UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

título del TFG
Documentación Técnica

Presentado por nombre alumno
en Universidad de Burgos — 9 de diciembre de 2021
Tutor: nombre tutor

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este anexo se tratará el plan de proyecto, es la base sobre la que se crea el proyecto. Desde el punto de vista de la temporalidad y la viabilidad. Es una parte fundamental del ya que permitirá visualizar el escenario en el que se desarrollará el proyecto, permitiendo hacer una alineación estratégica de todos los elementos que se deben completar para finalizar correctamente el proyecto.

Desde el punto de vista de la planificación temporal, el proyecto sigue la metodología ágil *Scrum*. Permitiendo definir cada uno de los objetivos que se desean alcanzar, los elementos que los componen y su respectiva prioridad.

Scrum, de manera muy resumida, trabaja con u *product backlog*, es una lista de prioridades en función del valor de cada tarea. Cuando comienza un *sprint*, se empieza a trabajar en las tareas que se encuentren en el *sprint backlog*, estas han sido extraídas del *product backlog*. En el caso de este proyecto se realiza una reunión de planificación, *sprint planning*, cada dos semanas aproximadamente.

Para el control y seguimiento se utiliza una herramienta externa, *Zenhub*, la cual permite la definición de las tareas, el seguimiento de cada una de ellas en función de la planificación póker, seguimiento de cada *sprint*, el versionado, etc.

Seguidamente se realizará un estudio de la viabilidad del proyecto, tanto a nivel económico como legal.

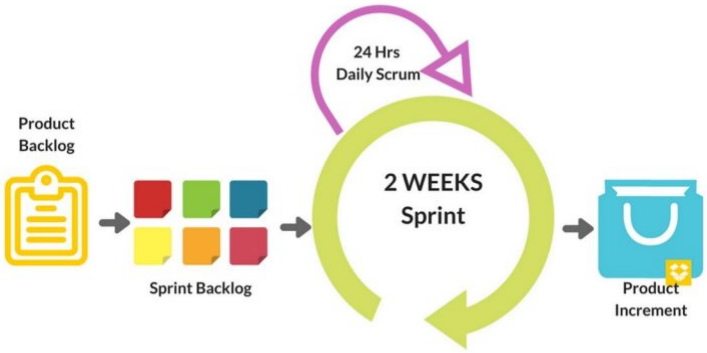

Figura A.1: Metodología *scrum*.

A.2. Planificación temporal

SCRUM

Scrum es un marco de trabajo que permite el trabajo colaborativo en equipos. Permite que los equipos que trabajan en proyectos con esta metodología se organicen por sí mismos, siendo ellos los que deciden cómo afrontar los problemas que van surgiendo.

Según [4], el modelo *Scrum* se basa en tres componentes principales: roles, procesos y artefactos. El *Scrum Master* es el puesto asumido por el director o gerente del proyecto, o en algunos casos el líder del equipo. Esta figura representa los valores y principios por los que se rige la metodología de *scrum*, manteniendo los valores y buenas prácticas, así como resolviendo los impedimentos que vayan surgiendo a lo largo del desarrollo del proyecto. Habitualmente los equipos están compuestos por entre cinco y diez personas que trabajan en el proyecto a tiempo completo. Siendo este equipo independiente y flexible en cuanto a jerarquía interna, no siendo representado el papel del "jefe" dentro de este por la misma persona siempre. Esto genera que el papel cambie en función de las necesidades del propio proyecto, la configuración del equipo cambia únicamente entre iteraciones, o *sprints*, no dentro de los mismos.

Sprints

Los *sprints* son periodos breves de **tiempo fijo** en el que el equipo trabaja para completar una cantidad de trabajo pre-establecida. Si bien muchas guías asocian los *sprints* a la metodología ágil, asociando la metodología ágil y la metodología seguida en *scrum* como si fueran lo mismo, cuando no lo son. La metodología ágil constituye una serie de principios, y la metodología *scrum* es un marco de trabajo con la única finalidad de conseguir resultados.

A pesar de las similitudes los *sprints* poseen un objetivo subyacente, entregar con frecuencia *software* de trabajo.

Sprint meetings

Dentro de la metodología *scrum* existen diferentes reuniones que favorecen la agilidad del proyecto y que todo el mundo sepa lo que tiene que hacer en cada momento.

- ***Sprint planning meeting.*** Esta reunión puede tener una duración de hasta de un día completo de trabajo. En ella deben de estar presentes todas las partes del proyecto, i.e. el *Scrum Master*, el equipo de desarrollo, y el *product owner*. Poseen dos partes, en la primera de ellas se define el *product backlog*, requerimientos del proyecto y se definen los objetivos para el *sprint* que comienza, i.e. lo que se espera "construir" o completar en el *sprint*. En la segunda parte de la reunión se trabaja en el *sprint backlog*, las tareas que se van a seguir en el *sprint* para completar el objetivo de éste.
- ***Daily meeting.*** Debido a que los requerimientos del proyecto no se pueden variar durante la vida de un *sprint*, existen las reuniones diarias que son organizadas por el *Scrum Master* en las que se comenta el trabajo del día previo, lo que se espera de ese día y qué está retrasando o impidiendo a un individuo el proseguir con sus tareas, esta reunión no debe tener una duración de más de quince minutos y se debe realizar "de pie". No es una reunión para ver quién retrasa el proyecto sino para ayudar a quién lo necesite entre todos los miembros del equipo y permitir esa agilidad.
- ***Sprint review meeting.*** Reunión fijada al final de cada *sprint* en la cual se hace una puesta en conocimiento de lo que se ha realizado en ese *sprint*, siempre que se pueda se hará una demostración funcional en lugar de una presentación al *product owner*. Esta reunión tiene un carácter informal.

Artifacts

Uno de los componentes más importantes de cara a la metodología *scrum* son los artefactos, o *artifacts* por su nombre en inglés. Éstos incluyen el *product backlog*, el *sprint backlog* y los *burn down charts*.

- ***Product backlog.*** Lista de trabajo ordenada por las prioridades para el equipo de desarrollo. Es generada a partir de las reuniones de planificación de los *sprints*, contiene los requisitos. Se encuentra actualizado y clasificado en función de la periodicidad asignada a las tareas, pudiendo ser de corto o largo plazo. Aquellas tareas que se deban resolver a corto plazo deberán estar perfectaemnte descritas antes de asignarlas esta periodicidad, implicnddo que se han diseñado las historias de usuario completas así como el equipo de desarrollo ha establecido las estimaciones correspondientes. Los elementos a largo plazo pueden ser abstractos u opacos, conviene que estén estimados en la medida de lo posible para poder tener en cuenta el tiempo que llevará desarrollarla.

  Los propietarios del producto dictan la prioridad de los elementos de trabajo en el *product backlog*, mientras que el equipo de desarrollo dicta la velocidad a la que se trabaja en *backlog.*[10]

  La estimación es una parte muy importante ya que es lo que permitirá al equipo de desarrollo mantener el ánimo y el trabajo al ritmo deseado. La estimación es realizada en la *sprint planning meeting*, en la que se estima para cada tarea/producto del *product backlog*. No se busca tener un resultado exacto del tiempo que va a llevar al equipo completar esa tarea, sino es una previsión. Para realizar correctamente la estimación se debe tener en cuenta el tamaño y la categoría de la tarea, los puntos de historia que se le van a asignar, así como el número de horas y días que van a ser necesarias para completar la tarea.

- ***Sprint backlog.*** Lista de tareas extraídas del *product backlog* que se han acordado desarrollarse a lo largo de un *sprint*. Este *backlog* es seleccionado por el propio equipo de desarrollo, para ello seleccionan una tarea del *product backlog* y se divide en tareas de menor tamaño y abordables. Aquellas tareas de menor tamaño que el equipo no haya sido capaz de desarrollar previo a la finalización del *sprint* quedarán almacenadas para próximos *sprints* en el *sprint backlog*.

Actores, roles y responsabilidades

Dentro de un equipo que sigue la metodología *scrum* encontramos diferentes actores, como ya se ha comentado el equipo de desarrollo suele estar compuesto por entre cinco y diez personas, además del *Scrum Master* y el *Product Owner*.[11]

- ***Product Owner.*** Encargado de optimizar y maximizar el valor del producto, es la persona encargada de gestionar las prioridades del *product backlog*. Una de sus principales tareas es la de intermediario con los *stakeholders*, partes interesadas, del proyecto; junto con recoger los requerimientos de los clientes. Es habitual que esta figura sea representante del negocio, con lo que aumenta su valor.

  Para cada *sprint* debe de marcar el objetivo de éste de manera clara y acordada con el equipo de desarrollo, lo cual hará que el producto vaya incrementando constantemente su valor. Para que todo fluya como debe, esta figura tiene que tener el "poder" de tomar decisiones que afecten al producto.

- ***Scrum Master.*** Figura con dos responsabilidades, gestionar el proceso *scrum* y ayudar a eliminar impedimentos que puedan afectar a la entrega del producto.

  1. Gestionar el proceso *scrum*. Su función es asegurarse de que el proceso se lleva a cabo correctamente, facilitando la ejecución de éste y sus mecánicas. Consiguiendo que la metodología sea una fuente de generación de valor.
  2. Eliminar impedimentos. Eliminar los problemas que vayan surgiendo a lo largo de los *sprints* con el fin de mantener el ritmo de trabajo dentro de los equipos de desarrollo para poder entregar valor, manteniendo la integridad de la metodología.

- **Equipo de desarrollo.** Formado por entre cinco y diez personas encargados del desarrollo del producto, organizados de forma autónoma para conseguir entregar las tareas del *product backlog* asignadas al *sprint* correspondiente. Para que funcione correctamente la metodología todos los integrantes deben de conocer su rol dentro del equipo, internamente se pueden gestionar como el equipo considere, pero de cara "hacia fuera" son un equipo con una responsabilidad.

Planificación por *sprints*

La organización temporal del proyecto se ha organizado siguiendo los estándares de la metodología *scrum*, i.e. usando *sprints*.

Inicialmente la *sprint planning meeting* es realizada cada dos semanas, debido a una falta de costumbre de trabajo con esta metodología se combina junto con la *sprint review meeting*, de forma que en una sola reunión se comenta tanto lo que se ha hecho como lo que está por realizarse en el siguiente *sprint*.

La velocidad de desarrollo del proyecto es una incógnita, debido a la no existencia de referencias previas del equipo de desarrollo del proyecto, en proyectos de ésta índole. Por lo tanto, la duración de los *sprints* puede que se vea ajustada a lo largo de la vida del proyecto.

No se utilizan *daily meetings* puesto que a pesar de que se invierte una media de tres a cinco horas diarias en el desarrollo, no es considerada necesaria. Si bien en caso de problemas se acuerda una reunión para el día siguiente con el fin de mantener la agilidad y no retrasar el proyecto.

Sprint 0: *Lights out and away we go!*

El *sprint* con el que comienza el desarrollo de este proyecto no ha seguido la metodología *scrum*, puesto que se formuló desde un punto de vista de toma de contacto inicial con el trabajo de investigación y todo lo que ello conlleva.

Los objetivos definidos han sido:

1. Lectura de *papers* relacionados con el ámbito de la inteligencia artificial. En concreto *SSL density peaks*[12], *Co-Training*[2], *Tri-Training*[14] y *Democratic Co-Learning*[13].

El tiempo empleado en la lectura y asimilación de estos conceptos ha sido de catorce horas, es la primera vez que se leen *papers* o artículos científicos completos procurando asimilar todos los conceptos de éstos. Se ha desarrollado entre el veintisiete de octubre y el cinco de noviembre, de dos mil veintiuno.

Sprint 1: Chad

- ***Planning meeting***

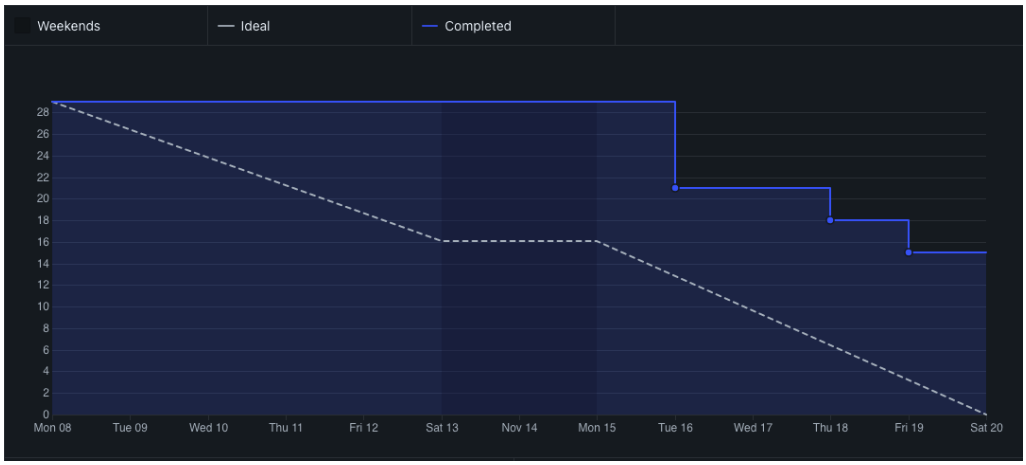

Figura A.2: *Burndown Chart Sprint 1.*

Objetivos del primer *sprint*:

1. Lectura del API de *scikit-learn*. Comprensión del funcionamiento de los transformadores y estimadores enfocado desde el punto de su programación.
2. Lectura de los *papers On issues instance selection*[9], *Comparison of instances seletion algorithms I. algorithms survey*[8] y *Comparison of instance selection algorithms II. Results and comments*[6].
3. Implementación de las técnicas de reducción del conjunto de entrenamiento, basados en k-NN.

- **Marcas temporales** El *sprint* se desarrolla entre el ocho y el diecinueve de noviembre de dos mil veintiuno. Han sido dedicadas al desarrollo del proyecto treinta horas.
- ***Burndown chart*** Durante este *sprint* el trabajo inicial comenzó ligeramente retrasado, motivos en el apartado *sprint review meeting*, por lo tanto podemos observar en la Figura A.2 como el trabajo completado dista del ideal o proyectado para este *sprint*.

  En el *sprint backlog* habían sido incluidos todos los algoritmos a programar, es por ello que indica que se ha completado aproximadamente la mitad del trabajo.
- ***Sprint review meeting*** El trabajo en este primer *sprint* ha salido adelante correctamente. Al ser el primer *sprint* ha habido un pequeño

error de configuración del repositorio junto con la herramienta ZenHub, de ahí que en el *burndown chart* de esta semana, Figura A.2, aparezca como que la primera semana del *sprint* no ha habido trabajo completado.

La adaptación a la metodología ágil ha resultado un poco compleja.

Sprint 2: Holleyman

- ***Planning meeting***

  Objetivos del segundo *sprint*:

  1. Finalizar implementación de los algoritmos basados en técnicas de reducción del conjunto de entrenamiento.
  2. Añadir la documentación correspondiente a los algoritmos implementados.
  3. Comprobar el rendimiento de los algoritmos implementados respecto a los resultados de una ejecución similar con el software *Weka*.

  El *sprint* se desarrolla entre el veintidós de noviembre y el tres de diciembre de dos mil veintiuno. Han sido dedicadas al desarrollo del proyecto treinta y ocho horas.

- ***Burndown chart***

  El trabajo realizado a lo largo de este *sprint* ya ha sido adecuado a la metodología *scrum*, obteniendo un *burndown chart*, Figura A.3, con más sentido que la que se había obtenido en el *Sprint* 1.

  El equipo de desarrollo se sigue habituando poco a poco a la metodología de trabajo y en este *sprint* se ha trabajo por debajo del "ideal" para el proyecto.

- ***Sprint review meeting***

  A lo largo de este *sprint* se descubrió un problema en la forma de identificar los k-NN en el algoritmo *Condensed Nearest Neighbor, CNN*[7], retrasando el trabajo cuatro horas, entre identificación y reprogramación. Este error se descubrió mientras se investigaba otro error, en este caso el algoritmo *Iterative Case Filtering, ICF*[3] terminaba en error buscando los k-NN de las últimas instancias.

  La implementación de los algoritmos *Reduced Nearest Neighbor, RNN*[5] y *Modified Selective Subset, MSS*[1] ha sido relativamente asequible una

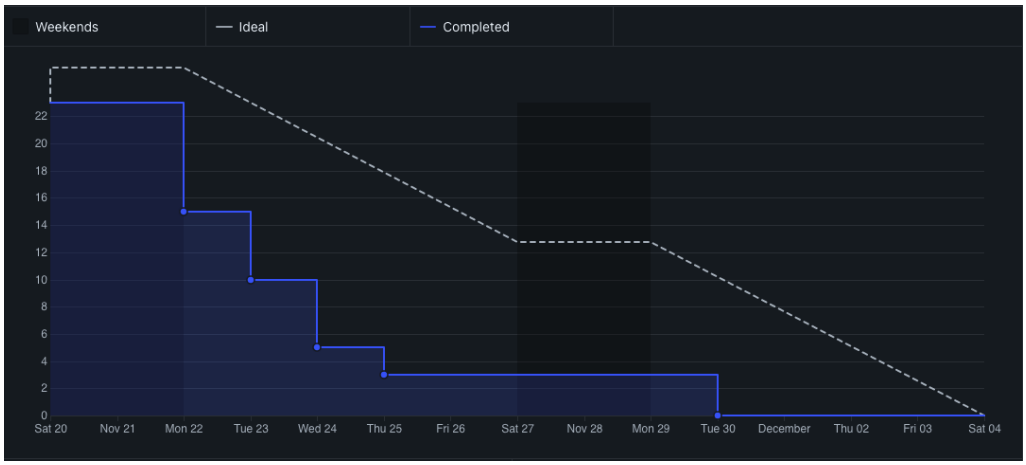

Figura A.3: *Burndown Chart Sprint 2.*

vez se comprendía el algoritmo en cuestión así como su funcionamiento (entradas, procesado, salidas...).

Sprint 3: Manion

- ***Planning meeting***

  Objetivos del tercer *sprint*:

  1. Comenzar la documentación del proyecto.
     - Comenzar la memoria por el marco teórico.
     - Comenzar los anexos por la planificación temporal.

     Se va a realizar en LaTeX.
  2. Aprender lo básico de LaTeXlo más rápido posible para poder trabajar con él.
  3. Buscar la precisión de los algoritmos implementados con conjuntos etiquetados de $[1\,\%, 5\,\%, 10\,\%, 20\,\%, 50\,\%]$ del conjunto total. En búsqueda de las asíntotas donde ya no mejora la clasificación.
  4. Validación de los algoritmos de selección de instancias con *Weka* y *KNN*.

- **Marcas temporales**

El *sprint* se desarrolla entre el seis y el diecisiete de diciembre de dos mil veintiuno. Han sido dedicadas al desarrollo del proyecto XYZ horas.

- ***Burndown chart***
- ***Sprint review meeting***

***Sprint* n: Name**

- ***Planning meeting***
    1. Primero
    2. Segundo
- **Marcas temporales**
- ***Burndown chart***
- ***Sprint review meeting***

A.3. Estudio de viabilidad

Viabilidad económica

Viabilidad legal

Apéndice B

Especificación de Requisitos

B.1. Introducción

B.2. Objetivos generales

B.3. Catalogo de requisitos

B.4. Especificación de requisitos

Apéndice C

Especificación de diseño

C.1. Introducción

C.2. Diseño de datos

C.3. Diseño procedimental

C.4. Diseño arquitectónico

Apéndice D

Documentación técnica de programación

D.1. Introducción

D.2. Estructura de directorios

D.3. Manual del programador

D.4. Compilación, instalación y ejecución del proyecto

D.5. Pruebas del sistema

Apéndice E

Documentación de usuario

E.1. Introducción

E.2. Requisitos de usuarios

E.3. Instalación

E.4. Manual del usuario

Bibliografía

[1] Ricardo Barandela, Francesc J Ferri, and J Salvador Sánchez. Decision boundary preserving prototype selection for nearest neighbor classification. *International Journal of Pattern Recognition and Artificial Intelligence*, 19(06):787–806, 2005.

[2] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the eleventh annual conference on Computational learning theory*, pages 92–100, 1998.

[3] Henry Brighton and Chris Mellish. Advances in instance selection for instance-based learning algorithms. *Data mining and knowledge discovery*, 6(2):153–172, 2002.

[4] H Frank Cervone. Understanding agile project management methods using scrum. *OCLC Systems & Services: International digital library perspectives*, 2011.

[5] Geoffrey Gates. The reduced nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 18(3):431–433, 1972.

[6] Marek Grochowski and Norbert Jankowski. Comparison of instance selection algorithms ii. results and comments. In *International Conference on Artificial Intelligence and Soft Computing*, pages 580–585. Springer, 2004.

[7] Peter Hart. The condensed nearest neighbor rule (corresp.). *IEEE transactions on information theory*, 14(3):515–516, 1968.

[8] Norbert Jankowski and Marek Grochowski. Comparison of instances seletion algorithms i. algorithms survey. In *International conference*

on artificial intelligence and soft computing, pages 598–603. Springer, 2004.

[9] Huan Liu and Hiroshi Motoda. On issues of instance selection. *Data Mining and Knowledge Discovery*, 6(2):115, 2002.

[10] Dan Radigan. El backlog del producto: la lista de tareas pendientes definitiva, 2021.

[11] Julio Roche. Scrum: roles y responsabilidades, 2020.

[12] Di Wu, Mingsheng Shang, Xin Luo, Ji Xu, Huyong Yan, Weihui Deng, and Guoyin Wang. Self-training semi-supervised classification based on density peaks of data. *Neurocomputing*, 275:180–191, 2018.

[13] Yan Zhou and Sally Goldman. Democratic co-learning. In *16th IEEE International Conference on Tools with Artificial Intelligence*, pages 594–602. IEEE, 2004.

[14] Zhi-Hua Zhou and Ming Li. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Transactions on knowledge and Data Engineering*, 17(11):1529–1541, 2005.