UNIVERSIDAD DE BURGOS
ESCUELA POLITÉCNICA SUPERIOR
Grado en Ingeniería Informática

TFG del Grado en Ingeniería Informática

título del TFG
Documentación Técnica

Presentado por Álvaro Ruifernández Palacios
en Universidad de Burgos — 11 de enero de 2019
Tutor: Jesús Enrique Sierra García

Índice general

Índice de figuras

Índice de tablas

Apéndice A

Plan de Proyecto Software

A.1. Introducción

En este apartado se especificará la planificación seguida en cuanto a tiempo se refiere para realizar cada una de las tareas que han conformado este proyecto, permitiendo gracias a esta planificación cumplir con los objetivos que se han establecido para el sistema desarrollado.
Debido a que gran parte de este proyecto se ha basado en la investigación (ya que, como se ha explicado en otros puntos, se desconocían algunos de los procedimientos a realizar), la planificación temporal no ha podido ser tan estricta como debería ser en un proyecto cuya organización se basa en la metodología ágil conocida como SCRUM como es este caso. Otro aspecto que ha impedido seguir esa planificación ha sido la aparición de una gran cantidad de contratiempos surgidos durante cada una de las fases de este proyecto, como se describirá más adelante.
En cuanto al estudio de viabilidad, la parte económica ha sido también un aspecto difícil de calcular ya que se ha necesitado probar una serie de implementaciones, tanto hardware como software, hasta dar con la idónea para cumplir los requisitos del proyecto.

A.2. Planificación temporal

Como bien se ha descrito en la introducción, en este apartado se especificará el desarrollo del proyecto descrito a partir de los periodos de tiempo que han llevado a desarrollar cada parte, o como se conoce en terminología

SCRUM, cada uno de los sprints.

Sprint 0: 05/10/2017 - 22/11/2017

Este es el sprint inicial, donde se explica y establece los conceptos sobre los que se va a trabajar y los elementos que se van a emplear para ello. Se plante la realización de este trabajo basandose en el sistema operativo Ubuntu, en su versión 14.04 LTS, para desarrollar el código usando la aplicación conocida como Eclipse en su versión 4.8 (conocida como Photon) y el leguaje de programación C++. Además se plantearon las librerías a utilizar para este proceso y la utilización del dispositivo MTX-GTW para la ejecución remota del sistema en este dispositivo.

Tareas

- Creación de la máquina virtual con el sistema operativo antes mencionado.
- Configuración de Eclipse con las correspondientes librerías.
- Preparación de los diagramas que servirán como guía para el desarrollo del sistema.
- Búsqueda de cables de alimentación para cada uno de los elementos.
- Comienzo de investigación sobre la realización de ejecuciones remotas.
- Creación del repositorio.

Backlog

En este sprint inicial se plantea cada uno de los aspectos del proyecto (requisitos, objetivos...) así como el planteamiento de la organización con el tutor en cuanto a reuniones que se iban a realizar. Además se empezó a organizar la estructura del programa con la realización de los diagramas de clases.

Investigación

Se comienza la investigación a cerca de las ejecuciones remotas de programas para poder realizarlo con el MTX-GTW. Tambien se comienza a

investigar a cerca del manejo de sockets y lecturas de puertos utilizando C++ para poder recoger los datos del láser y enviar mensajes por este mismo puerto para comunicarse con el láser (con la dificultad de que esta escucha también será remota).

Sprint 1: 22/11/2017 - 16/01/2018

En este sprint se plantean los cambios a realizar en el diagrama creado en el anterior sprint, se plantea el uso de LaTeXpara la realización de la memoria del proyecto y la creación del diagrama de secuencia para mejor comprensión del funcionamiento del sistema. Se comprueba también el estado de la máquina virtual y se detecta la falta de algunas librerías necesarias.

Tareas

- Creación del diagrama de secuencia.
- Mejora del diagrama de clases según la revisión realizada.
- Solucionar la ausencia de librerías antes mencionado.
- Realización de pruebas de conexión con el MTX-GTW.
- Continuación de búsqueda de material hardware.

Backlog

En este sprint se termina de crear y modificar los diagramas que servirán de base a la posterior creación del código del sistema, se pretende configurar Eclipse con las librerías que faltaban del sprint anterior y se implementa el programa que imprima por pantalla "Hola mundorealizando esta ejecución en el dispositivo remoto. Mientras se realizan estos procesos se prosigue con la búsqueda del cable de alimentación del láser ya que el cable del MTX-GTW ya ha sido encontrado.

Investigación

En este sprint se realizan varias tareas de investigación. En primer lugar, se deben encontrar las librerías que se necesitan para incluirlas en la configuración del proyecto en Eclipse. Esta tarea resulta costosa debido a que

algunas de las librerías que se habían acordado utilizar estaban desactualizadas y no se podían encontrar y se desconocía la organización de ficheros y el funcionamiento de las nuevas versiones con lo que se hacia complicado la configuración para algunas de las tareas futuras.
La otra tarea de investigación realizada fue la búsqueda de los diferentes cables de alimentación y esto es debido a la diferencia entre ambos ya que el dispositivo MTX-GTW necesita ser alimentado con una corriente alterna de 12 V. Debido a que, en la actualidad, la mayoría de los cargadores de los diferentes aparatos electrónicos cotidianos no suministran más de 7 voltios se necesitó fabricar un cargador a base de un transformador que suministrase la corriente deseada unido a dos cable los cuales irán conectados al cabezal del aparato debido a que tampoco posee un cabezal convencional. Por otro lado, el láser se han utilizado dos cableados distintos ya que se debe conectar tanto a la red eléctrica para alimentarse como al dispositivo antes mencionado. Para la primera de las conexiones descritas se ha empleado un adaptador a corriente alterna ya que este láser necesita alimentarse con 24 V de corriente alterna. Para la segunda conexión se necesita un cable con cabezal VGA y el otro extremos sin cabezal para poder conectar solo los cables que sean necesarios ya que solo se necesitan algunos de los pines de esta conexión para el envío y recepción de datos.

Pruebas

En este sprint se comienzan a realizar las primeras pruebas para la conexión del PC con el MTW-GTW para configurar de forma correcta esta conexión con el fin de tenerla de forma definitiva como conexión de ambos sistemas en el proyecto final.

Sprint 2: 16/01/2018 - 01/02/2018

Una vez se han creado los diagramas para aclarar en funcionamiento del sistema final, se plantean las tareas para este nuevo sprint. En este caso, una vez se posee una versión estable del programa que realiza las pruebas se empieza a plantear el inicio del desarrollo del código del sistema, así como la creación de una cuenta en la web Trello (mencionada en puntos anteriores) para la creación del panel con las tareas a realizar con lo que se consigue una version del panel de tareas para la organización de cada sprint.

Tareas

- Paso del proyecto de pruebas al repositorio para comenzar en el el desarrollo del sistema
- Creación de cuenta y tablón en Trello.
- Creacion de las primeras clases.

Backlog

En este sprint se ha comenzado con el desarrollo del sistema ya en el repositorio para mantener un control de las versiones que se van realizando. Tambien se comienza a dejar constancia de cada una de las tareas que se han plantado en los sprint, dividiéndolas en diferentes categorias segun se encuentren pendientes de realizar, en curso o si ya han sido realizadas.

Investigación

Se retoma la investigación a cerca de los sockets en el lenguaje de C++ para plantear la conexión con el láser con el objetivo de organizar la estructura final de la comunicación lógica de los diferentes elementos físicos que componen el sistema.

Sprint 3: 01/02/2018 - 20/05/2018

Este es uno de los sprints más largos debido a que ha sido en el que han surgido la mayoria de problemas. En este punto, al no poder plantear nuevas tareas debido al bloqueo de las ya presentes del anterior sprint, las reuniones se fueron separando unas de otras en el tiempo.

Tareas

- Solución de los problemas de Ubuntu y Eclipse.
- Continuación de la creación de las distintas clases y procedimientos.

Backlog

En este Sprint, como se ha dicho anteriormente, surgen la mayor parte de las incidencias que impiden el avance ene el proyecto por un largo periodo de tiempo. Una vez en proceso de creación del sistema, Eclipse comenzó a dar errores en cuanto a la comprensión de comandos básicos del lenguaje C++.

Junto con esta clase de problemas surgió el inconveniente del fallo completo de Eclipse, ya que en la máquina virtual donde se estaba realizando todo el proceso, la aplicación dejó de funcionar, permitiendo verse en pantalla únicamente la barra de tareas. Para solucionar este fallo se realizaron algunas investigaciones sin éxito.

Investigación

Las investigaciones realizadas en este periodo de tiempo fueron varias aunque todas con el mismo objetivo, solucionar fallos.
En un inicio, los fallos en cuanto al lenguaje de programación. Para solucionar este tipo de fallo se realizón un análisis exaustivo de la configuración completa del proyecto, incluyendo partes que no se habían visto hasta ese momento, por si se podría haber producido algún efecto en cascada que hubiese desencadenado el error (dicho análisis fue realizado de forma conjunta con el profesor). Tras comprobar que ese no era el problema se comenzó una búsqueda por diversas páginas web para encontrar soluciones de otros usuarios que hubiesen tenido el mismo problema. Al realizar esta búsqueda, no solo se descubrió que el número de personas cone este error era muy reducido, sino que además se pudo observar que las soluciones que se les ofrecía para solucionarlo o no funcionaban o no eran útiles para el problema que se planteaba en nuestro caso. Como conclusión a este problema se decidió crear otro proyecto y dejar para más adelante la configuración realizada con anterioridad.
En cuanto al problema de la visualización de Eclipse, la situación era similar a la del anterior problema. En este caso el número de personas con el mismo problema era mayor pero la solución que se daba en la mayoría de los casos era la eliminación de los recursos temporales generados al arrancar Eclipse para reiniciarlo y que los crease de nuevo, intentando solventar así los fallos que hayan podido aparecer durante el arranque. Al no conseguir solucionarlo de esta manera, se optó por la opción más drástica, extraer los cambios ya realizados en el proyecto creado para evitar el primer problema, eliminar la máquina virtual y comenzar la instalación y configuración de aplicaciones desde el principio.
Este método no funcionó, por lo que se decidió tomar un camino alternativo para el proyecto.

Sprint 4: 20/05/2018 - 13/06/2018

Tras el periodo anterior de cambios de máquina virtual y reconfiguraciones se tomó la decisión de cambiar el proyecto. A partir de este sprint el proyecto sería realizado en Windows 10 (sistema anfitrión del PC) para evitar problemas con los servicios de virtualización. Debido a esa decisión se hacia incompatible la utilización del MTX-GTW, ya que este aparato es incompatible con Windows, por lo que la ejecución del sistema se realizaría en el PC. Este aspecto solucionó otro de los problemas anteriores, las librerías. Al no tener que realizar ninguna ejecución remota, el proyecto solo iba a necesitar utilizar las librerías que ya vienen por defecto en C++, sin necesidad de descargar ninguna otra. En este periodo se comienza las primeras interacciones con el láser

Tareas

- Configuración del proyecto y desarrollo del código en Windows.
- Investigación sobre programas de manejo del láser.
- Cambios importantes en la memoria.

Backlog

En este periodo de tiempo se consigue la realización de gran parte del código que forma el sistema software ya que no se producen errores como los del anterior sprint. También se instala y empieza a investigar el funcionamiento del programa que se encuentra adjunto al láser en un CD presente en la misma caja del aparato. Junto con este programa, tambien se descarga e instala el programa RealTerm, el cual se usará para envir y recibir mensajes por el puerto USB, ya que el láser tiene la posibilidad de conectarse al PC por esta vía.

Investigación

En este aspecto se empieza a investigar la configuración del programa RealTerm antes mencionado, dejando para más adelante la programación de la parte restante del sistema, para centrarse en averiguar como conseguir la conexión con el láser además de conoces como realizar el intercambio de información.
Tambien se investiga la forma de configurar del programa del láser (UAM Proyect Designer) para poder ver el funcionamiento de este programa y la

visualización por pantalla de las lecturas del láser, lo cual se podrá tomar como guía en la representación del resultado de la ejecución del sistema objetivo. Esto se consigue gracias a la configuración de los puertos USB de PC para que permitan instalar drivers no firmados (como son los del propio láser, los cuales tambien se tuvieron que buscar por internet para evitar la dependencia con el CD) con lo que se pudiese reconocer al láser para que el programa funcionase correctamente. Este buen comportamiento también se produjo ya que el tutor facilitó un fichero de configuración básica el cual fue enviado vía USB al propio láser, el cual lo conserva en memoria desde entonces.

Sprint 5: 13/06/2018 - 19/11/2018

En este sprint, cuya duración se debe a la aparición del último de los problemas de gran relevancia para el proyecto y la presencia del periodo de verano lo cual supone una mayor dificultad para las reuniones. En este sprint se analizan los resultados del ciclo anterior, observando las lecturas del láser por primera vez desde el inicio del proyecto. El problema antes mencionado surge al intentar crear el socket lógico en C++ tras conocer como crear los mensajes y provocar la comunicación entre los dos aparatos y conseguir asi implementar una de las partes más importantes del proyecto. No se consigue (entre el alumno y el profesor) que el socket conecte el PC con el láser con lo que se dedica este periodo a solucionar este problema

Tareas

- Solucionar falta de comunicación con el láser desde el PC.
- Cambio de lenguaje del proyecto

Backlog

En este periodo se realizan algunas investigaciones con resultados nada satisfactorios y, tras no conseguir el bojetivo principal de este sprint, se decide cambiar de lenguaje. En este caso se pasa todo el código existente a Python. Este lenguaje, al ser más usado, posee un soporte mucho mayor, especialmente por parte de los usuarios que lo usan para desarrollar su código, por lo que la traducción del código se realiza de forma rápida y sencilla. Gracias a la utilización de este lenguaje se consigue configurar el socket y tenerlo plenamente operativo de forma muy sencilla, con lo que en

el periodo final de este sprint se consigue solucionar el priblema inicial y provocar un grán avance.

Investigación

En este periodo la primera investigación, como ya se ha comentado anteriormente, es la solución del socket en el lenguaje de C++, sin resultado satisfactorio. A parte de esta, la investigación principal se centró en la creación de sockets en Python lo cual se consiguió de forma muy sencilla gracias a la ayuda proporcionada por la propia web del lenguaje y los usuarios que poseian este problema en diferentes foros web.

Pruebas

Se realizan las primeras pruebas de comunicación y obtención de respuestas por parte del láser que da como resultado la implementación del procedimiento que permite extraer la lectura y conservar esos datos en el sistema para su posterior análisis.

Sprint 6: 19/11/2018 - 11/12/2018

En este periodo de tiempo se desarrolla el análisis de los datos recibidos por el láser (descripción del análisis en el punto 4 de la memoria). Con todos nuevos porcedimientos desarrollados, se crea la primera versión plenamente operativa del sistema software que conforma el objetivo del proyecto.

Tareas

- Crear procedimientos de análisis de datos.
- Crear representación gráfica del análisis.
- Integración de comentarios para mejor comprensión del código-

Backlog

Como se ha explicado antes se crea la primera versión operativa del proyecto, con interacción del usuario al introducir las coordenadas de las áreas que desea analizar (procedimiento el cual posee prevención de errores). Se añaden también comentarios al código para facilitar su lectura. Además de esto se implementa l amayor parte del código de forma modular para facilitar su escalabilidad futura.

Investigación

En este aspecto, este es es sprint en el que más se investiga ya que es en el que se desarrolla el comportamiento principal del sistema, por lo que se debe investigar la forma de hacerlo lo más eficiente y a la vez correcto.

Pruebas

En este sprint se han realizado pruebas con diferentes objetos a diferentes distancias para comprobar el correcto funcionamiento del programa. Tras algunas pruebas se demuestra que el programa es capaz de dibujar en la gráfica su entorno con mayor precisión cuanto más cerca estén los objetos que detecta.

Sprint 7: 11/12/2018 - 04/01/2019

En este sprint se pretende mejorar el comportamiento del sistema haciendo que funciones de forma continua, es decir, sin tener la necesidad de recargar el programa cada vez que se desea analizar una nueva lectura. A demás de esto, se comienza el proceso de creación de la versión final de la memoria.

Tareas

- Creación de comportamiento continuo.
- Creación de versión final de la memoria.

Backlog

Trascurrido este sprint no se ha conseguido la implementación continua asi que se ha devuelto el código a su forma funcional. Por la parte de la memoria se produce un avance muy significativo en cuanto a su versión final se refiere.

Investigación

Esta parte cuenta con la investigación en cuanto a las pruebas realizadas con otros comandos del láser usados para poder desarrollar el funcionamiento continuo, aunque, como ya se ha explicado en puntos anteriores, no se ha conseguido, teniendo que volver a la forma más funcional y estable que se tenía hasta el momento.

A.3. Estudio de viabilidad

En el desarrollo de este proyecto también se ha de estudiar si el sistema desarrollado es rentable y cumple con la legalidad vigente para poder instalarlo en los AGVs de cualquier empresa, es decir, se ha de estudiar su viabilidad, tanto económica como legal.

Viabilidad económica

En este caso se analizarán los costes de cada aspecto del proyecto los cuales tengan una importancia en este aspecto.

Coste de personal

En este caso se debe de hacer una estimación a cerca del coste que supone el trabajo del programador a lo largo de toda la duración del proyecto. Debido a que algunos periodos han sido vacacionales (en cuanto al calendario laboral estándar) y otros no se ha avanzado en el proyecto, se estimará un trabajo de 10 meses. Teniendo como salario (tambien obtenido de una estimación aproximada) de 9 euros la hora y una jornada laboral estándar de 6 horas al dia y trabajando 20 dias cada mes el suelo se calcularía como:
$9^{euros}/_{hora} * 4^{horas}/_{dia} = 36^{euros}/_{dia}$
$36^{euros}/_{dia} * 20^{dias}/_{mes} = 720^{euros}/_{mes}$
$720^{euros}/_{mes} * 10meses = 7200euros$
Con esto se estima el coste de personal en 7200 euros.

Coste de materiales

En este caso se ha de calcular tanto el coste del sistema hardware final como el de los materiales que han sido empleados a lo largo del proyecto para el desarrollo de las investigaciones y las pruebas de otras formas de implementación del sistema. Sabiendo esto y estimando Los costes de material de pruebas en 100 euros, el precio del láser en 400 euros, el del PC en 600 euros y 15 euros del cableado de alimentación y conexión entre los dos elementos finales del sistema, estimando una duración de 6 años para los finales y 6 meses para los materiales de prueba se pueden realizar los siguientes cálculos:
$(400euros/(12^{meses}/_{año} * 6años)) * 6meses = 33{,}33eurosLaser$
$(600euros/(12^{meses}/_{año} * 6años)) * 6meses = 49{,}99eurosPC$

$(15euros/(12^{meses}/_{\text{año}} * 6\text{años})) * 6meses = 1{,}25eurosCables$
$33{,}33eurosLaser + 49{,}99eurosPC + 1{,}25eurosCables + 100eurosPruebas = 184{,}57eurosMaterial$
Con esto se estima el coste de material en 184.57 euros.

Costes software

Tambien hay que tener en cuenta los costes en cuanto a los programas y aplicaciones que se han empleado. En este caso el único coste a tener en cuenta es la licencia del sistema oprativo final, Windows 10, con un valor de 120 euros con una vida útil de 4 años. El resto de software empleado no ha tenido coste económico alguno (a excepción de el programa UAM Proyect Designer el cual ya se encuentra incluido en el coste del láser previamente tenido en cuenta). Por lo tanto los cálculos a realizar son:
$(120euros/(12^{meses}/_{\text{año}} * 4\text{años})) * 6meses = 15eurosSO$
Con esto se estima el coste del software en 15 euros.

Costes totales

Con todos los cálculos anteriores se puede calcular de forma estimada (ya que, como bien se ha dicho, todos los cálculos se han realizado a través de estimaciones) el coste total del proyecto. Para ello, hay que hacer la suma de todos los costes y tener el cuenta los porcentajes correspondientes a la Cotización del desarrollador a la Seguridad Social. Se sabe que estos porcentajes son : 23.6 % de contingencias comunes, 5.5 % de desempleo y 0.6 % por la formación profesional. Estos porcentajes hacen un total del 29.7 %, con lo que los costes finales se calculan como:
$7200 + (7200 * 0{,}297) = 9388{,}4eurosDesarrollador$
$9388{,}4eurosDesarrollador + 15eurosSO + 184{,}57eurosMaterial = 9537{,}97eurosTotales$
Con esto se estima el coste total en 9537.97 euros.

Beneficios

Después de conocer en cuanto se estima el coste del sistema para la compañía, en este apartado se calculará el beneficio que este software proporciona. Debido a la orientación de este, el software se ha de distribuir instalado en los AGVs en los cuales se va a emplear.
Se estima un total de 500 vehículos en los cuales este programa va a ser instalado, con un precio de 40 euros por cada uno de estos vehículos cada

año. Con esto el beneficio sería de:
$500 vehículos * 40^{euros}/_{vehículo} = 20000^{euros}/_{año} de beneficio$
Con estos beneficios se estimará también el tiempo que e tiempo en el que la compañía será capaz de recuperar la inversión es de: $9537{,}97 eurosTotales / 20000^{euros}/_{año} = 0{,}48 años$
La compañía recuperará la inversión realizada en este proyecto en poco menos de 6 meses.

Viabilidad legal

Para poder comprobar que el sistema implementado no incumple ninguna ley o norma vigente hemos de analizar dos aspectos: las licencias software (comprobando que todos los programas y aplicaciones utilizados posean licencia) y la legislación en cuanto a los AGV en los que irá instalado el software.

Licencias software

Para comenzar esta sección, se van a investigar las licencias que poseen los distintos programas utilizados para el desarrollo de este proyecto.

- Numpy: licencia BSD.
- MatPlotLib: licencia BSD.
- Jupyter Notebook: licencia BSD.
- Spyder: licencia BSD.
- RealTerm: licencia BSD.
- Eclipse: licencia EPL.

Por la compatibilidad con la mayoría de los programas utilizados y por ser además una de las licencias adecuadas para un proyecto Python según su guía, se ha decidido que para este proyecto se va a escoger una licencia BSD (o Berkeley Software Distribution). Se ha escogido este tipo de licencias debido a la protección que ofrece, tanto al código del programa como al creador del mismo. A pesar de tener ciertos aspectos restrictivos, también posee algunas ventajas importantes:

- Esta licencia permite que otras versiones a crear por usuarios ajenos al desarrollo inicial puedan crear sus propias versiones con distintos tipos de licencia, e incluso, crear versiones y distribuirlas como software libre.
- Se permite realizar casi cualquier acción que el usuario quiera realizar sobre el software, incluyendo el uso este para productos propietarios, es decir, productos con licencias más restrictivas.
- Es una de las de las licencias que más se acerca a la definición de software libre, debido a la libertad ofrecida al usuario con respecto al software adquirido.

A parte de estas características, como ya se ha dicho con anterioridad, esta licencia también posee ciertas características no tan favorables al usuario o al software:

- Se debe incluir el reconocimiento del origen del software en cualquier anuncio o distribución realizada aunque no es obligatorio la colocación del nombre de los autores.
- No se incluye ninguna restricción en esta licencia que ayude a que las versiones sucesivas del software sean libres con lo que, como se ha comentado anteriormente, se podrían asignar licencias mucho más restrictivas a cualquiera de esas versiones.

En cuanto a la licencia de Eclipse, se ha tomado la decisión de no considerarla para tomar la decisión anterior debido a que solo se empleó esta aplicación para las fases iniciales del proyecto, por lo cual no tiene relevancia en el funcionamiento del sistema final.

Legislación de sobre la robótica

A nivel nacional no existe ningún tipo de legislación a cerca de los robots más allá de las leyes presentes para otras creaciones (Ley de Orgánica de Protección de Datos, Ley de la Propiedad Intelectual...), pero a nivel europeo se están empezando a desarrollar una serie de normativas en lo que se refiere a diferentes aspectos relacionados con la robótica las cuales, si en un futuro son transformadas en leyes, pueden afectar al desarrollo de este tipo de sistemas:

- Proteger a todo ser humano de los daños causados por un robot.

- Respetar el derecho de una persona a negarse a ser atendido por un robot (relacionado sore todo con el cuidado de personas mayores o la robótica aplicada al sector sanitario).
- Hacer que los robots respeten la autonomía de las decisiones tomadas por seres humanos.
- Protección de datos con respecto a los robots: es decir, hacer que estos sistemas respeten la privacidad del ser humano, o lo que es lo mismo, asegurar el derecho a la intimidad.
- Permitir al un ser humano gestionar los datos recopilados por el robot sobre si mismo para, como la anterior normativa, asegurar la protección de datos personales.
- Evitar la creación de vínculos emocionales con robots: sobre todo referido a las personas mayores, niños o personas con discapacidad. Esta ley pretende evitar el aislamiento de un ser humano del resto de la sociedad, es decir, evitar la sustitución de seres humanos por robots.
- Igualdad de acceso a las mejoras en la robótica tanto para los usuarios profesionales como los que no lo son.
- Restringir al ser humano mejoras en la robótica que permita la creación de los denominados cyborgs, lo que se augura que podría causar la extinción de la raza humana.
- Necesidad de implantar un sistema de registro, responsabilidad y cobro de impuestos para los robots: ya que al realizar un trabajo que copia al de un ser humano, se necesitaría tener un registro equivalente.
- Implantación de un botón de emergencia para revertir determinadas decisiones tomadas por el robot o desconectarlo directamente. Esta normativa viene impulsada por el objetivo de preservar la seguridad del ser humano.

En resumen, las reglas que se están desarrollando con las nuevas tecnologías desarrolladas es la de orientarlas a la ayuda del ser humano evitando daños al mismo o perjuicios con respecto al resto de la sociedad.

Apéndice B

Especificación de Requisitos

B.1. Introducción

En este apartado se presentan los requisitos funcionales especificados para este proyecto.
Como su propio nombre indica los requisitos funcionales son las funcionalidades básicas que debe cumplir el sistema desarrollado para este proyecto. En este caso se pueden diferenciar dos tipos: los objetivos generales y el resto de requisitos que se mostrarán en un catálogo debido a que su cumplimiento no es estrictamente obligatorio, es decir, que no es necesario que se cumplan todos.

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