

**Plan de Proyecto**

GRUPO

**El Colibrí**

CURSO

5K2

**UNIVERSIDAD TECNOLÓGICA NACIONAL**

**INGENIERÍA EN SISTEMAS DE INFORMACIÓN**

**Metodologías Ágiles en el Desarrollo de Software**

**DOCENTES**

Mgtr. Natalia Andriano

Ing. Juan Pablo Bruno

Ing. Mauricio Silclir

**ALUMNOS**

Herrera, Antonio leg. 57824

Pedrosa, Paula Melania leg. 58822

Rojas Amaya, M. Florencia leg. 58577

### 

21/05/2014

Tabla de contenido

[Plan de Proyecto 2](#_Toc388443694)

[Nuestro Producto 2](#_Toc388443695)

[Roles a Desempeñar 2](#_Toc388443696)

[Métricas a Tomar 2](#_Toc388443697)

[Recursos Necesarios 3](#_Toc388443698)

[Recursos humanos 3](#_Toc388443699)

[recursos tecnológicos 3](#_Toc388443700)

[Recursos de software 3](#_Toc388443701)

[Testing 4](#_Toc388443702)

[Criterio de Hecho 5](#_Toc388443703)

[Reglas de Nombrado 5](#_Toc388443704)

# Plan de Proyecto

La metodología con la que vamos a trabajar para llevar a cabo el desarrollo del proyecto será SCRUM.

Los Sprints con los que contará el proyecto tendrán una duración fija de dos semanas. Se llevarán a cabo las ceremonias que Scrum plantea como marco de trabajo ágil: Sprint Planning, Daily Meeting, Sprint Retrospective y Sprint Review. Las Daily Meeting tendrán lugar 3 veces por semana, en las cuales se tendrá en cuenta qué ha llevado a cabo cada integrante del equipo, qué irá hacer y si hubo algún inconveniente en su tarea. En caso que haya ocurrido algún inconveniente en la realización de alguna actividad, entre todo el equipo se tratará de llegar a una solución. Las Daily Meeting tendrán lugar los días martes, jueves y sábados. Se utilizará una plantilla de reuniones para la Daily Meeting donde se indicará la fecha de la reunión, los participantes, los temas que se abordaron, lo que se planificó y cuándo se realizará la siguiente reunión.

## Nuestro Producto

El producto a desarrollar, consiste en un Sistema destinado a la Gestión de torneos de Fútbol. El Sistema tendrá la capacidad de adaptarse a las distintas necesidades que presenten las organizaciones que administren torneos de futbol de cualquier dimensión.

## Roles a Desempeñar

**Scrum Master:** Antonio Herrera.

**Team:** Paula Pedrosa – Antonio Herrera – Florencia Rojas.

**Product Owner:** Mauricio Silclir.

## Métricas a Tomar

***Burndown Chart****:*

Llevaremos a cabo la realización del diagrama burndown Chart o diagrama de quemado que constituye en una representación gráfica del trabajo que queda por hacer en relación con el tiempo que queda por hacerlo. El BurnDown Chart se realizará para cada Sprint, en función de las historias de usuario que hemos determinado y planificado que tendrán lugar en esa iteración.

***Velocidad:***

Las historias de usuario están puntuadas. La velocidad constituye una métrica del número de puntos o valor de las historias de usuario que están siendo completadas. Definiremos la velocidad en función a historias de usuario completas en todo un Sprint.

Con esta métrica, podemos observar cuál es la productividad de nuestro equipo de trabajo, dado a que mientras mayor sea la velocidad del team, menores serán los tiempos para poder llevar a cabo la producción de más funcionalidades de nuestro Software.

***Capacidad:***

La capacidad del equipo está definida por la cantidad de horas que tiene el team en cada sprint para poder llevar a cabo las tareas asociadas a cada historia de usuario que tiene lugar en la iteración. Se toma esta métrica para poder hacer uso de esas medidas obtenidas en las estimaciones de los próximos sprints.

Para poder llevar a cabo la realización de estas métricas: BurnDown Chart, Velocidad y Capacidad del equipo, se utilizarán Planillas de Excel. Para el caso de la Velocidad, se procederá a indicar el número de puntos de historias de usuario que hemos completado en una iteración. Para el Burndown Chart, registraremos el avance en cuanto al trabajo para una iteración y construiremos el gráfico en Excel. Además se hará uso de la herramienta Team Fundation Server, que te permite hacer la administracion de un proyecto agil. La misma, nos facilita la definición del product backlog, la especificación de los sprint que tendrán lugar a lo largo del desarrollo del proyecto, las historias de usuario que se tratarán por cada sprint, la duración de cada sprint, nos permite descomponer cada historia en diferentes tareas, definir el board por sprint, la cantidad de horas disponibles de los integrantes por iteración, el estado de las tareas, visualizar rápidamente las tareas que han sido realizadas, y aquellas que faltan por llevar a cabo, como así también visualizar las horas quemadas por el equipo. Además, podemos obtener la toma de métricas de manera automática, tales como el burndown chart, la velocidad y la capacidad del equipo.

## Recursos Necesarios

Los recursos que son necesarios para llevar a cabo este proyecto son:

## Recursos humanos

Se requiere personalque cuente con conocimientos en metodologías de desarrollo de sistemas, gestión ágil, programación web con tecnología .Net y SQL Server.

## recursos tecnológicos

* 3 equipos con Sistema Operativo Windows 7 o superior
* Hosting para alojar el proyecto
* Hosting IIS para alojar el sistema y la Base de Datos

## Recursos de software

Las Herramientas de software que se utilizarán a lo largo del proyecto son:

* Lenguaje de Programación: **C# (Framework .net).**
* Base de Datos: **SQL Server 2012 y SQL Management Studio.**
* Herramienta de versionado: **Tortoise SVN y Ankh SVN.**
* Diseño de Interfaces de Usuario: **Divshot y Bootstrap.**
* Soporte para la comunicación: **Skipe, Teamviewer 9, Facebook, WhatsApp.**
* Respaldo de información: **GoogleDrive.**

Link del Repositorio: <https://riouxsvn.com/svn/quegolazo>

Para la gestión de la Documentación definitiva, utilizaremos Google Drive. Link de la carpeta de Google Drive: <https://drive.google.com/folderview?id=0B73_O7IPXDQPMTRmaXlScTU1NXc&usp=sharing>

Herramienta que estamos utilizando para gestionar las historias y los sprints. Se puede observer el burndown chart de cada sprint, y las tareas que se realizaron en cada sprint. Link: <https://quegolazo.visualstudio.com/DefaultCollection/ProyectoGestorDeCampeonatos>

## Testing

El Plan de Testing se llevará a través de:

* Casos de Prueba
* Ciclos de Prueba Manuales
* Pruebas Unitarias Automatizadas
* Tests Exploratorios
* Tests de Regresión
* Revisión de Código

|  |  |
| --- | --- |
| Casos de Prueba | Qué funcionalidad se probará, qué escenario se contemplará. La idea principal del desarrollo de los Casos de Prueba, es que los mismos cubran los criterios de aceptación de las historias de usuarios que fueron pactadas al comienzo del sprint. Se ejecutan primero aquellos casos de prueba que prueban las funcionalidades que agregan más valor al cliente. |
| Ciclos de Prueba Manuales | Se ejecutarán una cierta cantidad de veces los casos de prueba definidos. Una vez corregidos los errores, se ejecutará nuevamente el ciclo de prueba completo. |
| Pruebas Unitarias Automatizadas | Se desarrollarán Unit Test automatizados para verificar la lógica del producto. Cada historia de usuario debe tener al menos un Unit Test asociado. |
| Tests Exploratorios | Los tests exploratorios se llevarán a cabo en base a un escenario específico, en donde se probarán las distintas posibles combinaciones y variaciones del mismo. Realizaremos al menos un test exploratorio por historia de usuario y asociado algun aspecto de gran valor para el cliente. |
| Tests de Regresión | Cada dos sprint, se llevará a cabo la realización un ciclo de vida completo, donde se probarán todos los casos de prueba de los sprints anteriores. |
| Revisión de Código | Se revisarán las líneas de código en función a las reglas de nombrado previamente definidas por el equipo y con buenas prácticas de codificación, que el código sea eficiente y legible, y que esté comentado. Cada método debe estar comentado con una breve descripción, indicando los parámetros que recibe y lo que retorna. |

Se hacen usos de plantillas para la revisión de código, para la definición de los casos de prueba y para registrar los resultados de la ejecución de los ciclos de prueba.

## Criterio de Hecho

Para que una historia se considere como hecha y pueda ser presentada ante el Product Owner, la misma debe cumplir los siguientes requisitos:

* Todo el código fuente debe cumplir con las reglas de nombrado definidas en este documento.
* Debe pasar el 100% de los test funcionales planteados para la historia. (https://riouxsvn.com/svn/quegolazo/trunk/Documentacion/Testing).
* Debe pasar el 100% de los test unitarios funcionales planteados para la historia.
* La codificación debe ser revisada de a pares, dejando constancia formal de dicha revisión en el repositorio (https://riouxsvn.com/svn/quegolazo/trunk/Documentacion/Revisiones de Código).
* Cada historia debe tener un documento que explique, ya sea mediante diagramas UML, prototipos de interfaz o bien descripciones textuales, la manera en que el equipo desarrollará la solucion.

## Reglas de Nombrado

**Base de Datos:**

**Nombre de las tablas:** UperCamelCase y los nombres siempre en plural. (Equipos, Jugadores, JugadoresXEquipo)  
**Atributos de las tablas:** camelCase (nombre, goles, cantidadDePartidos)

**Claves primarias:** “id”+ el nombre de la tabla en singular (idJugador, idCampeonato )

**Claves foráneas**: exactamente igual que la clave primaria a la que referencia.

**Programación:**

**Nombre de las clases:** UperCamelCase en singular (Equipo, GestorEquipo, Cancha)

**Nombre de los atributos:** camelCase, siempre teniendo en cuenta que los atributos de aquellas clases que tengan persistencia datos deben ser exactamente igual en el código C# y en el nombre en la base de datos.

**Ennumerados y constantes:** mayúsculas separadas con guines bajos (CESPED, MOSAICOS, TIPO\_DE\_CANCHA)

**Vectores:** vector+ “Nombre de la clase que contiene” ( vectorEquipos)

**Listas:** lista+ “nombre del tipo de objetos que contiene” (listaJugadores)

**Métodos:** camelCase (agregarEquipo() ), uso obligatorio de la etiqueta <sumary> <sumary/> describiendo la funcionalidad del método.

**Métodos Gestores:** siempre deben estar implementados los métodos:

* obtenerTodos() : Lista o Datatable
* obtenerPorId(long id) : Object
* registrar(Object objetoAInsertar)
* modificar ()

**Controles de Formularios:**

**Controles de formularios:** abreviación del nombre del control + nombre significativo

**Textbox: txt**NombreDeJugador

**Grillas: gv**Partidos **Botones: btn**AgregarEquipo **Dropdownlist: ddl**Campeonatos

**Labels: lbl**Fecha **RadioButtons :** rbSexo

**CheckBox : cb**Eliminatorias  
**RadioButtonGroup: rbg**TiposDeCampeonato

**Panel:**  **panel**Jugadores

**Literal: lit**NombreDelJugador

**TextArea: txa**Descripcion:

**ListBox: lbox**Jugadores

**Consultas a la Base de Datos:**

**Sentencias INSERT, UPDATE, DELETE y SELECT, las palabras reservadas dentro de la consulta deben ir en mayúscula.**

**Las palabras reservadas que se utilizan en la sentencia SELECT, tales como SELECT, FROM, INNER JOIN, JOIN, LEFT JOIN, RIGHT JOIN, WHERE, HAVING, ORDEY BY, GROUP BY deben ir en mayúscula.**

**Las consultas en la Base de Datos, deben realizarse de la siguiente manera:**

**SELECT ……**

**FROM ……**

**WHERE ……**

**GROUP BY ……**

**HAVING ……**

**ORDER BY ……**

**Es decir uno abajo del otro.**

**Generales:**

**ABM los botones de las interfaces deben decir:**Alta:“Registrar”  
Modificación: “Modificar”

Baja: “Eliminar”

Consulta: “Consultar”.