UNIVERSIDAD NACIONAL AUTÓNOMA

DE MÉXICO

FACULTAD DE INGENIERÍA

Carrera:

Ingeniería en computación

Informe

Desarrollo de un servicio notificador para una asociación deportiva utilizando tecnología de Microsoft

Reporte de experiencia profesional

Presenta:

Héctor Sánchez Hernández

Asesora

M.I Norma Elva Chávez Rodríguez

México, D.F., 2016

Índice temático

Introducción

En el siguiente reporte, describo mi participación como programador en un proyecto que me fue asignado y que se encuentra funcionado actualmente. En dicho proyecto aplico los conocimientos adquiridos durante la carrera y a lo largo de mi vida profesional.

La empresa donde laboro es líder y reconocida en su ramo, cuenta con más de 800 empleados distribuidos en varias sedes. Debido a la cantidad de información que maneja, la institución requiere que los procesos sean automatizados para reducir los tiempos de respuesta y obtener información oportuna y precisa; es por ello que cuenta con un área robusta de tecnologías de la información (Sistemas).

Entre los sistemas que se han hecho para cubrir dichos propósitos está: “Solicitud de pedido”, en éste los empleados se encargan de solicitar diversos insumos para realizar sus actividades cotidianas; la dinámica del sistema consiste en ingresar y realizar una solicitud, después un administrador (dependiendo del área) autoriza o no la petición.

Sin embargo, cuando se requería alguna autorización urgente, debido a que los autorizadores no se encuentran todo el tiempo en la oficina ni revisando el portal, muchas veces los empleados no podían seguir con el flujo de sus pedidos y detenían la productividad ya que tenían que esperar hasta que la petición fuera autorizada, lo anterior representaba un obstáculo que afectaba a los empleados y a la empresa.

Por lo tanto, en el presente reporte expongo mi colaboración en la programación de un proceso para la automatización de notificaciones que son utilizadas para avisar sobre las peticiones de insumos a alguna de las áreas; para ello se implementó un “Windows Services”.

Capítulo 2

Descripción del puesto de trabajo

Por el tamaño de la empresa existen diferentes áreas de sistemas que pertenecen a la “Dirección de TI”. Yo formo parte de un equipo de cinco programadores como Programador Jr y constituimos el área de “Sistemas Administrativos”.

El objetivo de mi puesto es desarrollar sistemas de calidad, escalables y seguros, aplicando metodologías ágiles, por tal motivo he tomado las certificaciones CSM (Certified Scrum Master) y SFC (Scrum Fundamentals Certified), las cuales me han permitido alcanzar de manera satisfactoria los objetivos del puesto.

Mi trabajo consiste en desarrollar sistemas que ayuden a optimizar los procesos de las diferentes unidades de negocio de la empresa (contabilidad, recursos humanos, contraloría etc.), así como a realizar análisis de requerimientos, diseño e implementación de bases de datos y el mantenimiento a sistemas en producción.

En el área de sistemas, particularmente en desarrollo, te enfrentas a varios retos, uno de ellos es la seguridad de la información, para solventar lo anterior, se requieren aplicaciones seguras y confiables, por lo que cada aplicación que codifico lleva implícitos los conocimientos adquiridos en diversas asignaturas que tomé en la Facultad, por ejemplo: ingeniería de software, administración de proyectos, criptografía, bases de datos y algoritmos y estructura de datos.

Capítulo 3

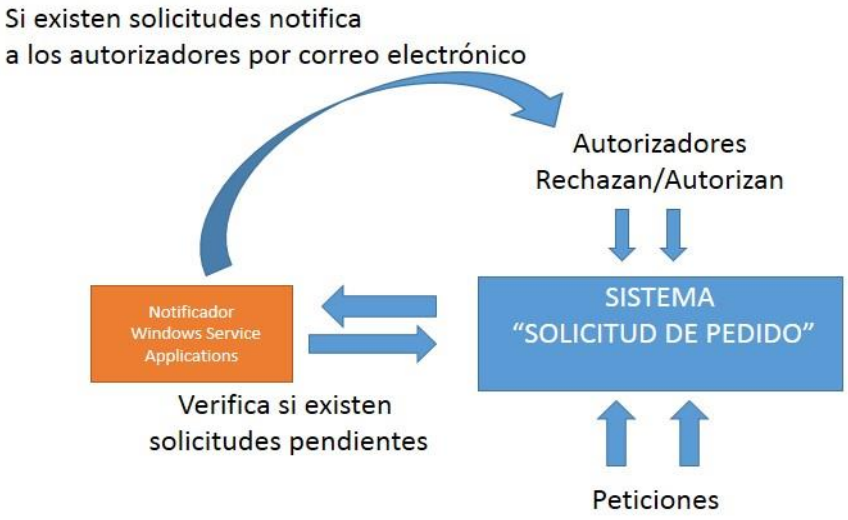
Participación dentro de la empresa

3.1 Descripción del problema

La empresa cuenta con un sistema llamado “Solicitud de pedido”, dicho sistema es el encargado de concentrar todas las peticiones de las diferentes unidades de negocio (contabilidad, finanzas, recursos humanos etc.). El sistema cuenta con autorizadores que dependiendo del presupuesto disponible autorizan o rechazan dicha solicitud.

El problema principal radica en que muchos autorizadores se encuentran fuera de sus oficinas, por lo que la mayor parte del tiempo, no están enterados de la urgencia que tienen algunas peticiones y por consecuencia detienen el trabajo de terceras personas, situación que representa un obstáculo para el correcto funcionamiento de las áreas.

Para solventar la situación anterior, me fue designado el desarrollo de un Windows Service Applications cuyo objetivo se centra en notificar por correo electrónico, en un tiempo determinado, a los autorizadores que se tiene una petición pendiente por revisar para que el flujo de los pedidos siga y no obstaculice las actividades de los empleados.



En la siguiente figura, se muestra el diagrama general del sistema

Figura 3. Diagrama sistema solicitud de pedido

3.2 Metodología

Para implementar el servicio se utilizó la metodología Scrum debido a que es rápida y adaptable al cambio, además la empresa fomenta su uso por ser práctica, sencilla y eficiente lo que ayudó a cumplir en tiempo con el desarrollo del servicio notificador.

3.2.1 Scrum

Scrum es un proceso en el que se aplican de manera regular un conjunto de buenas prácticas para trabajar colaborativamente, y obtener el mejor resultado posible de un proyecto. Estas prácticas se apoyan unas a otras y su selección tiene origen en un estudio de la manera de trabajar de equipos altamente productivos.

En Scrum se realizan entregas parciales y regulares del producto final, priorizadas por el beneficio que aportan al receptor del proyecto. Por ello, Scrum está especialmente indicado para proyectos en entornos complejos, donde se necesita obtener resultados pronto, donde los requisitos son cambiantes o poco definidos, donde la innovación, la competitividad, la flexibilidad y la productividad son fundamentales.

Scrum también se utiliza para resolver situaciones en que no se está entregando al cliente lo que necesita, cuando las entregas se alargan demasiado, los costes se disparan o la calidad no es aceptable, cuando se necesita capacidad de reacción ante la competencia, cuando la moral de los equipos es baja y la rotación alta, cuando es necesario identificar y solucionar ineficiencias sistemáticamente o cuando se quiere trabajar utilizando un proceso especializado en el desarrollo de producto.

3.2.2 El proceso Scrum

En Scrum un proyecto se ejecuta en bloques temporales cortos y fijos (iteraciones de un mes natural y hasta de dos semanas, si así se necesita). Cada iteración tiene que proporcionar un resultado completo, un incremento de producto final que sea susceptible de ser entregado con el mínimo esfuerzo al cliente cuando lo solicite.

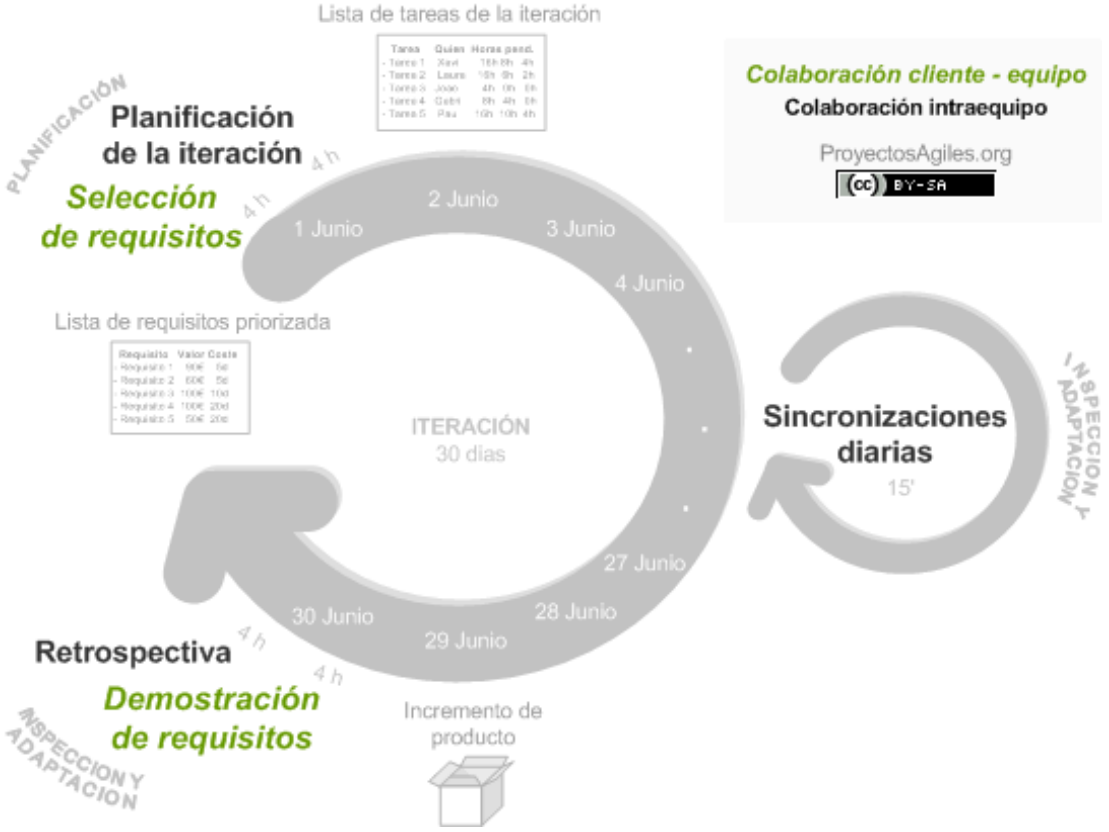


Figura 4 .Proceso Scrum

El proceso parte de la lista de objetivos/requisitos priorizada del producto, que actúa como plan del proyecto. En esta lista el cliente prioriza los objetivos

balanceando el valor que le aportan respecto a su coste y quedan repartidos en iteraciones y entregas.

Las actividades que se llevan a cabo en Scrum son las siguientes:

1. Planificación de la iteración

El primer día de la iteración se realiza la reunión de planificación de la iteración, consta de dos partes:

a) Selección de requisitos (4 horas máximo). El cliente presenta al equipo la lista de requisitos priorizada del producto o proyecto. El equipo pregunta al cliente las dudas que surgen y selecciona los requisitos más prioritarios que se compromete a completar en la iteración, de manera que puedan ser entregados si el cliente lo solicita.

b) Planificación de la iteración (4 horas máximo). El equipo elabora la lista de tareas de la iteración necesarias para desarrollar los requisitos a que se ha comprometido. La estimación de esfuerzo se hace de manera conjunta y los miembros del equipo se autoasignan las tareas.

2. Ejecución de la iteración

Cada día el equipo realiza una reunión de sincronización (15 minutos máximo), después cada miembro del equipo inspecciona el trabajo que el resto está realizando (dependencias entre tareas, progreso hacia el objetivo de la iteración, obstáculos que pueden impedir este objetivo) para poder hacer las adaptaciones necesarias que permitan cumplir con el compromiso adquirido. En la reunión cada miembro del equipo responde a tres preguntas:

- ¿Qué he hecho desde la última reunión de sincronización?

- ¿Qué voy a hacer a partir de este momento?
- ¿Qué impedimentos tengo o voy a tener?

Durante la iteración el Facilitador (Scrum Master) se encarga de que el equipo:

- Cumpla con su compromiso y de que no merme su productividad.
- Elimine los obstáculos que no puede resolver por sí mismo.
- Se proteja de interrupciones externas que puedan afectar su compromiso o su productividad.

Durante la iteración, el cliente junto con el equipo refinan la lista de requisitos (para prepararlos para las siguientes iteraciones) y, si es necesario, cambian o replanifican los objetivos del proyecto para maximizar la utilidad de lo que se desarrolla y el retorno de inversión.

3. Inspección y adaptación

El último día de la iteración se realiza la reunión de revisión y consta de dos partes:

a) Demostración (4 horas máximo). El equipo presenta al cliente los requisitos completados en la iteración, en forma de incremento de producto preparado para ser entregado con el mínimo esfuerzo. En función de los resultados mostrados y de los cambios que haya habido en el contexto del proyecto, el cliente realiza las adaptaciones necesarias de manera objetiva, ya desde la primera iteración, replanificando el proyecto.

b) Retrospectiva (4 horas máximo). El equipo analiza cómo ha sido su manera de trabajar y cuáles son los problemas que podrían impedirle progresar

adecuadamente, mejorando de manera continua su productividad. El Facilitador se encargará de ir eliminando los obstáculos identificados.

3.3 Tecnologías utilizadas

La empresa tiene preferencia por tecnología Microsoft, por ello cuenta con licencias de sus diferentes servicios y productos.

Por tal motivo se utilizaron herramientas de dicha compañía para desarrollar el servicio notificador.

3.3.1 Windows Service Applications

Los servicios son procesos que se encuentran en ejecución en segundo plano y pueden ser configurados para iniciarse junto con el sistema operativo o bien se pueden iniciar manualmente cuando sea necesario. Los servicios de Microsoft Windows, antes conocidos como servicios NT, permiten crear aplicaciones ejecutables de larga duración, que se ejecutan en sus propias sesiones de Windows. Estos servicios pueden iniciarse automáticamente cuando el equipo arranca, se pueden pausar y reiniciar, y no muestran ninguna interfaz de usuario.

Estas características hacen que los servicios resulten perfectos para ejecutarse en un servidor o donde se necesite una funcionalidad de ejecución larga que no interfiera con los demás usuarios que trabajen en el mismo equipo. También puede ejecutar servicios en el contexto de seguridad de una cuenta de usuario específica, diferente de la del usuario que inició la sesión o de la cuenta predeterminada del equipo.

3.3.2 Lenguaje C#

C# es un lenguaje orientado a objetos y con seguridad de tipos que permite a los desarrolladores compilar diversas aplicaciones sólidas y seguras que se ejecutan

en .NET Framework. Se puede utilizar C# para crear aplicaciones cliente de Windows, servicios Web XML, componentes distribuidos, aplicaciones cliente-servidor, aplicaciones de base de datos, etc. Visual C# proporciona un editor de código avanzado, depurador integrado y numerosas herramientas para facilitar el desarrollo de aplicaciones basadas el lenguaje C# y .NET Framework.

3.3.3 SQL Sever

Microsoft® SQL Server™ es un sistema de administración y análisis de bases de datos relacionales de Microsoft para soluciones de comercio electrónico, línea de negocio y almacenamiento de datos. Dentro de sus características fundamentales se encuentran:

• Soporte de transacciones

• Escalabilidad, estabilidad y seguridad

• Soporte de procedimientos almacenados.

3.3.4 Visual Studio

Microsoft Visual Studio es un entorno de desarrollo integrado (IDE, por sus siglas en inglés) para sistemas operativos Windows. Soporta múltiples lenguajes de programación tales como C++, C#, Visual Basic .NET,F#, Java, Python, Ruby, PHP; al igual que entornos de desarrollo web como ASP.NET MVC, Django, etc., Además cuenta con capacidades online bajo Windows Azure en forma del editor Monaco.

Visual Studio permite a los desarrolladores crear sitios y aplicaciones web, así como servicios web en cualquier entorno que soporte la plataforma .NET (a partir de la versión .NET 2002). Así se pueden crear aplicaciones que se comuniquen entre estaciones de trabajo, páginas web, dispositivos móviles, dispositivos embebidos, consolas, etc.

3.4 Implementación del notificador

Como se observa en la siguiente figura, en primer lugar se creó un nuevo proyecto en visual estudio (aplicación de consola).

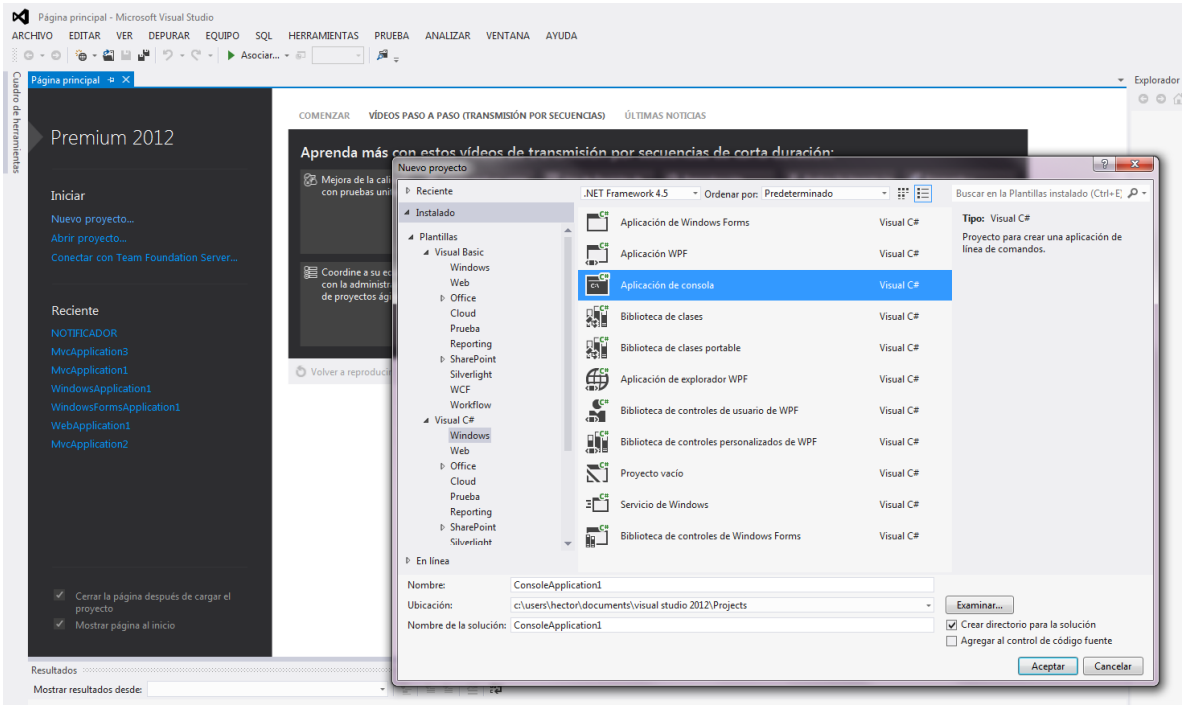


Figura 5 .Creación de un nuevo proyecto en visual estudio

Se comenzó implementando la clase ServiceProcessInstaller, misma que instala un ejecutable que contiene clases que extienden ServiceBase. Las utilidades de

instalación, como InstallUtil.exe, llaman esta clase al instalar una aplicación de servicio.

```
using System.ComponentModel;
using System.Configuration.Install;
using System.ServiceProcess;

[RunInstaller(true)]
public class InstallTLFN : Installer
{
    private ServiceProcessInstaller processInstaller;
    private ServiceInstaller serviceInstaller;

    public InstallTLFN()
    {
        processInstaller = new ServiceProcessInstaller();
        serviceInstaller = new ServiceInstaller();

        processInstaller.Account = ServiceAccount.LocalSystem;
        serviceInstaller.StartType = ServiceStartMode.Manual;
        serviceInstaller.ServiceName = "NOTIFICADOR";

        Installers.Add(serviceInstaller);
        Installers.Add(processInstaller);
    }
}
```

Figura 6 .Clase necesaria para instalar el servicio

La clase que se muestra en la figura 7, es una de las más importantes ya que en ella se definen los tiempos de ejecución, los días y los objetos que mandará llamar cada que se cumplan con las condiciones de tiempo, cabe mencionar que en este caso se establecieron dos horarios de ejecución, pero se pueden programar para que se esté ejecutando cada minuto en un servidor. Esto es de gran utilidad porque, como se demuestra más adelante, con este proceso se puede realizar la acción que se necesite; por ejemplo que cada15 minutos consulte la base de datos para saber si hay registros nuevos o si hubo algún cambio o actualización y si detecta algo que lo notifique.

Realmente un Windows Service es frecuentemente utilizado en las industrias ya que, por ejemplo en las instituciones bancarias cada mes se ejecuta un proceso

para generar un estado de cuenta y mandarlo sin necesidad de que alguna persona lo envíe, debido a lo anterior algunas veces los estados de cuenta suelen llegar por la madrugada ya que se programa a esa hora porque existe menor tráfico en la red.

```
namespace NOTIFICADOR
{
    public class SrvceLlmd : ServiceBase{

        private DateTime timeUnloading;
        private DateTime timeUnloadings;
        private Timer timer = new Timer();
       private int diase ;

        public SrvceLlmd(){
            this.ServiceName = "NOTIFICADOR";
            this.CanStop = true;
            this.CanPauseAndContinue = true;
            this.AutoLog = true;
        }
        protected override void OnStart(string[] args){
            // TODO: add startup stuff
            var timer = new Timer();
            timer.Enabled = true;
            timer.Interval = 1000;
            timeUnloading = DateTime.Parse("10:00:00");
            timeUnloadings = DateTime.Parse("18:00:00");

            //timer.Elapsed += timer_Elapsed;
            timer.Elapsed += timer_Tick;
            timer.Start();
        }

        protected override void OnStop(){
            // TODO: add shutdown stuff
        }

        private void timer_Tick(object sender, EventArgs e)
        {
            DateTime time = DateTime.Now;
            diase=(int) time.DayOfWeek;

            if ((time.Hour == timeUnloading.Hour && time.Minute == timeUnloading.Minute && time.Second == timeUnloading.Second) && (diase != 6 || diase != 0))

            {
              new solicitudespn().auto();
            }

            else if ((time.Hour == timeUnloadings.Hour && time.Minute == timeUnloadings.Minute && time.Second == timeUnloadings.Second) && (diase != 6 || diase != 0))
            {
            new solicitudespn().auto();

            }
        }
    }
}
```

Figura 7 .Clase donde se definen los tiempos de ejecución y acción a seguir

Una vez creada la clase donde se definen los parámetros de ejecución y el tiempo en el que se va a ejecutar, se implementa la siguiente clase que es donde se consultará a la base de datos si existen solicitudes nuevas; el proceso recolectará

la información del autorizador: correo electrónico, nombre, sistema solicitudes pendientes etc. y meterá la información en una lista de tipo empleado cuya clase es la siguiente.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace NOTIFICADOR
{

    public class empleado : IEquatable<empleado>
    {
        public string ClaveUser { get; set; }
        public string Nombre { get; set; }
        public string Correo { get; set; }
        public string SolPend { get; set; }
        public int Sistema { get; set; }
        public int Envio { get; set; }
        public string SolpenCom { get; set; }
        public string SolpenAnt { get; set; }



        public override bool Equals(object obj)
        {
            if (obj == null) return false;
            empleado objAsPart = obj as empleado;
            if (objAsPart == null) return false;
            else return Equals(objAsPart);
        }

        public bool Equals(empleado other)
        {
            if (other == null) return false;
            return (this.ClaveUser.Equals(other.ClaveUser));
        }

    }
}
```

Figura 8 .Clase que muestra los atributos que va recolectar el notificador

Después se implementó una clase para el envío de emails, a continuación se creará una instancia de la misma por cada correo que sea necesario mandar; al método que envía el email es necesario mandarle los siguientes parámetros[1]:

- Asunto
- Correo Destinatario
- Cuerpo del mensaje
- Correo para un adicional (Bcc)

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Diagnostics;
using System.Net;
using System.Security.Cryptography.X509Certificates;

namespace NOTIFICADOR
{
    class envia
    {
        public void envmail ( string asunto, string para, string cuerpo,string bcc)
        {

            /*-------------------------MENSAJE DE CORREO----------------------*/

            //Creamos un nuevo Objeto de mensaje
            System.Net.Mail.MailMessage mmsg = new System.Net.Mail.MailMessage();


            //Direccion de correo electronico a la que queremos enviar el mensaje
            mmsg.To.Add(para);

            //Nota: La propiedad To es una colección que permite enviar el mensaje a más de un destinatario3

            //Asunto
            mmsg.Subject = asunto;
            mmsg.SubjectEncoding = System.Text.Encoding.UTF8;

            //Direccion de correo electronico que queremos que reciba una copia del mensaje
            mmsg.Bcc.Add(bcc); //Opcional
```

Figura 9 .Clase que implementa el envío de correo electrónico

[1] La clase donde se consulta la base de datos no se muestra por seguridad de la información y se omitieron algunos parámetros de configuración por la misma razón.

Se cuenta con un método donde se construye el cuerpo del mensaje en HTML, se llega a esta acción cuando ya se tiene todo la información, en este método se ensambla la estructura del email y se personaliza.

```
public void BodyCompen(string Nombre,int solpencom, string cla, string mail)
{
    string cuerpo = "";
    query = "";

    if (solpencom == 1)
    {

        Console.WriteLine("Sr:" + Nombre + "   tiene: " + solpencom + " solicitud");
        cuerpo = "<strong style='font-family: arial, sans-serif'>Estimado(a) " + Nombre + ":<br><br> Le informamos que usted tiene <strong style=' font-size: 23px;font-family: arial, sans-serif'> " + solpencom +
    }

    else
    {

        Console.WriteLine("Sr:" + Nombre + "   tiene: " + solpencom + " solicitudes");
        cuerpo = "<strong style='font-family: arial, sans-serif'>Estimado(a) " + Nombre + ":<br><br> Le informamos que usted tiene <strong style=' font-size: 23px;font-family: arial, sans-serif'> " + solpencom +

    }

    cuerpo += "<br><br><strong style='font-family: arial, sans-serif'> Para autorizarlas dar click <a href=                                                                    AQUI</a> </strong>";
    cuerpo += "<br>";
    //cuerpo += "" + mail + "";
    cuerpo += "<br><br>";


    cuerpo += "<strong style='font-family: arial, sans-serif'>Atentamente.</strong><br><br>";
    cuerpo += "<strong style='font-family: arial, sans-serif'>Portal Administrativo.</strong><br><br><br>";
    cuerpo += "<strong style=' font-size: 13px;color: #808080;style='font-family: arial, sans-serif'>*Este correo fue enviado por un sistema de notificaciones automáticas, si no desea recibir más notificaciones
```

Figura 10. Método donde se construye el cuerpo del correo

Después de construir el objeto de la clase “envíamail” con la información recopilada se envía un correo por cada autorizador siempre y cuando tenga solicitudes pendientes. Este proceso se repite de manera cíclica hasta que recorra toda la lista de autorizadores.

```
// manda correo
 new envia().envmail("NOTIFICACIONES: Solicitudes pendientes de autorizar                    ,fusion[conta].Correo,
 Bodys(fusion[conta].Envio, fusion[conta].Sistema, fusion[conta].SolPend, fusion[conta].Nombre, fusion[conta].SolpenCom),                );
```

Figura 11. Se crea una instancia nueva de envía mail

Finalmente en el “Main” de la aplicación se detona el proceso que, dependiendo los tiempos definidos, se ejecuta de manera indefinida o hasta que se detenga el servicio. Más adelante se mostrará cómo instalar dicho servicio.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.ServiceProcess;
using System.Diagnostics;

namespace NOTIFICADOR
{
    class Program
    {

        static void Main(string[] args)
        {
            #if (_DEBUG)
            solicitudespn vv = new solicitudespn();

             vv.auto();

#else

ServiceBase.Run(new SrvceLlmd());
#endif

    }
}
}
```

Figura 12. Main del programa donde se detona el proceso

El proceso general del notificador se puede observar en el siguiente diagrama:

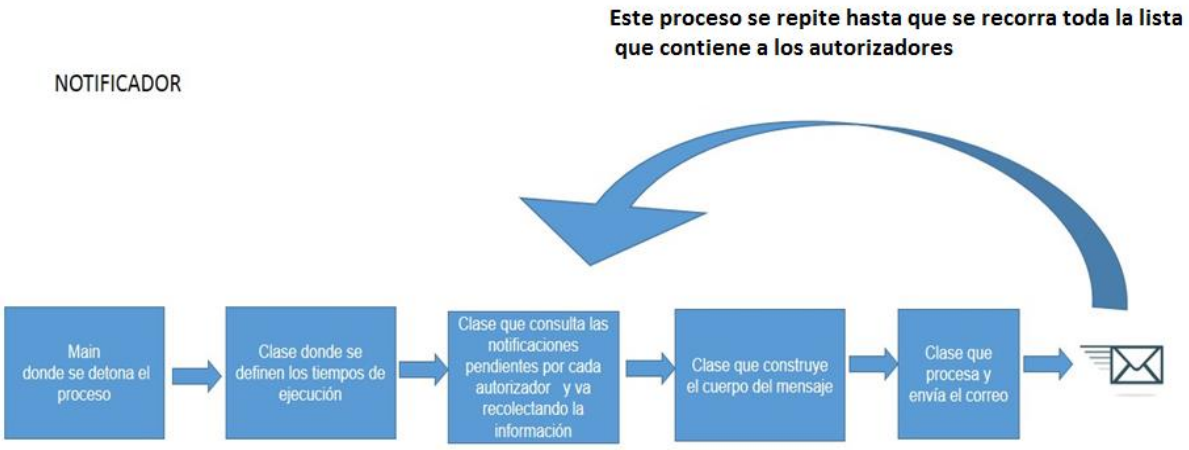


Figura 13.Esquema del funcionamiento general del notificador

3.5 Instalación del servicio

Una vez codificado el notificador es necesario instalar el servicio, para ello se requiere montarlo en un servidor (con sistema operativo Windows). En primer lugar se realiza la compilación del proyecto, al realizar dicha compilación se crea un ejecutable mismo que se copia a una carpeta del sistema ubicada en C:\Windows\Microsoft.NET\Framework\v4.0.30319

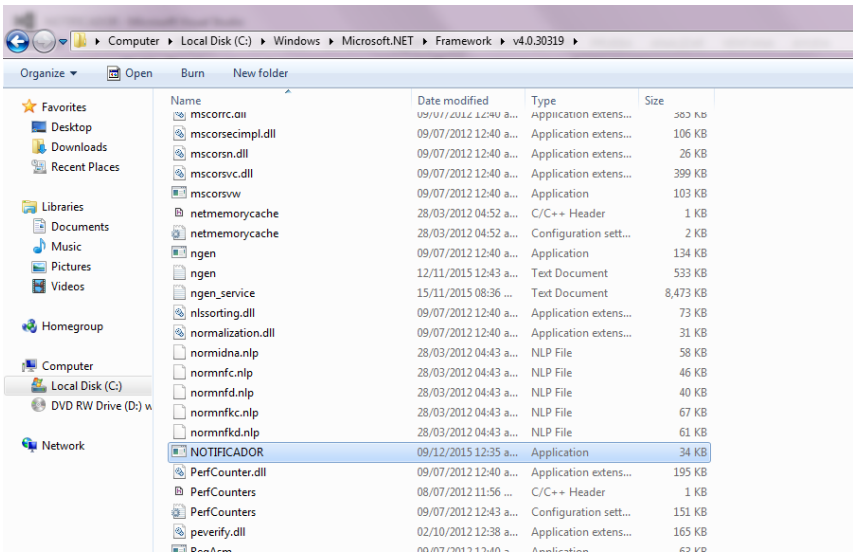


Figura 14.Creación de ejecutable

Posteriormente es necesario instalar dicho servicio, para ello se abre la consola de comando de Windows y se escribe Installutil “NOTIFICADOR.exe” tal y como como se muestra en la figura 15.

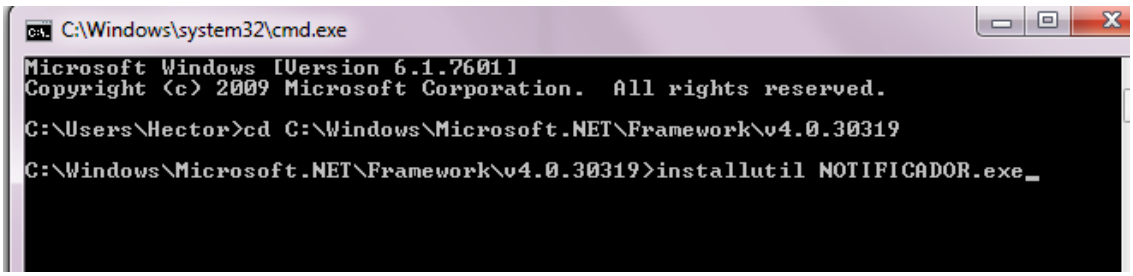


Figura 15. Instalación del servicio

Al finalizar la instalación es necesario iniciar el servicio, una vez activo dicho servicio se ejecutará en el tiempo que se estableció previamente.

Por último, se muestra como evidencia el correo que es enviado con el servicio de manera automática al autorizador siempre y cuando tenga solicitudes pendientes de autorizar.

Figura 16.Correo enviado de forma automática a los autorizadores

3.6 Análisis de los resultados

El resultado obtenido en la implementación del servicio notificador fue de gran impacto ya que dicho proceso optimizó considerablemente el sistema de solicitudes de pedido. El censar en tiempos determinados las peticiones pendientes de autorización y notificación permite a los autorizadores recibir un correo electrónico indicándoles que tienen solicitudes pendientes, además les permite aceptar o rechazar en el instante la petición ya que en el correo se implementó adicionalmente un link con un login seguro que los lleva al sistema de solicitudes de pedido.

El autorizar o rechazar casi al instante las peticiones, reduce significativamente el tiempo de respuesta y esto ayuda al flujo rápido de información, permitiendo que las unidades de negocio realicen su trabajo eficientemente.

Es importante mencionar que los autorizadores cuentan con un teléfono inteligente por parte de la empresa con acceso a internet que a su vez tiene configurado el correo corporativo. Por lo que no es un impedimento estar fuera de las instalaciones ya que ellos pueden revisar su correo electrónico en el teléfono.

Conclusiones

El aprendizaje al realizar este proyecto fue bastante amplio, ya que prendí a trabajar en equipo, analizar e implementar los requerimientos del cliente (en este caso la unidad de negocio contraloría). Es muy satisfactorio que algo que tú creaste sea utilizado y esté actualmente ayudando a optimizar un proceso cotidiano.

Pude desarrollar una aplicación segura y compacta con un lenguaje de programación muy robusto como C#, esto me ayudó a darme cuenta que tengo la capacidad para desarrollar e implementar aplicaciones a la medida de las empresas.

El proceso (Windows Service) se puede mejorar para una segunda versión ya que se pretende adjuntar al correo que se envía, un archivo pdf con un resumen de lo que tiene pendiente por autorizar, para que el autorizador lo pueda revisar sin entrar al sistema.

Este fue uno de los primeros proyectos que realicé, durante mi estancia en la empresa, actualmente he participado en más de seis proyectos diferentes, aprendiendo en cada uno de ellos cosas nuevas que, aunado a mi formación en la Facultad de Ingeniería, ha complementado mi formación profesional.

Al estar trabajando, me percato de que la Ingeniería en Computación es una profesión muy demandada por las empresas ya que cada día buscan optimizar sus procesos para tener un mayor control en sus operaciones y obtener reducción de tiempos y costos.

Me siento muy orgulloso de poner el nombre de la UNAM y de la Facultad de Ingeniería en alto ya que toda la formación que recibimos es de calidad y esto nos permite competir en el mercado laboral a gran nivel, realizando proyectos de exitosos y eficientes.

Índice de figuras

Bibliografía

- Ceballos Francisco, Javier. *Microsoft C# Curso de Programación*.2.ª Edición.RA-MA, 2011

- Pérez Marques, María .*Microsoft Sql Server 2008 R2: Motor de Base de Datos y Administración.,* RC LIBROS, 2011

- Sharp, John *Microsoft Visual C# Step by Step.*8th Edition. Microsoft Press, 2015.

- Orbegozo Arana, Borja. *Desarrollo de aplicaciones C# con Visual Studio .NET*. Alfaomega, 2015

Mesografía

- Microsoft.Creating a Windows Service Application in the Component Designer.Fecha de consulta: 06 Diciembre 2015.Disponible en: https://msdn.microsoft.com/es-es/library/zt39148a(v=vs.110).aspx

- Microsoft.Installutil.exe (Installer Tool).Fecha de consulta: 06 Diciembre 2015.Disponible en: https://msdn.microsoft.com/es-es/library/50614e95(v=vs.110).aspx

- Microsoft. Introducción to Windows Service Applications.Fecha de consulta: 05 Diciembre 2015.Disponible en: https://msdn.microsoft.com/en-us/library/d56de412(v=vs.110).aspx

- Microsoft SQL Server.Fecha de consulta: 06 Diciembre 2015.Disponible en: https://msdn.microsoft.com/es-mx/library/bb545450.aspx

- Microsoft. ServiceProcessInstaller Class Fecha de consulta: 06 Diciembre 2015.Disponible en: https://msdn.microsoft.com/en-us/library/system.serviceprocess.serviceprocessinstaller(v=vs.110).aspx