|  |
| --- |
| Raindrop  Installer |
| iInterchange Systems Pvt. Ltd.  Version 1.0 |

|  |  |  |
| --- | --- | --- |
| Date : Code : | | |
| **PREPARED BY** | **REVIEWED BY** | **APPROVED BY** |
| S.Thanganangai | Balaji | Balaji |

*Note: This is a privileged and confidential document of iInterchange Systems Pvt Ltd. No part of this publication may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), without prior written consent from* ***iInterchange***

Copyright 2010 by **iInterchange** all rights reserved.

(Subject to limited distribution and restricted disclosure only)

**REVISION HISTORY**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **VERSION NO** | **AUTHOR** | **RELEASE DATE** | **CHANGE DETAILS** | | **APPROVED BY** |
| **SECTION** | **DESCRIPTION** |
| 1 | Thanganangai | 17-Mar-2011 | All |  | Balaji |

# Table of Contents

[1 Table of Contents 3](#_Toc288129032)

[2 Introduction 4](#_Toc288129033)

[1.1. Purpose 4](#_Toc288129034)

[1.2. Scope 4](#_Toc288129035)

[1.3. Definitions, Acronyms and Abbreviations 4](#_Toc288129036)

[3 Tear Apart 5](#_Toc288129037)

[1.4. Page Architecture 5](#_Toc288129038)

[1.5. Spliting Of dll 5](#_Toc288129039)

[1.6. Namespace Resolution 5](#_Toc288129040)

[1.7. File Structure 5](#_Toc288129041)

[4 Generation of Patch 6](#_Toc288129042)

[1.8. Database scripts 6](#_Toc288129043)

[1.9. Validation 6](#_Toc288129044)

[1.10. Documentation 6](#_Toc288129045)

[1.11. Versioning 6](#_Toc288129046)

[5 Automation of Updates 7](#_Toc288129047)

[5.1 Rollback 7](#_Toc288129048)

[6 Roadmap 9](#_Toc288129049)

[7 References 10](#_Toc288129050)

# Introduction

## Purpose

Purpose of this document is to explain about the architecture involved in the development of Raindrop Installer(RDI).

## Scope

Scope of this document will be Project Raindrop

## Definitions, Acronyms and Abbreviations

|  |  |
| --- | --- |
| Tech Group | Technology Group |
| BL | Business logic |
| RDI | Raindrop Installer |
| Dll | Dynamic linking library |

# Tear Apart

## Page Architecture

All ASP.NET pages are composed of a designer file which contains all the markup for the controls and a code behind file which has the code for separate events like page load, button click etc. Now when a request for an asp.net page is received by the IIS, it transfers the request to the aspnet\_isapi.dll ISAPI extension, the aspnet\_isapi.dll extension in turn loads the HTTP Runtime and the processing of the request starts.

When the processing of the request starts, objects required for the processing of the request like HttpContext and HttpApplication etc are instantiated and the HTTP Modules are executed against the request until the request hits the ASP.NET page HTTP Handler, and then the page life cycle starts.

## Spliting Of Dll

Typically the web application of a project is compiled into a single assembly which brings down the number of assemblies produced while a build is taken; this makes it easy to obfuscate the codes in the code behind files of the ASP.NET pages. But this method has a major flaw which is, whenever a change is made in the code of a single page the whole project needs to be build and the complete build needs to be installed in the live environment. This is completely unnecessary and time consuming.

If we split the web application’s assembly into per page assembly, when the code in a page is changed only that assembly and the assemblies referenced by it needs to be replaced in the live environment. This drastically reduces the time it takes for us to make a change, take the build for the change and install the change in the live environment.

## Namespace Resolution

Even though each class file in Business logic is compiled into separate dll, the namespace remains same for all Business logic dlls. This way when there is change made to one of the business logic files only that business logic class and its referenced assemblies needs to be build for deployment.

## File Structure

All the aspx page in the project inherits Pagebase class and has reference for Script files and Business logic.

# Generation of Patch

A patch is a set of changes which needs to be effected in the live application.

When there is a new requirement or a change to be done anywhere in the application the changes are deployed to the live environment as a patch which replaces the necessary files to effect that change or requirement.

Generating patches has a lot of advantages over doing a complete build of the application for making a change in the live application.

All the files that are changed for a particular change or bug fix are compiled into a single executable, and the patch can be installed by running this executable. The GUID of the files are used to verify the identity of the files being changed.

In the event that the changes made by a patch needs to reverted, the patch can be uninstalled which would leave the application in the state where it was before installing the particular patch.

## Database scripts

Database scripts can be included in the patch for making any changes in the database. Before running these scripts in the live database server a backup of the database is taken and if the patch is uninstalled to rollback the changes, the database will be restored with the database backup taken before installing the patch.

## Validation

While installing a patch it validates whether the dependent patch is already installed and any patch not required for this is not installed. If all the validations are successful, the patch will be installed.

While uninstalling the patch, it validates whether any other patch is dependent on it. If there is no dependency for this patch, it can be removed.

## Documentation

Once the patch is installed successfully, build report is generated for the specified change and an email is sent to the concerned person.

## Versioning

The project version is updated depends upon the type of update provided in patch.

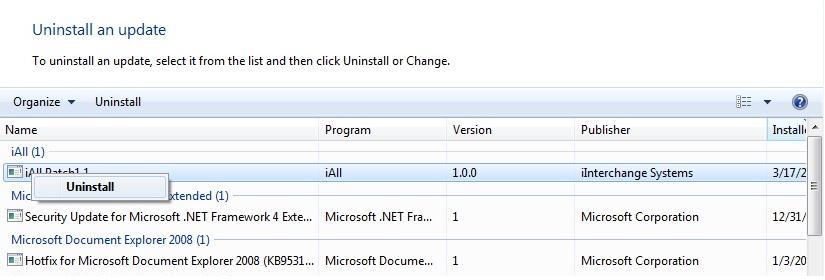
# Automation of Updates

Once the patch is created for the selective change set it is placed in a common folder which is accessed by the auto update service over internet via a secured layer. Auto update service should be running in server where the application is installed. It checks for updates periodically by monitoring the common folder. Once an update is found, the service downloads the patch and the manifest file for that patch which contains the scheduled installation time.

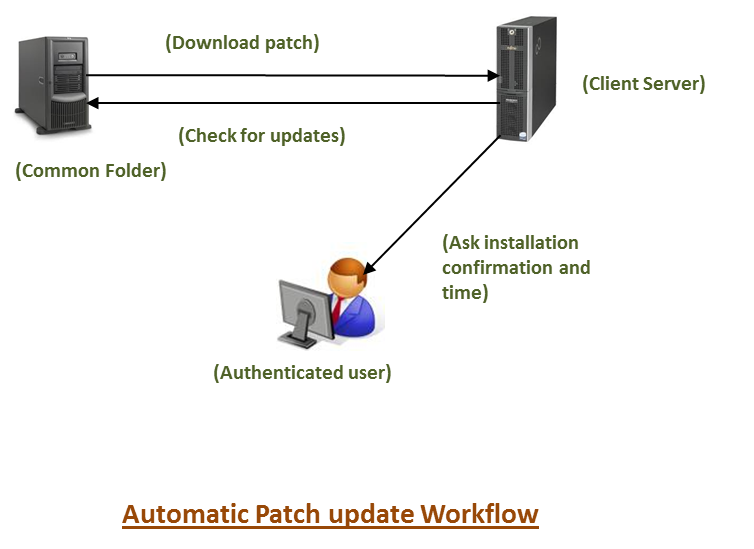
The user would be asked to confirm the installation time through email which was specified in the manifest. The user would be allowed to change the installation time. These updates are classified into different types. If the update is a critical update, it will be installed automatically. If the update is optional, user will be asked to install the update. This reduces the down time of the application. Automatic updates take care of the complete installation including the reboot of the server (if necessary). It also sends information on the successful installation of the update with the log from the installation via email (optional).

## Rollback

In the event that the changes effected by a patch needs to be rolled back the particular patch can be uninstalled from the control panel. When a patch is uninstalled, the files which were changed during the patch installation will be reverted back to the state it was before the patch installation. If any database changes were done during the patch installation, those changes will be reverted back when the patch is uninstalled.



Uninstalling a patch



# Roadmap

|  |  |  |  |
| --- | --- | --- | --- |
| **S.NO** | **Feature** | **1** | **2** |
| 1 | Patch install validation |  |  |
| 2 | Patch uninstall validation |  |  |
| 3 | Documentation |  |  |
| 4 | Versioning |  |  |
| 5 | Automatic Update |  |  |

# References

* [Code Project](http://www.codeproject.com/KB/aspnet/aspnetrequestarchitecture.aspx)
* [Dot Net Slackers](http://dotnetslackers.com/articles/iis/ASPNETInternalsIISAndTheProcessModel.aspx)