# SQL08r2_h_rgb.png

SQL Server Technical Article

**Writers:** Geoff Allix

**Technical Reviewers:** Joanne Hodgins, Omri Bahat, Morgan Oslake

**Published:** February 2010

**Applies to:** SQL Server 2008 R2, August CTP

**Summary:**

SQL Server 2008 R2 introduces new management tools to help improve IT efficiency and productivity. Investments in application and multi-server management will help organizations proactively manage database environments efficiently at scale through centralized visibility into resource utilization. Such investments can help streamline consolidation and upgrade initiatives across the application lifecycle—all with tools that make it fast and straightforward.

This paper introduces the new extensions in SQL Server Management Studio and the new Utility Explorer, and it discusses the simple process of setting up a SQL Server managed server group, including Control Point installation, enrolling an instance into central management, and upgrading SQL Server databases to SQL Server 2008 R2 using the data-tier application component.

Copyright

This is a preliminary document and may be changed substantially prior to final commercial release of the software described herein.

The information contained in this document represents the current view of Microsoft Corporation on the issues discussed as of the date of publication. Because Microsoft must respond to changing market conditions, it should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication.

This white paper is for informational purposes only. MICROSOFT MAKES NO WARRANTIES, EXPRESS, IMPLIED, OR STATUTORY, AS TO THE INFORMATION IN THIS DOCUMENT.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in, or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in any written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Unless otherwise noted, the example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

© 2010 Microsoft Corporation. All rights reserved.

Microsoft, PowerShell, SQL Server, Visual Studio, and Windows are trademarks of the Microsoft group of companies.

All other trademarks are property of their respective owners.

Contents

[Executive Summary 4](#_Toc252974419)

[Introduction 4](#_Toc252974420)

[New Terms 4](#_Toc252974421)

[Introducing the SQL Server Managed Server Group 5](#_Toc252974422)

[Creating a SQL Server Managed Server Group 7](#_Toc252974423)

[Scenario 7](#_Toc252974424)

[Setting Up a SQL Server Control Point 8](#_Toc252974425)

[Enroll an Instance into the SQL Server Control Point 8](#_Toc252974426)

[Understanding the Data within the SQL Server Control Point 8](#_Toc252974427)

[Administering a SQL Server Managed Server Group 10](#_Toc252974428)

[Monitoring Policies 10](#_Toc252974429)

[Utility Management Data Warehouse 10](#_Toc252974430)

[Considerations 10](#_Toc252974431)

[Data-tier Application Overview 11](#_Toc252974432)

[Advantages of Registering a Data-Tier Application to the Control Point Utility 12](#_Toc252974433)

[Data-Tier Application Upgrade Efficiencies 12](#_Toc252974434)

[Conclusion 14](#_Toc252974435)

# Executive Summary

Databases are increasingly used by applications and along with the data explosion, this had led to a proliferation of databases and additional complexity for database administrators (DBAs) as they try to manage the growing numbers of databases created by application developers. This increase in applications and SQL Server instances, combined with low-cost/high-storage hardware, has led to server sprawl across the organization with hundreds of servers at less than 2Gb utilization\* (footnote: \* <2GB is based on Microsoft customer surveying)

Application and multi-server management provides the DBA with the tools necessary to gain centralized insights into instance and database application utilization as well as a better way to develop, deploy, and manage data-tier applications.

# Introduction

Microsoft’s investment in application and multi-server management will help organizations manage database environments more efficiently at scale with visibility into resource utilization for consolidation and improved efficiencies across the application lifecycle. A core concept to application and multi-server management is the addition of the Utility Control Point, which enables a centralized view of Microsoft® SQL Server® instances and database applications and their utilization across the designated managed server group.

What’s more, for centralized SQL Server management to provide incremental value, DBAs and developers need a single unit of deployment for their database applications to accelerate changes, upgrades, and deployments. Having the ability to more easily package and move database applications is especially important for streamlining the tasks associated with consolidation initiatives. To this end, SQL Server 2008 R2 introduces a new concept, the Data-tier Application. A Data-tier Application is a container that defines and packages database schema and deployment requirements of an application into a single file.

This paper introduces the new extensions in SQL Server Management Studio, the Utility Explorer and Utility Control Point, and the new single unit of deployment concept, the Data-tier Application. The paper discusses the simple process of setting up a SQL Server managed server group, including setting up a Utility Control Point, enrolling an instance into central management, extracting Data-tier Applications from existing deployments, and deploying Data-tier Applications to the new managed server group.

## New Terms

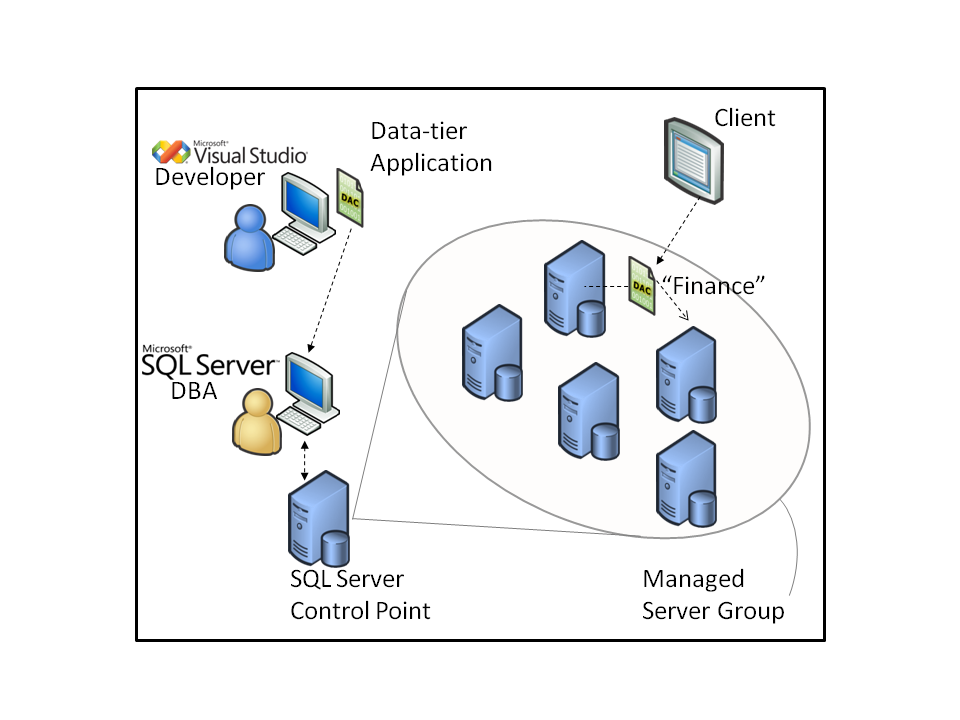
**Utility Explorer** – Accessed from SQL Server Management Studio, the Utility Explorer serves as the entry point to the multi-server management enhancements made available by the Utility Control Point.

**Utility Control Point** – Accessed via Utility Explorer, a Control Point is a SQL Server instance designated to maintain relationships with enrolled SQL Server instances within a managed server group.

*\* <2GB is based on Microsoft customer surveying*

**Managed Server Group** – Describes a group of SQL Server instances enrolled into a Control Point, where utilization and configuration data is collected and accessible through the dashboard viewpoints in the SQL Server control point.

**Data-tier Application (DAC)** – Interoperability with the Microsoft Visual Studio® development system introduces a new project template called Data-tier Application (DAC). This project template captures the database application schema (tables, stored procedures, and so forth) and packages it with application deployment requirements, enabling a single unit of deployment. The DAC serves as the file read by the new wizards within the Object Explorer that will unpack the application schema and deployment requirements for deployment as the Data-tier Application. Within Management Studio, SQL Server can also extract a DAC from an existing database.



**Figure 1:** Application and Multi-Server Management concepts

## Introducing the SQL Server Managed Server Group

Customers have an increasingly important requirement to manage their SQL Server environment as a whole, focusing more on managing all of their Data-tier Applications and less on managing individual computers and instances of SQL Server. SQL Server 2008 R2 addresses this requirement through the concept of the SQL Server managed server group. This models an organization’s SQL Server-related entities in a unified view. Entities that can be managed include instances of SQL Server, data-tier applications, database files, processor utilization, and storage utilization. This new way to organize and monitor SQL Server resource capacity enables administrators to have a holistic view of their environment.

The managed server group is managed through a Utility Control Point using the new Utility Explorer in SQL Server Management Studio (SSMS). The Control Point is configured on a SQL Server instance and provides the central reasoning point for a managed server group. It contains configuration and performance information collected by managed instances of SQL Server, and it stores this information in a central management repository. SQL Server configuration settings and performance data are collected and then compared to policy evaluation results on the Control Point to help administrators identify resource utilization bottlenecks and consolidation opportunities. The Control Point also contains data used for impact analysis and what-if scenarios.

The SQL Server managed server group model contains three layers:

 **Data-tier Applications**—the data-tier applications managed by the organization.

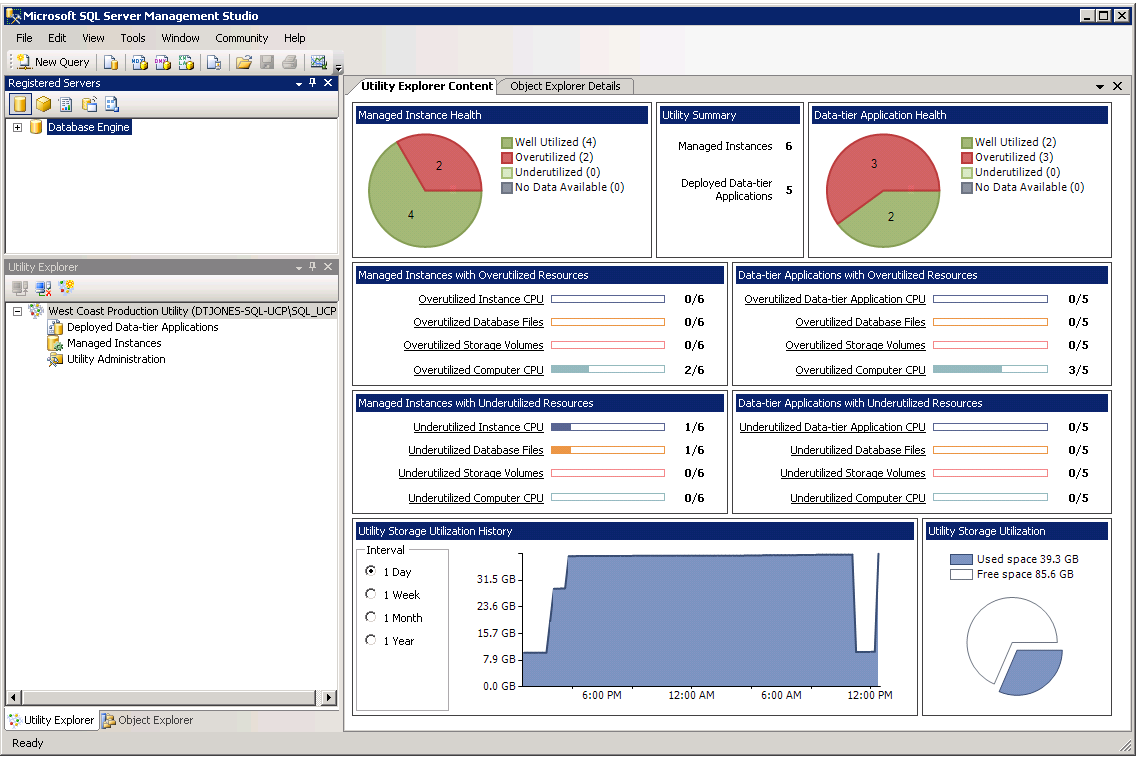
 **SQL Server Runtimes**—the instances of the Database Engine used by the organization.

 **Hardware Resources**—the resources used by the SQL Server Runtimes, like computers and disk storage systems.

A Data-tier Application is a container that defines and bundles the database schema, reference data, and deployment requirements of an application. The Data-tier Application forms a file that enables a single unit of deployment, for the full lifecycle of an application, including versioning. It further enables data-tier automation by providing a means to capture the intent of the developer and deployment-specific details. It abstracts the application data-tier by providing well-known endpoint names instead of computer and instance names, so a data-tier application can be moved between SQL Server runtimes without requiring application changes.

The SQL Server utility makes it easy to create the Control Point, enroll instances into the Control Point, and configure resource utilization policies for the data-tier applications and SQL Server instances.

The Utility Explorer within the SQL Server Management Studio user interface provides a hierarchical tree view for navigating through and managing the entities in the SQL Server managed server group. This contrasts with SQL Server Management Studio Object Explorer as Object Explorer displays each instance as a completely independent object at the top of the hierarchy. Viewpoints and dashboards provide views into the capacity of the elements in the managed server group.



**Figure 2:** Dashboard of instance and application resource health

# Creating a SQL Server Managed Server Group

This section will explain scenarios where a managed server group would be useful, describe setting up a SQL Server instance as a Control Point, explain new concepts and terminology associated with the Utility Explorer and a SQL Server control point, and explains the process of enrolling a SQL Server instance into the managed server group for insights into resource utilization.

## Scenario

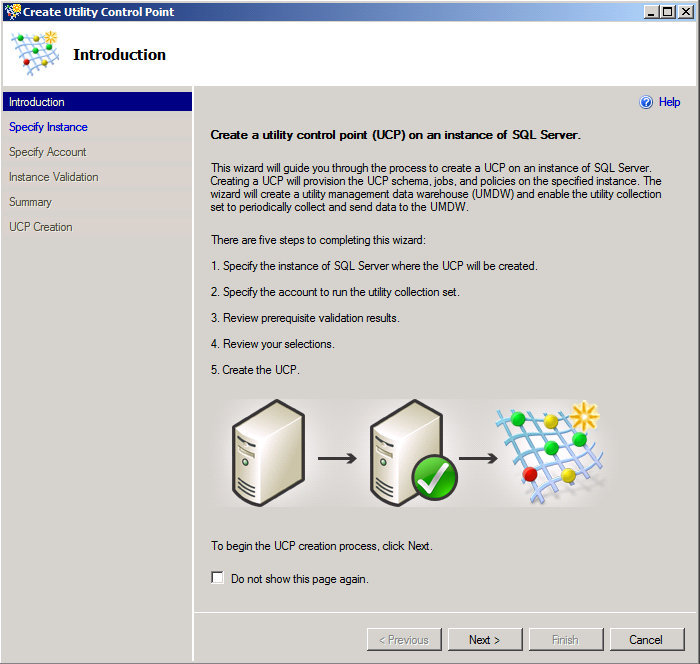
A team of DBAs from an investment bank need to manage many SQL Server instances. These instances have traditional databases and also databases whose only task is to support a specific application. To reduce power consumption, hardware costs, and to provide flexibility, the investment bank has consolidated multiple virtualized guest operating systems onto each host server using Microsoft Hyper-V™.

They have several issues at the moment. They would like to monitor instance utilization so that they can actively manage resource allocation in Hyper-V. They would like to monitor instance under-utilization to decide where to create new databases; and they would like a straightforward method to package and deploy databases and any object dependencies that are used by applications.

By installing SQL Server 2008 R2, they can now use Application and Multi-Server Management to monitor instance utilization and they can support the packaging and deployment of databases for applications without leaving the familiar SQL Server Management Studio interface.

## Setting Up a Control Point

In order to create the Control Point, you can invoke the Create Control Point Wizard in SQL Server Management Studio. The wizard creates a control point on an instance of SQL Server. This creation process includes provisioning the control point schema, jobs, and polices as well as creating a management data warehouse.



**Figure 3:** The Introduction page of the Create Control Point Wizard

## Enroll an Instance into the Control Point

After the control point is created, you can enroll instances that you want to manage.

In order to enroll an instance in the Control Point, you can invoke the Enroll Instance Wizard from the Control Point Explorer pane in SQL Server Management Studio. In the Control Point Explorer, right-click **Managed Instances**, and then click **Enroll Instance**. The Enroll Instance Wizard enrolls a SQL Server instance as a managed instance in the control point. This enrollment process will start the managed server group collection set, which will upload data to the control point once every 15 minutes.

You can also do a batch enroll using PowerShell, enrolling hundreds of instances at once to streamline the process.

## Understanding the Data within the Control Point

**Summary view** – Pie charts at the top of the dashboard provide at-a-glance summaries of resource health for managed SQL Server instances and data-tier applications. The summary at the top-center of the dashboard displays the total numbers of managed SQL Server instances and data-tier applications in the managed server group.

In the dashboard summary for managed instance health, a SQL Server instance is marked as overutilized if any of the following conditions are true:

* CPU resources for the instance of SQL Server are overutilized.
* CPU resources of the computer that hosts the SQL Server instance are overutilized.
* The instance contains data or log files with overutilized storage space.
* The instance contains data or log files that reside on volumes with overutilized storage space.

In the dashboard summary for managed instance health, a SQL Server instance is marked as underutilized if it is not marked as overutilized and any of the following conditions are true:

* CPU resources allocated to the instance of SQL Server are underutilized.
* CPU resources of the computer that hosts the SQL Server instance are underutilized.
* The instance contains data or log files with underutilized storage space.
* The instance contains data or log files that reside on volumes with underutilized storage space.

In the dashboard summary for managed instance health, a SQL Server instance is marked as well utilized if it is not marked as overutilized and it is not marked as underutilized.

Similar rules are used in the dashboard summary for data tier application health.

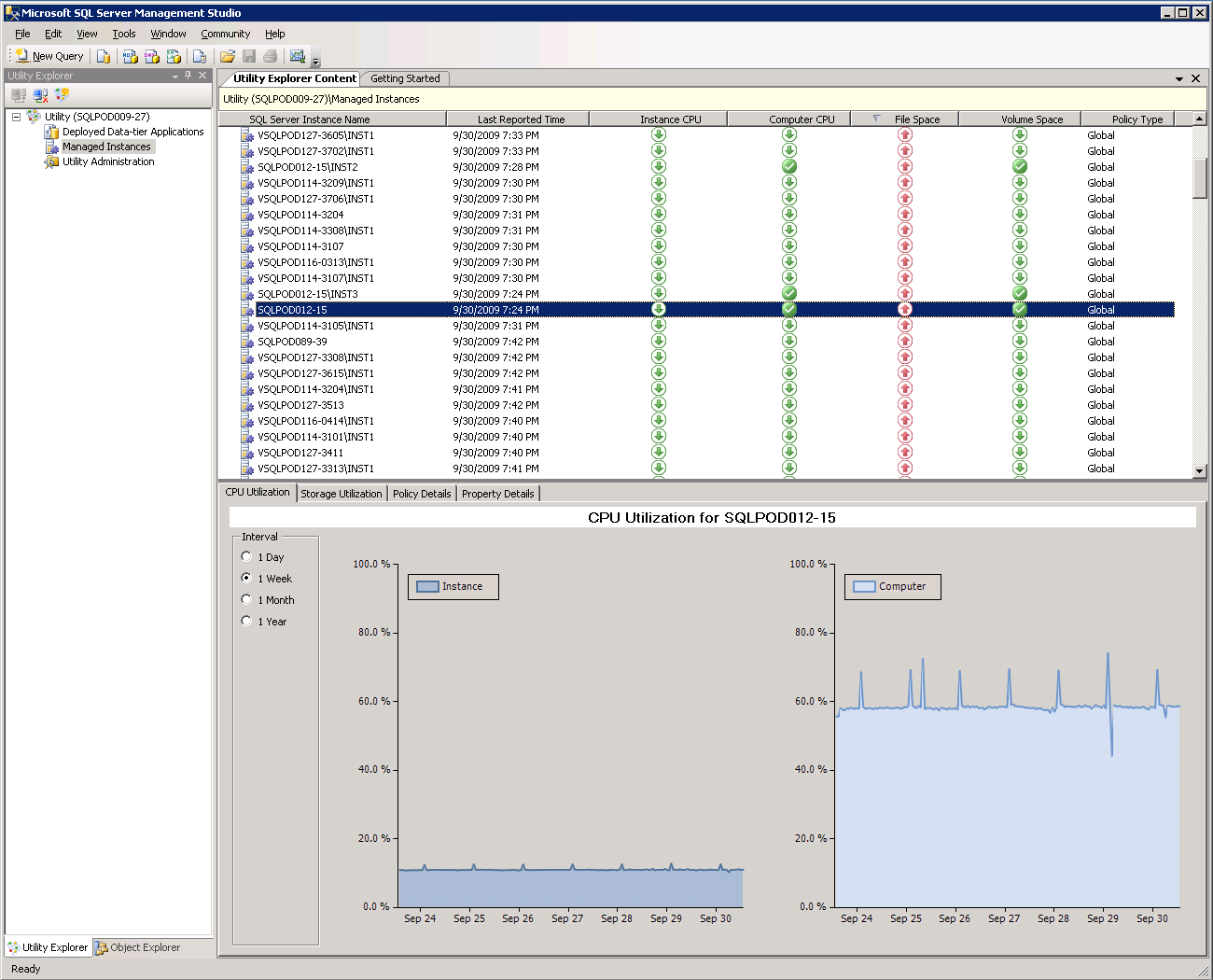
**Rollup view** - Sliding gauges below the pie charts show a summary of the number of managed instances of SQL Server and data-tier applications for each resource utilization dimension – for example, CPU utilization for instances of SQL Server, CPU utilization for the entire computer, file space utilization, and storage volume space utilization. Each of these rollup summaries is a link which you can click to seamlessly navigate to the detailed list view, which is filtered based on the rollup category.

**Storage Utilization view** – Graphic representations at the bottom of the dashboard show an aggregation of current utilization and utilization history of disk space use of storage resources across the entire managed server group.

**Detail view** – To view detailed information about resource utilization for specific computers, instances of SQL Server, and Data-tier Applications, navigate to the Utility Explorer navigation pane and then click **Managed Instances** or **Deployed Data-tier Applications**. The detailed list view displays the health states for data-tier applications and SQL Server instances across key resource utilization dimensions.

These dimensions include processor utilization and storage space utilization. The health states in the viewpoints represent either underutilized resources marked with a green down arrow icon, overutilized resources marked with a red up arrow icon, or resources that are neither underutilized nor overutilized marked with a green check icon. The underutilization and overutilization of a given resource are defined by resource utilization policies. The underutilization policy defines the underutilization threshold, and the overutilization policy defines the overutilization threshold – these policies have default settings that are easily adjustable with spinner controls.

The list view also has tabs with details about processor utilization, storage space utilization, and deployment properties for every managed instance of SQL Server and data-tier application in the managed server group.



**Figure 4:** Detailed list view of managed instances

# Administering a SQL Server Managed Server Group

## Monitoring Policies

Control Point health policies can be configured for data-tier applications and managed instances of SQL Server. Health policies can be defined globally for all data-tier applications and managed instances of SQL Server in the Utility Administration from within Utility Explorer, or they can be defined individually for each data-tier application and for each managed instance of the managed server group.

Using the Policy tab within Utility Administration, you can define global policies. For example, you could set the maximum and minimum values for managed instance processor utilization. This defines the levels at which instances are reported as over or under-utilized. These settings are defaults and each individual instance can have a specific setting. Furthermore, you can set global and individual policies for data-tier applications.

## Utility Management Data Warehouse

Built on Management Data Warehouse delivered in SQL Server 2008, data collected by managed instances of SQL Server are stored in the utility management data warehouse (UMDW) and you can configure the UMDW data retention period using Utility Administration.

## Considerations

In this release, the Control Point and all managed instances of SQL Server must satisfy the following requirements:

* SQL Server must be version 10.50 or higher.
* The SQL Server instance type must be Database Engine.
* The SQL Server Control Point Utility must operate within a single Windows domain, or domains with two-way trust relationships.
* The SQL Server service accounts on the Control Point and all managed instances of SQL Server must have read permission to Users in Active Directory.
* On Windows Server 2003, the SQL Server Agent service account must be a member of Performance Monitor User group.
* We recommend that the UCP is hosted by a case-sensitive instance of SQL Server.
* If the Control Point is hosted by a case-insensitive instance of SQL Server, then managed instances of SQL Server should also be case-insensitive.

In this release, the Control Point must satisfy the following requirement:

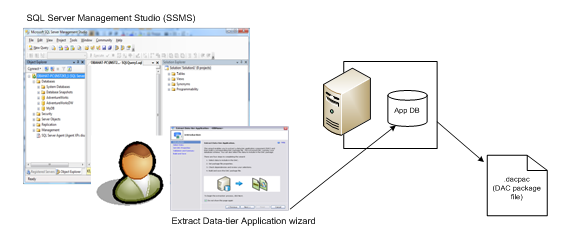
* The SQL Server edition must be Enterprise,Data Center, Developer, or Evaluation. If the edition is Enterprise, then licensing limits the number of managed instances to 25.

In a typical scenario, disk space used by the sysutility\_mdw database on the Control Point is approximately 2 GB per managed instance of SQL Server per year, but can vary depending on the number of database and system objects collected by the managed instance.

# Data-tier Application Overview

To simplify the development and deployment of the data-tier, Microsoft has introduced the concept of a data-tier application (DAC). A Data-tier Application is a single unit of deployment that captures data objects and data-tier application artifacts. In other words, it is a container that includes server and database schema objects that are used by an application (for example., tables, views, logins and users), as well as deployment prerequisites that set the requirements on the instances of SQL Server where Data-tier Applications can be deployed. The output file for a Data-tier Application is a Data-tier Application Component (.dacpac), this file is unpacked and deployed to a managed instance.

DBAs can create a DAC package file from within an existing database. The DAC package can then be opened and edited by Visual Studio developers in Visual Studio 2010 to make changes to the data-tier. They can then build an updated DAC package file and send the changes to DBAs, who in turn deploy the updates using SQL Server Management Studio.



**Figure 5:** Extracting a DAC from an existing database.

DAC extraction and deployment are also extremely useful for upgrading databases to the SQL Server 2008 platform. Users can move databases by extracting a DAC from an instance of SQL Server 2000, SQL Server 2005, or SQL Server 2008, and then deploy the extracted DAC to an instance of SQL Server 2008 R2, through either SQL Server Management Studio or the Windows PowerShell™ command-line interface. Because the DAC contains server objects that the database depends upon, this process is extremely straightforward.

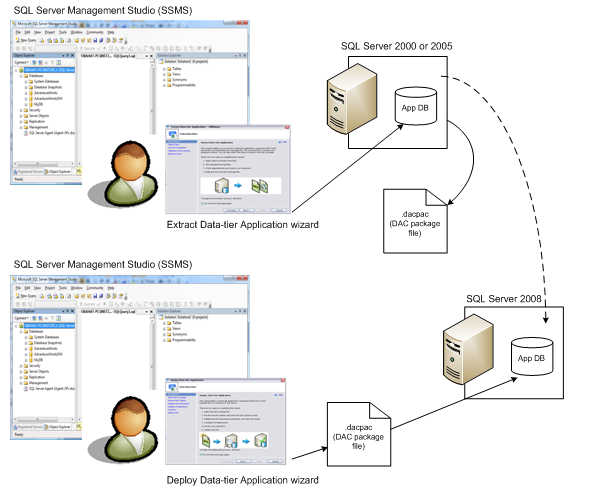
## Advantages of Registering a Data-Tier Application to the Control Point Utility

The key benefit of using DACs is that they are designed to offer data-tier automation, including:

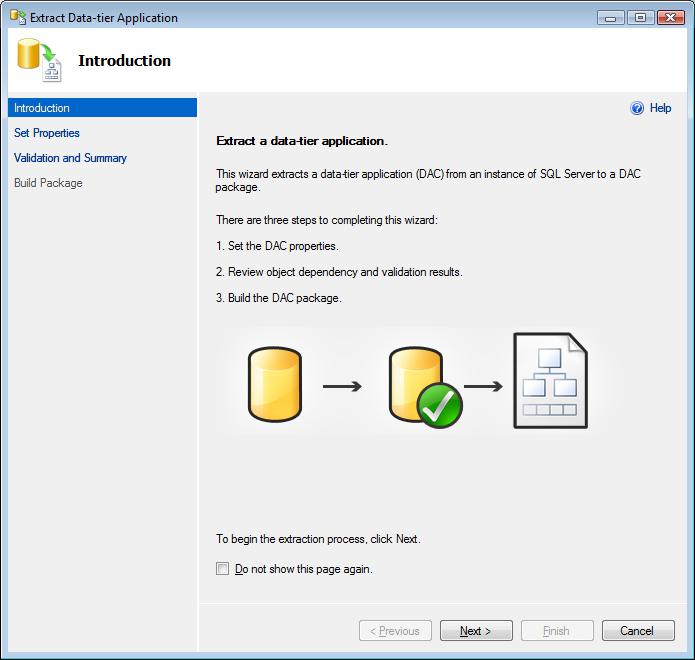
* Collecting large numbers of entities into one DAC that can be managed as a single unit through the full lifecycle of an application, including versioning.
* Automating the lifecycle of the data-tier by enabling developers to make changes to a data-tier application component, package it, and then pass it to DBAs for final deployment.
* Including policies that capture the intent of the developers, as well as deployment requirements on the instances of SQL Server where DACs can be deployed.

## Data-Tier Application Upgrade Efficiencies

DACs enable DBAs to upgrade the schema of SQL Server databases and instances from SQL Server 2000, SQL Server 2005, and SQL Server 2008 to SQL Server 2008 R2. Furthermore, the DAC contains any server level objects that the database depends upon such as logins. To upgrade, users can point to a SQL Server 2000, SQL Server 2005, or SQL Server 2008 database, extract a DAC and create the corresponding DAC package file, and then deploy the DAC to a target instance of SQL Server 2008 R2. Then, data can be transferred from the source database to the target database by using SQL Server Integration Services, the bulk copy utility, or many other data migration techniques.

**Figure 6:** Upgrading to SQL Server 2008 R2 by extracting a DAC from SQL Server 2000 or SQL Server 2005 and then deploying the DAC package file to an instance of SQL Server 2008 or 2008 R2

It is very straightforward to extract a DAC. The whole process runs from a wizard in Management Studio and you only need to supply a name and path for the DAC.



**Figure 7:** The Introduction page of the Extract Data-tier Application Wizard

The DAC package file can now be deployed to a SQL Server 2008 R2 instance or handed off to a Developer to make changes in Visual Studio 2010 through a new project type called Data-Tier Application. Policy-Based Management is extended from SQL Server to Visual Studio 2010 to enable developers to define deployment policies within the DAC package. New deployment policies help ensure deployment requirements aren’t lost in translation from developer to DBA and help significantly accelerate deployments and upgrades.

Deploying a DAC file is straightforward in Management Studio. A wizard guides you through the whole process and you only need to the DAC file and the name of the new database.

With this process there is no need for DBAs to write scripts or configure large numbers of parameters. The whole process is very straightforward and reduces time spent performing upgrades and potential mistakes with scripts. For example, previously a DBA at a manufacturing company upgraded every database using backup and restore. This was effective for the databases, but did not transfer the server level objects such as logins and these needed to be transferred separately. It was quite common for mistakes to be made and the databases were not functional until troubleshooting had been performed and the problems resolved. Now using DACs the database and any server level dependencies are transferred making the process more straightforward for the DBA and error free.

Furthermore, integration across SQL Azure enables DBAs to deploy Data-tier Applications directly from SQL Server 2008 R2 to the cloud and likewise developers from Visual Studio 2010 to the cloud.

# Conclusion

Familiar tools combined with new wizards help make setting up a SQL Server managed server group fast and easy. After managed server groups are set up, DBAs can easily assess capacity health and make decisions about consolidation to save money and better protect the health of their database environment. The introduction of the Data-tier Application introduces a single unit of deployment to accelerate consolidation and upgrade initiatives across the application lifecycle. Furthermore integration across SQL Azure enables seamless deployment of Data-Tier Applications on-premises or to the cloud for the ultimate in control and flexibility.

**For more information:**

<http://www.microsoft.com/sqlserver/>: SQL Server Web site

<http://www.microsoft.com/sqlserver/2008/en/us/r2.aspx>: SQL Server 2008 R2

<http://technet.microsoft.com/en-us/sqlserver/>: SQL Server TechCenter

<http://msdn.microsoft.com/en-us/sqlserver/>: SQL Server DevCenter

Did this paper help you? Please give us your feedback. Tell us on a scale of 1 (poor) to 5 (excellent), how would you rate this paper and why have you given it this rating? For example:

* Are you rating it high due to having good examples, excellent screen shots, clear writing, or another reason?
* Are you rating it low due to poor examples, fuzzy screen shots, or unclear writing?

This feedback will help us improve the quality of white papers we release.

[Send feedback](mailto:sqlfback@microsoft.com?subject=White%20Paper%20Feedback:%20SQL%20Server%202008%20R2%20-%20Application%20and%20Multi-Server%20Management).