

White Paper



ASP.NET to Silverlight Migration

Vidyadhar Parulekar, Atul Gupta

Abstract

Looking at the current landscape of the web technologies, RIA is the most sought of for the entire developer community. Thinking of Microsoft, when it comes to RIA, Silverlight is the platform of choice. You may have many existing ASP.NET based websites that you would want to migrate to Silverlight to reap the benefits of RIA. This whitepaper delves into the finer points to be considered when undertaking such a migration. It describes how the application architecture for ASP.Net and Silverlight differ and explains how the various features of a typical ASP.NET site can be converted to Silverlight. This paper can be used as a checklist for the migration.

Introduction

If you are building a web based application on Microsoft technology stack the typical options available today are ASP.NET, ASP.NET AJAX, ASP.NET MVC and Silverlight

Each of these options has their pros and cons. However, if you started a couple of years back, you pretty much had only one option and that was ASP.NET. Just as .NET versions keep changing and organizations typically look at migrating or upgrading to newer versions, it is very much possible that organizations will also want to move away from plain vanilla ASP.NET and use one of the other options.

With the advent of Web 2.0 Rich Internet Applications (RIA) are gaining public attention. More and more organizations are keen to jump the bandwagon and get their applications RIA enabled. If an organization decides to migrate from ASP.NET to a Silverlight [Reference 0] based application, then there are some basic considerations that should be kept in mind when embarking on this journey.

This white paper brings together key aspects that should be kept in mind and provides a checklist that can guide the migration effort.

Target Audience

This whitepaper is targeted to the people involved in the migration effort like project managers and developers. It could also be used by Architects or business decision makers to decide on if migration should be embarked on or not, however this paper doesn't delve in technology comparison between ASP.NET and Silverlight.

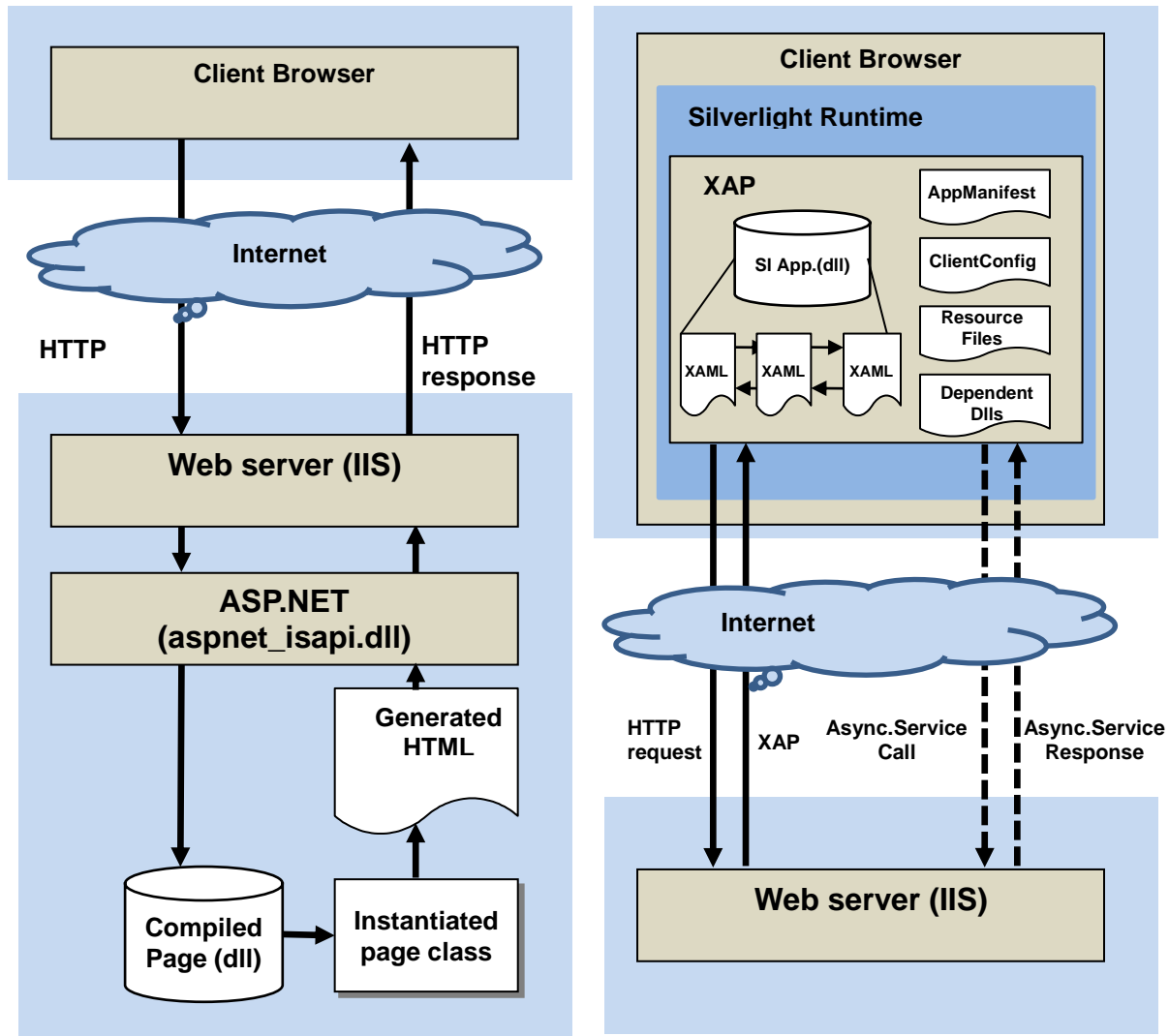
Migration Considerations

In this paper we will look at various features that are typically used for an ASP.NET Web Application and then look at how that can be reused (if possible) in Silverlight. We assume that if you are reading this paper, you have already factored in reasons to migrate to Silverlight and now looking at how to go about it. Some reasons could be making your site RIA enabled to add, i.e. media support, rich graphics, animations, and to have custom look and feel for the controls on the website. This paper will not get into feature comparison between ASP.NET and Silverlight.

Let us first explain the key architectural difference between ASP.NET and a Silverlight website. Most migration considerations are direct fallout of this.

Architectural Difference

Both ASP.NET and Silverlight are web based technologies however there is a key difference. Silverlight, unlike ASP.NET, runs on the client's machine via the plug-in. The plug-in i.e. the runtime is the Silverlight specific CLR that manages the execution of the Silverlight application on the client side, within the browser. Part of business logic can hence reside on client side along with the UI code. The data access would be still on the server side. The following figure compares Application Architecture for these technologies.



There are many more functional differences and these will get highlighted in the next section, where we take each point and explain the migration strategy for it.

Migration Checklist

Let's now look at the main features that are used in an ASP.NET Web application and see the migration impact. We will also touch upon some features that are not possible today with ASP.NET, but have been on organizations wish list. We will summarize the following in a table towards the end for easy reference.

Authentication

Most ASP.NET applications need authentication in some form or the other. The options are from using one of the many IIS based authentication mechanisms and/or couple it with those provided by ASP.NET itself [Reference 2]. Since Silverlight is hosted inside of an HTML or an ASP.NET page, it can pretty much reuse the authentication mechanism that is already configured. The user ID and other details however, would need to be passed to the Silverlight engine for further processing and use in application logic. This is done by using the concept of initialization parameters [Reference 3]. If we are thinking of building the entire application in Silverlight we can also use ASP.NET Authentication Services which were introduced with .NET Framework 3.5 version.

Browser Back Button Behavior

ASP.NET sites inherently support browser back button behavior or also the ability to deep link to particular page. Silverlight runs on client side and there is no change in browser URL even as you navigate within Silverlight pages. However with Silverlight 3 onwards deep linking is supported [Reference 4]. Navigation framework has been introduced in Silverlight 3. Using this framework you can partition the views into different XAML's and navigate to each of these file as you were doing it with .aspx pages in ASP.NET. Navigation framework also works with the browser's history journal to enable the back and forth of the XAML files using browser Back/Forward button. Using this framework will also enable some of the search enabled optimizations (SEO) for your Silverlight applications.

Client side scripting

Many ASP.NET applications rely on client side scripting and if they are built to use the AJAX concepts, then there will be heavy use of such scripts like AJAX library or jQuery etc. Since the client side is now managed by Silverlight CLR and the regular .NET programming languages like C# or VB.NET can be used, there is no need of these client side scripts. Based on the design of the Silverlight application, there may be a need to have a mix of Silverlight and ASP.NET pages in the same application. In this case some client side scripts may need to be retained. You will also possibly need to use HTML Bridge for communication between your Silverlight part and your ASP.NET part. Note that Html Bridge also involves some amount of client side scripting.

Control data binding

In ASP.NET, data would be either bound to the UI controls by using the ItemSource property or directly assigned to the relevant properties of the control. In some cases, expressions were used in ASPX pages itself to parse and extract the value and associate with the UI control. Silverlight extends this model significantly and provides for rich data binding [Reference 5]. With the data binding support via dependency properties, values can easily flow between the UI control and the backing field in the code. For example - If new value is retrieved from database, the UI control will automatically display it without need for any additional call as the data binding engine will take care of it. Similarly values typed in by user in say a text box control will be available in the code again via the binding engine.

Cross domain policy

With ASP.NET, calling cross domain services was not an issue as it was triggered from the server side. In Silverlight, however the service call could be triggered from the client itself and for security reasons cross domain calls aren't allowed. The cross domain calls have been prohibited to provide security and to prevent Silverlight applications from initiating malicious connections. Potential networking threats would be Denial of Service attacks, DNS Rebinding attacks and Reverse tunnel attacks. The options are to either call an in-domain WCF service and via that invoke the cross domain service (similar to ASP.NET model) or deploy a cross domain policy file in case the call has to be made from Silverlight client side code [Reference 6 and 7].

Deployment

Deploying a Silverlight application would be similar to how you deploy traditional ASP.NET applications to web server (IIS). For Silverlight two new mime types need to be added to the IIS Server:

```
.xap application/x-silverlight-app
```

```
.xaml application/xaml+xml
```

Silverlight code is packaged and deployed in a XAP file and hence appropriate configuration is required for the IIS to serve such content. Another important point to note while deployment is the use of cross-domain or client access policy file. This is required if you are making cross domain service calls from your Silverlight application (as explained in previous point). The cross domain policy file should be placed at the root of where your service is placed for your cross domain calls to succeed. As in case of ASP.NET application you can either publish from your Visual Studio environment or just do an XCopy deployment.

HTTP Request object

In ASP.NET the Request object is used to retrieve the values that the client browser passed to the server as part of page/post back [Reference 8]. Silverlight applications execute on the client browser and directly make WCF service calls on the server. Hence the Request object is not used. All the values like the form values reside in the client memory and are accessible throughout the application if stored at the application level. Variables can be stored at the Application level in the app class and can be accessed throughout the application. HTTP Request object can still be used in the ASPX hosting page for Silverlight application, but those values need to be passed as initparams to the Silverlight application.

LINQ

LINQ was added along with .NET 3.5 and hence if you have an ASP.NET 3.5 site, very likely you would have already used LINQ in some sense. If you are on ASP.NET and moving on to Silverlight, then do consider using LINQ for processing the data returned by the WCF services. LINQ with its SQL like syntax makes working with objects very easy and intuitive. Though Silverlight runtime getting downloaded at the client is subset of the .NET framework it supports LINQ to object as well as LINQ to XML. If your application uses limited amount of data LINQ to objects might play a major part in functionalities like search, sort and manipulating a particular object in your data collections.

Master Pages

Master page is a technique used in ASP.NET to create a standard page template and then derive all other pages from it, to provide a consistent look and feel for the website. While Silverlight does support similar concept, the master page cannot be reused as is and will have to be redone. Silverlight primarily relies on user control and along with that the styles and template feature to create consistent look and feel. Some parts of master page may still be usable, as part of the ASPX page in which the Silverlight application is hosted like if the existing master page has static header it can be re-used. This can be achieved by placing your Silverlight plug-in just below the header content of your master page. But if you have a dynamic header for

example, if you have a menu placed inside your header then it's better to have a new master page created for your Silverlight application. A master page in a Silverlight application would be nothing but a user control with a header and footer and other user controls placed in between these two.

Message Window

Many times messages are displayed in modal message boxes in ASP.NET using client side scripting. Since Silverlight runs on client side and supports .NET languages, this is fairly easy to manage. Silverlight 3 added support for ChildWindow that can be used for this purpose as a modal dialog or a non-modal dialog [Reference 9]. The ChildWindow when used as modal dialog also gives you disabled visual affect. You can display anything on your ChildWindow as you do with your User control. Apart from the ChildWindow you can still use a MessageBox if it's a simple message and it is similar to the modal Window shown using scripting in ASP.NET. Invoking a client side scripting modal box is also possible using HTML Bridge.

Offline Viewing

ASP.NET supported only connected model i.e. to access an ASP.NET site, you needed to be connected to the network. Silverlight primarily also works in connected mode but it also supports offline experience. The out of browser feature [Reference 10] of Silverlight allows you to install parts of the Silverlight application on the client machine and have user access some functionality of the site without being connected to the net. It can work with isolated storage on client side to store relevant data when working offline. The application will detect when it goes online and then can synchronize the data. This is possible as Silverlight supports detecting internet connectivity [Reference 10]. The developer has a choice to have a different UI for out of browser working as against when working in connected mode. This is a new feature in Silverlight which is not present in ASP.NET applications and is quite useful in scenarios where the end user does not have internet connectivity.

Patterns

ASP.NET has been using patterns like Model-View-Presenter (MVP) and the more recent Model-View-Controller (MVC), to enable separation of concerns. These patterns allow the UI to be decoupled from its business logic allowing isolation of the View from the model which enables enhanced Testability. Considering the different feature set of Silverlight especially around dependency properties and data binding, a new pattern called Model-View-ViewModel is used. MVVM [Reference 12] is an architectural pattern largely targeted for modern UI platforms like Silverlight and WPF. MVVM does derive its concepts from MVC. MVVM was introduced as a standardized way to leverage core features of Silverlight to simplify the creation of user interfaces and facilitate separation of concerns. Using this pattern the User Interface can be independently developed by the UX developers. It also increases the testability of the application as the ViewModel can be tested independent of the actual View. The application is loosely coupled, which in turn results in more flexibility and improved code quality.

Response.Redirect / Server.Transfer

For navigation from one ASPX page to another, options available in ASP.NET are typically either to do a Response.Redirect (this had an overhead of additional client navigation) or Server.Transfer (this has an issue that the URL on the address bar won't update). Since Silverlight runs on client side, it really doesn't need to do either of these and follows a different navigation model. Either one can build a Silverlight Navigation framework based application or manage the navigation across screens (user controls) using custom logic, like displaying and hiding relevant user controls, or loading and unloading specific user control from parent content control.

Serializable objects

In ASP.NET application, you would be dealing with many entities (service entities, business entities etc.) These entities may also need to be cached or stored in session based on application requirements. In case the application uses out-proc session

state or has business logic being accessed as services (web service/remoting/WCF services) these entities would have been marked serializable. In Silverlight the business logic will reside behind WCF service calls and for the objects to be transferred over the wire, they should be serializable. So if entities are already marked serializable, you can reuse your entities as is and if not marked, it should be a trivial task to mark them appropriately.

Session State

Since ASP.NET runs over HTTP (which is stateless) and is a server side implementation, session objects are used in order to maintain user state across page post backs. Based on need for load balancing or not, these could be in-proc or out-proc. On the other hand, Silverlight runs on client side browser and in some sense behaves like a thick client application. So while the application is executing, it will retain all data in memory and hence no separate session maintenance is required. If the application has a mix of ASP.NET pages and Silverlight pages, the ASP.NET session data can be passed to Silverlight using InitParams.

Services

Any server side interaction like database interactions in ASP.NET can happen directly from the ASP.NET code behind page (though not recommended as per architectural best practices). With Silverlight, services are pretty much mandatory as they are the gateway to the data access on server side. So it would be easier to migrate a layered SOA based ASP.NET application to Silverlight. You can directly consume existing Service layer in your Silverlight application resulting in greater code re-use and faster migration. Silverlight has host of options on how data can be accessed namely WCF Services, WCF Data services and WCF RIA services. The decision on using anyone of these depends on many factors (that discussion is out of scope of the current document) [Reference 13].

User Controls

ASP.NET uses ASPX pages and within them ASCX user controls. The user controls are used either to package some functionality together or as a reusable entity across the site. Silverlight entirely uses user control, but these are Silverlight specific user controls (XAML as front end and .CS as code behind). The user controls will need to be completely re-developed in Silverlight as XAML itself has different syntax compared to an ASPX or Html page and the controls available in Silverlight are different from those in ASP.NET.

Validation controls

ASP.NET validation controls are used to do input validation for ASP.NET pages. Validation can be of two types Client-Side validation or Server-Side validation. ASP.NET engine decides the type of validation to be done depending upon the browser that is making the request. ASP.NET validation controls are the only controls which generate a client-side script [Reference 14]. Silverlight applications execute on the client machine and hence the validation controls also validate the input on the client irrespective of the browser settings. Silverlight validation controls are not provided out-of-box and come with the Silverlight Toolkit. With Silverlight, you may have to work with IDataErrorInfo as well for validation of business entities.







View state

In ASP.NET View state is the technique that the ASP.NET page framework uses to preserve page and control values between server round trips [Reference 15]. Overtime this has caused severe heart burn for ASP.NET developers as it increases the page size and thus impacts the performance, but at the same time cannot completely be done away with, as some features rely on view state. Silverlight applications execute on the client browser and hence there are no round trips or post backs and hence no need of ViewState.

The table below summarizes the features that we discussed and can be used as checklist when embarking on the migration path. Based on the discussions above, we explain if a feature can be migrated as is or with some rewrite or needs to be completely discarded. The remarks column highlights the key reasoning behind this choice.

Note that the list is not organized in order of importance, but is alphabetically sorted for easy reference.

Migration to Silverlight				
ASP.NET/Web Application Feature	Reuse possible	Obsolete	Needs Rewrite/New Feature	Remarks
Authentication				IIS settings can be reused, but will need to pass information to Silverlight object via InitParams
Browser Back Button Behavior				No coding required. Silverlight 3 onwards deep linking is supported via Navigation framework
Client Side Scripting				Silverlight application runs on the client machine using the Silverlight specific CLR. However there could be HTML interaction specific scenarios
Control Data binding				Binding works differently in Silverlight as compared to ASP.NET
Cross domain policy				Silverlight client side service calls will need this if invoking services on another domain
Deployment				Silverlight XAP package deployment is similar to ASP.NET application deployment. However additional mime types need to be configured
HTTP Request object				Silverlight runs on the client machine and does not need HTTP Request object
LINQ				LINQ would be used exactly the same in Silverlight as in ASP.NET
Master Pages				If the header/footer is mostly static, reuse may be possible, else need to rewrite Master Pages
Message Window				Message window can be completely customized in Silverlight and need a re-design
Offline Viewing				New feature for Silverlight applications
Patterns				MVC/MVP pattern need to be replaced with MVVM pattern
Response.Redirect / Server.Transfer				Silverlight purely runs on the client and hence the page navigation also takes place on the client

Serializable objects				All objects/entities need to be marked serializable. Reuse may be possible depending on if ASP.NET app was SOA enabled or used out-proc session state
Session State				Can be passed to Silverlight as Initparams
Services				Most services can be consumed as is. May need changes in WCF service bindings
User Controls				UserControls need to be rewritten to be used in Silverlight
Validation Controls				Validation controls are different for Silverlight and need to be handled separately for Silverlight
View State				Silverlight runs on client machine and there are no page post back happening

Conclusion

In this paper we have highlighted key factors that have an impact on the migration of your existing ASP.NET application to Silverlight. The points discussed above will guide in making a choice for migration and can be used a check list when embarking on the migration path.

Reference

1. Silverlight Overview
<http://www.silverlight.net/getstarted/overview.aspx>
2. ASP.NET Authentication
<http://msdn.microsoft.com/en-us/library/eeyk640h.aspx>
3. How to: Specify and Retrieve Custom Initialization Parameters
[http://msdn.microsoft.com/en-us/library/cc189004\(VS.95\).aspx](http://msdn.microsoft.com/en-us/library/cc189004(VS.95).aspx)
4. Silverlight 4 + RIA Services - Ready for Business: Search Engine Optimization (SEO)
<http://blogs.msdn.com/b/brada/archive/2010/03/17/silverlight-4-ria-services-ready-for-business-search-engine-optimization-seo.aspx>
5. Data Binding
[http://msdn.microsoft.com/en-us/library/cc278072\(VS.95\).aspx](http://msdn.microsoft.com/en-us/library/cc278072(VS.95).aspx)
6. Making a Service Available Across Domain Boundaries
[http://msdn.microsoft.com/en-us/library/cc197955\(VS.95\).aspx](http://msdn.microsoft.com/en-us/library/cc197955(VS.95).aspx)
7. Silverlight cross domain policy file helpers
<http://timheuer.com/blog/archive/2008/04/06/silverlight-cross-domain-policy-file-snippet-intellisense.aspx>
8. Request object in ASP.Net
[http://msdn.microsoft.com/en-us/library/ms524948\(VS.90\).aspx](http://msdn.microsoft.com/en-us/library/ms524948(VS.90).aspx)
9. Refactoring Silverlight ChildWindow for a non-modal use
<http://timheuer.com/blog/archive/2009/05/10/silverlight-childwindow-non-modal-refactor.aspx>
10. Building An Out-of-Browser Client With Silverlight 3
<http://msdn.microsoft.com/en-us/magazine/dd882515.aspx>
11. NetworkInterface.GetIsNetworkAvailable Method
[http://msdn.microsoft.com/en-us/library/system.net.networkinformation.networkinterface.getisnetworkavailable\(VS.95\).aspx](http://msdn.microsoft.com/en-us/library/system.net.networkinformation.networkinterface.getisnetworkavailable(VS.95).aspx)
12. Model-View-ViewModel In Silverlight 2 Apps
<http://msdn.microsoft.com/en-us/magazine/dd458800.aspx>
13. Comparison on using Data access technologies
http://wildermuth.com/2010/07/08/State_of_Data_Access_in_Silverlight_4
14. Validation Controls in ASP.NET
<http://msdn.microsoft.com/en-us/library/aa479013.aspx>
15. View State in ASP.Net
<http://msdn.microsoft.com/en-us/library/ms972976.aspx>

Authors

Vidyadhar Parulekar

He is a Technology Lead with Product Competency Center (PCC) in Microsoft Technology Center (MTC). He has 9+ years of experience on different Microsoft Technologies including 2.5 years of experience on Silverlight. He is a contributor on the official Microsoft Silverlight forum at: <http://forums.silverlight.net/members/Vidds.aspx>. He currently works on building applications with Silverlight 4 and also explores its new features. He can be reached at Vidyadhar_parulekar@infosys.com

Atul Gupta

He is a Principal Technology Architect with Product Competency Center (PCC) in Microsoft Technology Center (MTC). He has spent 15+ years with Infosys and has been working on various Microsoft Technologies. He leads a team working on user experience creation using technologies like Windows Presentation Foundation and Silverlight. He blogs on latest technologies at: <http://blogs.infosys.com/microsoft>. Working on latest technologies is his passion and he was Microsoft's [Most Valuable Professional \(MVP\)](#) for six consecutive years. He can be reached at AtulG@Infosys.com.



Infosys among the world's top 50 most respected companies

Reputation Institute's Global Reputation Pulse 2009 ranked Infosys among the world's top 50 most respected companies.



About Infosys

Many of the world's most successful organizations rely on Infosys to deliver measurable business value. Infosys provides business consulting, technology, engineering and outsourcing services to help clients in over 30 countries build tomorrow's enterprise.

For more information about Infosys (NASDAQ:INFY), visit www.infosys.com.

Global presence

Americas

[Brazil](#)
[Nova Lima](#)
[Canada](#)
[Calgary](#)
[Montreal](#)
[Toronto](#)
[Mexico](#)
[Monterrey](#)
[United States](#)
[Atlanta](#)
[Bellevue](#)
[Bentonville](#)
[Bridgewater](#)
[Charlotte](#)
[Detroit](#)
[Fremont](#)
[Hartford](#)
[Houston](#)
[Lake Forest](#)
[Lisle](#)
[New York](#)
[Phoenix](#)
[Plano](#)
[Quincy](#)
[Reston](#)

Asia Pacific

[Australia](#)
[Brisbane](#)
[Melbourne](#)
[Perth](#)
[Sydney](#)
[China](#)
[Shanghai](#)
[Hangzhou](#)
[Hong Kong](#)
[Central](#)
[India](#)
[Bangalore](#)
[Bhubaneshwar](#)
[Chandigarh](#)
[Chennai](#)
[Gurgaon](#)
[Hyderabad](#)
[Jaipur](#)
[Mangalore](#)
[Mumbai](#)
[Mysore](#)
[New Delhi](#)
[Pune](#)
[Thiruvananthapuram](#)
[Japan](#)
[Tokyo](#)
[New Zealand](#)
[Wellington](#)
[Philippines](#)
[Metro Manila](#)
[Singapore](#)
[Singapore](#)

Europe

[Belgium](#)
[Brussels](#)
[Czech Republic](#)
[Brno](#)
[Prague](#)
[Denmark](#)
[Copenhagen](#)
[Finland](#)
[Helsinki](#)
[France](#)
[Paris](#)
[Germany](#)
[Frankfurt](#)
[Stuttgart](#)
[Walldorf](#)
[Ireland](#)
[Dublin](#)
[Italy](#)
[Milano](#)
[Norway](#)
[Oslo](#)
[Poland](#)
[Lodz](#)
[Spain](#)
[Madrid](#)
[Sweden](#)
[Stockholm](#)
[Switzerland](#)
[Geneva](#)
[Zurich](#)
[The Netherlands](#)
[Amsterdam](#)
[United Kingdom \(UK\)](#)
[London](#)

Middle East and Africa

[Kingdom of Saudi Arabia](#)
[Riyadh](#)
[Mauritius](#)
[Reduit](#)
[UAE](#)
[Dubai](#)
[Sharjah](#)

For more information, contact askus@infosys.com

www.infosys.com

© 2011 Infosys Technologies Limited, Bangalore, India. Infosys believes the information in this publication is accurate as of its publication date; such information is subject to change without notice. Infosys acknowledges the proprietary rights of the trademarks and product names of other companies mentioned in this document.