

**Plan de Métricas**

GRUPO

N°3

CURSO

5K2

**DOCENTES**

Ing. Zohil, Julio

Ing. Liberatori, Marcelo

Ing. Jaime, Natalia

**ALUMNOS**

Allemand, Facundo leg. 58971

Herrera, Antonio leg. 57824

Pedrosa, Paula Melania leg. 58822

Rojas Amaya, M. Florencia leg. 58577

### 

**UNIVERSIDAD TECNOLÓGICA NACIONAL**

INGENIERÍA EN SISTEMAS DE INFORMACIÓN PROYECTO FINAL

**PROYECTO**



**Que Golazo!**

Sistema de Gestión de Torneos de Fútbol

24/05/2014

|  |  |  |  |
| --- | --- | --- | --- |
| HISTORIAL DE VERSIONES | | | |
| VERSION | **FECHA** | **RESPONSABLE** | **OBSERVACION** |
| 1.0 | 11/05/2014 | Antonio Herrera | Creación Documento |
| 1.1 | 24/05/2014 | Paula Pedrosa | Se actualizaron las métricas a tomar: Burndown chart, velocidad, capacidad, porcentaje de casos de pruebas exitosos y líneas de código. |
| 1.2 | 24/05/2014 | Antonio Herrera | Corrección de versión 1.1, falta agregar Release Burndown chart |
| 1.3 | 31/05/2014 | Paula Pedrosa | Corrección de versión 1.2, se completó reléase Burndown chart. |

Tabla de contenido

[Introducción 2](#_Toc388720630)

[Métricas a Medir 2](#_Toc388720631)

[Sprint Burndown Chart 2](#_Toc388720632)

[Release Burndown Chart 3](#_Toc388720633)

[Velocidad del Equipo 3](#_Toc388720634)

[Capacidad del Equipo 4](#_Toc388720635)

[Cobertura de Líneas de Código de Unit Test 5](#_Toc388720636)

[Porcentaje de Casos de Prueba Exitosos 5](#_Toc388720637)

# Introducción

El objetivo de este documento es detallar el plan de Métricas a utilizar en el proyecto y cada cuánto tiempo se medirán, así como también presentar un resumen sobre los resultados que se esperan obtener de estas métricas, incluyendo una descripción sobre como se utilizarán para poder controlar el avance del proyecto.

# Métricas a Medir

Durante el desarrollo del proyecto, se tomaremos las siguientes métricas:

* Sprint Burndown Chart
* Release Burndown Chart
* Velocidad del Equipo
* Capacidad del Equipo
* Cobertura de Líneas de Código de Unit Test
* Porcentaje de Casos de Prueba Exitosos

## Sprint Burndown Chart

|  |  |
| --- | --- |
| Métrica: Sprint Burndown Chart |  |
| ¿Qué es? | Representación gráfica de las horas de trabajo que quedan por hacer en relación con el tiempo que queda por hacerlo. Esta métrica constituye el gráfico de trabajo remanente. |
| ¿Cada cuánto se medirá? | Mediremos el trabajo remanente ***por Sprint***. |

Un Burndown Chart a lo largo del tiempo, muestra la velocidad a la que se está completando los objetivos/requisitos planteados. Permite verificar si el equipo podrá completar el trabajo en el tiempo estimado, y tomar medidas correctivas para cumplir el objetivo en caso que se identifiquen inconvenientes.  
En el caso del **Sprint** Burndown Chart se tiene en cuenta el total de horas estimadas para las tareas de un sprint determinado (en el eje vertical) y los días transcurridos en el sprint (en el eje horizontal), comparando las horas restantes de trabajo del equipo con el ritmo ideal (línea recta que une la cantidad de horas estimadas con el día de fin del sprint).

Este gráfico permite identificar rápidamente si el equipo llega a terminar con las tareas programadas para un sprint específico, o si es nceseario cambiar el rumbo de acción o renegociar los requerimientos del sprint para poder cumplir correctamente los objetivos planificados del mismo.

La responsabilidad de que el Sprint Burndown Chart sea llevado a cabo correctamente es del Scrum Master, quien debe verificar que el equipo mantenga actualizado el gráfico indicando diariamente el estado de sus tareas asignadas, es decir si están realizadas, si se encuentran en progreso o todavía restan por desarrollarlas.

Para el desarrollo de este proyecto, se obtendrá un Burndown Chart por cada sprint que tenga lugar durante el desarrollo e implementación de nuestro producto.

## Release Burndown Chart

|  |  |
| --- | --- |
| Métrica: Release Burndown Chart |  |
| ¿Qué es? | Representación gráfica de los puntos de historia que quedan por quemar en relación con el tiempo que queda para hacerlo. |
| ¿Cada cuánto se medirá? | Se medirá para cada release que tendrá lugar a lo largo del proyecto. Nuestro producto se compondrá de 3 releases a desarrollar. Se mide ***al final de cada sprint.*** |

Un release puede tener lugar durante varios Sprint, esta métrica nos permitirá visualizar los puntos de historia que restan por quemar para obtener el release. Es decir que al final de cada sprint, podremos observar cuanto trabajo queda por realizar para obtener el release.

## Velocidad del Equipo

|  |  |
| --- | --- |
| Métrica: Velocidad del Equipo |  |
| ¿Qué es? | Cantidad de trabajo realizado por el equipo. Sólo cuenta trabajo completado para el cálculo de la velocidad |
| ¿Cada cuánto se medirá? | Mediremos la velocidad del equipo de trabajo ***por Sprint***. |

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. Representa la cantidad de trabajo realizado por el equipo. Definiremos la velocidad del equipo 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.

La forma de cálculo consiste básicamente en tomar el número de puntos o valor de historias de usuarios completados en cada Sprint.

## Capacidad del Equipo

|  |  |
| --- | --- |
| Métrica: Capacidad del Equipo |  |
| ¿Qué es? | Cantidad de horas disponibles del equipo de trabajo. |
| ¿Cada cuánto se medirá? | Mediremos la capacidad del equipo de trabajo ***por Sprint***. |

La capacidad del equipo está definida por la cantidad de horas que tiene el equipo en cada sprint para poder llevar a cabo las tareas asociadas a cada historia de usuario que tiene lugar en la iteración. Constituye una estimación de cuanto trabajo puede completarse en un período de tiempo dado. Está basado en la cantidad de tiempo ideal disponible del equipo. Se toma esta métrica para poder hacer uso de esas medidas obtenidas en las estimaciones de los próximos sprints. La capacidad del equipo se medirá en función de la cantidad de horas disponibles por sprint. Es muy importante, que los integrantes del equipo calculen su capacidad de trabajo de manera realista y a conciencia.

Para poder llevar a cabo la realización de estas métricas: Sprint BurnDown Chart, Release BurnDown Chart, Velocidad y Capacidad del equipo, se utilizarán Planillas de Excel. Para el caso de la Velocidad, se indicará el número de puntos de historia que hemos completado en una iteración. Para el Sprint Burndown Chart, registraremos el avance en cuanto al trabajo para una iteración y construiremos el gráfico en Excel, al igual que para el Release BurnDown Chart. Además se hará uso de la herramienta Team Fundation Server, que permite hacer la administracion de un proyecto ágil. La misma, nos facilita las siguientes tareas:

* 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.
* La descomposición de cada historia en diferentes tareas.
* La definición del board por sprint.
* La administración de la cantidad de horas disponibles de los integrantes del equipo por iteración.
* La visualización del estado y los responsables de cada tarea.
* La rápida visualización de las tareas que han sido realizadas, y aquellas que faltan por llevar a cabo, como así también la distinción de las horas quemadas por el equipo.
* La obtención de la toma de métricas de manera automática, tales como el Burndown chart, la Velocidad y la Capacidad del equipo.

## Cobertura de Líneas de Código de Unit Test

|  |  |
| --- | --- |
| Métrica: Cobertura de Líneas de Código de Unit Test |  |
| ¿Qué es? | Porcentaje de cobertura de líneas de código que está destinado a los unit test. |
| ¿Cada cuánto se medirá? | Mediremos la cobertura de líneas de código de unit test ***por Sprint*** y se acumulará sprint a sprint y luego obtendremos un porcentaje. |

En primer lugar, se calculará la cantidad de líneas de código escritas. Esta métrica no la utilizaremos para medir complejidad, ya que no constituye una medida representativa, sino que se utilizará como entrada de cálculo para obtener otra métrica: Cobertura de líneas de código de Unit Test. Esta cobertura de líneas de código de unit test se usará fundamentalmente para obtener una medida representativa del esfuerzo extra que le estamos dedicando a las pruebas unitarias automatizadas y poder obtener de esta manera determinar qué porcentaje de cobertura está destinado a los unit test.

La cobertura de líneas de código de Unit test se calculará por Sprint, y se recalculará sprint a sprint para obtener un porcentaje de la misma.

## Porcentaje de Casos de Prueba Exitosos

|  |  |
| --- | --- |
| Métrica: Porcentaje de Casos de Prueba Exitosos |  |
| ¿Qué es? | Porcentaje de casos de prueba que, una vez ejecutados, se obtuvo el resultado planificado en un principio. |
| ¿Cada cuánto se medirá? | Mediremos el porcentaje de casos de pruebas ***por Sprint*** para definir si una historia de usuario cumplió con los criterios de hecho definidas. |

Otra de las métricas que tomaremos, es el porcentaje de casos de pruebas ejecutados con un resultado exitoso. A través de esta métrica, mediremos la cantidad de casos de prueba planificados para ejecutar que fueron ejecutados y obtuvieron el resultado que habíamos planificado en un principio. El porcentaje de casos de pruebas exitosos se tomará por Sprint, esta métrica es una medida de la calidad de software en ese sprint. Nos ayuda principalmente para poder comprobar si las historias de usuario que tienen lugar en un sprint específico cumplen con los criterios de hecho que han sido definidos.

Para que una historia de usuario se considere como *hecha* y pueda ser presentada ante el Product Owner, la misma debe cumplir los siguientes requisitos: a nivel de casos de prueba, el Testing de cada sprint se considerará completo y que pasó óptimamente cuando no haya presencia de errores de severidad mayor ni menor, es decir que todos los errores mayores y menores encontrados, fueron corregidos. Sólo se admite que el testing pasó, cuando estamos en presencia de errores cosméticos que deben ser corregidos para la siguiente iteración.