**CONCEPTOS:**

**Ambiente:** un ambiente es un servidor donde ha sido instalado el sistema SAP. Se configuran en SAP tres ambientes, uno de desarrollo (DEV), uno de pruebas(QAS) y otro de producción(PROD), que es donde se utilizan las aplicaciones desarrolladas y probadas satisfactoriamente.

[**BADIs:**](http://orekait.com/blog/que-son-las-badis-de-sap/)las **Business Add-Ins** son un nuevo tipo de extensión al sistema SAP, basado en la utilización de objetos ABAP para la implementación de validaciones y ampliaciones al código estándar de SAP, sin tener que modificar el código estándar.

**Juego de datos:** es la técnica de Batch Input en donde la ejecución se realiza en forma Batch, es decir de fondo y queda almacenada en SAP para su posterior ejecución mediante la transacción estándar SM37, la cual nos permite lanzar el juego de datos.

[**BAPIs**](http://orekait.com/blog/que-es-una-bapi-en-sap/): las Business Application Programming Interfaces son las interfaces estándar de SAP. En concreto, son piezas de código en forma de módulo de funciones, usadas en la integración entre componentes de SAP, así como con componentes externos. Las BAPIs están ligadas en forma de método a los objetos de negocio de SAP y se prefieren sobre los Batch Input porque son mantenibles por el mismo SAP.

**Batch input:** es una técnica automática mediante la cual podremos realizar la carga masiva de datos en el sistema utilizando las validaciones de las transacciones.

**Breakpoints:** son puntos de control que se colocan en el programa, para que al ejecutarse se detenga el procesamiento en esos puntos, de modo de poder analizar un posible error o la razón por la cual ciertas variables adquieren determinados valores.

**Buffer**: un búfer (del inglés, buffer) es un espacio de memoria, en el que se almacenan datos de manera temporal, normalmente para un único uso (generalmente utilizan un sistema de cola FIFO); su principal uso es para evitar que el programa o recurso que los requiere, ya sea hardware o software, se quede sin datos durante una transferencia (entrada/salida) de datos irregular o por la velocidad del proceso.

**Cabecera:** es la sección principal del documento que almacena todos los datos únicos del mismo, esto es válido para todo el documento. Por ejemplo, número y tipo de documento, fecha, proveedor, cliente, etc.

**Campo quemado y código quemado:**

**Control de cambio Vs Extensión:** ambas son modificaciones, lo que varía es que el control de cambios se le hace a un objeto Z, en cambio la extensión se hace a código estándar.

**Dato maestro:** son registros en SAP que se utilizan en las transacciones. Estos registros no se modifican constantemente y permiten al usuario trabajar de forma más ágil. Los datos maestros son la fuente de información central para una empresa. Podemos encontrar diferentes tipos de datos maestros: clientes, proveedores, materiales, personales, etc.

**Objeto:** es la representación en un programa de un concepto de la vida real y contiene toda la información necesaria para abstraerlo: datos que describen sus atributos y operaciones que pueden realizarse sobre los mismos.

**Objeto de negocio:** es un tipo de entidad inteligible que es un actor dentro de la capa de negocio de un programa de ordenador basado en n capas. Se distinguen en que no realizan nada por sí mismos, sino que albergan un conjunto de atributos y asociaciones con otros, tejiendo un mapa de jugadores que representan las relaciones de negocio. Los objetos de negocio se recogen en un repositorio (BOR – Business Object Repository) accesible desde la transacción SWO1.

**Objetos de bloqueo:** son objetos que se crean en SAP, a través de la transacción estándar SE11 para permitir que varios usuarios puedan actualizar datos de una tabla base de datos en forma concurrente sin que se pierdan datos de modo de garantizar la consistencia de los datos. Se utilizan sólo en las transacciones concurrentes de SAP, es decir aquellas transacciones en donde varios usuarios acceden al mismo tiempo a modificar datos almacenados en la base de datos del sistema.

**La calidad del *software*:**es el conjunto de cualidades que lo caracterizan y que determinan su utilidad y existencia. La calidad es sinónimo de eficiencia, flexibilidad, corrección, confiabilidad, mantenibilidad, portabilidad, usabilidad, seguridad e integridad.

**Legacy:** un sistema heredado (o sistema legacy) es un sistema informático (equipos informáticos y/o aplicaciones) que ha quedado anticuado, pero continúa siendo utilizado por el usuario (típicamente una organización o empresa) y no se quiere o no se puede reemplazar o actualizar de forma sencilla.

**Clase:** abstracción de un objeto con métodos, propiedades y atributos. Pueden ser declaradas **globalmente o localmente**. Las primeras se crean mediante el Class Builder (transacción SE24). Estas clases son almacenadas centralmente en pools de clases en la biblioteca de clases dentro del repositorio de SAP. Todos los programas ABAP en un sistema SAP pueden acceder a las clases globales. Las clases locales son las que se definen dentro de un programa ABAP. De esta manera, las interfaces y las clases locales pueden ser usadas solamente en el programa en donde están definidas. Cuando se usa una clase en un programa ABAP, el sistema primero busca una clase local con el nombre especificado. Si no se encuentra ninguna, busca una clase global.

**Componente de software:** es la unidad más pequeña, instalable y que puede ser mantenida.

**Cross application:** aplicación transversal que impacta varios o todas las áreas del negocio.

**Tabla interna:** es un concepto similar a los arrays en otros lenguajes de programación. Permiten guardar en memoria grandes cantidades de registros de datos con la misma estructura.

**Debug:** es el proceso de identificar y corregir errores de programación.

**Debugger:** es una herramienta de programación que se utiliza para ejecutar los programas ABAP, por línea o por sección, de modo de analizar malos funcionamientos o funcionamientos no deseados. Permite colocar breakpoints y watchpoints. Actualmete hay dos versiones: clásica y el nuevo debugger.

**Estructuras Append:** es un objeto de ABAP que se crea para agregar datos y configuraciones adicionales a las tablas y estructuras estándar de SAP. También son conocidas como Ampliaciones de tablas de base de datos.Los campos que agreguemos en una Estructura Append deben empezar por TT(tipo de dato), ya que de esta forma se previene que existan conflictos con los nombres de los campos estándar de SAP.

[**Dynpros:**](http://tumyeto-dev.blogspot.nl/2012/01/programacion-por-pantallas-2-creacion.html) Es el sinónimo de pantalla dentro de SAP, permite ingresar datos por medio de campos de entrada u elementos varios. Cada dynpro tiene asignado un número compuesto por cuatro dígitos numéricos.

Las Dynpros son los componentes fundamentales de un ***programa de diálogo*** en ABAP, el cual está compuesto por una serie de pantallas por las cuales el usuario navega mientras interactúa con el programa y realiza determinadas acciones.

**Estándar de programación:** determina la forma en la que codificaremos nuestros programas ABAP. Asegura que cualquier miembro del equipo pueda trabajar y mantener de forma fácil con el desarrollo de otro compañero.

**Esquema estrella:** es un modelo de datos que tiene una tabla de hechos (o tabla fact) que contiene los datos para el análisis, rodeada de las tablas de dimensiones.

**ETL**: proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos y limpiarlos, y cargarlos en otro repositorio para apoyar procesos de negocios. Su abreviación, significa: Extracción, Transformación y Carga (Load en inglés).

**Eventos:** Los eventos son los mensajes levantados por un objeto. Siempre que se cumpla alguna condición. Un evento es una etiqueta que identifica una sección de código dentro de un programa. Los nombres de eventos son palabras reservadas. No podemos crear nuevos eventos, solamente podemos usar los que ya existen. En objetos ABAP hay ciertos métodos que se conocen como disparadores (triggers) y otros que se conocen como manejadores (handlers). Los triggers son los métodos que disparan un evento, mientras que los handlers son los métodos que se ejecutan cuando ocurre un evento. Los eventos se declaran en la parte declarativa de una clase o en una interface.

**GAPS:** se refiere a los requerimientos funcionales o técnicos no contemplados en el diseño inicial que luego de un análisis detallado salen a la luz.

**Grupo de basis:** grupo de IT de una organización dedicado a cuestiones técnicas de seguridad y programación (como aplicación de notas por la transacción SNOTE).

**Integración:** hace referencia al intercambio de información entre el software, sistemas internos y externos a él.

**In-memory computing:** consiste en llevar a memoria RAM los datos de la base de datos en lugar de hacerlo en discos, tal como lo hacen las bien conocidas bases de datos relacionales.

**Include:** palabra clave del sistema que se utiliza para la declaración de objetos (~~programas~~) cuyo objetivo es modularizar los programas ABAP de manera que sea más fácil su mantenimiento. Por lo general se crean Includes de datos con las declaraciones de datos e Includes de Subrutinas con las rutinas que se utilizan en el programa. Los includes son considerados una buena práctica puesto que reducen el número de líneas, facilitan el mantenimiento y favorecen la reutilización. Se deben usar cuando el No. de líneas de código es mayor a 1000.

**Index**: cláusula ABAP que es utilizada en muchas sentencias para especificar un registro en particular dentro de una tabla interna.

[**Índices**:](http://www.tufuncion.com/indices-mysql) una estructura de datos que mejora la velocidad de las operaciones, permitiendo un rápido acceso a los registros de una tabla en una base de datos. [(ver como se crean en SAP)](http://www.blogdesap.com/2012/01/definicion-un-indice-de-una-base-de.html)

**Interface**: es una estructura independiente que se puede implementar en una clase para extender el ámbito de esa clase. Las interfaces en ABAP se crean a través de la transacción estándar SE80.

**Interfase**: unión provista entre dos sistemas.

**Mandante:** Es una instancia creada dentro de un ambiente, que se utiliza para configuración, desarrollo o pruebas. Los datos que se visualizan en cada mandante son independientes dentro del mismo ambiente. El mandante es el nivel jerárquico más alto en el Sistema SAP.

[**Módulos de función:**](http://www.abap.es/Descargas/Modulos_de_funciones.pdf)son objetos que se crean para ser utilizados en distintos programas, de modo de no tener que generar dicho código en cada uno de los programas y son manejados en una biblioteca central de funciones, que es la transacción SE37. Siempre que se define un módulo de función será necesario crear primero un ***Grupo de Funciones*** que lo pueda contener. Una buena práctica consiste en agrupar funciones relacionadas por algún tópico. El grupo de funciones se creará también mediante la transacción SE37. Debe documentarse ya que son reutilizables. También deben documentarse los parámetros de la función.

**Programas de dialogo o module pool:** tipo de programa ABAP que está formado por una serie de Dynpros o pantallas por las que el usuario interactúa mientras navega por la aplicación. Cada dynpro tiene una lógica de procesamiento que está especificada en módulos. Dentro de cada dynpro se define lo que se conoce con los nombres de PBO y de PAI.

**Objeto local:** programa o cualquier otro objeto de SAP es grabado como "Objeto Local", es decir no es asignado a una clase de desarrollo o paquete, cuando dicho objeto es de pruebas, es decir no se transportará dicho objeto al ambiente de testing. “*Los objetos locales pueden comenzar con "Z" o con "Y", dependiendo la empresa en donde estemos, se utilizará una u otra letra”.*

**Framework:** define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar (infraestructura, armazón, marco).

**Object:** es la clase maestra vacía que agrupa a todas las clases existentes en SAP. La clase OBJECT no tiene componentes y tiene la misma función para las variables referenciadas que el tipo de datos ANY para las variables normales.

**Netweaver:** es una plataforma de tecnología integrada para todas las aplicaciones SAP en el plano técnico. Es conocida como una aplicación orientada a servicios y a la integración.

**Field exits:** también llamados ampliaciones de campo, son validaciones de los elementos de datos asociados a campos de pantallas o dynpros de SAP.

**Field Symbol:** nos permiten acceder y pasar datos cuyos nombres y atributos no conocemos hasta el momento de la ejecución (de forma dinámica). Son como punteros.

**Grupo de función:** un grupo de función es un objeto que concentra uno o más módulos de función que tiene algo en común para compartir.

**Interfaces:** las interfaces son estructuras independientes que se pueden implementar en una clase para extender el ámbito de esa clase.

**PAI** (Process After Input): el PAI es el evento que se ejecuta luego de introducidos los datos en la dynpro y realizada alguna acción sobre ella, como puede ser presionar un botón.

**PBO** (Process Before Output): es el evento que se ejecuta antes de que se visualice una dynpro. Aquí se define que se mostrará en la Dynpro.

**Paquete:** también llamado "clase de desarrollo", es una forma de organizar todos los nuevos objetos que se crean en SAP, ordenándolos generalmente por módulos. Permite transportarlo a diferentes entornos o sistemas. Para ello debemos ir al menú> entrada catálogo de objetos> y allí asignarle un paquete Z que no sea el correspondiente a los objetos locales que es $TMP. Podemos crear un objeto como local y luego si lo necesitamos pasarlo a transportable. Ruta: SE80>seleccionar paquete.

Nota: los desarrolladores NO tienen que crear nuevos paquetes. Los paquetes son previamente creados por el coordinador de Basis y ABAP.

**Transacción:** es un código único que existe en el sistema SAP y que es ejecutado por un usuario del sistema para acceder a una funcionalidad particular del sistema SAP.

**Datos maestros:** representación en SAP de cada una de las distintas instancias de una categoría cuya información va a ser compartida en todo o parte del sistema. Característica: varían poco en el tiempo. Ejemplo: materiales, clientes, proveedores, etc.

**Datos transaccionales:** distintos documentos, operaciones u acciones que creamos en SAP como consecuencia del funcionamiento normal de la empresa. Ejemplo: movimientos de materiales, órdenes de compra y de venta, etc. Característica: crecen y se crean, modifican y eliminan constantemente por las operaciones diarias de la empresa.

**Estructuras:** es un conjunto de definiciones de campos almacenado en el diccionario de SAP. Una estructura nunca podrá “llenarse” con ningún registro de información asociada en la base de datos. Su uso se sitúa solamente dentro de la programación ABAP.

[**Tabla transparente**:](http://www.blogdesap.com/2011/05/crear-tablas-transparentes-y.html) al igual que una estructura, una tabla transparente también corresponde a un conjunto definido de campos estructurados almacenada en el diccionario de SAP. La diferencia es que una tabla transparente puede ser “llenada” con información. De hecho, una tabla en general es la unidad mínima necesario para introducir información en nuestra base de datos.

**Landscape:** indica el compuesto de un ambiente SAP, cuantos servidores, con qué aplicativos, etc. En SAP, podemos elegir varios tipos de landscapes, por ejemplo: singles donde solo contamos con una instalación de un servidor en este caso un productivo, landscape de 2 instancias, por ejemplo un desarrollo y un productivo y el recomendado por las mejores prácticas un landscape de 3 fases, lo que lo convierte en el más típico y cuenta con una instalación de un ambiente de sap para desarrollo, otro para calidad y un último productivo.

**Layout:** término que hace referencia a la distribución de columnas e información en una pantalla de selección (filtrado de información) o salida de un reporte. Suele utilizarse en español con la denominación "disposición

[**Dominio:**](http://www.blogdesap.com/2011/05/crear-tablas-transparentes-y.html) el dominio se define como el rango de valores que puede tener un campo. Es decir, el tamaño máximo que puede tener un campo (desde 0 caracteres o números hasta lo que se nos ocurra) y el tipo de información que podrá albergar (caracteres, números enteros, números decimales, etc.).

**Tipos de** [**tablas SAP:**](http://ciogestion.com/tipos-tablas-sap/)(darle clic al enlace)

[**Tipos de tablas internas:**](http://formacion.desarrollando.net/cursosfiles/formacion/curso_390/sap-05-03.pdf) (dar clic en al enlace)

Vista:

**Match code:** o ayuda: es una ayuda de búsqueda dentro del Sistema SAP. Esta ayuda de búsqueda se activa al presionar la tecla F4 sobre un campo en un formulario en el cual estemos posicionado.

ITAB: (ver tabla interna)

**Menú:** de ámbito los menús de ámbito son utilizados para agrupar transacciones en una estructura de árbol tal como se observa en el Menú SAP (Corresponde a la estructura de árbol donde se despliegan todas las opciones de usuario).

**Field symbol:** son una sentencia ABAP que nos permite acceder a los datos de los programas en forma dinámica. Los Field Symbols son como los punteros del lenguaje C ya que sirven para asignar el contenido de una variable al field symbol. No reservan espacio físicamente para un campo, sino que “apuntan” a su contenido. Un field symbol puede apuntar a cualquier objeto de datos.

Los Field Symbols aumentan las posibilidades de error o de incorrectas asignaciones de datos durante la ejecución de un programa, dado que los campos se asignan recién en el momento de ejecución del programa y las verificaciones de sintaxis y chequeos de seguridad son muy limitados. Es por ello que se recomienda el uso de los field symbols sólo si no se pueden alcanzar los mismos resultados usando otras sentencias ABAP.

[**Variante:**](http://www.blogdesap.com/2010/12/crear-variantes-de-report.html) cuando en SAP hablamos de Variantes nos estamos refiriendo a una imagen fija de la pantalla de selección de una transacción de modo que podemos introducir los valores que deseamos en cada uno de los campos de la pantalla y grabar dicha pantalla para no tener que volver a cargarlos, una y otra vez con cada ejecución que realicemos de la transacción. Las variantes se asocian a los programas de tipo "reporte", los cuales requieren interacción con el usuario a través de la pantalla. Cuando hablamos de reportes nos referimos a los llamados "Reportes Clásicos", también a los llamados "Reportes interactivos" o a los llamados "Reportes ALVs".

Las variantes son de gran utilidad en las etapas de pruebas de los objetos, debido a que nos ahorran mucho tiempo y esfuerzo en realizar tareas repetitivas y tediosas como son las cargas de las pantallas de selección de transacción que tienen muchos campos en las mismas.

Las variantes se crean en la misma pantalla de selección del reporte. Allí luego de completar los campos presionamos el botón "Grabar" y a continuación visualizaremos en la pantalla los campos "Variante" y "Significado", introducimos el contenido de estos dos campos y volvemos a grabar.

Lo más común es crear en cada entorno en donde trabajamos una variante nueva para cada transacción, pero también es posible transportar una variante de un ambiente a otro. Para crear una variante que pueda ser transportada simplemente debemos crearla con un nombre que empiece por CUS&.

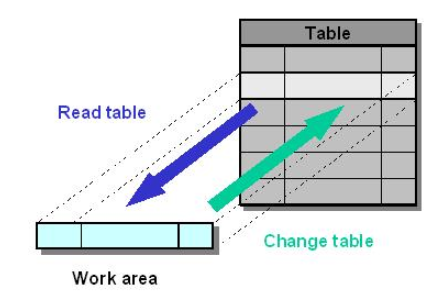
**LSMW** (Legacy System Migration Workbench**):** La LSMW es una herramienta que sirve para cargar y modificar datos maestros o transaccionales en SAP de manera masiva desde un fichero externo.

[**Orden de transporte (OT):**](https://abapers.wordpress.com/tag/orden-de-transporte/) Es un número único en SAP que se utiliza para agrupar objetos que van a ser transportados entre ambientes. Para la administración de las órdenes de transporte se utilizan las transacciones SE10, SE01 y SE09. Una OT puede tener dos estados: modificable (permite agregar o quitar objetos) y liberada (no se puede modificar, lista para ser transportada). Y puede ser de dos tipos:

* De workBench: son aquellas que se utilizan para el desarrollo de aplicaciones.
* De customizing: son aquellas que se utilizan para configuración del sistema.

[**Tabla interna**](https://www.codejobs.biz/es/blog/2013/11/22/continuacion-aprendiendo-a-desarrollar-en-abap-para-sap-tablas-con-cabecera-y-work-area)**:** variable dinámica temporal que permite almacenar datos del mismo tipo en memoria durante el tiempo de ejecución. Las tablas internas tienen un cuerpo (1 a N registros) y una cabecera (estructura) que puede ser opcional.

* ¿Para qué es el cuerpo de la tabla interna? El cuerpo es quién almacena los registros de la tabla interna.
* ¿Para qué es la cabecera de la tabla interna? La cabecera (header) no es más que un único registró que se utiliza para agregar y recoger datos del cuerpo del cuerpo de la tabla interna.
* ¿Qué es el Work área en ABAP? El Work área viene siendo cómo la estructura de una tabla, pero que almacenará un sólo registro de la misma.



*Tipos de tablas internas*

Hay dos tipos de tablas internas:

1. Tablas internas con HEADER
2. Tablas internas sin HEADER.

*Tablas internas con encabezado:*

Aquí el sistema crea automáticamente el work area.

* El work area tiene el mismo tipo de datos que la tabla interna.
* El work area se llama HEADER.
* Es aquí donde se realizan todos los cambios o cualquiera de las acciones sobre el contenido de la tabla. Como resultado de esto, los registros se pueden insertar directamente en la tabla o acceder directamente desde la tabla interna.

*Tablas internas sin encabezado:*

* Aquí no hay work área asociado con la tabla.
* El work area debe ser explícitamente especificado cuando necesitamos acceder a dichas tablas.
* Por lo tanto, no se puede acceder directamente a estas tablas.

**Diferencias entre work areas, tablas internas con cabecera y field symbols.**   
  
Las work areas y las tablas internas con cabecera, necesitan que se copien los datos desde la tabla interna a ambas. Mientras que, si se usa un field symbols declarado con la misma estructura de una tabla interna para recorrerla, no copiamos ninguna información, tan sólo nos desplazamos por las distintas posiciones de la tabla (un puntero). Por lo que el desplazamiento es mucho más rápido.   
  
Pero cuando se busca que un programa sea más rápido, optimizarlo en cuanto a la velocidad, hay que estudiar donde se está perdiendo el tiempo. Usando la herramienta estándar de SAP con test buenos para ello y en condiciones concretas. Lo más habitual es que el mayor tiempo se pierde en la selección de datos, mientras que el recorrido de los datos ya en una tabla interna no suponga mucho tiempo en comparación y ni lo note el usuario si lo mejoramos, aunque siempre está bien tener el mejor código que optimice los recursos en todos los puntos.

**Delivery unit (DU):** es un contenedor usado en la administración del ciclo de vida (LCM) para transportar un repositorio de objetos entre los ambientes de SAP Hana. El nombre del DU debe contener una mayúsculas, dígitos (0-9) y guion bajo.

**Tipos de tablas internas**

Se recomienda hacer uso delos tipos de tabla hashed y/o sorted antes de hacer uso de la estándar.

**Reporte:** es un tipo de programa ABAP, una pantalla de visualización de datos, donde generalmente existe una pantalla de selección, en la cual se introducen ciertos parámetros, luego a partir de estos parámetros se realiza una selección de datos a tablas SAP o tablas Z y finalmente se muestra los datos obtenidos por pantalla mediante la utilización de la sentencia write.

Lo que describimos anteriormente se conoce en ABAP como "Reporte Clásico" es decir un reporte generado a partir de la sentencia WRITE.

También existe la posibilidad de crear reportes ALV (ABAP List Viewer).

Este tipo de reporte es el más utilizado debido a lo fácil que es su creación. Otro punto fuerte que tienen los reportes ALV es que muchas funcionalidades tales como ordenar, filtrar, sumarizar, etc ya están incorporadas en el reporte por lo que no es necesario crearlas.

¡Ojo! Existen varios tipos de ALV: list, grid, jerárquicos, etc. Y varias formas de programar ALV (puedes utilizar la programación orientada a Objetos).

Diferencia entre reporte y formulario???

Inner join mas de 3 malo

**Jerarquías:**

Module pool SAPMZAAnnn

Dynpro SAPMZAAnnn

Definición de variables: MZAAnnnTOP

Includes:MZAAnnnInn

Donde I:

O: Mod. PBO

I: Mod.PAI

F: procedimiento

Grupo de función ZAAnnn

Módulo de función ZAA\_xxxxx

Paquetes ZAA

**Flujo de cambios en SAP**

Parametrizar>Ampliación>Copia a objeto estándar>Modificación

**Pruebas del programador:**

* Ejecutar debugger: verifica EF.
* Usar la transacción ST05: consultas ejecutandose en donde y TE.
* Usar transacción SCI: code inspector, seguridad del código y buenas prácticas.
* Verificar rendimiento: Utilidades>más utilidades> Run time analysis.

**Pruebas unitarias:**

Verifican que el programa desarrollado se integre con su módulo y sus programas. Verifican el correcto flujo de datos.

**Pruebas integrales:**

Verifican que el desarrollo se integre con otros módulos y sus programas.