**毕业设计开题报告**

课 题 名 称 ：基于 .NET 的跨语言通用版本控制管理系统的设

计与实现

学 院 ： 计算机科学与技术

专 业 ：　　 软件工程

姓 名 ： 　张允祺

学 号 ：　　　　 131320224

指 导 教 师 ：　　　　　 　陈德华

**二〇一七 年 一 月 三 日**

**基于 .NET 的跨语言通用版本控制管理系统的设计与实现**

# 项目背景

随着计算机和互联网的发展，越来越多的行业将这二者融入到办公和业务流程之中。自上个世纪的软件技术迅速发展以来，软件的地位逐步攀升，越来越多的行业在业务流程中融入了软件技术的相关应用，如ERP系统、OA系统、CRM系统等等，它们把人们从大量、繁琐、复杂的工作中解放了出来，并且承担了许多常人无法胜任的工作。

如今，在医院的医疗流程中，HIS系统已经成了必不可缺的部分，无论是医生、患者，还是管理人员等等，HIS给他们都带来了极大的便利。但目前的HIS仍存在着问题，因为数年前PB在C/S架构开发的优势，大量的HIS系统采用了PB作为开发技术。但随着技术的发展，PB逐渐开始跟不上技术的发展，同时在微软的大力支持下，已经有部分医院开始采用.NET作为核心技术进行新一代的HIS开发。

但因为技术的差异过大，使得前后两代如果需要更新，则只能通过循序渐进的过程，否则可能会导致很严重的问题。因此，为弥补以下环境时的不足，本项目得以设计、开发。

## 传统版本控制的思路劣势

传统的版本控制具有如下几点劣势：

1、系统的更新模块与系统的联系（非耦合）过于紧密，一个系统的更新模块仅供该系统使用。在这种情况下，如果同一企业拥有多个产品，则需要服务器内配置多个与各个系统相关的更新相关服务，这样便会导致服务器资源存在浪费的问题。

2、系统的模块与系统语言需保持一致，且要在系统的开发阶段进行开发。因此，功能一致的更新模块需要在根据系统的语言进行移植修改，增加了编码过程的工作量。

3、传统桌面系统的更新模块需要通过用户手动从系统中启动，从而进入更新流程。也有一部分系统的更新模块能够实现自动检查更新，但下载以及之后的流程仍然需要手动启动，而不能像手机端应用一样将更新流程放入后台中。

## 本项目版本控制的思路优势

1、消耗服务器资源小。一个企业内部的所有系统只需要在服务端配置一组服务与数据库即可，减轻了性能压力、增加了服务器资源的利用率。

2、与系统编写语言无关。系统的开发过程中可以排除更新模块部分，减少了开发过程中的任务数量。并且在系统出现技术转换时无需进行重复劳动，可兼容不同技术实现的版本差异。

3、实现了自动的后台更新过程。更新无需手动启动，在系统启动后会自动进行更新工作。

4、具有独立的管理工具，无需直接对数据库进行操作。通过用户与角色等的分配，将版本控制业务化、规范化。

# 研究现状

自二十世纪以来，计算机科学的发展取得了非常卓越的成果。计算机、软件、网络等等，无一不在科学研究到日常生活的各个领域大放光彩。而且计算机科学的成果，一方面提供了计算机科学飞速发展的动力，另一方面，成果之间相互结合、相辅相成，又孕育出了更加丰富、强大的成果。本项目作为计算机科学成果的应用之一，自然离不开计算机科学的发展，本项目主要得益于两项技术的发展成果：以太网以及软件体系结构。

## 以太网

### 发展历程

二十世纪七十年代，Xerox公司提出并实现了最初的以太网模型，此时，作为一项新技术的以太网正式诞生于世。10年后，IEEE工作组发布了IEEE 802.3标准。作为最早的以太网正式标准，它的出现，意味着以太网技术开始普及，也正是从此时开始，以太网进入了快速发展的阶段。

自此之后，以太网以低廉的端口价格和优越的性能获得了市场的青睐。每一次新规格的出现，都意味着以太网的速度提升了一个数量级。

也正是因为以太网速度的提升，使得大数据、云计算等新理论技术的研究计划提上日程，分布式系统的设计也能够得以实现。

### 发展趋势

以太网技术的发明人Bob Metcalfe博士曾经在访谈中对以太网技术的下一部发展方向进行过归纳，如下：

1、Up

在速度上进一步提升。

2、Down

在嵌入式系统上发展，将以太网技术应用到CPU技术领域。

3、Over

在无线领域上的技术改进。

4、Across

在宽带接入技术上的发展，扩大宽带接入领域市场。

## 体系结构

### 发展历程

在计算机的发展初期，计算机的形式还只是大型中央主机，而在大型中央主机中运行的软件对应的也只有一种集中的结构，Mainframe结构。由于当时计算机的硬件技术与工艺所限，大型机有着大型、集中、独立等特点，这些特点意味着大型机上的软件有着很多的局限性，尤其是远程访问。

当时软件的设计重点往往在于数据结构和算法的选择上。但随着计算机的普及与发展，软件的规模越来越庞大，结构越来越复杂，此时的问题数据结构和算法的选择已经无法满足开发工作的需要，直至软件危机出现时，这些问题仍旧没有解决。

为了解决软件危机，人们重新审视了软件的整个生命周期，发现此时对系统的总体设计已经比算法和数据结构的选择重要得多，于是提出了软件工程的概念。而软件的体系结构也脱胎于软件工程，并借鉴了计算机科学中的许多研究成果，成为了计算机科学的一个独立学科分支。

因此，在二十世纪八十年代，Client/Server的分布式计算结构出现了。C/S结构使得应用程序能够分布式地部署在服务器与客户端上，分别处理服务端和用户的信息，支持模块化开发，与关系型数据库间更加安全，并且具有出众的灵活性。因此，C/S结构得到了相当广泛的应用。

得益于Internet的飞速发展，基于C/S结构的新的更具生命力的结构得以诞生，即是客户层、服务层、数据层的三层结构。

在三层结构中，用户请求、业务逻辑和数据三层被物理上的隔离，灵活性、可扩展性都得到了前所未有的提升，并且还具有良好的移植性和更加稳定的安全性。

### 发展趋势

软件体系结构如今仍处于迅速发展的阶段中，目前有如下几个发展趋势：

1、结构间交流

将针对于各个领域的ADL使用一种公共的形式将各自包含的信息进行整合，使体系结构之间能够无障碍地交换各自包含的信息。

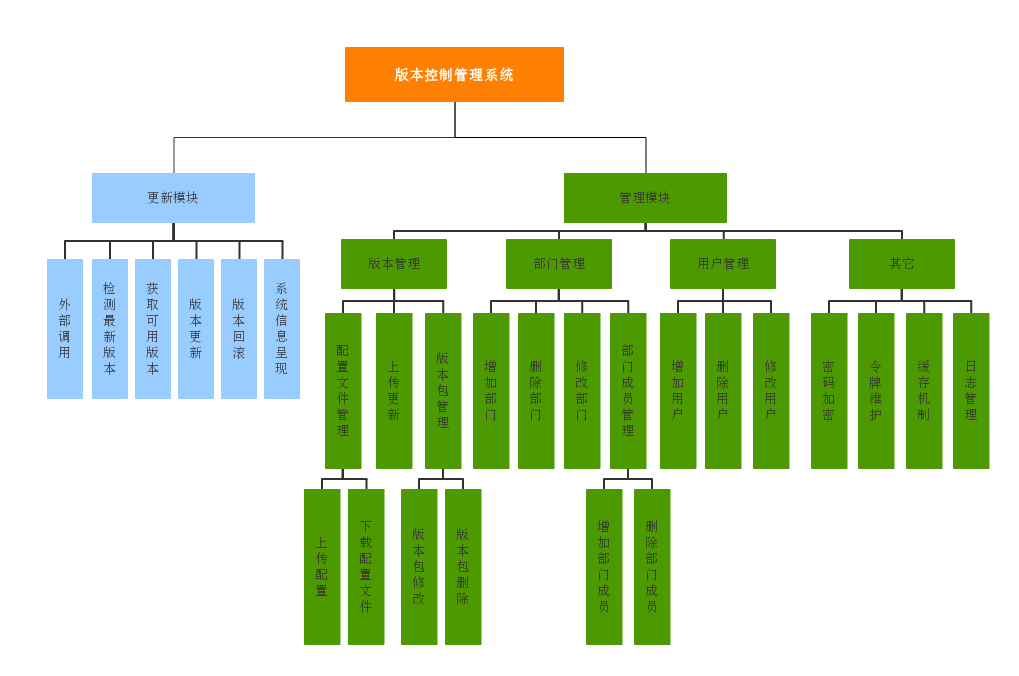
2、辅助工具

即将软件工程的各个环节成果与体系结构的描述、分析进行互相转换的工具。

3、再工程

再工程是将遗留系统转换为可进化系统的途径，即为特定领域提供一种能够快速构造新系统的体系结构框架。

# 主要研究内容



## 更新层

### 版本更新模块

1. 更新模块的启动时机：

更新模块需要在用户系统启动时自动调用。

1. 更新模块的功能：

更新模块能够通过调用服务端的服务进行检查更新、下载更新等操作。

### 版本回滚模块

1. 回滚模块的启动时机：

回滚模块能够在任何用户系统有需要的时刻对其进行手动调用。

1. 回滚模块的功能：

回滚模块能够通过调用服务端的服务进行查询回滚列表、下载指定版本等操作。

## 管理层

### 版本管理系统

1. 管理系统的功能：

管理系统能够针对不同的用户提供生成配置文件、系统版本控制、角色管理、用户管理等功能的入口。

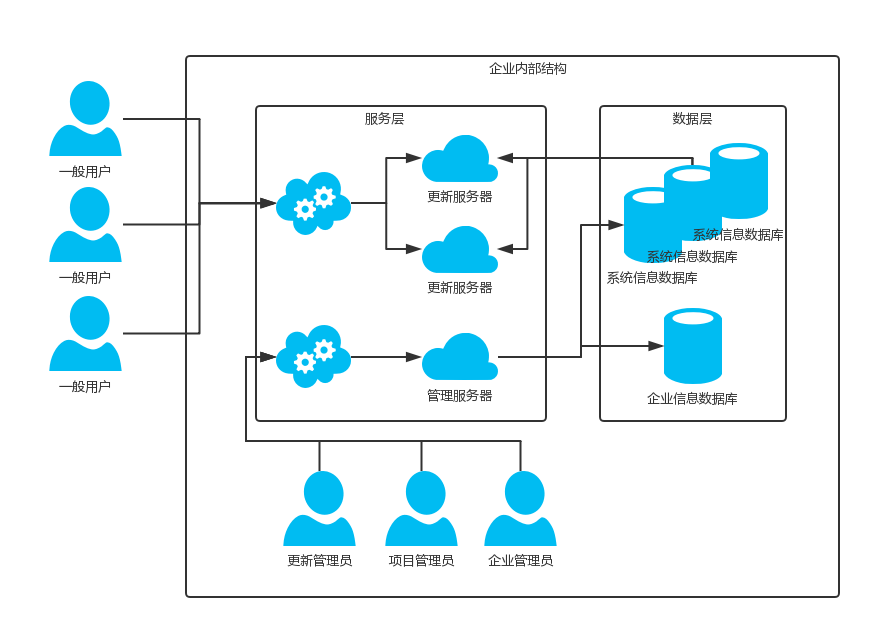
1. 管理系统的权限控制：

本系统用户有系统管理员、企业管理员、项目管理员、版本管理员、系统用户。

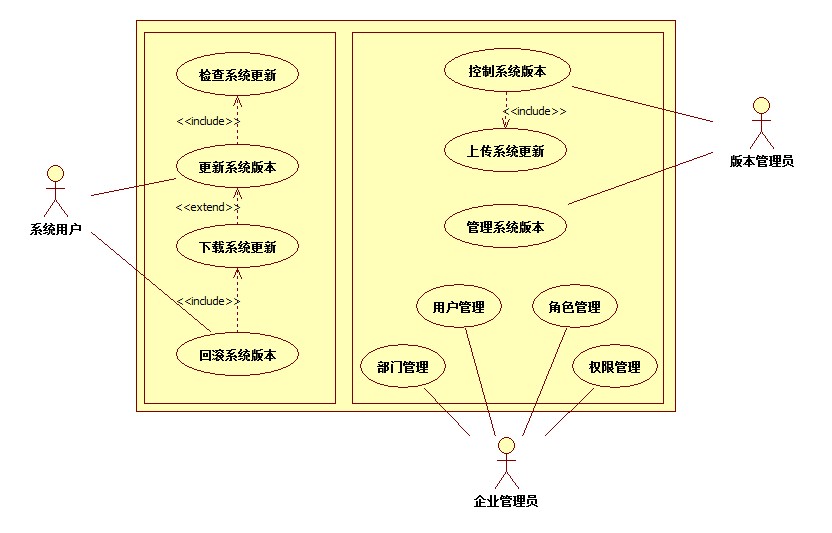
* 系统管理员负责对整个系统的维护、后台数据的维护以及元数据的管理。该角色一般情况下企业内有一位。
* 企业管理员负责对在本系统中进行版本管理的系统项目信息进行维护、对项目下的项目管理人员进行管理。该角色一般情况下企业内有一位。
* 项目管理员负责对其所属项目的信息进行维护、对所属项目的版本管理人员进行管理。该角色一般情况下企业内人数与企业内需要版本管理的项目数量相同，可以多于，但必须保证每个项目至少有一位。
* 版本管理员负责对其所属项目的版本进行管理、对版本包进行管理。该角色一般情况下在每个项目下至少有一位。
* 系统用户负责使用该系统对包含该系统的系统进行更新。该角色与上述角色不冲突，即所有该系统的使用者。

# 技术路线

## 系统架构图



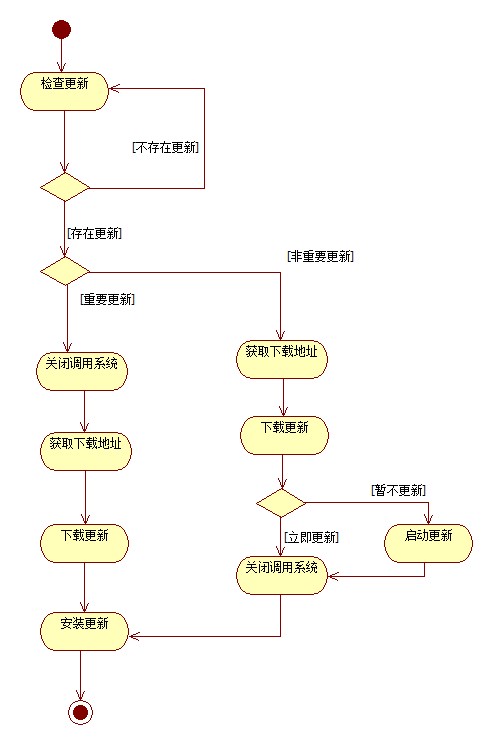
## 系统用例图



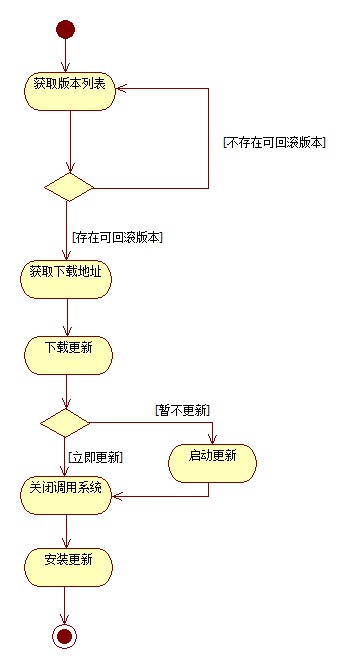
## 业务流程图

### 开发层

#### 版本更新

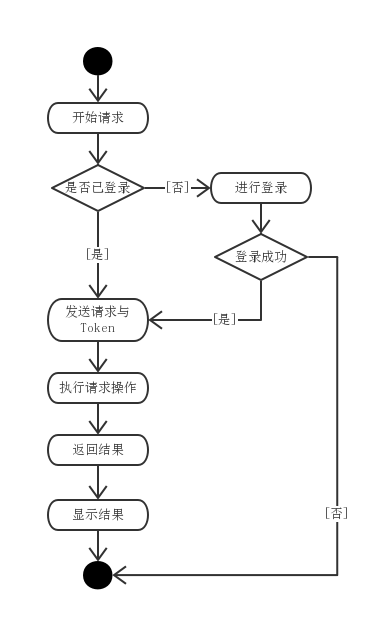


#### 版本回滚



### 管理层

#### 管理业务



## 采用技术

1. 本项目采用迭代式开发方法中的增量模型，讲本系统分为多个模块组件，以需求分析得出的基础需求作为原型，在原型的基础上对其余模块进行迭代增量开发。
2. 本项目在生存周期中，融入UML技术，使UML模型贯穿整个项目的生存周期。绘制工具采用StarUML 5.0
3. 客户端开发采用winform技术，服务端开发采用.net web api技术。
4. 开发工具采用Visual Studio 2015，开发语言为C# 6.0，开发环境为.Net Framework 4.6。
5. 数据库采用Microsoft SQLServer 2012，管理工具为SQL Server 2012 Management Studio。
6. 服务器系统使用windows server 2012 R2，部署工具为IIS 8.0。

# 关键技术介绍

## .NET Framework

.NET框架（英语：.NET Framework）是由微软开发，一个致力于敏捷软件开发（Agile software development）、快速应用开发（Rapid application development）、平台无关性和网络透明化的软件开发平台。.NET是微软为2000年代对服务器和桌面型软件工程迈出的第一步。.NET包含许多有助于互联网和内部网应用迅捷开发的技术。

.NET框架是微软公司继Windows DNA之后的新开发平台。.NET框架是以一种采用系统虚拟机运行的编程平台，以通用语言运行库（Common Language Runtime）为基础，支持多种语言（C#、F#、VB.NET、C++、Python等）的开发。

.NET也为应用程序接口（API）提供了新功能和开发工具。这些革新使得程序设计员可以同时进行Windows应用软件和网络应用软件以及组件和服务（web服务）的开发。.NET提供了一个新的反射性的且面向对象程序设计编程接口。.NET设计得足够通用化从而使许多不同高级语言都得以被汇集。

## Windows Forms

Windows Forms是微软的.NET开发框架的图形用户界面的一部分，该组件通过将现有的Windows API（Win32 API）封装为托管代码提供了对Windows本地（native）组件的访问方式。虽然该组件看起来是为先前较复杂的基于C++的微软基础类库（Microsoft Foundation Classes，MFC）的替代品，但是它并没有提供与Model View Controller "Document/View"架构相应的特色。"Document/View"架构已经被“MDI（多文档接口）”所替换。

该实现位于.NET框架的System.Windows.Forms名字空间，并且它试着纠正自己在Windows XP背景之下扮演的角色。然而，许多有关于标签表格与控制单元置于标签表格上的支持性的显著问题悬而待决。

许多非微软官方实现的.NET框架，如Mono开发平台，提供了Windows Forms的实现。

## Web服务

Web服务是一种服务导向架构的技术，通过标准的Web协议提供服务，目的是保证不同平台的应用服务可以互操作。

根据W3C的定义，Web服务（Web service）应当是一个软件系统，用以支持网络间不同机器的互动操作。网络服务通常是许多应用程序接口（API）所组成的，它们透过网络，例如国际互联网（Internet）的远程服务器端，执行客户所提交服务的请求。

尽管W3C的定义涵盖诸多相异且无法介分的系统，不过通常我们指有关于主从式架构（Client-server）之间根据SOAP协议进行传递XML格式消息。无论定义还是实现，WEB服务过程中会由服务器提供一个机器可读的描述（通常基于WSDL）以辨识服务器所提供的WEB服务。另外，虽然WSDL不是SOAP服务端点的必要条件，但目前基于Java的主流WEB服务开发框架往往需要WSDL实现客户端的源代码生成。一些工业标准化组织，比如WS-I，就在WEB服务定义中强制包含SOAP和WSDL。

SOAP：一个基于XML的可扩展消息信封格式，需同时绑定一个网络传输协议。这个协议通常是HTTP或HTTPS，但也可能是SMTP或XMPP。

WSDL：一个XML格式文档，用以描述服务端口访问方式和使用协议的细节。通常用来辅助生成服务器和客户端代码及配置信息。

UDDI：一个用来发布和搜索WEB服务的协议，应用程序可借由此协议在设计或运行时找到目标WEB服务。

这些标准由这些组织制订：W3C负责XML、SOAP及WSDL；OASIS负责UDDI。

## Microsoft Visual Studio 与 IIS

Microsoft Visual Studio（简称VS）是微软公司的开发工具包系列产品。VS是一个基本完整的开发工具集，它包括了整个软件生命周期中所需要的大部分工具，如UML工具、代码管控工具、集成开发环境（IDE）等等。所写的目标代码适用于微软支持的所有平台，包括Microsoft Windows、Windows Phone、Windows CE、.NET Framework、.NET Compact Framework和Microsoft Silverlight。

而Visual Studio .NET是用于快速生成企业级ASP.NET Web应用程序和高性能桌面应用程序的工具。Visual Studio包含基于组件的开发工具（如Visual C#、Visual J#、Visual Basic和Visual C++），以及许多用于简化基于小组的解决方案的设计、开发和部署的其他技术。

互联网信息服务（英语：Internet Information Services），是由微软公司提供的基于运行Microsoft Windows的互联网基本服务。最初是Windows NT版本的可选包，随后内置在Windows 2000、Windows XP Professional和Windows Server 2003一起发行，但在普遍使用的Windows XP Home版本上并没有IIS。

IIS包括FTP/FTPS、NNTP、和HTTP／HTTPS、SMTP等服务。 IIS可设置的内容包括：虚拟目录及访问权限、默认文件名称、以及是否允许浏览目录。

# 要解决的技术问题

1. 模块与用户系统的耦合度要尽可能的低。
2. 独立的更新模块要保证足够高的安全性。
3. C/S三层架构的开发。
4. HTTP协议下文件的上传和下载。
5. 加密存储用户信息。
6. 处理尽可能多的异常情况。

# 日程安排

|  |  |  |
| --- | --- | --- |
| **序号** | **各阶段名称** | **起止日期** |
| **1** | 系统开发环境配置 | 2016/11/17~2016/12/20 |
| **2** | 完成系统的概要设计 | 2016/12/21~2017/01/19 |
| **3** | 完成系统的详细设计 | 2017/01/20~2017/01/31 |
| **4** | 系统的实现 | 2017/02/01~2017/04/20 |
| **5** | 程序的测试以及修改 | 2017/04/20~2017/04/24 |
| **6** | 撰写毕业论文 | 2017/04/25~2017/05/15 |

# 参考文献

[1] Jon Skeet. C# in Depth[M]. 北京：人民邮电出版社，2014

[2] Jeffrey Richter. CLR via C#[M]. 北京：清华大学出版社，2015

[3] 杨传伟, 孟凡荣. C/S客户端软件自动更新系统的设计与实现[J]. 计算机工程与设计, 2007, 28(18):4561-4563.

[4] 王广兴, 周伟. C/S模式智能客户端自动更新功能的设计与实现[J]. 电脑知识与技术:学术交流, 2007, 3(13):156-156.

[5] 程浩. 基于.NET的客户端自动升级系统的设计与实现[J]. 计算机与网络, 2014(10):52-54.

[6] 唐教兵, 郭四稳. 基于Web服务的应用程序自动更新系统的设计与实现[J]. 电脑知识与技术:学术交流, 2007, 2(11):1294-1297.

[7] 赵宏伟, 孙永清, 崔宇寅. 一种软件自动更新方法[J]. 电脑知识与技术:学术交流, 2012, 08(1):74-76.

[8] 陈坚, 宋安平. 自动更新系统的设计与实现[J]. 计算机时代, 2010(8):19-21.

[9] Frieder O & Segal M E. On dynamically updating a computer program: from concept to prototype[J]. Journal of Systems & Software, 1991, 14(2):111-128.