**Project Server 2010 Architecture**

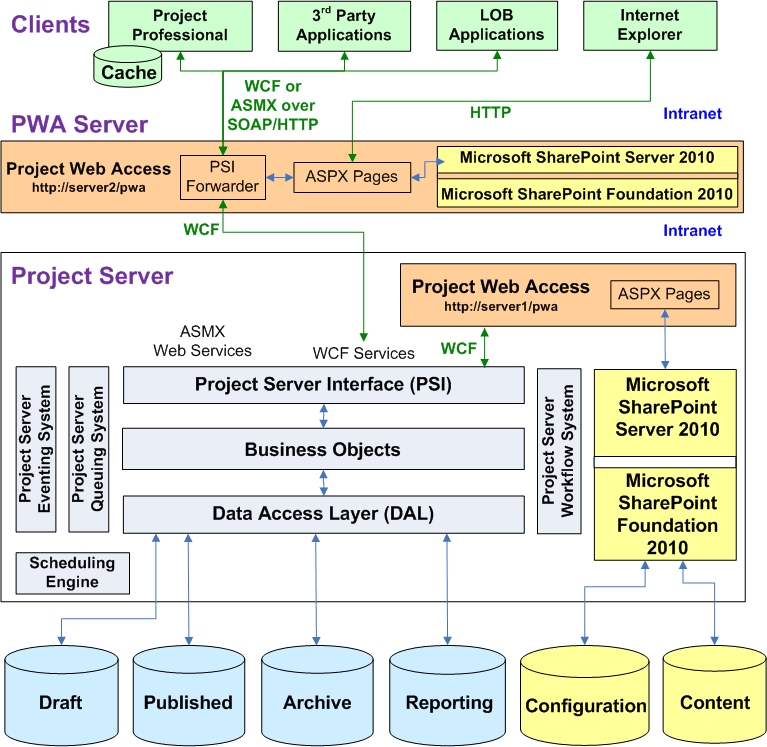
**General Architecture**

The major differences between Project Server 2010 and Office Project Server 2007 are the following:

* Project Server 2010 requires Microsoft SharePoint Server 2010 to be installed on the computer running Project Server and on all Project Web Access servers within a SharePoint farm.
* The PSI includes both the Windows Communication Foundation (WCF) interface and the ASMX interface for Web services.
* The Project Server workflow platform is integrated in the PSI, business object layer, and DAL, and built on Windows Workflow Foundation (WF) in the SharePoint platform.
* Project Server task, assignment, and assignment status data is integrated with Microsoft Exchange Server (not shown in Figure 1), instead of with a Microsoft Outlook add-in.

Figure 1 shows a generalized view of the Project Server 2010 architecture, including an optional Project Web Access front-end server and a Project Web Access instance on Project Server. As in Project Server 2007, there can be multiple instances of Project Web Access running on one server.

Figure 1: General Project Server architecture:



|  |
| --- |
| **Note:** |
| * In Project Server 2010, you can use the WCF or the ASMX interface of the PSI. The Project Web Access and Project Professional 2010 clients both use the WCF interface. * The PSI Forwarder in Figure 1 consists of two components, a WCF Forwarder and a Web Service Forwarder. Clients that use the ASMX interface call the PSI through the Web Service Forwarder. Clients that use the WCF interface call the PSI through the WCF Forwarder. * Figure 1 does not show that a Project Web Access server can be isolated by an additional firewall in a perimeter network (also known as a "demilitarized zone" or DMZ). * The SharePoint Web Services site is part of SharePoint Server 2010 and is not shown in Figure 1. The SharePoint Web Services site includes the Project Service Application with the PSI virtual directory for the ASMX and WCF services. The Project Service Application in Project Server 2010 replaces the Shared Services Application in Project Server 2007. |

The front-end tier includes third-party applications, Project Professional, and Project Web Access. Project Web Access uses Internet Explorer to display Microsoft ASP.NET 3.5 pages. The Project Web Access pages use Project Server Web Parts that communicate with the PSI and also use standard SharePoint Server 2010 Web Parts.

Client applications on separate computers call the PSI through service proxies. External clients that use the WCF interface access the PSI through http://ServerName/ProjectServerName/\_vti\_bin/psi/ProjectServer.svc. Clients that use the ASMX Web service interface use one of the Project Web Access URLs, such as http://server1/pwa/\_vti\_bin/psi/project.asmx. If applications do not have intranet access to Project Server, they can use a Project Web Access server in a perimeter network (not shown in Figure 1).

If they can directly access the Project Server computer, client applications and back-end components of line-of-business (LOB) Web applications can use PSI proxies that use the SharePoint shared service URL to the PSI Web services, such as http://server1:32843/ProjectServiceApplication/project.asmx. Port 32843 is the default port for the SharePoint Web Services application in SharePoint Server 2010. Direct access to the Project Service Application virtual directory should be used only when an application needs to use impersonation or run with elevated permissions.

The middle tier includes the PSI and the business logic layer, which consists of logical business objects that represent Project Server business entities. Business objects include Project, Task, Resource, Assignment, and so forth. The PSI and the business logic tier are tightly coupled and are located on the same server. A client application calls the PSI Web services, and the PSI invokes business objects in the business logic tier.

The DAL provides communication between the middle tier and the database. All Project Server data is stored in Microsoft SQL Server databases. The Project Server databases are factored into the following stores: Draft, Published, Archive, and Reporting. Client applications can read the Reporting database for project data. Clients should use only the PSI to access the Draft, Published, or Archive databases. The Reporting Data Service (RDS, which is not shown in Figure 1) updates the Reporting database from published data in nearly real time. In Project Server 2010, all of the Project Server databases can be located on separate servers.

The Project Web Access components of Project Server also use the SharePoint Foundation 2010 configuration database for project site setup and the content database for project site content such as custom pages, workflows, management settings, documents, and lists of issues, risks, and commitments. The SharePoint configuration and content databases support additional features for project management, such as project templates and workspaces, custom lists for team collaboration, and reports.

**Project Web Access Server**

You can install one or more Project Web Access servers within a corporate intranet to allow load distribution for intranet clients. When a client application uses a separate Project Web Access server, PSI calls are routed through a PSI Forwarder to the PSI Web services on the Project Server computer. The PSI Forwarder (either the WCF Forwarder or the Web Service Forwarder) performs the following functions:

* Optimizes calls to the PSI from remote clients.
* Includes a server-based cache that works with the client-side active cache in Project Professional to reduce roundtrip calls to Project Server.

After a user receives an authentication cookie from Project Server, the PSI Forwarder transparently sends requests to the PSI Web services on the Project Server computer. The PSI Forwarder improves performance and reliability over both the LAN and a WAN.

Project Web Access is developed with ASP.NET 3.5. The visual elements in .aspx files (HTML, server controls, and static text) are separate from the programming logic in code-behind classes that are in compiled assemblies (.dll files). Site pages in Project Web Access, such as the top-level page, Project Center, and Report Center, can be customized by using Web Parts. Application pages that do not have an Edit Page option in the Site Actions menu cannot be edited, such as the Server Settings page and Review Timesheet page.

**Project Server Interface**

The PSI is the API of Project Server. The PSI object model exposes Project Server functionality to all external applications. Project Professional 2010, Project Web Access, LOB, and other third-party applications use the PSI to access Project Server data in the Draft, Published, and Archive databases. The PSI is available through WCF services and through ASMX Web service calls by back-end LOB applications, or through a PSI proxy.

Web methods in the PSI typically produce or consume typed DataSet objects as the means to exchange information with the business objects. The PSI reference includes DataSet documentation.

**Business Objects**

The internal object model of Project Server includes the business objects. Client applications access business objects only through the PSI, and only business objects can call the DAL.

Business objects are logical entities that can be classified into three types:

* Core entities are objects such as projects, tasks, assignments, resources, and calendars. The core entities include basic business logic such as permissions and naming rules.
* Business entities are objects such as timesheets, portfolios, and models. Business entities include additional business logic and usually are built from a combination of the core entities.
* Support entities are objects such as security and validation.

Developers need not be directly concerned with business objects. The PSI handles mapping of the API to business objects.

**Data Access Layer and Databases**

The DAL is internal to Project Server and is not exposed to external applications. The DAL translates between the logical business entity representation of the data and the physical database tables. Each logical entity is stored in a number of different tables. The DAL encapsulates the work required to manage connections, execute queries, and begin/commit/roll back transactions.

Project Server data is partitioned into four databases in SQL Server.

* The Draft database contains tables for saving unpublished projects from Project Professional and other applications. Project Web Access does not show project data in the Draft database.
* The Published database contains all of the published projects and enterprise resources, the enterprise global template, and other project templates. Published projects are visible in Project Web Access. The Published database also contains tables that are specific to Project Web Access (timesheets, models, views, and so on), and global data tables (custom fields, lookup tables, security, and metadata).
* The Archive database saves backup versions of projects and other data.
* The Reporting database (RDB) is the staging area for generating reports and OLAP cubes. Data in the Reporting database is comprehensive and is updated nearly in real time. The tables and views are optimized for read-only report generation; for example, the RDB tables are denormalized to provide redundant data and reduce the number of relational tables.

Entities such as Resource or Project can span multiple tables, and all tables for a particular entity have the same primary key. The primary key is a single column that uniquely identifies one instance of a particular entity. Unique identifiers are GUIDs.

Only the Reporting database schema is documented. You should access the Draft, Published, and Archive databases only through the PSI. You can add data tables, fields (properties), and entities that are not defined in the Project Server 2010 database schema to the Reporting database. If you add tables to the core databases, you must also provide the full stack of a custom assembly, Web service, business objects, and data access. You can easily modify the Reporting database; we recommend that you do not modify the core Project Server databases.

**Publishing and Server-Side Scheduling**

Project Server 2010 supports both manual and automated project schedule updates. The default process is to update projects manually. That is, the project manager opens the project in Project Professional, applies the changes, and then saves and publishes the project to make the changes available to everyone. The scheduling engine in Project Professional calculates scheduling changes for manual updates.

The scheduling engine in Project Server enables automated project updates by using the PSI. Project Server allows the published version of a project to be updated while a project manager is using the draft version, using the following steps:

1. Project Server applies updates and reschedules the published version automatically.
2. Project Server saves the update to apply to the draft version when either of the following events occur:
   * Project Professional opens the project.
   * Project Professional tries to publish the project.
3. If there is a conflict, the project manager must resolve it before the draft version can be published.

|  |
| --- |
|  |
|  |

**Project Server Programmability:**

# Project Server and SharePoint Server

Project Web Access is built upon SharePoint Server 2010, and uses master pages and Web Parts to make it easier to build custom Project Web Access solutions. SharePoint Server 2010 also provides core infrastructure services for Project Server such as administration, security, the reporting center using Excel services, and common workflow activities. Microsoft SharePoint Foundation 2010 hosts the Windows Workflow Foundation run-time engine.

Project Server 2010 integrates deeply with SharePoint Server 2010 as the platform for project collaboration, reporting, site administration, and workflow management. Project workspace sites include specialized SharePoint lists for tracking issues, risks, project deliverables, and the team calendar, along with the document library and team discussions. Project Server 2010 provides flexibility for team collaboration; for example, you can locate project workspaces anywhere within the SharePoint server farm where Project Server is installed. To use other core services of SharePoint Server 2010, such as Excel Services and Enterprise Search, an administrator needs to enable and configure the services.

Project Server 2010 installation also provisions the Project Service Application in the SharePoint Web Services site, which includes the local ASMX Web services and Windows Communication Framework (WCF) services for the PSI. Shared service applications in SharePoint Foundation 2010 replace the Shared Service Provider in Windows SharePoint Services 3.0. The change enables third-party developers to create custom shared service applications using the Service Application Framework. In addition to the Project Service Application, other examples of shared service applications include SharePoint Search and SharePoint Document Management. For more information, see the SharePoint Server 2010 SDK.

The Project Service Application is a logical service provider that handles one or more instances of Project Web Access. Project Server provisioning creates a specific Project Web Access site within a Web application. Project Web Access contains the Project portal pages (Project Center, Resource Center, Report Center, and so on). For an illustration of the Microsoft Internet Information Services (IIS) view of the Project Service Application for a typical installation, see [Overview of WCF and the PSI](mk:@MSITStore:D:\\Documentation\\pj14SDK_beta.chm::/html/1bceeeef-269b-4c80-b9ab-09608b0ddee6.htm).

### Workflow Engine

Windows Workflow Foundation (WF) is a Microsoft .NET Framework 3.0 technology. The WF run-time engine hosted in SharePoint Foundation 2010 is the workflow engine for SharePoint Server 2010 and is also part of the core Project Server platform.

Integrated workflow solutions help to automate an organization's business processes and can provide tracking, auditing, and alerts. Examples of workflow solutions include change management and demand management systems, specialized timesheet applications, and integration with other LOB applications such as customer relationship management.

SharePoint Foundation 2010 builds on the WF platform introduced in Windows SharePoint Services 3.0, and adds the following features:

* New workflow activities, which are the building blocks of all workflows.
* Pluggable workflow services enable workflows to interact with external sources.
* New workflow pre- and post-events. For example, one workflow can call another workflow and wait for it to finish or respond to an error.
* Workflow event receivers can be registered with a specific SharePoint site, list, or content type, to limit the scope.

WF provides an extensible workflow run-time engine that enables partners, ISVs, and customers to build custom business workflow solutions (workflows) on Project Server. Project Server adds workflow activity and integration classes in the Microsoft.Office.Project.Server.Workflow namespace. Developers can create custom workflows using Microsoft Visual Studio 2010.

|  |
| --- |
| **Note:** |
| Microsoft SharePoint Designer cannot create Project Server workflows. We recommend Visual Studio 2010, although you can also use Visual Studio 2008. See [Developing Project Server Workflows](mk:@MSITStore:D:\\Documentation\\pj14SDK_beta.chm::/html/735bbb04-a8c1-46c0-a346-42050f0ac9b1.htm). |

For more information about SharePoint workflows, see the SharePoint Foundation 2010 SDK.

# General PSI Features

The Project Server Interface (PSI) enables all client applications, including Project Professional 2010, Project Web Access, and back-end LOB applications, to access Project Server data. The PSI is built and programmed with the Microsoft .NET Framework 3.5 and provides all of its advantages, such as ease of development, security, error handling, and garbage collection.

### Types of APIs

All of the PSI is accessed through ASMX Web services or WCF services. Each PSI service typically contains a base class with "CRUD" methods that create, read, update, and delete items within that class. Items are specified by related DataSet classes. For example, the CustomFields service contains the CustomFields class with methods such as CreateCustomFields. Data for one or more enterprise custom fields are specified in the CustomFieldDataSet.

The PSI contains three general kinds of APIs:

* Generic APIs that expose the Project Server functionality with an emphasis on ease of use and organization for third-party developers. These APIs are well-documented. There are 22 generic services in the PSI, with base classes such as Project, PortfolioAnalyses, and TimeSheet. In addition, the LoginForms and LoginWindows Web services provide generic methods for programmatic Project Server logon and logoff.
* Project-specific APIs that are geared for performance or special purposes such as upgrading from Project Server 2007 and synchronizing with Microsoft Exchange Server. The Project-specific APIs may combine or batch multiple API calls. The Authentication, ExchangeSync, P12Upgrade, PWA, and View services are designed specifically for use by Project Web Access and Project Professional.
* APIs that are limited to use only by Project Professional. The WinProj class includes methods with parameters that can contain large amounts of binary data.

Project Web Access and Project Professional use the generic PSI services as well as the limited and Project-specific APIs.

|  |
| --- |
| **Note:** |
| The Authentication, ExchangeSync, P12Upgrade, PWA, View, and WinProj services are not documented or supported for third-party development. These interfaces can change in service packs or later versions of Project Server. |

The PSI is factored to match the business objects. That is, each PSI method is associated with a business object such as Calendar or Resource. The PSI is the interface to the business objects. Because the business layer provides reusable business logic components, different applications that interact with Project Server data use the same business logic. The PSI helps to solve the problems in Project Server 2003, where developers sometimes had to re-implement Project Server business logic in PDS extensions.

Each PSI method is implemented with a separate interface that interacts with the client by using strongly typed data. For example, the QueueCreateProject method in the Project service accepts the dataset parameter of type ProjectDataSet. The ProjectDataSet class is derived from the DataSet type. Type checking in the .NET Framework and Intellisense completion in Visual Studio help to reduce errors in development with the PSI.

Project Server 2010 uses exception handling of the .NET Framework. All errors are logged in the server, at the top of the PSI stack. Some errors send a simple report to the client, such as a SoapException object that is recorded in the application event log, and some errors also record a detailed report on the server in the Unified Logging Service (ULS) trace logs.

The PSI is also extensible. You can add a .NET assembly with its own Web service that provides new functionality, uses the same Project Server security infrastructure, and calls other PSI methods or inherits from PSI classes. A PSI extension also needs to provide the business logic and database access needed for the new functionality.

For an introduction to the detailed reference for PSI namespaces, classes, methods, properties, events, and related assemblies, see [PSI Reference Overview](mk:@MSITStore:D:\\Documentation\\pj14SDK_beta.chm::/html/d3c33089-0cbe-48c3-bfc0-0be819ca4d73.htm).

### Porting Applications Built for Project Server 2003

In Project Server 2003, much data and functionality is available only with Project Professional 2003 or by direct database access. The PSI, introduced in Project Server 2007, removes much of that restriction. Unlike the Project Data Service (PDS) in Project Server 2003, the PSI provides a comprehensive interface to business objects in Project Server.

Applications developed for the Project Data Service (PDS) in Project Server 2003 are not compatible with Project Server 2010. The PSI provides functional parity for the PDS, but does not match PDS methods or parameters. To port a PDS application to Project Server 2010, you must do the following:

* Set a WCF reference to http://ServerName/ProjectServerName/\_vti\_bin/psi/ProjectServer.svc, or set a Web reference to each PSI Web service you need, such as http://ServerName/ProjectServerName/\_vti\_bin/psi/resource.asmx.
* Change all PDS calls to PSI calls
* Change the processing of PDS XML requests and replies to processing of the PSI DataSet objects and other parameters.

The PSI does not support PDS extensions that were built for Project Server 2003, because those extensions usually use direct database access. For more information about PDS compatibility and guidelines for porting PDS extensions, see [PDS Parity in PSI Web Services](http://msdn.microsoft.com/en-us/library/ms197081.aspx" \t "_blank).

#### Project Versions

Project Server 2003 used project name extensions for different versions of a project. To keep track of project versions in Project Server 2010, you can do following:

* Create a Version custom field and lookup table, available through the CustomFields and LookupTable PSI Web services, to access project level metadata.
* Use task-level publishing flags and project-level availability settings to control the impact of a project on the organization.

# Project Server Events

In Project Server 2003 and earlier versions, SQL Server triggers are the only way to invoke custom code based on events. Triggers require knowledge of Project Server internal processes, and can reduce system performance. Project Server 2010 provides public events that enable development of custom processes such as adding and enforcing business rules, validation, data processing, notification services, and workflow. Server-side events greatly increase the value of Project Server as a development platform.

The Project Server eventing platform includes the Project Server Eventing Service and a Project Web Access administration page for viewing a list of events and registering application event handlers. With the Project Server eventing platform, developers can:

* Extend Project Server functionality with custom event handlers and associate them with events raised by Project Server.
* Send data to other applications or receive data from applications using event handlers.
* Cancel a Project Server process, with a pre-event handler that implements custom business logic.
* Associate multiple event handlers with an event, and determine the sequence of execution.
* Invoke workflows and send or receive messages from a workflow.

Client applications can raise server-side events when they interact with Project Server 2010. Clients can interact with Project Server through the PSI and through the workflow platform in SharePoint Foundation 2010. PSI calls invoke business objects that act as event sources. Third-party developers can also override base abstract classes for event handlers. The assembly needs to make only the class name, not the method name, available to the Project Server eventing platform. The implementation class also enables type checking when registering an event handler and during run time.

Project Server does not provide transactions across an event. To help insulate the core Project Server process from badly written event receivers, all event receivers run in a different application domain than Project Server. Event handler crashes do not affect the Project Server process. It is the developer's responsibility to manage exceptions, perform cleanup, and maintain data consistency if an exception occurs.

# Project Scheduling on the Server

In general, Project Server scheduling works the same way as scheduling in Project Professional. However, there are a few differences and limitations in server-side scheduling.

* Earned value not calculated   The scheduling engine on Project Server does not calculate the earned value fields: ACWP, BAC, BCWP, BCWS, CPI, CV, CV%, EAC, SPI, SV, SV%, TCPI, VAC, Duration Variance, Start Variance, Finish Variance, Cost Variance, and Work Variance. If a project has values for these fields and the project is updated on the server, the field values do not change. However, you can duplicate some of the earned value fields by using custom fields, such as the variance fields. The Project Server scheduling engine does update custom fields.
* Single project scheduling only   Project Server schedules only the current project, when changes are made through task status updates, changing projects with the PSI, or with Project Web Access. If the current project has links to other projects, subprojects, or master projects, the linked projects are not changed.

You can handle the server-side scheduling limitations in the following ways:

* Open projects in Project Professional and save them back to Project Server.
* In reports, do not include fields that are not updated by Project Server.
* Add a note in reports about data that may be stale.

There are flags in the Reporting database and the cubes that allow you to detect when some project data is not updated. For information, see the MSP\_EpmProject table in the Reporting database schema reference (ProjectServer\_ReportingDB.chm in the Project 2010 SDK download).

* ProjectWbsIsStale   Indicates whether the work breakdown structure (task outline hierarchy) is stale.
* ProjectEarnedValueIsStale   Indicates the earned value fields are stale.
* ProjectRollupsAreStale   Indicates that a subproject is updated in the Draft database, but the master project is not updated. The rolled-up values from the subproject are stale.
* ProjectHierarchyNotSynchronized   The master project is not synchronized with its children. That happens when the child projects are published explicitly, not as part of the master project publishing.
* ProjectCalculationsAreStale   Project Professional saved a project without calculating the schedule (that is, the calculation mode is set to Manual on the Schedule tab in the Project Options dialog box).
* ProjectGhostTaskAreStale   Similar to ProjectHierarchyNotSynchronized, but warns on cross-project link data. It is possible that no master project exists, but the project data on one side of the link is newer than on the other side.

Following are the other differences between the way scheduling works in Project Server and in Project Professional:

* Roll-down of actuals   Project Server 2010 and Project Professional 2010 both roll down actual values for material resource assignments to the material resources. In contrast, Project Server 2007 rolls down actuals for material resource assignments to work resources.
* Summary tasks   Summary tasks are generally read-only on Project Server. For example, assignments for summary tasks cannot be created, and percent completion cannot be modified. However, Project Server does support editing the dates and duration of manually scheduled summary tasks.

Actuals on Project Server are not added automatically to a summary task assignment, because that would bypass the approval process in Project Server. In Project Professional, when you add actuals to a subtask, the actuals are also added for an assignment for the summary task. The difference in behavior can be confusing for a user.

Project Server deletes actuals on a summary task assignment if the subtask duration shortens or the finish date is changed. Project Professional 2010 changes the roll down of actuals and addition of actuals to summary task assignments to match the behavior of Project Server.

|  |
| --- |
| **Caution:** |
| Although Project Professional can do it, we recommend that you do not make assignments on summary tasks. |

* Inactive tasks   Project Server cannot deactivate or reactivate tasks.
* Outline numbers   Project Server does not calculate task outline numbers.
* WBS values   Project Server does not update work breakdown structure values.
* Resource overallocation   Project Server does not correctly calculate values for the overallocated flag.

# About Accessing the Project Server Databases

Developers are strongly discouraged from directly accessing the Draft, Published, or Archive Project Server database through Microsoft SQL Server queries. Making direct changes in the Project Server database tables can damage referential integrity and interfere with database access through the Project Server Queuing Service.

|  |
| --- |
| **Important Note:** |
| There is nothing to actively prevent you from using direct programmatic database access to update data. You should be aware that the Project Professional cache, the Publishing database and the Reporting database all rely on a cache synchronization protocol that can be disrupted by direct data editing. If you damage your Project Server databases or corrupt Project Professional client-side caches by using direct access to change data, be warned that product support won't be able to help! |

Applications that directly access the Draft, Published, or Archive database also are dependent on the database schemas, which can change in service packs or later versions of Project Server 2010. Furthermore, applications that directly access the databases lose the built-in Project Server security, common business logic, tracking, audits, error checking, reporting, workflow, and other features. You would likely need to rewrite such an application for Project Server 2010 updates.

For all of these reasons, Project Professional and Project Web Access do not make direct calls to the Draft, Published, or Archive database; neither should any other application that integrates with Project Server.

The Draft, Published, and Archive database schemas are not documented. You can use the Reporting database to help generate reports, and the Reporting database schema is documented in the Project 2010 SDK download. You can also extend the Reporting database to include external data for integrated reporting solutions.