Concepts of Portal and its Architecture

**Portal**
The portal is a single point of access to integrated information, from heterogeneous sources through Web services. (Yahoo is a portal and gmail is not a portal)

Sources of information ranging from the Internet to enterprise back-end systems deliver content presented in the portal. Portal components and services are the mechanisms that provide content as iViews, which are programs that run in the portal client, such as a Web browser.

To run the Portal Platform, the end user needs a standard Web browser on the desktop; no additional software components are required.

Once logged on, users can access the information in the iViews to which they have privileges through the security mechanisms for authentication and authorization.

In addition, end users can configure and personalize the portal, making it unique to their individual preferences and working styles. For example, users may configure the portal to display only the information they wish to see. Information available to any one individual can be different, depending on his or her authorization levels.

**Portal Architecture:**

**Overview**
The Portal Platform has an open architecture that enables the integration of SAP NetWeaver components such as Knowledge Management and Collaboration.

The portal environment is for both developing iViews for distribution to end users and the administration of iViews, including connectivity and communication processes with back-end systems.

Portal Platform includes objects that can be accessed using a number of protocols. Although some protocols request individual objects directly, portal resources and objects are referenced by instructions that are encapsulated in a Uniform Resource Locator (URL) as encoded strings to form the address of the object or resource.

As a part of SAP NetWeaver, the portal is responsible for all processes, including components that pass messages to other platforms, and interfaces such as Knowledge Management and Collaboration.

**Components of Portal Platform in SAP NetWeaver:**

The application architecture of the portal is comprised of integrated SAP technologies that run other technologies and processes. In SAP NetWeaver, the Portal Platform provides the Portal Framework, which has the Portal Runtime.

All operations and processes of the portal occur within the portal runtime, which among other systems provides the Unification infrastructure, and the Connector Framework.

You can integrate and implement other SAP NetWeaver components, such as the Knowledge Management platform and Collaboration, into the portal through services that enable end users to search for both structured and unstructured information, and to collaborate with each other in their work.

**The Portal on SAP Web Application Server**
The portal is deployed on the J2EE Engine, which is a component of the SAP Web Application server, based on the Java 2 Enterprise Edition (J2EE) standards. The Portal Platform relies on the J2EE architecture implemented by the application server.

The portal is comprised of software that provides core functionalities, such as applications that are loaded at runtime, before other applications. Such applications provide central functionality to the Portal Platform, such as integration with the User Management Engine and the Portal Content Directory.

In addition to the runtime libraries, configuration files, core applications and so on, the portal sets up other services that interface with the J2EE Engine such as, PRTBridge: a service containing functionalities and services for inter-cluster communication. This facilitates the integration processes between the runtime environment of the portal and J2EE Engine.

**The following provide the interfaces between the portal and SAP Web Application Server:**

- **Portal Runtime Container**
  A container implemented for the runtime processes of the portal to manage:
  - The deployment of Portal Applications: these consist of Java software that is programmed for the portal.
  - The communication between Portal Applications and the J2EE application.
  - The robust interaction and integration between Portal Applications and the J2EE Engine.
  - PRTBridge Service

The PRTBridge is the link between the runtime processes of the portal and the J2EE Engine cluster configuration. It is installed for the portal as a J2EE Engine service.

The PRTBridge allows communication among J2EE Engine cluster elements, including communication with the J2EE cluster groups. For example, the number of cluster servers, the name of each server, and so on.

**Portal Framework**
The Portal Framework is a logical environment that consists of a collection of software components.* Some of these enable implementation of
iViews. Others are interfaces of the portal infrastructure that run processes such as:
Creating pages and applying their defined layouts
Managing styles and themes
Providing access to the resources needed at runtime
These components are contained in the following:
Portal Runtime (PRT), an entity that manages iViews. For more information, refer to the section, Portal Runtime.
1. Portal Content Directory (PCD), service-based implementation of the portal infrastructure that interfaces with the SAP Web Application and the portal. For more information, see Portal Content Directory.
A database shared with SAP Web Application Server. It serves as a central location in which an aggregation of specific portal data is maintained and selectively obtained at runtime for further processing. The PCD has interfaces to the database. For additional information, see Database. In addition, the Portal Framework is comprised of standard technologies, such as HTML, WML, and XML, as well as protocols such as the following:
Hypertext Transport Protocol, and Secured Sockets Layer (HTTP and HTTPS)
Authentication protocols, including access to directory servers such as Lightweight Directory Access Protocol (LDAP), Active Directory Services, digital certificates and logon tickets
Java Database Connectivity (JDBC), a database protocol for connecting Java software to data in a database system
Simple Object Access Protocol (SOAP), a protocol that interfaces with Hypertext Transfer Protocol (HTTP) and Extensible Markup Language (XML), to enable exchange of information across different platforms.
SAP Java Connector (JCO), an SAP proprietary interface that allows communication with SAP systems using Java applications
P4, an SAP proprietary RMI-based protocol in the J2EE Engine that enables passing messages to components.
Hyperrelational Navigation Protocol (HRNP), an SAP proprietary protocol that enables Drag&Relate functionality in iViews.
The following figure shows the components in the Portal Framework
The following are the main tasks of the Portal Framework:
Running development tools for creating portal content, such as iViews, pages, roles, and worksets

2. Running iViews and pages

**Portal Runtime (PRT)**

The Portal Runtime (PRT) is the system that gathers and runs iViews. It is open and extensible, allowing changes to its components, and supports different programming and user interface models.

Access to the PRT is done by means of the PRTBridge service and Portal Runtime Container that are implemented in the SAP Web Application Server. The integration between the PRT and the SAP Web Application Server enables the portal to leverage the J2EE standardization and implementation.

In addition, the PRT is a host to Portal Applications. A portal application consists of Java software that is programmed for the portal. The PRT manages the following objects for portal applications:

1. **Portal Components**
   A portal component is an object that can be executed in a portal application. For more information, see Portal Components.*
   The Page Builder component assembles the content of a page according to the defined page layout. For example, when you run the portal for the first time, the Page Builder assembles the initial portal page, and the content of its iViews.

2. **Portal Services**
   A portal service is an object that is available to every portal component. For example, the User Management service connects to the User Management Engine (UME). For more information, see Portal Services.*
   The Portal Runtime container enables the PRT to analyze any URL request, and to find out if a request is intended for the portal. It then handles connection for that request only if it is intended for the portal.

Default functionality in the PRT is that it recognizes only two messaging mechanisms:
A servlet connection - the default connection of the portal
A SOAP connection - for handling SOAP messages

A request consisting of commands, instructs the PRT to perform some specific tasks such as, running a specific routine, or fetching a set of attributes or values for a specific component.

The PRT directly or indirectly obtains two types of requests from the Portal Framework:
A set of queries
A set of commands

A request in the form of queries may ask for a specific portal component, information about that component, and the content for its iViews. For example:
1. Create a static list that describes all the iViews relating to a specific portal component.
2. Check whether a specific user requesting the content of a specific iView is authorized to access the iView.
A logical relationship exists between the portal runtime and iViews. This relationship facilitates separation between the communication requirements of the portal from the implementation of iViews. For example, an iView can be implemented to use any suitable portal component.

**How the Portal Works**

On initializing the portal for the first time, a set of predefined portal applications, such as the security and authentication mechanisms, user management functions, notification and other tools, are loaded by the PRT.

When you start the portal, you initiate a request for an initial page and the content of iViews in that page. Within the page, you can navigate and launch requests for the content of other iViews, or for other pages.

The two types of requests sent to the PRT are for either a page or an iView.

When a request is launched for an iView, the Web server activates the request cycle. There are components responsible for passing incoming requests to the PRT.

The PRT then loads the implementation class and instantiates the corresponding object. Internally the portal application broker checks whether the object is already available and then returns it.

If the object is not yet available, PRT tries to get it from the local deployment. If a new version is available in the repository then the local deployment is updated. Afterwards, the broker loads the class and instantiates the object.

The deployment process gets the PAR file from the repository and deploys it on the file system.

The following is an illustration of the processes that are set in motion in the portal when it receives a request from the browser:

1. When a user launches an iView, the client sends an HTTP or HTTPS request to the Web server.
2. The Web server recognizes the URL as a PRT request and then it triggers the PRT.
3. The PRT parses the request and identifies the following:
   4. Requested object from the PCD, either a page or an iView
   5. User related data, such as personalization data, logon language, page personalization, iView personalization.
   6. Additional pieces of information.
   7. If the user that made the request has permissions for the requested object, the following information is obtained:
      8. Portal component to be executed, for example the Page Builder.
      9. Profile (set of properties) to be passed to the portal component. This may be a set of properties of an iView or page.

The PRT obtains an instance of the portal component to be executed, and provides it together with other information obtained from the PCD.

If the request is for a page, the activated portal component is the Page Builder. The Page Builder component parses the information, assembles the page, and produces the HTML.

If the request is for the content of an iView, the PRT activates the appropriate components, which in turn may use portal services to produce the HTML.

For example, the initial portal page, including its content, are assembled and produced by the Page Builder. Navigation in the portal page can activate a request for either the content of an iView or other pages.

10. The HTML is then returned to the browser.

**Portal Components**

A portal component is custom Java code that is executed according to user requests, and generates HTML output for display on the client. Portal components can interrogate one another, expose their properties, identify other components and respond to events. One portal component can detect the profile and properties of another component, for instance, whether a component is based on Java Server Pages (JSP), or is a Java servlet. When necessary a component can invoke and load the class file of another component.

For example, when a user requests an iView, the portal runtime (PRT) first generates a corresponding portal component, if one does not exist already. The portal component that is invoked is then returned to the PRT, before being passed on to the client.

Usually a portal component serves a specific purpose, such as generating a static list of attributes. Several portal components can be combined to create a page, define the page layout, and fill iViews with content.

There are several portal components, such as Page Builder, which assembles pages, and the logger, which is comprised of user interface messages.

You integrate JSP into a portal component in order to separate the presentation data from the content to be supplied by the Java class.

For detailed information on portal development, go to the SAP Developer Network at [sdn.sap.com](http://sdn.sap.com).

**Page Builder**

The Page Builder component assembles the content of a page according to the defined page layout. For example, when you run the portal for the first time, the Page Builder assembles the initial portal page, and the content of its iViews.

A portal page is comprised of the following:
1. List of possible layouts, and a default active layout
2. iViews located in the page

At design time, each page is assigned one or more layouts. Later, when iViews are added to the page, their placement is determined by the specific active page layout. This information is all stored in the Portal Content Directory (PCD).

SAP Enterprise Portal contains a set of pre-defined page layouts based on layout templates. In addition, new layouts can be developed. The definition of a layout is implemented by proprietary tags used in Java Server Pages (JSP) to define HTML elements for pages. Layout templates are created by portal components, such as HTML-Business for Java (HTMLB). The HTMLB elements manage themes and styles, and provide a full set of easy-to-use Web controls. The Web controls describe the HTMLB controls, their types, usage, and attributes, and how to set the attributes with the JSP-tag libraries and the class libraries.

The Page Builder can assemble iViews in a page as follows:
3. Inserts the output of the iView in the HTML for the page while retrieving the iViews.
4. Alternatively, IFRAMES in the page make the request for the content of iViews either from the portal framework, or from other sources, such as Internet Transaction Server (ITS) or any Web site.

In addition, the Page Builder executes iViews in parallel and manages timeout operations for pages.

**Portal Content Directory (PCD)**

The PCD is the central storage mechanism of the portal. It stores data from portal content objects such as roles, pages, worksets, system landscape, and many more. The details of such data are in portal archive (PAR) files that have been deployed on the portal.

This storage mechanism enables separation of data that describes content, such as, names, fields, values, and so on, from the technical implementation of the content.

From the architectural point of view, the PCD is a logical layer of services that interfaces with portal components on one hand, and with the database shared by the portal and the SAP Web Application Server, on other.

The PCD is comprised of the following:

1. **Semantic Layer**

   The Semantic layer is comprised of services (portal services) that define metadata describing portal content objects; however, the information on the origins of the content is located in the database implemented for the PCD. Also in this layer are portal applications that interface with the portal client. The semantic layer creates data for the generic layer.

2. **Generic Layer**

   The Generic layer (portal services) consists of services that enable the portal to implement the content. The location of various routines, other services, components and portal applications, form part of the implementation details. Origins and sources of specific data, such as code to be implemented, are located in the database.

3. **Persistence layer**

   The Persistence layer (portal services) is also a service that interfaces with the Generic layer and the database for the portal. This service obtains details of objects from the database, parses the information, and then encapsulates it for further processing in the portal at runtime.

   The PCD storage mechanism enables content developers to concentrate on creating and adding business processes to the portal, without worrying about how it is implemented.

   The following illustration shows the various layers in the PCD:

   The PCD is Java Naming Directory Index (JNDI) compliant and features schema support for object type definition for creating object hierarchies.

   As an extension to the JNDI, PCD offers a feature called delta link, enabling one object to inherit properties from another so that changes to the source object are automatically updated in the target object (the delta link object). In short, the PCD simply holds a hierarchy of objects with properties.

   In addition to storage of objects and attributes, the PCD offers several features such as:

4. **Attributes of properties**

   Properties such as personalization can have extended attributes. Such a property can be modified by end users. The attributes control what the portal does with the values.

5. **Personalization**

   Personalization objects are user-based, relating to the specific instance of the object. When a user modifies the object, the changes apply only to that user's instance of the object. The changes do not apply to the same object being used by others.

   Personalized objects are stored in a way that enables personalization changes to be merged automatically when a specific user asks for a specific personalized object, that is, their view of an object.

6. **Internationalization**

   Objects can have 'text' properties, which may have multiple values associated with multiple languages. Thus, when an object is requested, the
The PCD interfaces with several services, such as User Management Engine, Access Control Lists (ACLs), messaging (for cache invalidation notifications), and transport mechanism.

The PCD is capable of working with several portals distributed over several physical machines, and connecting to the same database. It handles the complexity of maintaining a valid single cache for the distributed portals. **Portal Services**

Portal services usually include some combination of programming, data, and resources that are accessible to portal applications from the portal runtime (PRT).

The portal services act as the middleware, that is, interfaces that are enabled to exchange procedures and data. They offer functionality to portal components, and other services.

The PRT enables portal services to be used as web services, and also allows portal components and portal services to access external WEB services via Web Services Description Language (WSDL) files.

The portal services are enabled in the PRT to receive SOAP messages. This implementation includes J2EE Engine, and the Java API for XML Messaging (JAXM) version 1.0.

SOAP messages are included in the HTTP request to the PRT. As such, integrity checks and authentication processes in the PRT, such as the use of HTTPS protocol and single sign-on (SSO) identification are performed on them. When a user does not have the proper permission to services, calls in the SOAP message are denied.

For detailed information on portal development, go to the SAP Developer Network at [sdn.sap.com](http://sdn.sap.com).

The Portal Development Kit (PDK) available on the SAP Developer Network provides APIs that allow developers to enable portal components to access portal services, and to obtain content or invoke other applications.

For example, using an aggregate of portal services, an iView for suppliers can obtain prices from several vendors, submit an order to a specific vendor, track the status of an order at each processing stage until its arrival.

There are two groups of portal services: the services that are part of the PRT, and the external services that extend the procedures and functionality of the PRT.

External services include the following:

- **Client Eventing**
  Enables iViews to communicate with each other on the client side, by means of JavaScript and applets.

- **Logger**
  Used to diagnose problems and to generate log messages.

- **URL generator**
  Provides functions for generating URLs to address.

- **JCO client service**
  Defines all the actions needed to connect to an SAP system through RFC connections.

- **HTML Business for Java (HTMLB)**
  Enables the portal to manage themes and styles.

- **Caching**
  Offers services for caching objects in memory. Caching services take care of cluster notification of cache invalidation.

- **Notification**
  Enables communication among portals in a cluster. It is also used for cache invalidation, and deployment notifications.

- **iView service**
  Enables the creation of new iViews, gets list of objects, such as systems, layouts, templates, and so on.

  It provides the object representing pages, layouts, and iViews (this object is stored in the PCD as JNDI content).

- **Look & Feel**
  Interfaces with the profile for themes and styling elements obtained from the PCD.

- **Application Repository**
  This is an internal service for managing the deployment of portal archive (PAR) files.

- **System Landscape**
  Manages the sources of information, such as find, or create data source.

- **Role, pages, worksets**
  This service connects to the navigation service for roles, including adding and deleting operations. It also represents the object for roles and worksets in the PCD.

- **Web services**
  Enables publication, and invocation of a set of services in the Web server, using XML and standard web-based protocols, such as HTTP for transport mechanisms.

  Via services, the portal has access to the User Management Engine, Access Control Lists (ACLs), external Web services, connectivity to back-end applications, and other API-based functionality.

  The following illustration shows various services exposed in the portal framework:

  There are other services not featured in this figure which play an equally important role in the architecture of the portal.

Portal components such as the Page Builder depend on services at runtime to coordinate and process portal elements, including the following:

1. User management and security tasks, for authentication and Single Sign-On
2. Requested object obtained from the Portal Content Directory (PCD)
3. Objects that manage the theme and styles of the portal
4. Framework for determining wizards in the portal
5. Administration of iViews

### Connector Framework

Connectors enable integration of Enterprise Information Systems (EIS) with application servers and enterprise applications into the Portal Platform. The portal runtime environment supports the creation and running of connectors.

The portal runtime provides various services, including a Web service that makes connectors available to external clients, and allows these clients
to access an enterprise system as if they are local. To do so, the PRT implements the SOAP XML protocol to pass requests and responses between applications, such as iViews, and specific connectors.
In addition, other portal services act as an intermediary for user authentication and security, to make available a wizard framework for the creation of iViews for specific connectors, and their connection to back-end applications using existing connectors.
A software development kit (SDK) is available, offering APIs for creation of local connectors. The SDK is part of the Portal Development Kit (PDK) and can be obtained from the DevZone at: iviewstudio.com

**Unification:**

The unification infrastructure is a set of services that exposes business objects and the relations between them in back-end applications. It enables the portal to interpret, illustrate, and present the relations in the business objects, which represent data in a back-end system.

Using a set of design tools, content developers can create Drag&Relate enabled iViews and facilitate object-based navigation (OBN) for business objects. Drag&Relate, which is based on the relation resolving features in unification, is both a navigation method as well as the technology for sending queries and receiving results in the form of a URL.

From the same application or different applications in a back-end system, the content developer can collect data into an iView that is enabled to carry out Drag&Relate processes using the SAP proprietary Hyper relational Navigation Protocol (HRNP).

The Drag&Relate functionality for an iView is stored as part of the iView's meta-data in the PCD. At runtime, an end user can drag an item from the Drag&Relate enabled iView, and drop it on to another Drag Relate enabled iView to obtain information about the relations, and the related data between the business object and the target iView.

**Unification in the Portal**

Unification in the portal consists of portal services that implement interfaces for use both at runtime and at design time to enable business object modeling, structuring of relations for business objects, and Drag&Relate capabilities in the portal.

The set of portal services and components responsible for unification include tools, such as, the query iView wizard, which is used for creating query-based iViews.

In addition, you can create applications, such as iViews that use the unification services to communicate with back-end systems through unifiers. You can customize the generic behavior of the applications that you create to the requirements of different back-end systems.

The services and interfaces responsible for unification in the portal include:

- **Drag&Relate Service**
  Implements and manages default Drag&Relate processes at runtime. In addition, it provides the main interface for Drag&Relate methods.
  Content developers can modify the default implementation of the DragAndRelate processes.

- **Relation Resolving Service**
  Receives two business objects and finds the relation between them. First, it attempts to find manual relations, and then automatic relations.
  In addition, it uses the Autocomplete service to determine the lightest path between two business objects. Relationship between objects can be based on relations from the back-end application, or autocomplete relations defined by the content developer.

- **Autocomplete Service**
  Finds the lightest path between two business objects within a set of relationships.

- **Unification Controller Service**
  Drives and coordinates the relations resolving processes and the operations.
  It has an interface to the Distributed Query Engine (DQE), which communicates with the JDBC driver to dispatch distributed queries (from multiple data sources), and provide results to the target iViews.
  In addition, the service interfaces with the Relation Resolving (RRS), and Autocomplete services, which work with the actual metadata obtained from back-end systems, and the manually defined relations.

- **Business ObjectSemantic Layer**
  Provides an infrastructure that communicates with the PCD and DQE, and exposes unification objects to the unification services, such as editing, and caching business objects, and their relations.
  From the PCD, the DQE, and the back-end enterprise systems, the business object semantic layer enables interfaces that extract business objects, attributes of unification objects, and information about relations created and defined at design time.

- **Unification Design Time Tools**
Responsible for running the set of tools used at design time to define meta-data such as, relations and business objects for an iView: including iView wizards, relationship editors, and unification elements in the portal administration environment.

The following figure shows some of the services that facilitate unification in the portal.

**Database (Repository):**

The portal shares the same database and schema installed for the SAP Web Application Server (SAP WebAS). In this schema, it stores unprocessed information about objects and properties, including meta-data relating to lists, profiles of property files and their attributes, and other data.

Interfacing with the database is a set of portal services in the PCD that access the stored information and other data needed by the portal at runtime.