Slide 4

**Evolution of EDI**

Let us consider a typical business scenario where a customer who wants to purchase an item and he creates a purchase order and then faxes or mails it to the vendor.

The vendor receives the purchase order and he manually keys in a sales order.

After that vendor's system generates a confirmation date that is sent back to the customer via fax or mail.

The vendor then ships the goods via a carrier. The carrier delivers the products to the customer.

When the goods are shipped, the vendor invoices the customer.

The customer makes the payment by check, and the vendor deposits the check in the bank.

Finally, funds are transferred from the customer's account to the vendor's account.

This simple scenario requires the exchange of various documents between several business partners at

different times. A business process is a series of actions or functions that bring about a business result, and as

such there are some inherent problems with this scenario. It:

Is highly inefficient and laborious

Cannot be tracked easily

Gives no visibility into the process

Has a very long lead time

Includes redundant data entry at various points

To circumvent some of the trouble spots, business partners started exchanging data electronically by means of floppy

disks and other storage devices.

This led the business partners to adopt standard formats.

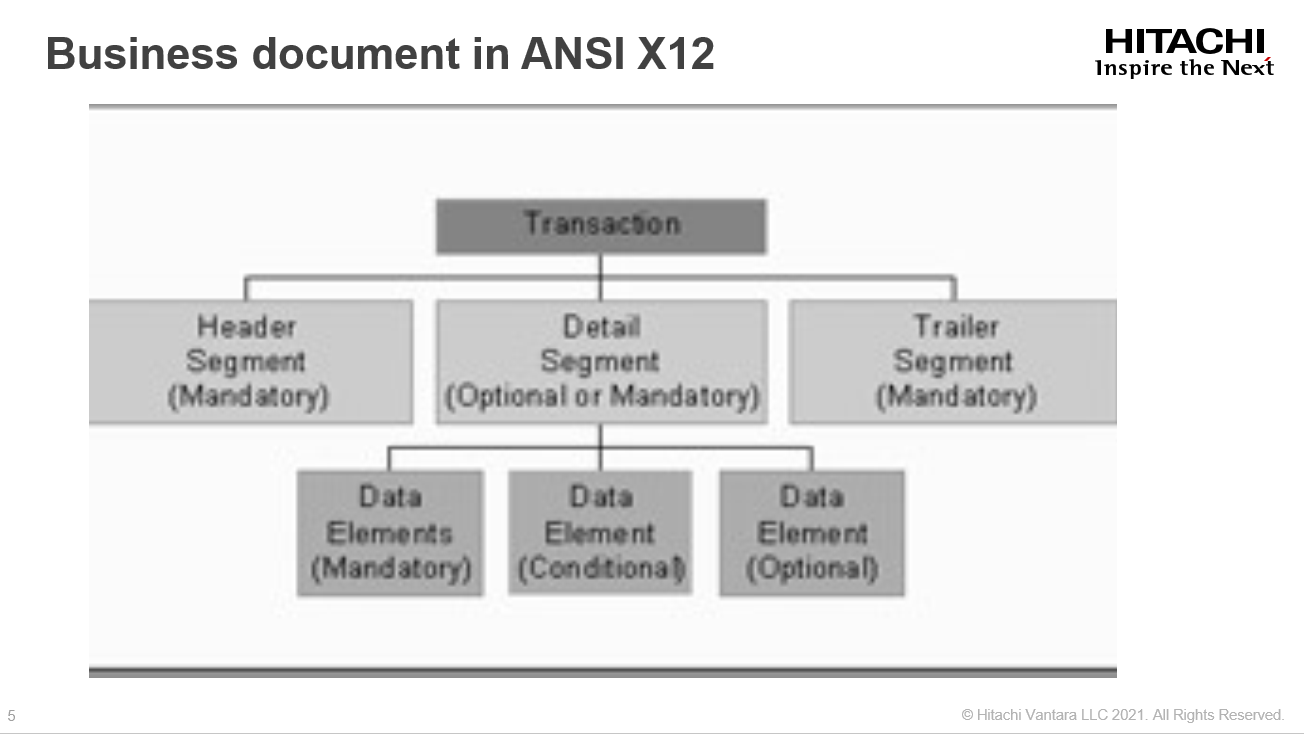
However, no commonly adopted standard existed among the companies involved, which resulted in multiple electronic

formats for the same type of business document.

The industry eventually saw a need for standardization, and a committee named *ANSI ASC X12* was formed to address this issue.

So, ANSI committee was formed to define the standards and

Ultimately, the electronic exchange of business documents in a standard format gave birth to what is now known as *EDI*.

Slide 5

A *transaction represents the* electronic equivalent of a business document.

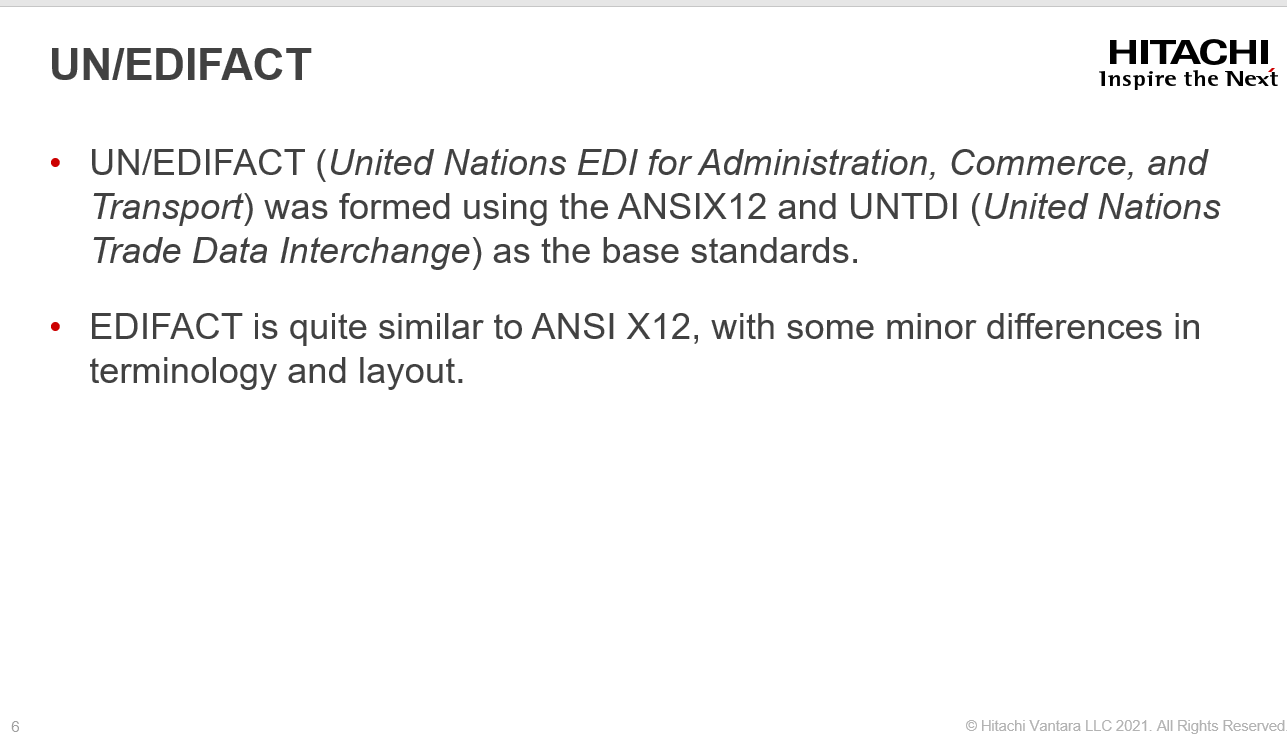
The transaction is usually divided into three areas like, header area, detail area, and summary area.

The *header area* consists of segments that apply to the entire document, and is usually mandatory.

For example, in a purchase order, vendor number and vendor address are part of the header segments.

The *detail area* contains document details. The items being ordered and their quantities are considered detail segments. The *summary area* consists of data that summarizes the entire document, and the total amount and taxes

are part of the summary segments.

SLIDE 6

EDIFACT calls business documents messages and represents them by a name such as *ORDERS* for a

purchase order, whereas ANSI X12 calls them *transactions* and represents them by a number such as

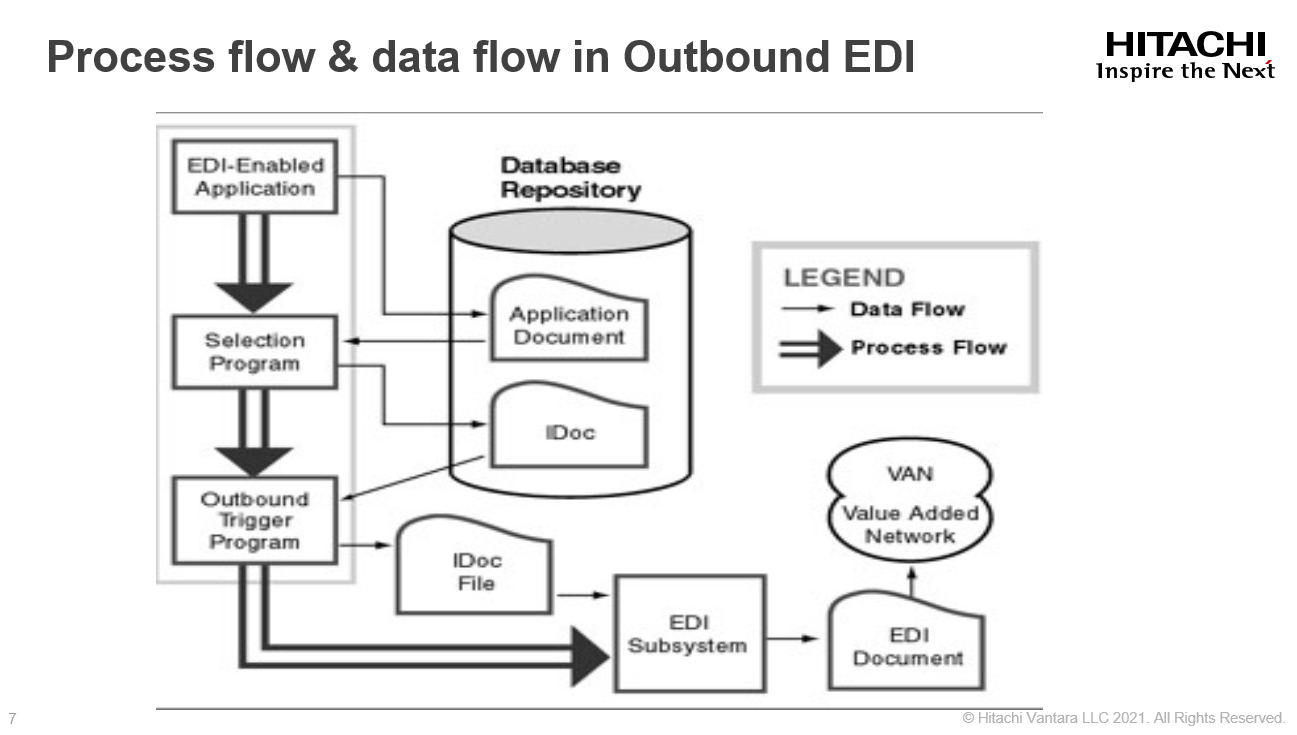
850 for a purchase order.



EDIFACT uses the same segment in multiple places, whereas ANSI has a specific use for each

segment.

EDIFACT has additional segments that apply to international trade.

Slide 7

The first step in the outbound process involves creating an application document such as a purchase order or sales order and saving it in the database tables.

The document is created and leaves some hooks for the EDI process to begin.

The application document just created is now formatted to an IDoc format.

At this point you can think of IDoc as yet another format in which the application document has been

represented.

The IDoc created resides in the SAP database repository. This IDoc document must be passed down to the operating

system layer for processing by the EDI subsystem and in this Step, the IDoc is transferred to the operating

system as a text file. The document is still in an IDoc format. The only difference is the medium of

storage.

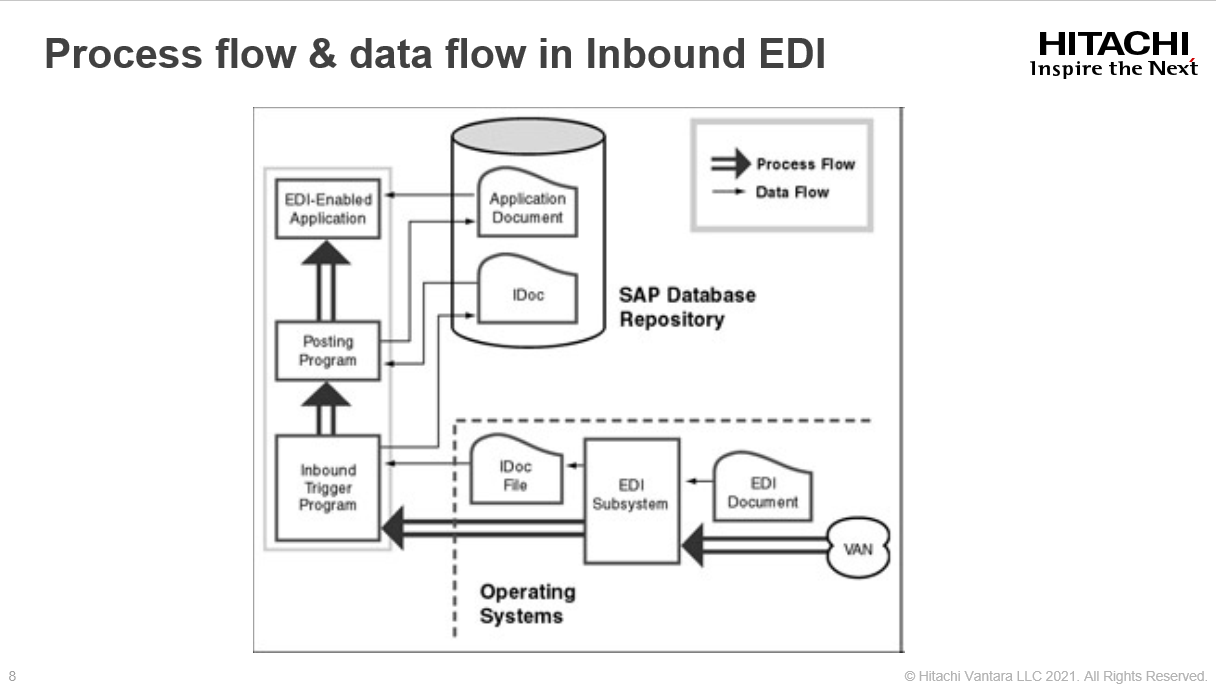
The IDoc format is an SAP proprietary format. For EDI purposes, the document in IDoc format has to be converted into an EDI standard format. Third−party software called a *translator* carries out the transformation process and reports status back to the SAP system. SAP refers to these translators as *EDI subsystems*, and has certified several subsystems for

connectivity to SAP. SAP takes no responsibility for translation. Thus, from SAP's perspective, after

the IDoc is delivered to the subsystem, SAP does not have control over the process, but it maintains the status reported by the EDI subsystem.

After the document is converted to an EDI standard format, it is transmitted to a trading partner based on the partner's EDI settings.

When an IDoc is under the control of the EDI subsystem, the subsystem can optionally report the state of processing at various milestones back to the SAP system.

 SLIDE 8

EDI documents are received from a business partner via the VAN. These documents are in one of the EDI standard formats. The documents are deposited in a common repository for the subsystem. This part of the process is not part of the SAP EDI architecture.

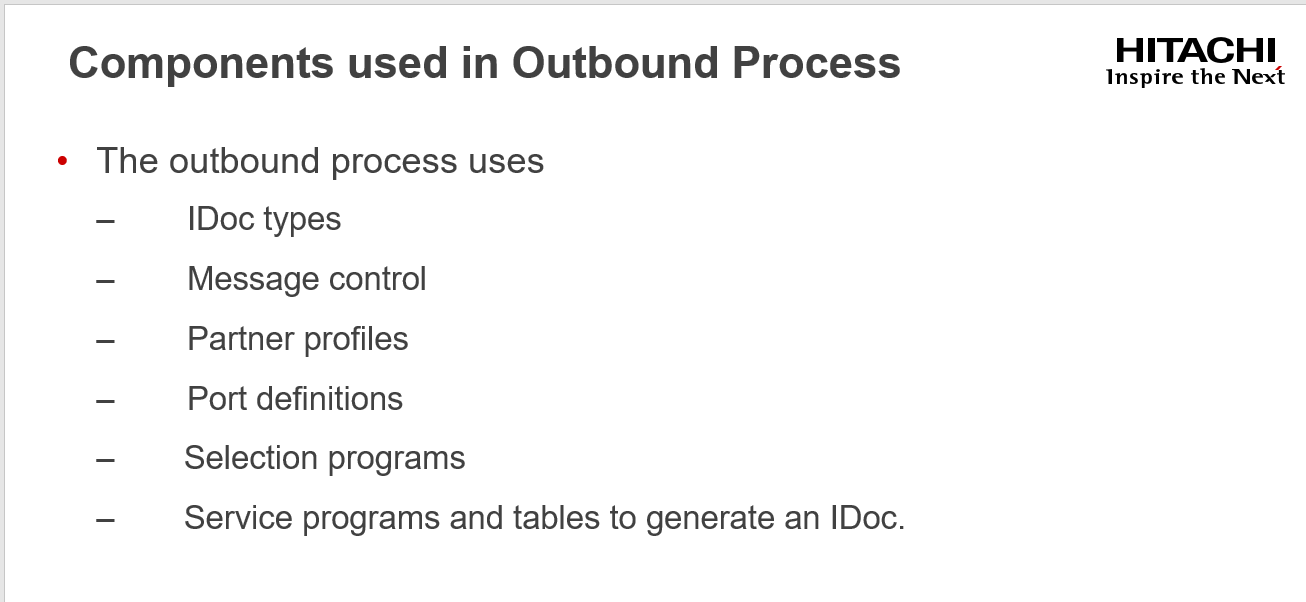
The EDI−specific headers and trailers are stripped off, and the document is converted into an IDoc format suitable for SAP applications. The process is carried out at the EDI subsystem level.

The IDoc created is stored in a text file at the operating system layer. The subsystem starts an inbound program in the SAP layer. This program reads the IDoc file and creates an IDoc in the SAP repository for further processing.

The IDoc received from the subsystem is passed to a posting program. This program creates an application document such as a sales order, purchase order acknowledgment, invoice, or shipment notice.

The application document created via EDI is the same as any document created manually in the system

For example, if an incoming sales order was created via EDI, you could view the sales order document via transaction VA03.

 Slide 9

For the EDI document to be generated has an equivalent message type defined in the SAP system.

The message type is based on an IDoc structure.

For example, if you are going to generate a purchase order, the message type ORDERS is assigned in SAP to purchase orders. This message is based on IDoc type ORDERS01 and ORDERS02.

Message control determines and processes the various outputs associated with an application document (for example, it can be EDI, printed output, fax, or mail).

A partner profile is created for each business partner, and a record exists for each outbound message sent to a business partner. For example, if two outbound messages (purchase order and purchase order change) are being sent to vendor number VEN001, a partner profile must exist for VEN001, and two outbound records (one for each message type) must exist in the partner profile. The partner profile is an important and frequently referenced component.

A port is used in the outbound process to determine the name of the EDI subsystem program (if installed), the

directory path where the IDoc file will be created at the operating system level, the IDoc file names, and the

RFC destination.

Selection programs, which are typically implemented as function modules, are designed to extract application

data and create an IDoc. A selection program exists for each message type. The programs are generally named

with the following naming convention:

IDOC\_OUTPUT\_<message type>

These function modules have a standard interface for input and output parameters. A process code is assigned

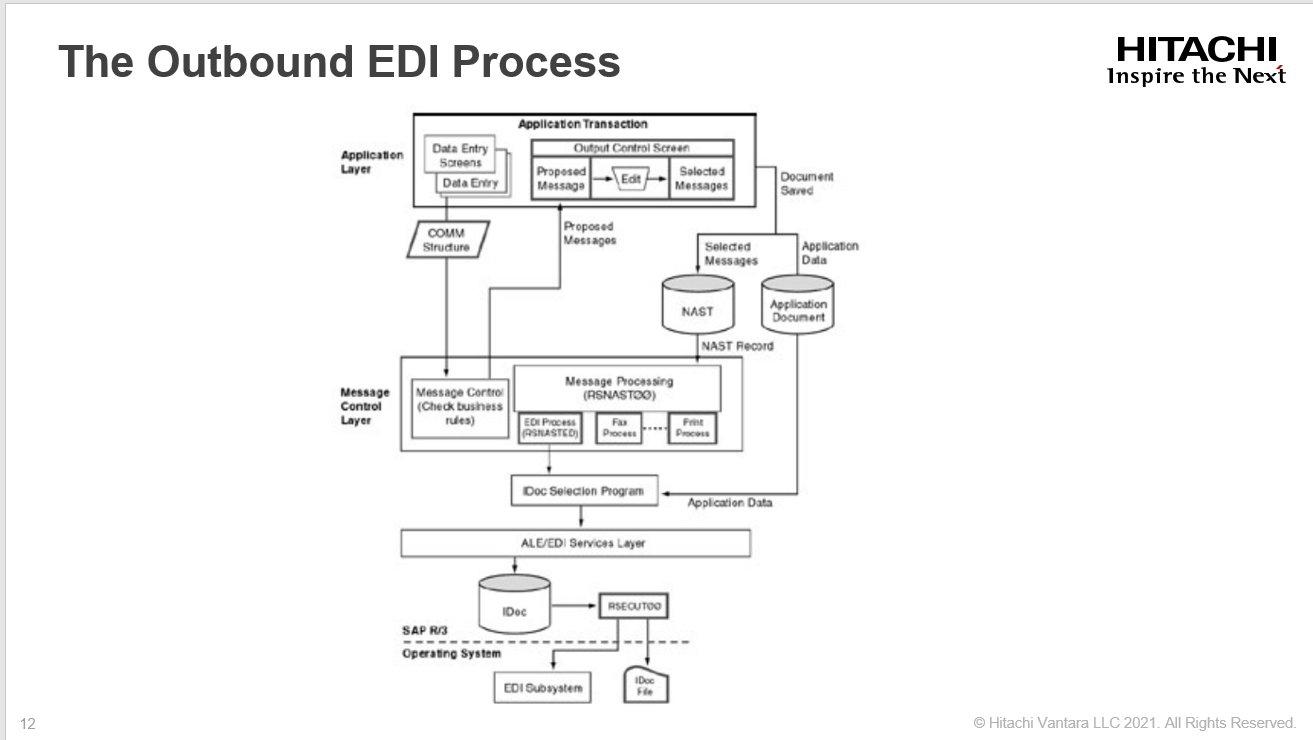
to each selection program that executes under Message control. Because process codes are flexible, you can

assign any processing option to a process code. A process code can point to a function module or a workflow.

In the standard system, process codes always point to a function module.

For example, the selection program for message type ORDERS is IDOC\_OUTPUT\_ORDERS. A

four−character process code, ME10, has been assigned to this function module.

 Slide 10

**Processing in the Application Layer**

An application document is entered via a transaction; for example, a sales order is entered via VA01. Before

an application document is saved, the Message control component is invoked. An application passes several

key elements of the application data to the Message control component via communication structures.

**Processing in the Message Control Layer**

The Message control component checks various business rules defined in the Message control configuration.

Based on the business rules, Message control proposes the output type (sales order response, internal notification, and so on), timing (immediate or batch), medium (EDI, ALE, printout, mail, fax, and so on), and language of the message.

Multiple outputs are possible from Message control. Each output is processed independently. A simple example of a business rule is to send a sales order response immediately via EDI for a sales order from Customer 1111 and generate an internal message for the production department.

The proposed outputs can be seen on the output control screen of the transaction. Some applications provide

output control at both the line−item level and the header level. In most applications, you can reach the output

control screen in one of the following ways on the application transaction screen.

By choosing Header, Messages

By choosing Header, Output

For example, you reach the output control screen for a purchase order by selecting Header, Messages. For a

sales order, you reach the output screen at the header level by selecting Header, Output and at the item level

by selecting Item, Output.

The output proposed by the Message control component can be edited. The user can delete, add, or modify

any outputs proposed on the output control screen.

When the application document is saved, the outputs proposed on the output control screen are saved as a

record in the NAST table. An entry in the NAST table stores the key of the application document, output type,

timing, processing status, and language.

If the timing for an output is set to 4 (Immediate), the system immediately starts the processing of the output

type by executing the RSNAST00 program. If the timing is set to 1 (Batch Mode), the entries are processed

when the RSNAST00 program is executed explicitly. The RSNAST00 program is usually scheduled to run on

a regular basis. It selects entries that can be processed based on their status and calls the appropriate program

for the selected medium. Each output medium (EDI, ALE, fax, printed output, and so on) has a corresponding

program to generate the output in the required medium. SAP provides standard processing programs for each

output medium.

In the case of EDI, the processing program is a form routine: EDI\_PROCESSING in the RSNASTED

program. This program is a generic program for all EDI outputs and reads the partner profile configuration to

determine the selection program for a particular message.

In ALE, the processing program is also a form routine: ALE\_PROCESSING in the RSNASTED program.

This program is a generic program for all ALE outputs and reads the partner profile configuration to

determine the selection program for a particular message.

**Processing in the Selection Program**

The selection program is specified in the partner profile with a process code. The RSNASTED program calls

the appropriate selection program.

One of the parameters for these function modules is the NAST entry, which contains the key of the

application document. These programs extract application data, format it into an IDoc format in an internal

table, and then pass control to the ALE/EDI layer.

**Processing in the ALE/EDI Layer**

The ALE/EDI layer is responsible for creating the IDoc in the SAP database. By now, a tangible IDoc that can

be viewed using the various monitoring tools has been created in the system. The IDoc gets a status record

with a status code of 01 (IDoc Created).

The IDoc, before being saved in the database, goes through a syntax check and other validations. If there are

no errors, the IDoc gets a status of 30 (IDoc Ready for Dispatch to ALE Service).

**Dispatching the IDoc**

The next step is to pass the IDoc to the operating system layer. The settings of the Output Mode fields in the

partner profile are read to determine the timing of this step, and the port definition is read to determine the

directory location in which the file will be created. If the mode is set to Do Not Collect IDocs, the IDoc is immediately passed to the operating system layer automatically, using program RSEOUT00. If the field is set

to Collect IDocs, the IDoc is not passed to the operating system layer until RSEOUT00 program is executed

explicitly. The RSEOUT00 program is typically scheduled to run on a regular basis. If IDocs are collected,

RSEOUT00 creates a single file with all the IDocs in it. After the IDoc file is created successfully, the IDoc

gets a status code of 03 (Data Passed to Port OK). The timing of this step is important from a performance

point of view.

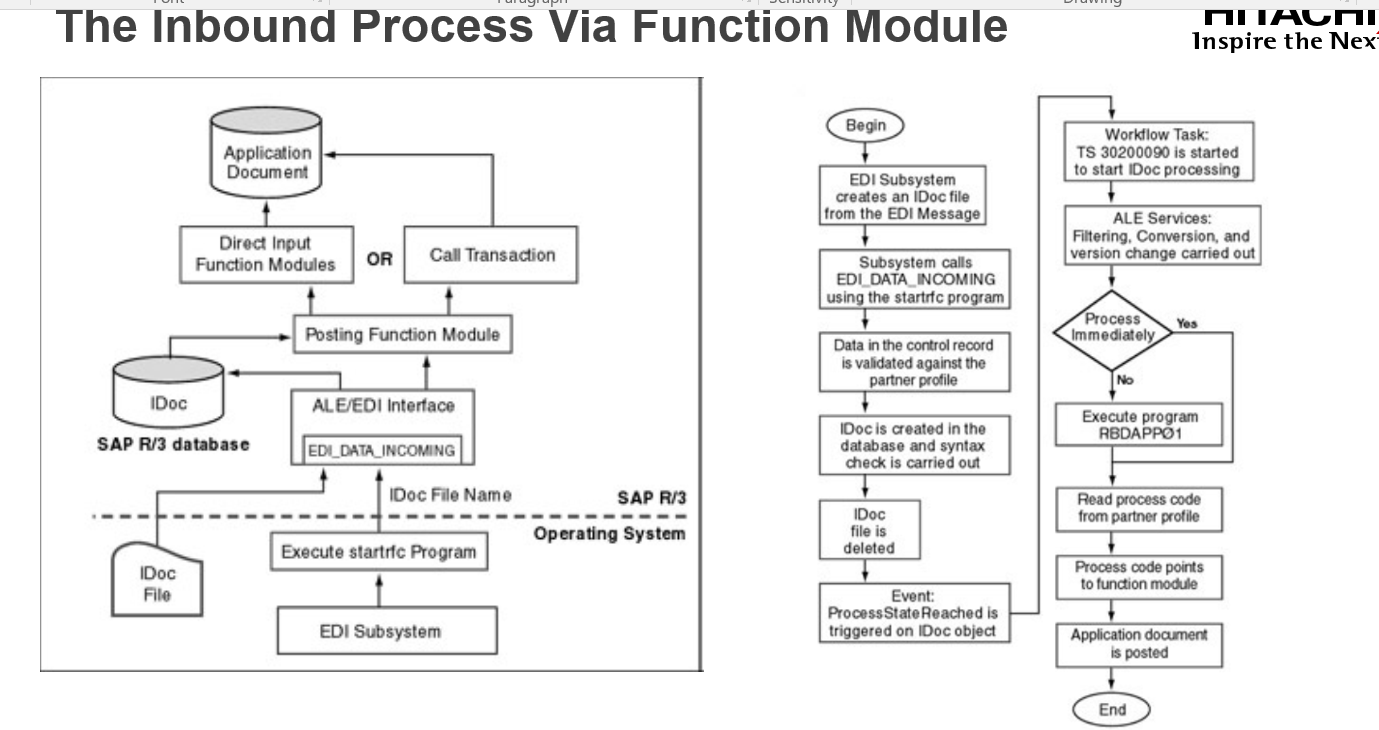
Now the system determines whether the subsystem is to be started. This parameter is also specified in the

partner profile in the Output Mode settings. If the value is set to Do Not Start the Subsystem, the process ends

here. If the flag is set to Start the Subsystem, the subsystem program name is read from the port definition and

the subsystem program is started. The name of the IDoc file is passed to the EDI Subsystem program. If the

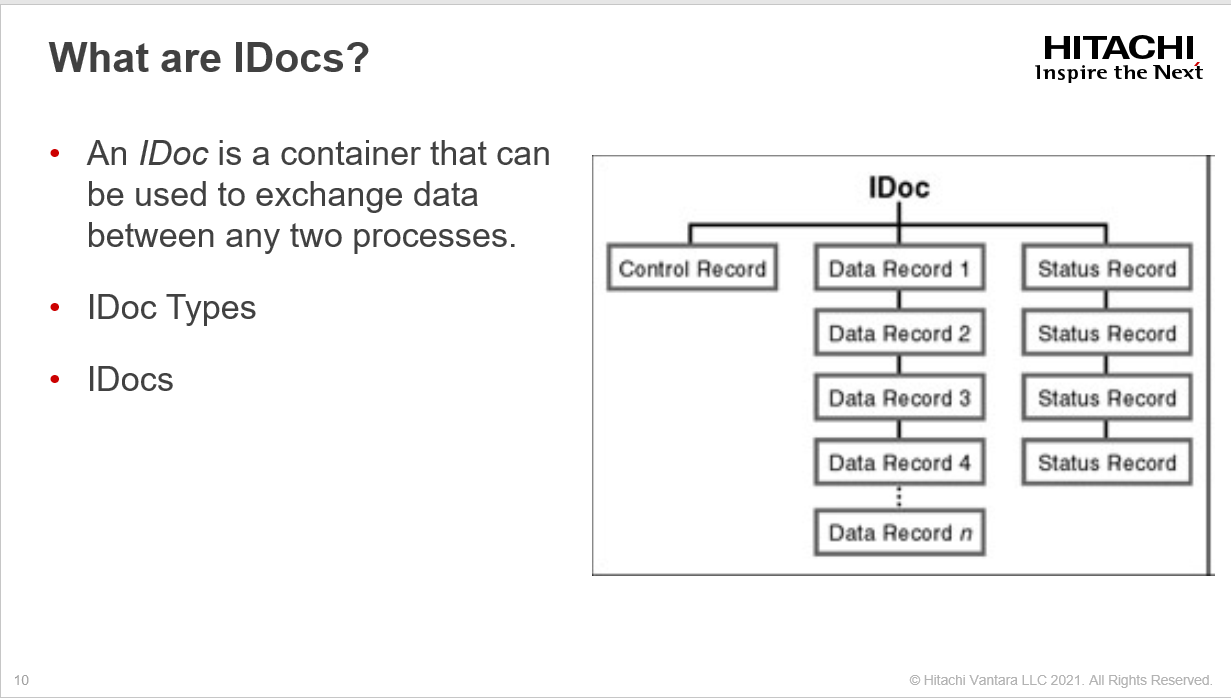
subsystem is started successfully, the IDoc gets another status code 18 (Triggering EDI Subsystem OK).

 Slide 11

In this process, the IDocs are transferred from the EDI subsystem to SAP, and then they are passed to the

posting function module to post an application document.

Explain the Slide Steps.

 Slide 13

An *IDoc* is a container that can be used to exchange data between any two processes

An IDoc represents an IDoc type and IDoc data, depending on the context in which the word *IDoc* is used. An IDoc type defines the structure and format of the data being exchanged. For example, the IDoc type INVOIC01 defines the format of an invoice document. IDoc data can be seen as an instance of an IDoc type. For example, an actual invoice received from a vendor in electronic form is converted into an IDoc.

IDocs types are based on EDI standards (ANSI X12 and EDIFACT). They are closer to the EDIFACT

standards than to ANSI X12.

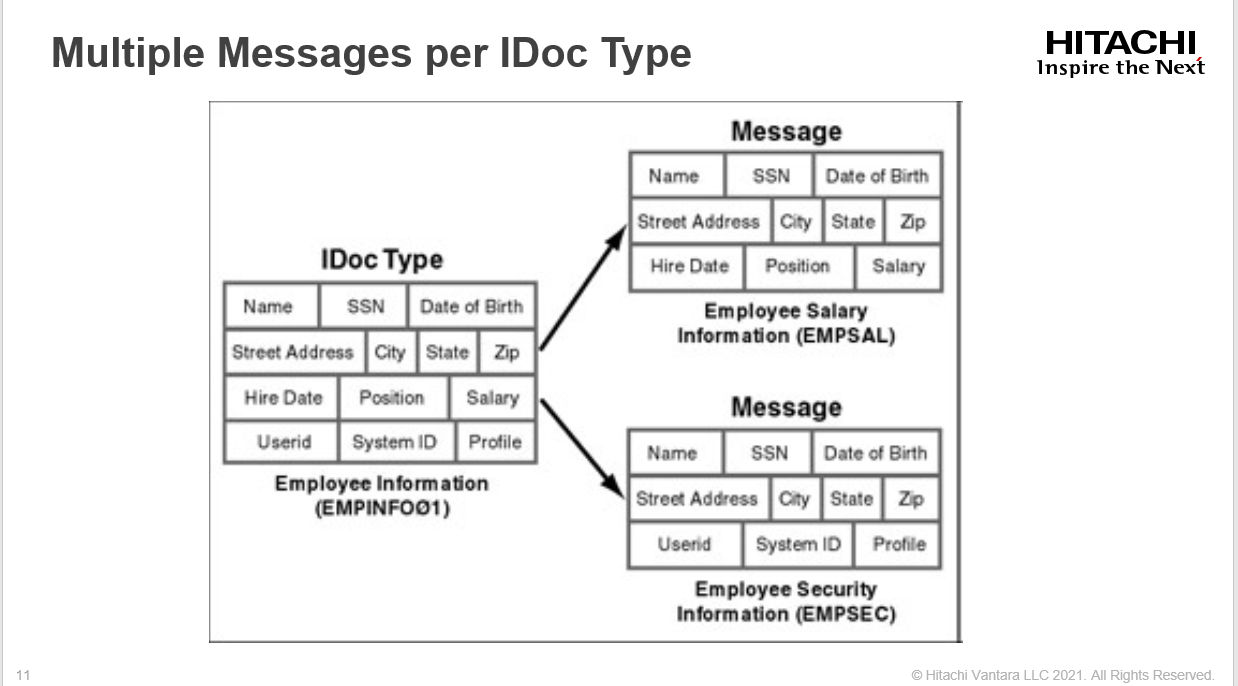
An IDoc structure consists of several segments, and segments consist of several data fields. The IDoc

structure defines the syntax of the data by specifying a list of permitted segments, the arrangement of the

segments, and optional versus mandatory segments. Segments define a set of fields and their formats.

An IDoc is an instance of an IDoc type. Each IDoc is assigned a unique number for tracking and future

reference. An IDoc serves as a focal object for tracking the state of the process that generated it.

 Slide 14

A message represents a specific type of document that is transmitted between two partners. Orders, order

responses, order acknowledgments, invoices, and shipment notices are examples of messages.

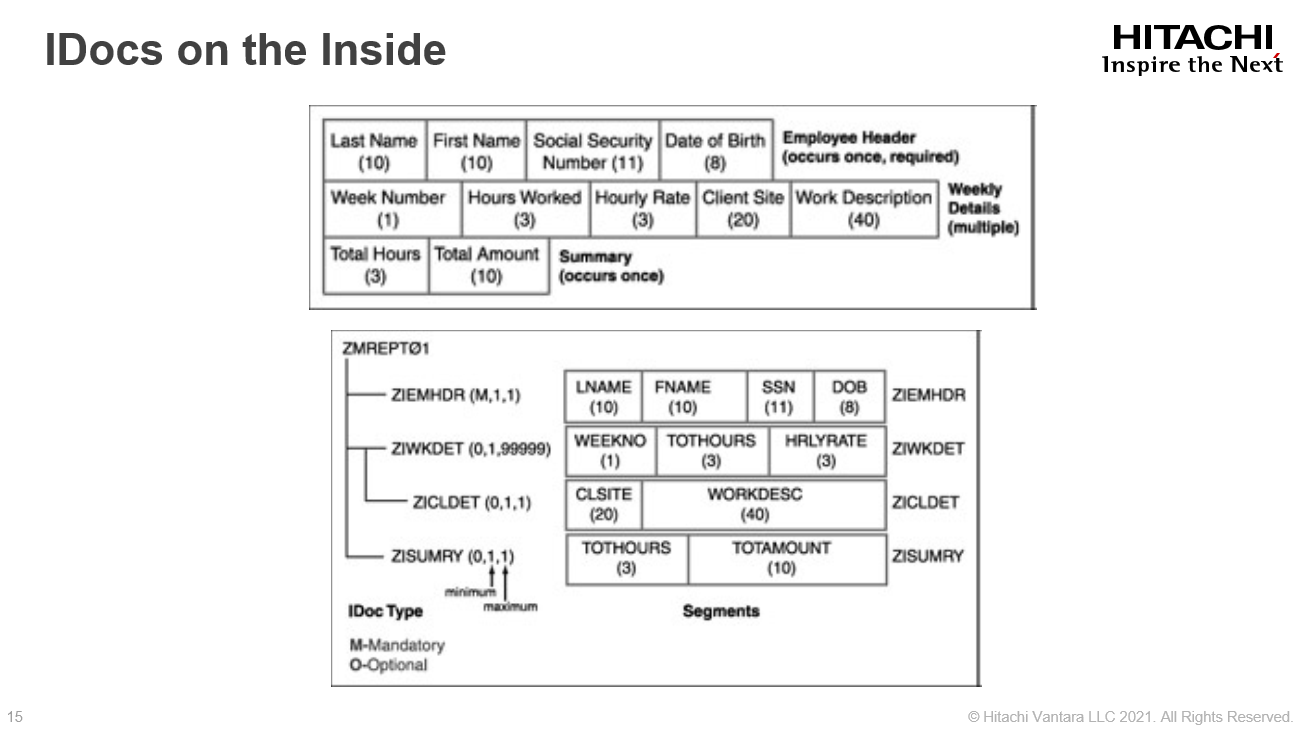
An IDoc type in SAP can be used to represent several messages or business documents.

Of course, the messages must be logically related.

In this slide we see, an IDoc type represents all possible information about an employee.

This IDoc type is being used to send two separate messages to two separate applications. One message is the Employee Salary Information; the other is the Employee Security Information.

The difference between the two messages is the set of segments used.

 Slide 15

Assume that you have an application that records an employee's weekly hours. At the end of the month, a file

containing the monthly report data for each employee is sent to an external system.

This application has been replaced by the SAP system, and a standard IDoc has been developed to support the process.

Following are some properties of the file.

It has three types of records, including employee header information, weekly details, and monthly summaries.



Each record type has certain properties, such as whether it's optional or mandatory, number of times it

can be repeated, field names, data type for each field, and length of each field.



The client site and work description in the weekly details are not always available.