch interfaces can still make use of the middleware functionality for data transformation, routing, etc.

Interface Usage Types

SAP to SAP

MySAP.com is an integrated product and SAP provides standard protocols for communication among the mySAP.com components. These protocols have evolved over time. Some of them are very stable and well established. Others, particularly related to the New Dimension products, are in a state of flux as SAP moves to a more distributed, internet-based model. Internal data synchronization is controlled via configuration and generally does not require special development or extra tools.

The classical solution of exchanging data between two SAP core systems is an ALE coupling. By means of data distribution via ALE, both synchronous (BAPIs) and asynchronous interfaces can be established. Since the data formats in such an exchange are identical (with the only possible exception of different release versions), these interfaces are expected to have no requirement for external middleware.

SAP to Client complementary applications

Client complementary applications become part of the CLIENT Template solution because they represent necessary functionality required in all or most Markets that is not provided by SAP. Interfaces to these applications will be supported by the standard CLIENT technology (middleware, etc.) Because they represent permanent applications, it might be justifiable to expend the development effort to make them as integrated as possible with SAP. For example, they could be made more directly compatible with SAP data formats and less dependent on middleware transformation, cross-references, etc...

SAP to Market applications

Market applications provide business functionality required only at the local level. As the scope of the CLIENT Template functionality expands with each version, it is expected that the number of Market applications will trend downward. The interfaces to these applications will be supported by the standard CLIENT technology (middleware, etc.) These interfaces will be constructed during implementation with the necessary developments in the Market applications being performed by Market IS.
Temporary Interfaces

Temporary interfaces are constructed only for phased implementations. For example, all factory locations might not receive the CLIENT solution at the same time. The non-CLIENT sites will require interfaces for a short time until CLIENT arrives. We will take the simplest possible approach to these interfaces, including manual workarounds, spreadsheet uploads, etc.

EDI

EDI stands for Electronic Data Interchange, and refers to the exchange of business transactions between a company and one or more of its business partners using a standard predefined format.

In the context of a SAP implementation there are two distinct activities that need to take place in order to complete the setup of an EDI subsystem:

SAP configuration and development

The configuration side is a business task, whereby business partners are defined and configured as EDI partners, and triggers for the generation of EDI messages are configured in the business transactions. The development side refers to any additional processing that is required during the generation of the SAP EDI output, such as for instance the recalculation of prices to avoid sending net prices or discount levels. The output of an EDI-enabled transaction is in the form of a SAP IDoc.

Translation and transmission

The EDI IDocs generated by SAP do not fully conform to the EDI standards. There is still the need of using an additional tool called an EDI translator, which maps the IDoc into the standard EDI formats and routes it to the business partners through a VAN (Value Added Network).

After the implementation of the pilot markets of Template Version 1, CLIENT will select a standard EDI translator tool, which will be deployed as part of the future versions of the template. CLIENT will provide standard mappings between the SAP IDocs and the EDI formats used within the markets, as well as standard guidelines for the configuration of market business partners and assistance through the implementation phase.

Web

The CLIENT solution will include interfaces between SAP and web-based applications or business partners using the Internet as communication channel. One example is integration with Internet Marketplaces. A tool that is specialized for this type of integration will support these interfaces. This tool could either be provided by SAP directly, or will be a third party tool selected after evaluation by CLIENT. The web side of the interface will be based on XML format. On the SAP side the integration is typically supported by BAPIs.

Data conversion

Data conversion represents a specialized instance of a batch interface that provides the mechanism for formatting and loading large volumes of Master Data, historical business data, etc. These interfaces are typically scheduled less regularly than other interfaces; often they executed manually. We will support data conversion with specialized tools that provide the necessary performance and database logic functionality. The same tools could be applied to any large batch files if necessary.
Basic Design

Plug-in Point

A plug-in point is an interface that is expected to be similar in all Markets, although it may interface to different systems. An example could be an interface to receive payroll values: it is expected that some Markets will have a non-CLIENT payroll system; however the CLIENT SAP system will need to receive the same information (e.g. personnel number, amounts, cost centers) from each and every one of the payroll systems.

In the case of an inbound plug-in point it is expected that CLIENT will define the format of the data required, and will leave it up to the implemented Markets to define their standards to populate the interface with data inbound into CLIENT.

In the case of an outbound interface, CLIENT will make available all the data elements that are in a standard or custom IDoc, and leave it to the implemented Markets to define how to best use them.

In both cases, CLIENT should have one and only one extract or load program, and make it a Market responsibility to adapt the data so that it is consistent with the CLIENT provided formats. It is possible that in some cases the standard CLIENT format will be a generic version of an EDI structure, however it would not be specific to any particular business partner.

Protocols

The SAP interface techniques and its usage are described in more detail in the document SAP Components Integration protocols.

Relevant SAP Protocols are:

**ALE (Application Link Enabling)** provides a mechanism for integrating business processes in a distributed system environment. ALE enables business-controlled asynchronous and synchronous messaging between distributed applications on loosely or narrowly coupled SAP or non-SAP business solutions. An important application of ALE is the replication of master data and transactional data.

**BAPI (Business Application Programming Interfaces)** is a method for accessing business functionality. Based on the object-oriented view of the business objects within mySAP.com components, BAPIs are the methods to create, access, and change these business objects.

**RFC (Remote Function Call)** is a SAP interface protocol simplifying the programming of communication processes between systems. RFCs enable to call and execute predefined functions. Types of RFC include the tRFC (transactional RFC).

**IDoc (Intermediate Document)** is the SAP standard format for electronic data interchange between systems.

The IDoc type indicates the SAP format that is to be used to transfer the data for a business transaction. An IDoc is a real business process in the form of an IDoc type. An IDoc type can transfer several message types: These are the "logical messages" that correspond to different business processes.

An IDoc type is described using the following components:

- A control record: the format of the control record is identical for all IDoc types.
- One or more data records: a data record consists of a fixed administration part and a data part (segment). The number and format of the segments can be different for each IDoc type.
- Status records: the status records describe the processing stages that an IDoc can pass through and have an identical format for each IDoc type.

Connectivity

Connectivity for CLIENT data movement will be enabled by common, standard technologies and products. The toolset will be broad enough to support all necessary platforms and applications, but CLIENT will not support all possible connection technologies.
Message-based

Much of the connectivity between applications and components will be based on messaging. Messages represent information flows containing both core data and control information. Messages are transferred by a message-queuing system with guaranteed delivery based on a store-and-forward model. If necessary, dedicated channels for specific messages and message types can be provided e.g. to separate the transfer of large data sets and small transaction sets.

The CLIENT messaging tool is expected to be IBM MQSeries due to the clear market leadership of this product and Client Market experience.

File Transfer

File transfer refers to the functionality of moving a file from one system to another ensuring that the file is delivered correctly and completely. Additional requirements of a file transfer protocol are the encryption of data and the security that surrounds its execution both on the sending and on the receiving file systems.

File transfer tools need to be able to operate across different operating systems, for instance transferring from NT to UNIX or MVS. If possible, the messaging tool described earlier will also be used to handle file transfers.

Web Services

Web services provide access through the web to SAP applications and if necessary to non-SAP applications. See section 5.4.6.

VAN (Value Added Network)

VANs are infrastructure put in place to guarantee a private dedicated connection between business partners. In general VANs are made available by large infrastructure providers (e.g. IBM or GE) and are not used continuously, but are available on demand. A typical example for the use of a VAN is EDI, which uses a VAN as a secure and private mechanism to connect two business partners without needing an open connection at all times. For the purpose of this strategy we will assume that the markets will define and implement their own VAN requirements without involvement from CLIENT.

Technical Standards

Technical standards provide guidelines on how interfaces will be developed. Such guidelines will be provided in a separate document to be developed by the Custom Development group within IS. In addition, the Development Support team will prepare standards specific to the technical setup of relevant tools.
### Analysis of specific topics

#### Code Page

Different countries use different character sets for special characters in their languages (like French é, à; German ö, â, ü etc.). The code pages with character sets of European languages use one byte for each character. For some Asian countries, such as Japan that uses Kanji, the code page used by SAP requires two bytes for every character. Multiple code pages cannot be used in the same SAP system; therefore the decision over code pages has a significant implication on the overall system architecture.

The interface strategy does not address code page, particularly does not provide support for transformations from one character set to another. CLIENT’s language strategy should provide additional guidelines for the handling of language and character issues.

#### Flow Control

The flow of information is defined according to business purposes. Business processes define in which cases which information has to be passed from system to system. Business processes must also define the procedures to be followed in case a business message could not be processed because of missing or inaccurate information. The information flow consists of business messages containing data and control information.

#### Process Logic & Dependencies

The majority of business logic resides in the sending and receiving applications that produce and consume the interface data. Only limited, specific process logic will be incorporated into the middleware, as described here.

Process logic in the middleware defines business rules for transformations and routing. The process logic needs to be provided according to business purposes. This includes rules for transformation of data from one format to another and information about the purpose of the information flow, its source system and its destination or destinations. This type of logic will be captured in the interface specifications.

Wrappers will be used to facilitate processing logic. A wrapper is a tag attached to the data being interfaced across systems. Typically a wrapper will specify the source and target system, and any special processing requirements that can be understood by the middleware.

CLIENT middleware will be generally “stateless”, that is it will not maintain an awareness of the individual content of data messages that have passed through. If different sending systems produce message types with a dependency between them, one design possibility is that they can share the same interface channel and be processed in sequence. This achieves the goal of keeping related messages in order during transmission and middleware processing. More complex dependencies will not be directly supported by the middleware.

#### Sequencing

The sequencing of messages is considered a special type of process logic and must be handled appropriately in the middleware as well as both source and destination systems. Sequencing preserves the order of messages – also of several units of work. Wrappers and channels have to be aligned in order to provide sequencing.

#### Logical Unit of Work

A logical unit of work is either processed completely or not at all. According to the possible communication styles there are two different cases:

- **Synchronous**: If two application components are tightly coupled the communication is done synchronously and a unit of work may span both components. In addition, transaction monitors can be used to guarantee the unit of work. (Note – we do not anticipate the use of transaction monitors for CLIENT.)
- **Asynchronous**: If two applications – named A and B – communicate in an asynchronous communication style having loose coupling, three units of work must be considered:
  1. The unit of work that is performed by application A, which ends when the message is handed over to the middleware;
  2. The unit of work that is performed by the middleware, which starts when the message is taken over from A and ends when the message – possibly transformed by the...
middleware – is delivered to B (or several Bs). This unit of work does span the middleware components, but does not span the applications;

3. The unit of work that is performed by application B, which begins when the message is handed over from the middleware.

In every unit of work no partial delivery will be allowed; either all or nothing will be performed.

The following consequences need to be considered:

- If one message consists of 100 work items in one unit of work and the middleware is unable to transform record 100 then the whole processing for the first 99 records will be rolled back and no delivery will be done.
- If 100 separate messages consist of 100 work items, each work item representing one unit of work – which may be sequenced in this case – and the middleware is unable to transform message 56, then the processing for the first 55 messages is successful and the results will be delivered to the target systems. However, the failure of Record 56 will stop the processing – because of the sequencing – such that message 56 to message 100 can not be processed and delivered to the target system(s).

Units of work must be defined according to business needs.

Ack/Nack

Ack (Acknowledgement) and Nack (Negative Acknowledgement) are results of flow control and can be the results of transformations that the middleware has performed.

As a principle, acknowledgements are generally to be provided by the application that receives the data to inform the sending application that the processing is complete. The middleware will not acknowledge delivery of messages, since at this stage it cannot guarantee that the processing has been completed successfully or unsuccessfully. Ack and Nack are branches in an information flow and the reaction on these results needs to be defined according to business processes. The expectation is that the receiving system sends acknowledgement to source system after successful processing as a separate message.

Routing

Routing determines the path of a message and is performed according to the definition of the information flow. Every message contains both data and control information. Control information includes information that is used to perform the routing, e.g. information about message types, source system and target system. It is preferable to use a wrapper to hold the control information separately from the business data. Wrappers facilitate the operation of the middleware in the following ways:

- Data can be routed to specific mapping logic based on wrapper values.
- Wrappers can be used to track messages through the middleware.
- Wrappers provide clear source and destination information.

Note that the wrapper must be added at the source application side. Since the standard SAP IDoc structure does not include a wrapper, IDoc messages from SAP must be combined with a wrapper by a connector at the SAP system side. Lookups to get routing information within the middleware can work only with relatively static lookup data.

There are three primary types of routing:

- **Point to Point**
  - System A sends message to designated System B
- **Fanning out**
  - System A sends message. Middleware parses individual data records and transmits to designated System B, C, D, etc.
- **Replication**
  - System A sends message. Middleware sends copy to System B and C.

Business requirements will determine the type of routing to be used. In general, CLIENT will avoid as much as possible the point-to-point routing, and rely as much as possible on fanning and replication.
Status Control

Error Handling

Errors are classified in two categories: system errors and content errors.

System Errors

System errors are handled by the middleware. There will be a system monitoring facility in place to report whether systems are running or not and identify incidents. Errors are handled in such a way that responsible maintenance staff can be notified. The technical strategy relies on the provision of the monitoring facility by a standard tool, expected to be Tivoli.

Content Errors

Content errors occur when the information is incomplete or inaccurate and they must be managed at the application layer. These errors occur on individual interfaces and must be managed by interface procedures (either automatic or manual). Capturing and reporting of the error can be supported by the monitoring facility. Note that the development approach needs to provide agreed common guidelines for the handling of content errors.

There will be a common solution in form of an error handling system. Both system and content errors resulting from interface processing will be reported to the error handling system. The recipient of error notifications will have to be defined for each interface, along with resolution procedures. A single tracking tool is proposed to capture status information, including errors, from source system through the middleware to destination system.

Tracking

Messages from application to application will be tracked by the middleware; main items to be tracked are record contents, delivery/completion status and time stamps. The aim is to assure that the end-to-end delivery can be verified, checked and used for audit purposes. The tracking does not capture application knowledge but provides a logging of messages.

It is assumed that tracking can be done with a standard tracking tool that is part of the middleware. Responsibilities for performing the tracking will be assigned within the GCs according to procedures to be defined as part of implementation.

Status Reporting

Status reporting can be provided based on information captured by tracking. The information to be reported will be defined as part of the implementation. Examples of the contents of a status report could be information regarding number of processed records, completion status, timestamps etc. Status reporting does not address application-specific reporting, such as the correctness of the data.

Auditing

A special case of status reporting is auditing. Legal considerations and obligations defined by Client Controllers will specify the cases where audit requirements may be placed on interfaces. Auditing can be done by providing appropriate status reports (completion status, time of processing). These reports can provide information about messages passed, their contents, and their processing status. Note that application-specific auditing must be done by the applications.

Archiving

Currently no specific requirements about archiving are known. Archiving will be based on tracking to identify the completed transmissions. The data files archived will be kept online for some period of time, then later stored on magnetic tapes or similar media and kept as long as needed to fulfill legal requirements.

The data will be archived both before and after it goes through transformation. Moreover, the maps used to transform the data will also be archived to ensure that the processing can be reproduced fully. Maps will be archived when they are modified as part of normal development or maintenance activities.

Note that an archiving policy must be in place to refine archiving within the scope of the technical strategy. Moreover, archiving needs to be aligned with change control because modifications in software may cause changes in message formats. In addition, there could be special legal requirements for EDI transmissions that specify the amount of time that data files have to be kept.
Reprocessing
Reprocessing can be supported on the technical base provided by tracking, which includes logging. Reprocessing cannot be done if data is incorrect, i.e. reprocessing does not solve rejects of messages by applications because of inaccuracy of data. Reprocessing of unsuccessful transmissions will be done using the archived files sitting on the middleware rather than starting a new transmission from the sending system.

Note that reprocessing is also part of backup-and-recovery and needs to be addressed in appropriate operational guidelines.

Data Transformation
Technical Strategy takes into account the following use of terms (the first two definitions are common within the industry – the illustration is taken from the Common Warehouse Meta-model standard):

- **Transformations** occur among structures and concepts on the same logical level, e.g. from any physical object (like data structures) to any other physical object.
- **Mappings** map logical structures and concepts onto physical implementations.

The term **Conversion** is used for specific data transformations (on physical level)

**Translation** has been used for specific code page transformations

The technical strategy relies on the provision of data transformation within the middleware. If the data transformation needs additional lookups in the source system then the data transformation should be performed at the source system side. Moreover, if messages must be composed then the composition should be done at the source system. The middleware is suitable to perform format transformations and data conversion based on conversion tables.

Therefore the preferred location for data transformation is the middleware when translation is simple and lookup requirements are static. Exceptions must be decided on a case-by-case basis. Source or destination systems will do mapping when the translation is complex and lookup requirements are large and/or dynamic.

Two ways of updating lookup files in middleware (when lookup data is from another source) need to be considered:

- Business process (change management) invoked when contents of lookup change.
- Technical solution (automated process) where change in contents sends update to lookup files.
Security

Security can be divided into four separate components. The example below is for an interface outbound from SAP. The same logic applies for interfaces inbound into SAP.

- **SAP level security** – At the SAP level we will rely on the security offered by the application to ensure that an interface is triggered only by the appropriate jobs or dedicated resources, and that only the correct steps are executed in the correct order and with the correct programs.

- **Messaging and queuing security** – In this case we will rely on the capabilities of the messaging and queuing system to prevent unauthorized manipulations of the data, and to ensure that the data is delivered correctly across queues.

- **Middleware (mapping and transformation) security** – In this case the highest risk is posed by the possible manipulations of maps and mapping data. In terms of maps, we will make use of the level of security that is provided by the middleware vendor (e.g. security at the file system level to ensure that maps are not accessible). In terms of mapping data, we will rely on the security and logging capabilities offered by the RDBMS where the mapping data is stored.

- **Receiving system security** – From a CLIENT point of view there is typically no knowledge of the security offered by the receiving non-CLIENT systems. We will assume however that once a file has been successfully and securely received from the messaging and queuing system, the application will provide the integrity necessary to ensure that the data is posted correctly without external manipulation in the receiving application.
Development Responsibilities

The following items represent the primary activities necessary to construct an interface between SAP and other applications. This technical strategy does not attempt to define organizational responsibilities, but multiple teams will be involved.

- Establish development standards and guidelines.
- Set up connectivity between SAP and middleware, including SAP configuration (partner profile, ALE distribution model, etc.)
- Set up connectivity between middleware and non-SAP application (complementary, Market).
- Establish technical specifications based on functional requirements.
- SAP design and development.
- Non-SAP application design and development.
- Develop data mapping between applications.
- Define and configure routing in middleware.
- Test connectivity and interface functionality.
Glossary

This section provides some further explanation of some key interface terminology and concepts.

Transactions
Transaction can have several meanings. In general a transaction is a unit of work.
1. In the context of business logic a transaction describes the fulfillment of some related working steps (e.g. supported by a SAP workflow). Business messages define the information transfer between applications. Therefore business messages can contain several information units. Moreover, a transaction can be composed of several sub-transactions.
2. A technical transaction is a unit of work that has to be accomplished according to the ACID properties (Atomicity, Consistency, Independence, and Durability), e.g. assuring that money transfers are either done completely or not-at-all.

Components
Applications are composed by software components. Every component fulfills some functionality and provides interfaces for inbound and outbound communication. Components may be distributed over several machines, usually there are components running on desktops communicating with components on servers and mainframes in order to fulfill one task.

Interface
For our definition of the term interface see Section 5.

Communication Styles
The manner in which applications and components interact is critical for their flexibility. The two basic choices for the communications model are synchronous and asynchronous communications.
1. Synchronous Communication requires the sender of a request to wait until a reply is received before continuing to process. It typically needs to wait until the response is received in order to continue processing. Interactive systems require synchronous communication because a user expects to see the information based on actions he or she takes. Synchronous communication is also required where two (or more) applications or software components must work together to accomplish a task. Synchronous communication is the key for client/server applications where presentation layer components act together with server and mainframe components in order to fulfill one business transaction. There are three popular types of synchronous communication:
   - Request/reply
   - One-way
   - Polling
2. Asynchronous Communication does not require the sender and receiver to coordinate their processing with any communication and therefore provides a lower degree of coupling than synchronous communication. Asynchronous communication occurs when the communication between a sender and a receiver is accomplished in a manner that allows each of them to operate independently of the other. There are three popular types of asynchronous communication:
   - Message passing
   - Publish/subscribe
   - Broadcast

Flow (of Information)
Interactions and flows of information between application components can be described independent of any specific attributes of a particular deployment, whether that is a workflow, messaging service, etc. [OMG-1] Flow composition is performed with components, Nodes and Connections. A node can be a decision node, a mapping node, a branch node, or a join node. Source nodes and sink nodes are special nodes.
Messages
The communication between applications and components is mainly based on messages. Business processes together with core data blocks form business messages, which represent information flows between business components (being implemented by software components). Therefore every message has a semantically meaning that is transported with data (information can be seen as data together with its semantics).

Integration
There exist several types of integration – on the presentation, functional or data layer.

1. Presentation Layer: Integration on the presentation layer is accomplished through an application user interface like web portal. An application user interface integrates several views on several subsystems in one presentation, which appears as one single application interface to the user. However, presentation layer integration requires that a consolidation of data have been done in order to present consistent information.

2. Data Integration: Integration of the data layer is accomplished by access to the same database or by replicating data.

3. Functional Layer: Integration on the functional layer addresses the fact that several applications and/or components have to act together in order to fulfill a task. This type of integration reflects the business processes and their business logic. It requires a description of the purpose of the integration and the tasks to fulfill.

Middleware
Middleware is the application-independent software that provides services that mediate between applications. [EAI-2]

Workflow
Workflow supports execution of long running, interruptible, potentially interactive business processes. [IBM-1] Workflow is managing a set of requests or messages across a series of components in a prescribed order as a single action. Originally workflow systems concentrated on automating paper flows among people. [EAI-2] Workflow is used if the business process does involve interaction with different human users or roles. Workflow is used if the duration and consistency of the business process is critical.

XML
XML is currently the most popular format for messages and plays an important role in e-business and for many other types of information exchange. However, XML is a meta-language defining the structure of messages. The Internet community creates XML dialects/languages that define common business semantics to business documents – like ebXML.
References


Disclaimer and Liability Notice

This document may discuss sample coding or other information that does not include SAP official interfaces and therefore is not supported by SAP. Changes made based on this information are not supported and can be overwritten during an upgrade.

SAP will not be held liable for any damages caused by using or misusing the information, code or methods suggested in this document, and anyone using these methods does so at his/her own risk.

SAP offers no guarantees and assumes no responsibility or liability of any type with respect to the content of this technical article or code sample, including any liability resulting from incompatibility between the content within this document and the materials and services offered by SAP. You agree that you will not hold, or seek to hold, SAP responsible or liable with respect to the content of this document.