Universidad de Oviedo


# ESCUELA POLITÉCNICA DE INGENIERÍA DE GIJÓN

## MÁSTER UNIVERSITARIO EN INGENIERÍA INFORMÁTICA

### ÁREA DE PROYECTOS DE INGENIERÍA

### TRABAJO FIN DE MÁSTER

# HERRAMIENTA EN LÍNEA PARA FOMENTAR EL ENVEJECIMIENTO ACTIVO Y SU GESTIÓN A TRAVÉS DE UNA EMPRESA BASADA EN SOFTWARE LIBRE

D. Guillermo Álvarez Martínez

Tutor: D. Vicente Rodríguez Montequín

Cotutor: D. José Antonio Labra Pérez

Fecha: 15 de junio de 2022.

Universidad de Oviedo


# DECLARACIÓN DE ORIGINALIDAD

D. ...**GUILLERMO ÁLVAREZ MARTÍNEZ**... con DNI. ..... como alumno/a del Máster Universitario en Ingeniería Informática de la Universidad de Oviedo, DECLARO que el Trabajo Fin de Máster titulado ……...**HERRAMIENTA EN LÍNEA PARA FOMENTAR EL ENVEJECIMIENTO ACTIVO Y SU GESTIÓN A TRAVÉS DE UNA EMPRESA BASADA EN SOFTWARE LIBRE**……. es de mi autoría, es original y las fuentes bibliográficas utilizadas han sido debidamente citadas.

En GIJÓN, a 15 de junio de 2022.

Firmado:……..

**Dedicatoria**

*A mis familiares, amigos, socios, colaboradores, profesionales y usuarios vinculados con la Mancomunidad "Comarca de la Sidra" que me acompañaron durante el camino que marca un punto y seguido en mi etapa formativa, esta vez asociada al Máster Universitario.*

**Agradecimientos**

*Gracias a mi tutor, y especialmente gracias a José Antonio por haber confiado en mí a nivel personal y posteriormente también a nivel empresarial para retomar esta maravillosa idea de ayudar a las personas mayores desde un punto tecnológico y humano diferente.*

*Gracias a todos los profesionales y participantes que han estado trabajando codo con codo para validar la herramienta dentro de la plataforma de RETEMANCOSI.*

*Gracias a Pablo y a Sofía por haber participado en la fundación de ESTUDIOS PAWI SL y todo lo que eso conlleva.*

*Gracias a la comunidad de software libre que, de manera desinteresada y relativamente anónima, comparte sus conocimientos y esfuerzos en beneficio de toda la sociedad.*

*Gracias a toda la gente que, de una u otra forma, me han aportado su granito de arena en esta etapa.*

**Resumen**

Las herramientas de teleasistencia ofrecen un valioso servicio a todas las personas que las usan, especialmente en casos de soledad o durante la vejez. Son una solución fantástica que permite poner en contacto a miembros de diversos colectivos vulnerables y mantenerlos activos.

Desde hace dos años los habitantes de la Mancomunidad "Comarca de la Sidra" cuentan con una renovada red de teleasistencia. Esa renovación se realizó, primero, como parte del Programa de Prácticas con la Universidad de Oviedo en el curso 2020/2021, y derivando posteriormente en la fundación de ESTUDIOS PAWI SL, empresa encargada de su gestión. Una de las principales actividades que realiza esta empresa es el desarrollo de software basado en software libre.

Actualmente, la red de teleasistencia de la Mancomunidad "Comarca de la Sidra" promueve el envejecimiento activo a través de talleres en línea. Gracias a los esfuerzos realizados durante su gestación y construcción, se consiguieron minimizar o eliminar multitud de barreras técnicas y generacionales que habitualmente rodean a este tipo de implantaciones. Tras la buena acogida que ha tenido la herramienta, tanto por los usuarios como por los profesionales que organizan en ellas diferentes talleres, se han puesto sobre la mesa diferentes propuestas de mejora con la que ampliar la funcionalidad de la herramienta. Entre todas esas ideas, se solicitó a la empresa adjudicataria por parte del organismo público encargado de su explotación la incorporación en la herramienta de un módulo adicional que permitiera crear un nuevo tipo de actividades con las que se pudiera incluir más interactividad a los talleres que se imparten actualmente. Sabiendo que dotar de esta funcionalidad a una herramienta de este estilo supondría una clara oportunidad para mantener el carácter innovador que busca la Mancomunidad, se ejecuta por parte de ESTUDIOS PAWI SL el proyecto solicitado durante los primeros meses del año 2022.

Este documento describe, a modo de Memoria, los pasos dados para satisfacer la demanda solicitada por los clientes de la herramienta desde su gestación hasta su uso en un entorno real.

**Palabras clave**

Teleasistencia, programación genérica, lenguajes, desarrollo web, html5, javascript.

## Abstract

**Online tool to promote active aging and its management through a free-software company.**

Telecare tools offer a valuable service to everyone who uses them, especially in cases of loneliness or in old age. They are a fantastic solution for connecting members of various vulnerable groups keeping them active.

The inhabitants of the "Comarca de la Sidra" community have had a renewal telecare network for the past two years. This renovation was carried out, firstly, as part of the Internship Program with the University of Oviedo in the 2020/2021 academic year and, subsequently, from then on, through a contract with the company ESTUDIOS PAWI SL. One of the main activities this software company carried out is the development based on free software.

Currently, the telecare network of the Mancomunidad "Comarca de la Sidra" promotes active aging through online workshops. Due to the efforts made during its development and construction, it was possible to minimize or eliminate many technical and generational barriers usually appears on this situations. After a positive reception of the solution by both users and professionals who organize different workshops in the tool, several improvement proposals to expand the functionality of the tool came up. Among all those ideas, the public body in charge of its operation asked the contractor the incorporation of an additional module to create a new type of activities that would include more dynamism to the workshops currently being taught. Knowing that providing this functionality to a tool of this style would be a clear opportunity to maintain the innovative character that the Mancomunidad is looking for, ESTUDIOS PAWI SL executed the requested project during the first months of the year 2022.

This document describes all steps taken to satisfy this client request from the creation of the tool to its use in a real environment.



**Keywords**

Telecare, generic programming, languages, web development, html5, javascript.

## Prefacio

En este documento se plasma la Memoria de mi **Trabajo Fin de Máster** correspondiente al **Máster Universitario en Ingeniería Informática** que cursé en la Escuela Politécnica de Ingeniería de Gijón (EPI), Universidad de Oviedo. El proyecto ejecutado lo llevé a cabo durante los últimos meses del curso **2021/2022** como socio fundador de ESTUDIOS PAWI SL. Este trabajo es una continuación de mis Prácticas de Empresa asociadas al propio Máster Universitario, donde realicé, por entonces como estudiante en prácticas, el desarrollo original de la herramienta de teleasistencia sobre el cual se sustenta la funcionalidad incorporada ahora. Dejo aquí[1] un enlace a la Memoria de dichas prácticas a modo de complemento previo que permita contextualizar con más detalle el punto de partida de este documento.

Toda la información contenida en este trabajo es original, elaborada a partir de apuntes, notas, informes y conversaciones mantenidas con todas las partes interesadas durante el desarrollo del trabajo. Para el maquetado final se ha utilizado principalmente la suite LibreOffice[2] en su versión 7.3; concretamente se usaron los módulos de Writer para la redacción final del documento junto con el módulo de Draw para la creación de los diferentes diagramas y esquemas incluidos en el mismo.

El formato y la estructura para los capítulos y demás secciones que conforman esta memoria siguen las recomendaciones académicas generales; en particular, las directrices para la redacción de Trabajos Fin de Máster en la EPI[3] y las normas APA[4]. Las referencias externas, mostradas como una secuencia de números en superíndice entre corchetes, se ubican en las últimas páginas del documento acompañadas de un hiperenlace a la correspondiente página web para su consulta.

Hasta la obtención de la actual versión definitiva, se han generado seis versiones parciales con el objetivo de mantener un histórico donde ver la evolución que ha tenido la documentación a medida que el trabajo se completaba. También utilicé esas versiones intermedias a la hora de solicitar su revisión tanto al tutor como al cotutor del TFM. Todas esas versiones están disponibles junto con ésta en el repositorio del proyecto[5], enumeradas de manera incremental.

**Índices de contenidos**

## SUMARIO

## LISTA DE FIGURAS

## LISTA DE TABLAS

# Capítulo 1:

# Introducción

## 1.1 – Contexto

RETEMANCOSI[6] (**Re**d de **Te**leasistencia de la **Man**comunidad **Co**marca de la **Si**dra) es un proyecto iniciado en el año 2005 con el objeto de probar y analizar diferentes soluciones tecnológicas con las que ampliar y mejorar la atención a las personas mayores del medio rural asturiano. Durante el curso 2020-2021, gracias a la colaboración de la **Consejería de Servicios y Derechos Sociales** del Principado de Asturias a través del programa "Rompiendo Distancias" y de la **Fundación La Caixa**, y en el marco que define el Convenio de Colaboración que tiene la **Mancomunidad "Comarca de la Sidra"** con la **Universidad de Oviedo** y que habilita el desarrollo de Prácticas de Empresa dentro de su Departamento de Informática entre otros, se consiguió relanzar, optimizar y evolucionar el sistema de videoconferencia que a día de hoy ya se utiliza para realizar distintas actuaciones de envejecimiento activo y saludable a través de Internet semanalmente. Este desarrollo software se publicó bajo una **licencia libre** que habilita la implantación y explotación del sistema por cualquier persona u organismo.

El objetivo establecido para ese desarrollo solicitado completamente renovado fue dotar a los usuarios de diferentes municipios asturianos de una herramienta de comunicación sencilla con la que, de manera extremadamente simplificada, pudieran ponerse en contacto con familiares y/o profesionales que les reunirían semanalmente o bajo demanda con el objetivo de mantener el contacto y estimularles a través de talleres donde se realizarían diferentes entrenamientos cognitivos y demás actividades psicosociales compatibles con el uso de las nuevas tecnologías. Estos talleres los realizaron psicólogos, personal de centros asistenciales y diversos estudiantes de la Facultad de Psicología de la Universidad de Oviedo y de Cantabria. Para ello, a los usuarios se les ofrece una tablet digital debidamente configurada para conectarse a Internet e iniciar automáticamente la aplicación de RETEMANCOSI. A través de este

dispositivo, iniciado de manera totalmente autónoma, los usuarios pueden comenzar a realizar diferentes actividades, elaboradas por los profesionales, a las que se encuentren suscritos.

Tras el relanzamiento a finales del año 2020, entre 20 y 30 usuarios se empezaron a dar cita en la herramienta semanalmente en grupos de 3 a 5 personas. Estos usuarios consiguieron realizar satisfactoriamente diversas actividades, guiados siempre por un profesional que planificaba las sesiones y ejercicios previamente. Desde enero de 2021 hasta mayo del 2022, las estadísticas muestran más de 500 sesiones y casi 600 horas de videoconferencias que se complementaron con alrededor de 900 actividades. Estas actividades fueron predominantemente en formato imagen, pero también se realizaron en menor medida otras basadas en audio o vídeo.

A raíz de la liberación del código asociado a la herramienta de teleasistencia utilizada en RETEMANCOSI durante ese nuevo desarrollo, en febrero del 2021 se funda ESTUDIOS PAWI SL[7], empresa dedicada a varias actividades en el marco de las nuevas tecnologías. Entre las distintas labores que se realizan en la empresa se encuentra el servicio de instalación, mantenimiento, soporte y mejora de la herramienta de teleasistencia que fue liberada por la Mancomunidad “Comarca de la Sidra”. Esta empresa actualmente es la única que, de manera oficial, ofrece ese servicio de manera integral bajo el nombre comercial de PAWI CONECTA. Por ello, tanto en el año 2021 como durante el año 2022, es la empresa encargada del mantenimiento y evolución del sistema de videoconferencia adaptado a personas mayores utilizado en RETEMANCOSI.


*Figura 1.1.* Logo de PAWI CONECTA.

Por otro lado, puesto que entre las diferentes partes que componen el servicio PAWI CONECTA se encuentra el desarrollo de nuevos casos de uso dentro de la herramienta de teleasistencia, también resulta posible que un cliente solicite a la empresa nuevas características a través de un contrato de prestación de servicios digitales. Por construcción, la herramienta de teleasistencia permite a nivel técnico incorporar evolutivos a través del concepto de "módulos". Los módulos son archivos que empaquetan código y recursos complementarios. Con su instalación en la herramienta, se incrementan o modifican las funcionalidades disponibles en la misma gracias a un mecanismo interno con el que la herramienta es capaz de unir un módulo dado al resto de características disponibles.

## 1.2 – Descripción del problema

Tras casi un año de utilización ininterrumpida de la herramienta de teleasistencia en RETEMANCOSI, es lógico que aparezcan opiniones y sugerencias de boca de todas las personas involucradas en la herramienta; especialmente, de sus usuarios, tanto los participantes como de los profesionales que monitorizan y guían los talleres. Como cualquier aplicación en constante evolución, algunas de estas opiniones se pueden materializar con diferentes ampliaciones de la funcionalidad. Debido a la buena previsión que se hizo a la hora de diseñar la herramienta de manera modular, la mayoría de evolutivos se pueden enmarcar como complementos dentro del sistema de módulos que soporta la herramienta.

Por ese motivo, y respondiendo a la principal demanda por parte de los profesionales canalizada a través de MANCOSI, el primero de los objetivos que se decidió resolver fue: **dotar a la herramienta de teleasistencia la capacidad de incorporar actividades genéricas que permitan diseñar, ejecutar y evaluar tareas avanzadas que realizarán los participantes de un taller a través de un módulo complementario que se pudiera desplegar en la plataforma de RETEMANCOSI**.

Estas tareas avanzadas se construirán a partir de un conjunto de elementos y operaciones por el personal encargado de los talleres. Se podrán mostrar diferentes recursos (imágenes, formas geométricas y texto) que podrán ser ubicados en pantalla

a través de un código que los monitores elaborarán e incorporarán en la herramienta. También los colaboradores podrán establecer en las actividades genéricas ciertas acciones y detectar la ocurrencia de eventos concretos. Las primeras, ejecutarán operaciones que podrán modificar total o parcialmente la actividad en curso, llegando incluso a finalizarla; los segundos, se podrán utilizar para desencadenar acciones (las definidas u otras predefinidas) a partir de algunas situaciones que se podrán dar durante el transcurso de una actividad; situaciones como la interacción persona-máquina o incluso hasta otros factores externos como por ejemplo el paso del tiempo.

El ámbito del problema en lo que respecta al **formato** de este código se restringe a la definición del mismo, priorizando la facilidad de ejecución por un programa de ordenador frente a otros factores como podrían ser la adaptación a personal con escasos conocimientos técnicos en el ámbito de la programación o incluso la legibilidad o la compatibilidad entre otras herramientas. Para paliar esta necesidad potencial que evidentemente terminará surgiendo más pronto que tarde, se podría implementar adicionalmente dentro de la herramienta un pequeño entorno de desarrollo que utilice la programación gráfica como característica facilitadora, o bien se estudien y establezcan herramientas externas que permitan exportar un código compatible con el que se definirá en este trabajo.

### 1.3 – Estudio de la situación actual

Como se explicaba anteriormente, la herramienta de teleasistencia que actualmente dispone la Mancomunidad es una sencilla solución tecnológica que permite desarrollar talleres básicos a través del módulo principal de videoconferencia, pudiéndose apoyar con el uso de actividades interactivas basadas en imágenes, fragmentos de audio, vídeos, animaciones, o secuencias de estas. Prácticamente cualquier aplicación web dentro de la categoría de “comunicaciones” ofrece de manera más o menos análoga estas posibilidades. Existen en el mercado soluciones generalistas como **Microsoft Teams**, **Zoom**, **Skype**, **Whereby** o incluso **Telegram**. Todas ellas disponen de las funcionalidades elementales para establecer una comunicación basada en audio y vídeo; algunas ofrecen características adicionales

que facilitan en mayor o menor medida esa comunicación o permiten compartir recursos adicionales. Es el caso de, por ejemplo, las presentaciones o las pizarras que incluyen muchas herramientas globales. En función de los usuarios de cada taller, la viabilidad al utilizar una aplicación generalista puede llegar a comprometerse si se requieren pasos para iniciar esa aplicación o incluso si se precisan ciertos sistemas que deban ser configurados o adquiridos por los usuarios. Las personas mayores o personas que no disponen de los conocimientos tecnológicos suelen tener problemas para utilizar este tipo de aplicaciones. Por supuesto, existe también un amplio abanico de aplicaciones enfocadas especialmente al sector de la teleasistencia, incluso de la telemedicina. Esta categoría de soluciones se compone, por lo general, de aplicaciones propias o internas utilizadas por diferentes entidades (tanto públicas[8] como privadas[9]) donde se ofrecen incluso sensores y actuadores específicos que interaccionan con la herramienta, facilitando la monitorización de la salud, ubicación o estado de cada usuario por parte de los profesionales o servicios de emergencia.


*Figura 1.2.* Soluciones más próximas a la telemedicina que incluyen multitud de dispositivos.

En otras ocasiones, algunas entidades recurren al uso de aplicaciones generalistas. Estas entidades logran darles un uso muy rebuscado puesto que no están realmente pensadas para ejecutar los ejercicios concretos que desean realizar. En lo referente a la necesidad de disponer de ciertos conocimientos previos por parte de los usuarios cuando una entidad opta por el uso de una aplicación generalista, estas entidades suelen incluir en su servicio también el despliegue, el soporte técnico y, sobretodo, la formación necesaria para facilitar el correcto aprovechamiento de la herramienta por parte de los usuarios. Eso en ocasiones es un verdadero reto cuando los usuarios son personas muy alejadas de las nuevas tecnologías.

Por resumirlo, las aplicaciones utilizadas en el ámbito de la teleasistencia a día de hoy se pueden agrupar, especialmente en función del público al que están dirigidas, de la siguiente forma:

- **Generalistas**: ofrecen el servicio a gran escala, pudiendo incorporar algunas características diferenciadoras. Su soporte es global. Algunas permiten extensiones que amplían la funcionalidad, pero esa personalización tiene un límite que, en ocasiones, no permite ni tan siquiera cambios relacionados con la accesibilidad. Por lo tanto, pueden dejar de lado algunas particularidades importantes para ciertos usuarios.
- **Específicas**: dan servicios concretos a un conjunto de usuarios determinados. Se suelen complementar con dispositivos que son capaces de conectarse entre ellos para trabajar conjuntamente. Logran una experiencia de usuario más fiel y profunda. Tienen un soporte más concreto y pueden ofrecer adaptaciones para situaciones aún más particulares.

A la vista de esta clasificación, aunque se puede constatar que existen diferencias importantes entre ambos grupos, se pone en evidencia una característica común en ambos grupos: **la flexibilidad a la hora de establecer personalizaciones** es un factor que abre las puertas a ampliar sustancialmente las distintas posibilidades de uso de una aplicación para la comunicación entre personas. Es por ello que se puede concluir que ofrecer más personalización a los usuarios de este tipo de plataformas concretas mejoraría de manera significativa la experiencia de uso y las posibilidades de

explotación para segmentos específicos de la población. En concreto para las herramientas de teleasistencia, parece interesante buscar mecanismos con los cuales se facilite la incorporación de características genéricas en las actividades que se puedan realizar; como por ejemplo, la posibilidad de distribuir el contenido por pantalla (tamaño, colores, posición, etc.) en función de condiciones que se pueden establecer a modo de programa informático, así como posibilitar diferentes flujos de ejecución a partir de ciertas condiciones, y desde luego, un largo etcétera.

En esta misma línea, desde hace décadas, existe entre los paradigmas de programación lo que se denomina "**programación visual**"[10]. Esta forma de crear software permite a usuarios definir flujos de ejecución a partir de construcciones que incluyen fundamentalmente elementos gráficos en lugar de utilizar exclusivamente texto como ocurre en la programación tradicional basada en código fuente escrito. Este paradigma es una idea a tener en cuenta a la hora de incorporar a las soluciones informáticas esas cualidades previamente mencionadas ya que permiten acercar a los usuarios sin conocimientos profundos en nuevas tecnologías al mundo del desarrollo de software. De entre los numerosos entornos disponibles donde se implementa la programación visual o gráfica, se pueden encontrar soluciones orientadas a:

- Escuelas o universidades, ya que son una herramienta fantástica para introducir los conceptos de la programación a estudiantes con cierto interés por la tecnología. Existen publicaciones[11] que buscan la formación básica en programación con el fin de preparar a los jóvenes desde una edad temprana para un futuro donde esas habilidades serán cada vez más demandadas.
- Entornos específicos para escenarios concretos. Por ejemplo SCRATCH[12], para programación de video juegos, con módulos para programación de drones; o VISUALINO[13], para programar sobre Arduino[14].
- Entornos realmente profesionales. Es en este último grupo donde se podría ubicar perfectamente BLOCKLY[15]: se trata de un entorno de programación visual desarrollado por Google. Una de las características más interesantes que provee esta solución es la capacidad de convertir un diseño de bloques a código nativo de diferentes lenguajes, entre ellos, JavaScript.

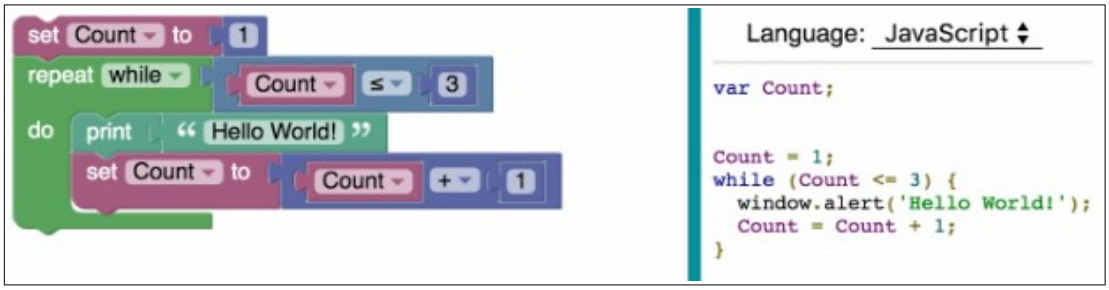


*Figura 1.3.* Captura de pantalla de la aplicación BLOCKLY.

Como todos los entornos de desarrollo, una vez generado el diagrama de bloques, el código fuente asociado ya traducido en un lenguaje concreto se puede exportar para ser introducido en los sistemas que lo vayan a procesar. Si el lenguaje es interpretado, se podrá evaluar directamente por el intérprete; si el lenguaje es compilado, deberá ser procesado e incluido en el programa principal.

Como punto adicional, algunos de estos entornos de programación visual pueden empotrarse dentro de las propias soluciones, pueden incluirse parcialmente incorporando solo un conjunto reducido de operaciones, o pueden llegar incluso a no ser incluidos, delegando la tarea de la programación de esos fragmentos de código a herramientas externas como las mencionadas antes o similares. Cuando se da este último caso, para conseguir toda la interoperatividad requerida, las soluciones que no implementan un entorno de programación internamente, suelen permitir la **importación del código** realizado externamente a través de interfaces que van desde un sencillo cuadro de texto, pasando por diálogos para cargar ficheros de código fuente, hasta entornos de integración conectados con repositorios de código.

Complementariamente, desde el punto de vista interno de los entornos de programación, la propia estructura de bloques se guarda con un formato concreto que no se parece en algunos casos a ningún código de programación funcional. Por lo general, se utilizan lenguajes de marcado (XML o JSON predominantemente) donde se guardan el conjunto de elementos que componen el escenario construido junto con los atributos que parametrizan cada uno de esos elementos, relaciones entre ellos, disposición gráfica de los mismos, y anotaciones o comentarios. Actualmente **no existe ningún estándar o metalenguaje** que establezca un formato único para

representar un programa genérico elaborado desde un entorno de programación visual; por consiguiente, no existen tampoco librerías que lo implementen.

Por último, el análisis del estado actual de la herramienta RETEMANCOSI vislumbra dos características a destacar. Por una parte, la posibilidad de ampliar la funcionalidad de la herramienta gracias a su diseño modular; por otra, la experiencia previa muy satisfactoria haciendo uso de actividades particulares orientadas al trabajo de corte psicosocial. Estas actividades resultan especialmente interesantes dado el nicho de mercado donde se explota la herramienta dentro de la Mancomunidad "Comarca de la Sidra": la mayoría de usuarios son personas mayores que viven en zonas rurales y en ocasiones muy aisladas del resto del mundo.

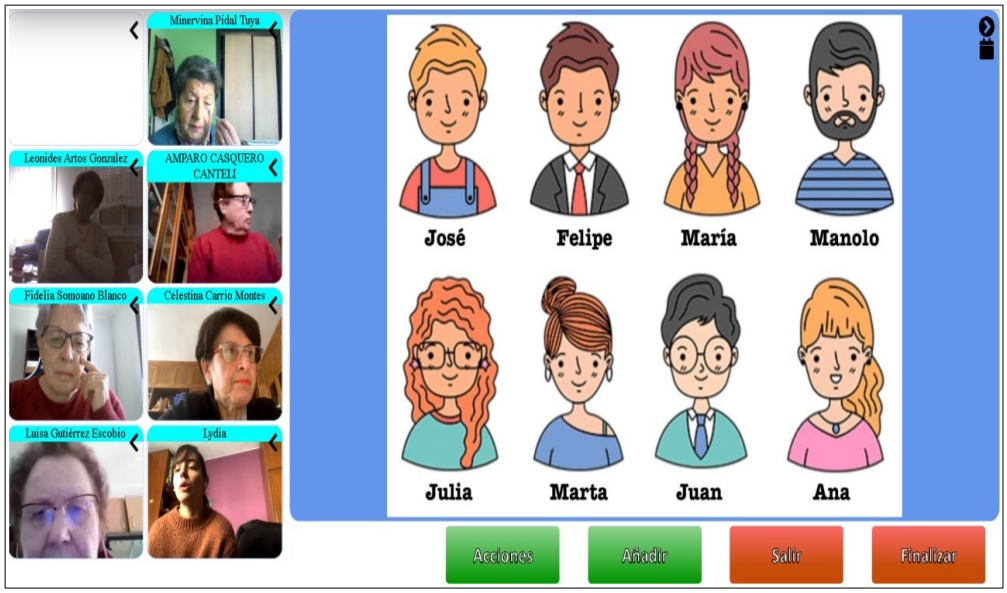


*Figura 1.4.* Actividad "caras" durante un taller.

En lo relativo a la operativa con la herramienta, como ya se comentó previamente, se permite a los profesionales crear actividades básicas y establecer talleres con secuencias fijas compuestas por parte de las actividades previamente creadas. Internamente las actividades se categorizan. Además, se les da un identificador único. Una vez creadas las actividades de manera independiente o en conjunto a través de las secuencias de actividades, los profesionales pueden iniciarlas dentro de los talleres en curso para que los usuarios puedan ejecutarlas bajo la supervisión del monitor.

Además de las actividades de corte general basadas en recursos sencillos como son imágenes, vídeos o sonidos, también existen en RETEMANCOSI un subconjunto de actividades particulares. Algunas de estas actividades son complementos para los talleres que aportan ciertas funcionalidades en el plano digital. Por ejemplo, los profesionales pueden incluir un en un tallar un "dado". Esta actividad permitirá mostrar un valor de entre un conjunto de ellos que podrán escoger los profesionales durante la planificación del taller. La actividad se acompaña de una animación y sonido al tirar el dado.

Otras actividades complementarias buscan evaluar el nivel cognitivo de los participantes a través de diferentes tareas: *EMAV*, *D2*, *5 dígitos*, *BADS*, por ejemplo. Estas pruebas fueron adaptadas al entorno digital por ESTUDIOS PAWI utilizando código nativo JavaScript e incluidas a través de un módulo de "entrenamientos cognitivos" que se desarrolló a mediados del año 2021.

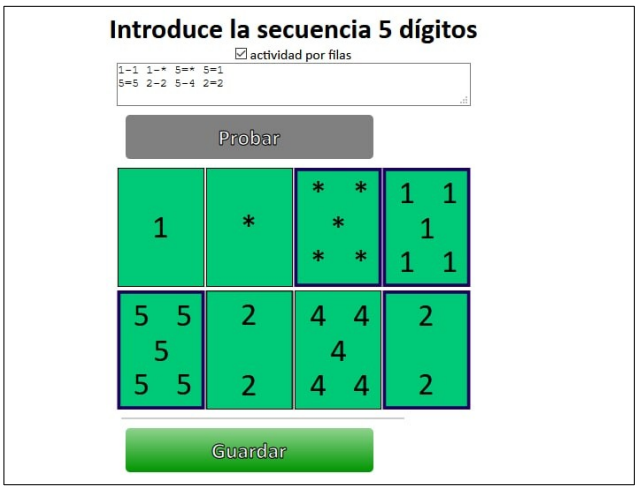


*Figura 1.5.* Resultado al configurar una actividad "5 dígitos".

Este nuevo conjunto de actividades particulares fue utilizado como parte del Trabajo Fin de Grado "Propuesta de intervención psicomotriz a través de medios telemáticos para personas mayores del medio rural" del Departamento de Psicología de la Universidad de Oviedo en el curso 2020/2021. En este módulo de entrenamientos cognitivos se permitía además la parametrización parcial de cada ejercicio (número de elementos, colores, distribuciones, …) a través de un formulario

vinculado con la actividad que los monitores podían editar desde de la ficha de datos de la correspondiente actividad. También el profesional podía analizar los valores relativos a los tiempos de respuesta y nivel de acierto de los participantes en los informes que se elaboraban al finalizar la actividad para cada taller.

Los módulos adicionales implementados para la Mancomunidad "Comarca de la Sidra" fueron desarrollados igualmente bajo una licencia de software libre. Esta particularidad permite a otras entidades utilizar las nuevas funcionalidades que incluyen los módulos sin ningún coste adicional. Esta característica posibilitaría la realización de estos ejercicios en otras regiones y con otros usuarios. Los resultados de las diferentes intervenciones se podrían comparar y compartir entre profesionales para detectar diferentes comportamientos entre las poblaciones estudiadas.

## 1.4 – Metodología de trabajo

En el plan docente actual[16] la carga total de trabajo para un TFM se estima en **450 horas** (18 ECTS). Por ese motivo, se han establecido tanto el alcance como los objetivos con la intención de que la planificación se aproxime a ese valor. Es evidente que antes del inicio de este proyecto ya se han trabajado en múltiples áreas del proyecto RETEMANCOSI, tanto como alumno en prácticas como desarrollador en ESTUDIOS PAWI SL. Es por ello muy necesario resaltar que toda la labor de formación e investigación previa que permite conocer y optar por ciertas decisiones tomadas en este trabajo está excluida del cómputo de tiempo de dedicación al trabajo.

La metodología elegida para este proyecto ha seguido un ciclo de vida tradicional. Para cada fase del proyecto (formalización, análisis, diseño, implementación, pruebas, y despliegue) se establecen hitos que se deben cumplir para dar por finalizada la misma. A nivel operativo, la finalización de cada fase se acompaña con un informe o acta que recoge estas conclusiones, aprendizajes y mejoras de cara al futuro a modo de retrospectiva. Al cliente se le entrega una versión final validada acorde con los requisitos establecidos finalizando la ejecución del proyecto. Ese momento da pie al inicio del periodo de mantenimiento donde se ofrecerá un servicio de soporte técnico en los términos que se establezcan en un contrato complementario.

# Capítulo 2:

# Formalización

*Al tratarse de un proyecto de carácter oficial ante un organismo público, se deberá formalizar la prestación del servicio requerido a través de una oferta que incluirá todas las actuaciones a realizar en el marco del objetivo planteado.*

## 2.1 – Prediagnóstico

A partir de la solicitud realizada por la Mancomunidad "Comarca de la Sidra" a la empresa ESTUDIOS PAWI SL en enero del 2022 donde se solicitó la realización de un módulo para la herramienta de teleasistencia que incorporaría la posibilidad de crear, editar, ejecutar y revisar actividades genéricas, se inició el proceso de análisis previo con el fin de poder dar una respuesta firme a dicha solicitud. Para ello, se establecieron las diferentes fases por las que pasaría el desarrollo de la solución y se definieron los objetivos asociados a cada una de ellas. Se optó por un desarrollo clásico compuesto por las fases de ANÁLISIS, DISEÑO, IMPLEMENTACIÓN, PRUEBAS, y DESPLIEGUE. Para cada fase se fijaron los siguientes objetivos a alto nivel:

- **ANÁLISIS**: acordar los requisitos específicos que se deberán cumplir.
- **DISEÑO**: definir las diferentes partes que componen la solución y las relaciones que tendrán con el resto módulos disponibles.
- **IMPLEMENTACIÓN**: codificar las diferentes funciones y estructuras que componen la solución a desarrollar.
- **PRUEBAS**: ejecutar el plan de pruebas acordado y verificar el cumplimiento de todos los requisitos que se han fijado.
- **DESPLIEGUE**: instalar la solución en el sistema informático de la Mancomunidad "Comarca de la Sidra" y dar formación a los monitores.

Tras la finalización del desarrollo comenzará el mantenimiento en los términos acordados con el cliente. El objetivo de esta labor posterior es dar soporte adicional y garantía tras la entrega del módulo respetando los acuerdos a nivel de servicio. Esta parte queda excluida de la Memoria por quedar fuera del ámbito del proyecto.

## 2.2 – Elaboración del presupuesto

De manera complementaria al análisis previo de la parte técnica, se realizó un presupuesto que marcaría el precio que tendría que abonar el cliente por el desarrollo del módulo. Se usaron las tarifas vigentes en la empresa junto con los precios de referencia de anteriores trabajos análogos, con su correspondiente adaptación a la situación actual, y se añadió también un margen para imprevistos. Se incluyeron también los impuestos que se aplicarían en cumplimiento de la legalidad vigente. Al tratarse de una **aplicación libre**, el cliente no incurre en gastos por licencias de uso.

En el apartado de desarrollo se incluye la creación del módulo. Se añade también la sesión formativa y un apartado de documentación que incluyen tres manuales:

- **Instalación**: instrucciones para incluir el módulo en la herramienta original.
- **Usuario**: explicación sobre cómo usar las nuevas funcionalidades.
- **Desarrollador**: claves para entender lo que hace internamente el módulo.

| Apartado | Concepto | | Precio |
|---|---|---|---|
| Desarrollo | Módulo de aplicaciones genéricas | | 1.200,00 euros |
| Formación | Sesión formativa con ejemplos (5 horas) | | 150,00 euros |
| Documentación | Manual de Instalación | | 100,00 euros |
| | Manual de Usuario | | 150,00 euros |
| | Manual de Desarrollador | | 250,00 euros |
| Licencias | Derechos de uso de PAWI CONECTA | | Gratuito |
| **Subtotal** | | | 1.850,00 euros |
| **Margen para imprevistos** | | 10 % subtotal | 185,00 euros |
| **Base imponible** | | | 2.035,00 euros |
| **I.V.A.** | | 21% base imponible | 427,35 euros |
| **Total** | | | **2.462,35 euros** |

*Tabla 2.1.* Presupuesto.

### 2.3 – Envío de la propuesta

Una vez definidas las diferentes acciones a realizar y el presupuesto asociado a esas tareas, se envió la propuesta a la Mancomunidad "Comarca de la Sidra". El envío constaba por una parte de la Oferta Técnica y por otra parte del Presupuesto asociado al trabajo. Ambos documentos, firmados digitalmente, se entregaron a la dirección de correo electrónico de la Mancomunidad. Se solicitó acuse de recibo y fue dado el mismo día de la presentación de la oferta por la Secretaría de MANCOSI.

### 2.4 – Aceptación e inicio del trabajo

Tras las oportunas gestiones que tuvo que realizar internamente la Mancomunidad "Comarca de la Sidra", finalmente se aceptó la oferta presentada. Se firmó el pertinente contrato de presentación de servicios de desarrollo vinculándolo con la oferta que se envió al cliente. De esa forma, los trabajos acordados podrían iniciarse de inmediato. Debido a problemas de agenda, el desarrollo real del proyecto comenzó, un par de semanas después de la aceptación de la oferta, con la primera entrevista dentro de la fase de ANÁLISIS.

# Capítulo 3:

# Análisis

*Acordar los requisitos que se deberán cumplir.*

## 3.1 – Entrevistas para definir nuevos requisitos del sistema

Puesto que la propuesta parte de la propia Mancomunidad, se fijan dos entrevistas para concretar los detalles sobre el desarrollo con el responsable del servicio y dos profesionales que tendrán el rol de monitor en la herramienta. En estas entrevistas se persigue conocer las necesidades en relación a las funcionalidades que se deben incorporar o modificar o eliminar respeto a la versión actual de RETEMANCOSI.

En la primera entrevista se buscó específicamente conocer con más detalle la operativa que se querría incorporar: ideas para esos nuevos tipos de actividades, ejemplos de otras soluciones similares en las que basarse, ejercicios a digitalizar que tradicionalmente se realizan en formato analógico, y demás funcionalidades y/o recursos que deberían incorporarse en la herramienta. Toda esta información es anotada y sirve para elaborar la respuesta que se expondrá en la siguiente reunión.

De cara a esa segunda entrevista, con la respuesta elaborada, se plantea una operativa básica que se va concretando con todas las partes interesadas. Ya que por parte de los solicitantes de esta funcionalidad hubo, desde el primer momento, una clara predisposición a abordar el reto de la creación de las nuevas actividades desde un punto de vista mucho más cercano a la programación[17] (entendida como **la tarea cuyo resultado permite controlar el flujo de ejecución que realizará un sistema informático, particularmente con el control de los eventos y acciones que estén involucradas en el desarrollo de una actividad, a alto nivel**), el planteamiento giró alrededor de cómo habría que presentar a los monitores la interfaz que les permitiría crear esas nuevas actividades genéricas, así como el conjunto de acciones y eventos disponibles en ellas, y finalmente la sintaxis y nomenclatura utilizada en el proceso.

Tras sendas entrevistas, se alcanzó un acuerdo materializado en las listas de requisitos que se describirán a continuación.

### 3.2 – Requisitos funcionales

| | |
|---|---|
| R-0.0 | Se podrán **crear** nuevas actividades de tipo "*Actividad genérica*". |
| R-0.1 | Este nuevo tipo de actividad aparecerá en el desplegable de la zona de creación de actividades junto con el resto de tipo de actividades disponibles. |
| R-0.2 | Al crear una actividad de tipo "*Actividad genérica*" se asignará un identificador único asociado a su nombre y tipo en el catálogo de actividades disponibles guardándose de manera análoga a como se hace con el resto de actividades. |
| R-0.3 | El nombre de una actividad genérica no podrá superar los 20 caracteres y solo podrá incluir caracteres alfanuméricos, espacios o guiones. En caso de no cumplirse este requisito, el usuario recibirá un mensaje de error describiéndolo. |
| R-0.4 | El identificador único de una actividad de tipo "*Actividad genérica*" se generará según las normas comunes con el resto de actividades. |
| R-0.5 | No se podrá modificar el identificador único, ni el nombre, ni el tipo de una actividad de tipo "*Actividad genérica*" una vez se haya creado. |
| R-1.0 | Una actividad genérica se podrá **editar** una vez creada a través de la ficha de datos asociada a la misma. |
| R-1.1 | Nada más crear una actividad genérica correctamente, se visualizará la ficha de datos de la misma. |
| R-1.2 | Se podrá editar una actividad de tipo "*Actividad genérica*" presente en el listado de actividades pulsado sobre ella. |
| R-2.0 | Dentro de la ficha de datos de una actividad de tipo "*Actividad genérica*" se mostrará el **identificador único** correspondiente a la misma y su **nombre**. |
| R-3.0 | En la ficha de datos también se presentará un campo de texto donde aparecerá el **código genérico** asociado a la actividad de tipo "*Actividad genérica*". |
| R-3.1 | El campo de texto con el código genérico asociado a una actividad de tipo "*Actividad genérica*" se podrá editar solamente como **texto plano**. |
| R-3.2 | El campo de texto con el código genérico se podrá seleccionar total o parcialmente con teclado o con el cursor. |
| R-3.3 | Se podrá **cortar** o **copiar**, total o parcialmente, el contenido en texto plano presente en el campo de texto con el código genérico. |
| R-3.4 | Se podrá **pegar** texto plano incorporando en cualquier punto del campo de texto con el código genérico o sustituyendo una sección parcial o total del mismo. |
| R-4.0 | Debajo del campo de texto con el código aparecerá un **botón** con el texto "Probar" y fondo sólido de color gris. |
| R-4.1 | Al pulsar el botón de "Probar" se intentará generar una actividad genérica con el código del campo de texto **que haya en ese momento**. |

| | |
|---|---|
| R-4.2 | En caso de que el código sea **correcto**, se mostrará la actividad asociada al código procesado en un área de pruebas que aparecerá justo debajo del botón "Probar". Ese área se considerá el lienzo de la aplicación genérica de similar manera al área que aparecerá durante un taller. |
| R-4.3 | El área de pruebas aparecerá siempre vacío pero **visible** al acceder a la ficha de datos de una actividad de tipo "*Actividad genérica*". |
| R-5.0 | Se considerará que un código genérico es correcto si cumple todos los requisitos sintácticos y semánticos. |
| R-5.1 | Un código genérico debe cumplir el formato **JSON** estándar. |
| R-5.2 | La estructura JSON debe incluir una propiedad llamada "elementos". |
| R-5.3 | La propiedad "elementos" debe ser un **array de elementos**. |
| R-6.0 | Cada elemento debe incluir los siguientes atributos: "**figura**", "**id**", "**x**" o "**xxx**", "**y**" o "**yyy**", "**ancho**" y "**alto**. |
| R-6.1 | Adicionalmente, cada elemento puede incluir **más atributos** que podrán ser usados o no, pero cumpliendo las directrices del formato JSON. |
| R-6.2 | El valor para cada atributo "id" debe ser **único**. |
| R-6.3 | Los valores posibles para el atributo "figura" son: "**imagen**", "**rectangulo**" (sin tilde), "**circulo**" (sin tilde) o "**texto**". |
| R-6.4 | Las figuras "imagen" cargarán la imagen asociada a la actividad de tipo "imagen" que tenga por identificador en el sistema el valor del atributo "**src**". |
| R-6.5 | Las figuras "**rectangulo**" y "**circulo**" se visualizarán como un rectángulo o círculo en el lienzo de la aplicación; se rellenarán del color indicado como una terna RGB[18] (0-255) separada por comas que aparezca en el valor del atributo "**rgb**"; tendrán análogamente como color de borde el valor del atributo "**rgb_borde**". Un valor inválido conllevará un fondo negro y un borde negro. |
| R-6.6 | Las figuras "texto" se visualizarán en el lienzo de la aplicación como un texto con una tipografía por defecto que corresponderá al valor del atributo "**texto**". |
| R-6.7 | Los elementos con atributo "x" se ubicarán en la **posición horizontal** relativa al lienzo de la aplicación genérica que corresponda con ese valor; de manera análoga, el atributo "y" determinará la **posición vertical** relativa al lienzo. Estos atributos prevalecen sobre los "xxx" o "yyy". |
| R-6.8 | En caso de aparecer el atributo "xxx", la posición horizontal se determinará **evaluando la función** que aparezca como valor del atributo; de manera análoga se procederá con el atributo "yyy" para la posición vertical. |
| R-6.9 | Los elementos con atributo "ancho" se dispondrán **horizontalmente** en pantalla ocupando el valor relativo al lienzo de la aplicación; de manera análoga se procederá con el atributo "alto", que corresponderá con la disposición **vertical**. |

| | |
|---|---|
| R-7.0 | En caso de que el código no sea correcto, se mostrará el **error** correspondiente en la zona de pruebas. |
| R-7.1 | Si el código no cumple con la **sintaxis** correcta (formato JSON), se indicará el error con el mensaje "Error: formato incorrecto". |
| R-7.2 | Si el código no cumple con la **semántica** definida (elementos incorrectos o inválidos), se indicará el error con el mensaje "Error: código inválido". |
| R-8.0 | El código asociado a una actividad de tipo "*Actividad genérica*" se podrá guardar desde su ficha de datos. |
| R-8.1 | En la parte inferior aparecerá un botón con el texto "Guardar" y fondo sólido de color verde. |
| R-8.2 | Como en el resto de actividades, al pulsar en el botón "Guardar" se guardará (dentro de la correspondiente **base de datos**, en el registro vinculado a la actividad que se está editando) el código presente en el campo de texto. |
| R-8.3 | El código se guardará **independientemente** de si es erróneo, válido o no. |
| R-9.0 | Se podrá salir de la ficha de datos **sin alterar** la actividad visualizada. |
| R-9.1 | En la parte inferior de la sección aparecerá un botón con el texto "Volver" y fondo sólido de color rojo. |
| R-9.2 | Al pulsar en el botón "Volver", se **regresará** al listado de actividades sin alterar la actividad que se estaba editando. |

*Tabla 3.2.* Requisitos funcionales.

### 3.3 – Requisitos tecnológicos y de infraestructura

| | |
|---|---|
| T-0.0 | El módulo necesitará una conexión a internet estable para realizar las tareas de carga, ejecución, edición, y guardado de las actividades genéricas. |
| T-0.1 | El módulo solo funcionará estando el servicio de teleasistencia disponible. |
| T-1.0 | El código generado deberá ser y producir **HTML5 estándar** (junto con JS/CSS), para ser utilizado en cualquier navegador web moderno. |
| T-1.1 | El código fuente del módulo debe cumplir con la guía de estilo anexa [10.2]. |
| T-2.0 | Se podrán almacenar ciertos datos en la caché del navegador local durante la edición de una actividad genérica. |
| T-2.1 | Se limitará el acceso a las variables de estado al ejecutar una actividad genérica. |
| T-3.0 | Cualquier usuario podrá ejecutar actividades genéricas. |
| T-3.1 | Cualquier usuario, debidamente autentificado y que tenga el rol de monitor, supervisor o administrador, podrá crear o editar actividades genéricas. |

*Tabla 3.3.* Requisitos tecnológicos y de infraestructura.

### 3.4 – Requisitos de usabilidad

| | |
|---|---|
| U-0.0 | El módulo se deberá poder usar tanto en ordenadores de sobremesa como dispositivos portátiles (con pantallas de más de 10 pulgadas y entrada táctil) con las mismas funcionalidades. |
| U-0.1 | Para una experiencia adecuada, es recomendable el uso de ratón y teclado para confeccionar (editar y probar) una actividad genérica. |
| U-0.2 | Se deberán usar diseños adaptables (*responsive design*). |
| U-0.3 | El código respetará las proporciones relativas a las dimensiones de cada dispositivo donde se ejecute favoreciendo la legibilidad. |
| U-1.0 | El código que ejecutará el navegador cumplirá con las normativas creadas por la W3C[19] relativas a los estándares del requisito [T-1.0]. |
| U-2.0 | Al definir la experiencia visual con las funcionalidades del módulo se deberá primar el rendimiento frente a la inclusión de animaciones complejas. |
| U-3.0 | Los textos mostrados en la aplicación podrían ser traducidos a otros idiomas. |
| U-3.1 | Por el momento, los textos solo se deben presentar en castellano. |
| U-4.0 | Todos los elementos visualizados en una actividad genérica podrán ser modificados a través de hojas de estilo (CSS) a través de su clase y/o id. |

*Tabla 3.4.* Requisitos de usabilidad.

### 3.5 – Demás requisitos necesarios

| | |
|---|---|
| N-0.0 | Los monitores son los **encargados** de diseñar las actividades genéricas. |
| N-0.1 | Los monitores podrán incluir en los talleres o secuencias las actividades genéricas; alternativamente, podrán iniciarlas habiéndolas creado previamente desde el menú de control de un taller al igual que el resto de actividades. |
| N-0.2 | Es **responsabilidad** del monitor comprobar el correcto funcionamiento de una actividad genérica antes de ejecutarla. |
| N-0.3 | Los monitores deberán contar con un **plan de contingencia** si una actividad genérica no funciona adecuadamente durante un taller. |
| N-1.0 | El código de una actividad genérica podría dejar de ser compatible en futuras versiones o llegar a tener un funcionamiento anómalo. |
| N-1.1 | Se recomienda añadir un atributo "**version**" (sin tilde) dentro de la estructura JSON con un valor acordado para mitigar este posible efecto adverso. |

*Tabla 3.5.* Demás requisitos necesarios.

### 3.6 – Requisitos delegados[20]

Por último, y sabiendo que se trata de una parte con una relativa importancia dentro de la fase de ANÁLISIS ya que persigue evitar malentendidos a la hora de verificar si se alcanzaron los objetivos definidos, se determinarán también aquellos puntos íntimamente relacionados con los requisitos a cumplir pero que están fuera del ámbito de actuación del proyecto. Estos requisitos excluidos pueden serlo, bien porque estén delegados en otros contratos o proveedores o componentes, o bien porque simplemente se han pospuesto para más adelante.

En este trabajo, como se ha comentado anteriormente, la parte de la **interfaz avanzada** donde potencialmente los monitores podrían elaborar gráficamente el código que finalmente utilizará el módulo de aplicación genéricas queda excluido del desarrollo actual: los profesionales dispondrán de documentación y ejemplos de código reales para adaptarlos a sus necesidades bajo demanda. Por consiguiente, el código elaborado por el profesional se podrá asociar a una actividad genérica exclusivamente a través de una interfaz web básica según lo descrito en los requisitos; la inclusión de entornos de desarrollo más elaborados u otras opciones para la construcción del código genérico no se contemplarán por el momento.

Para toda la parte de **persistencia** de las actividades de tipo "*Actividad genérica*" se aprovechará el componente que ya es utilizado por el resto de actividades. El módulo simplemente rellenará la estructura de datos en el momento de guardar una actividad con los campos que se requieran y el guardado en BBDD quedará delegado.

Por otra parte, aunque el módulo se presta a admitir teóricamente una infinidad de códigos, todas las posibles **construcciones** que no siguieran de manera clara las directrices recogidas en los términos de uso quedan excluidos del catálogo de nuevos servicios asociados al módulo y su uso se hará exclusivamente bajo responsabilidad de los usuarios del módulo. Por este motivo, cualquier uso válido pero diferente a los recogidos en la oferta no estará dentro del ámbito de actuación de este proyecto.

Finalmente, la parte relativa al **diseño gráfico** de los elementos visuales y su estilo para estar adaptado a las necesidades específicas de los usuarios y la adaptación a dispositivos particulares queda fuera del ámbito de actuación de estas tareas.

# Capítulo 4:

# Diseño

*Definir las diferentes partes que componen la solución y las relaciones que tendrán con el resto módulos disponibles.*

## 4.1 – Casos de uso

Con el objetivo de establecer con detalle las diferentes operaciones a través del nuevo módulo, se fijan un total de **siete** casos de uso. Uno de ellos está subdividido en dos subcasos de uso. Se toma como guía el catálogo de requisitos funcionales descritos en la fase anterior. De esta forma, los casos de uso mantendrán cierta vinculación con los requisitos, lo que facilitará mucho la validación de los mismos.

| **Caso** | **Descripción** | **Requisitos** |
|---|---|---|
| #0 | **Crear** actividades de tipo "*Actividad genérica*". | R-0.* |
| #1 | **Editar** una actividad de tipo "*Actividad genérica*". | R-1.* |
| #2 | **Alterar** una actividad de tipo "*Actividad genérica*". | R-2.*, R-3.1 |
| #3 | **Probar** una actividad de tipo "*Actividad genérica*". | R-4.*, R-5.*, R-6.*, R-7* |
| #3.1 | **Validar** una actividad de tipo "*Actividad genérica*". | |
| #3.2 | **Lanzar** una actividad de tipo "*Actividad genérica*". | |
| #4 | **Exportar** una actividad de tipo "*Actividad genérica*". | R-3.2, R-3.3 |
| #5 | **Importar** una actividad de tipo "*Actividad genérica*". | R-3.2, R-3.4 |
| #6 | **Actualizar** una actividad de tipo "*Actividad genérica*". | R-8.*, R-9.* |

*Tabla 4.6. Casos de uso.*

Cada uno de estos casos de uso corresponde con los nuevos pasos que un usuario con el rol de monitor podrá ejecutar en la herramienta. Todos los casos de uso requieren de unas condiciones previas para poder ejecutarse. Esas condiciones se definirán para cada uno de los casos de uso en concreto más adelante.

Antes de eso, se presenta un diagrama general donde se muestra las relaciones entre todos los casos de uso definidos. A la vista de este diagrama, se puede observar que es posible iniciar el flujo de ejecución asociado a este nuevo módulo creando una

actividad de tipo "*Actividad Genérica*" con la ejecución del caso "#0 Crear" o no. Con lo que sea, posteriormente habría que ejecutar el caso "#1 Editar" antes de realizar cualquiera de los casos "#2 Alterar", "#4 Exportar", "#5 Importar" o "#3 Probar" o nada. Posteriormente, se podrá ejecutar o no el caso "#6 Actualizar", finalizando el flujo principal proveído por el nuevo módulo desarrollado. Internamente, el caso "#3 Probar" se compone de dos subcasos: el "#3.1 Validar" y "#3.2 Lanzar" que, de irse a ejecutar este último, deberá hacerlo siempre tras el otro.

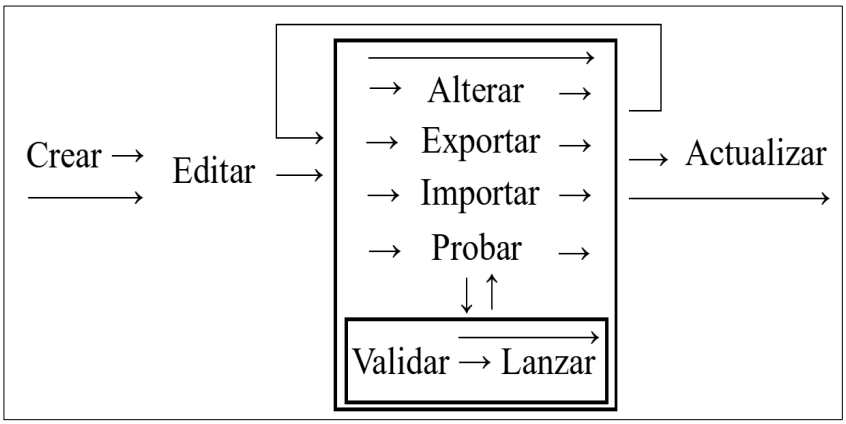


*Figura 4.6.* Relaciones entre casos de uso.

Estas relaciones constituyen un flujo de ejecución que se puede repetir por el usuario tantas veces como lo desee siempre y cuando se cumplan las condiciones establecidas para iniciarse.

Las operaciones opcionales dentro del flujo de ejecución se representan con una flecha larga horizontal. Por ejemplo, el primer caso de uso ("#0 Crear") es optativo porque para utilizar esta funcionalidad, o bien se empezará creando una actividad de tipo "*Actividad genérica*" a través de ese caso de uso, o bien se utilizará una actividad de este tipo previamente creada pasando directamente al caso de uso "#1 Editar". Análogamente, para finalizar el flujo de ejecución se podrá de manera opcional ejecutar el caso "#6 Actualizar" o no. Por otro lado, la flecha larga horizontal dentro del bloque central representa que un usuario puede ejecutar o no cualquiera de los casos incluidos en la caja. Finalmente, la flecha larga horizontal en los subcasos del caso de uso "#3 Probar" indica que, si bien se deberá iniciar el subcaso "#3.1 Validar", si este da un resultado erróneo, el subcaso "#3.2 Lanzar" no se ejecutará.

Las flechas verticales en ambas direcciones indican que un caso de uso se compone de dos o más subcasos. En el diagrama estas flechas aparecen solamente debajo del caso de uso "#3 Probar", puesto que es el único que contiene subcasos.

Para iniciar el flujo de ejecución asociado a este módulo, el usuario deberá estar dentro del módulo de gestión de actividades. Concretamente:

a) En la sección de creación de actividades, donde podrá ejecutar el caso "#0 Crear" una vez seleccionado el tipo de actividad "*Actividad genérica*".

b) O bien, en la sección que muestra el listado de actividades, donde podrá ejecutar el caso "#1 Editar" pulsando sobre una actividad (previamente creada) que corresponda con el tipo de actividad "*Actividad genérica*".

Para finalizar este flujo de ejecución, el usuario volverá al listado de actividades habiendo ejecutado o no el caso de uso "#6 Actualizar".

A continuación, se define por separado cada uno de los casos de uso:

#### 4.1.1 – Creando actividades genéricas

Para realizar cualquier otra acción relacionada con las actividades de tipo "*Actividades genéricas*" es necesario crear al menos una. En este caso de uso se crea una actividad genérica desde el *Gestor de Actividades* de la herramienta.

- **Precondiciones**: Estar en la sección de *Creación de Nuevas Actividades* dentro del *Gestor de Actividades* de la herramienta, indicar un nuevo nombre para la actividad y haber seleccionado el tipo de actividad "Actividad genérica" en el desplegable.
- **Poscondición**: Se creará la correspondiente actividad y se inicia inmediatamente el caso de uso "#1 Editar" para esa actividad.
- **Actores**: *Usuario* y *Servicio de Gestión de Actividades*.
- **Descripción**: Crea una nueva actividad de tipo "*Actividad genérica*" con el nombre indicado y un identificador único.
- **Excepciones**: si no se cumplen los requisitos asociados al nombre de la nueva actividad a crear, no se realiza ninguna operación y se regresa al punto de partida mostrando al usuario un mensaje de error.

#### 4.1.2 – Editando actividades genéricas

Una vez creada correctamente una nueva actividad de tipo "*Actividad genérica*" se accederá inmediatamente a su **ficha de datos**. Alternativamente se podrá acceder a la ficha de datos de una actividad de tipo "*Actividad genérica*" previamente creada seleccionándola del listado de actividades. La ficha de datos de una actividad que verá el usuario estará compuesta por el identificador único de la aplicación, su nombre y un campo de texto editable con el código fuente de la actividad en cuestión. Esos datos se cargarán a través del *Servicio de Gestión de Actividades* a partir del identificador de la actividad involucrada en el proceso.

- **Precondiciones**: Justo haber creado antes de manera correcta una actividad de tipo "*Actividad genérica*" o haber seleccionado una de ese tipo desde el listado de actividades disponibles en la herramienta.
- **Poscondición**: Se mostrará la ficha de datos de la actividad dada.
- **Actores**: *Usuario* y *Servicio de Gestión Actividades*.
- **Descripción**: Carga y visualiza la ficha de datos de la actividad dada.
- **Excepciones**: si no existe la actividad dada o no es posible cargarla, se mostrará un mensaje de error al usuario y se volverá al listado de actividades.

#### 4.1.3 – Alterando actividades genéricas

Cuando el usuario ha accedido a la ficha de datos de una actividad de tipo "*Actividad genérica*" concreta, podrá alterar el contenido de la misma. El único dato que se puede alterar en una actividad genérica es su código fuente asociado. Para ello, el usuario dispondrá de un campo de texto que podrá modificar a su antojo haciendo uso del teclado (físico o en pantalla) tras seleccionarlo (con ratón u pantalla táctil). Se resaltará el campo de texto para indicar que se está editando y un cursor parpadeando.

- **Precondiciones**: Haber cargado la ficha de datos de una actividad de tipo "*Actividad genérica*" a través del caso de uso "#1 Editar".
- **Poscondición**: El código fuente mostrado en la ficha de datos estará alterado.
- **Actores**: *Usuario*.
- **Descripción**: El usuario podrá editar el campo de texto.

#### 4.1.4 – Exportando actividades genéricas

Paralelamente, cuando el usuario ha accedido a la ficha de datos de una actividad de tipo "*Actividad genérica*" concreta, podrá exportar el contenido de la misma. El usuario podrá seleccionar total o parcialmente el campo de texto correspondiente al código fuente asociado a la actividad y lo podrá copiar o cortar para exportarlo como texto plano en cualquier otra aplicación. El contenido en texto plano puede guardarse en un documento para posteriormente poderlo importar. El código exportado no incluye ninguna meta-información adicional.

- **Precondiciones**: Haber cargado la ficha de datos de una actividad de tipo "*Actividad genérica*" a través del caso de uso "#1 Editar".
- **Poscondición**: El código fuente habrá sido exportado.
- **Actores**: *Usuario*.
- **Descripción**: El usuario podrá exportar el contenido del campo de texto correspondiente al código fuente asociado a la actividad dada.

#### 4.1.5 – Importando actividades genéricas

De forma análoga, cuando el usuario haya accedido a la ficha de datos de una actividad de tipo "*Actividad genérica*" concreta, podrá pegar total o parcialmente en el campo de texto correspondiente al código fuente asociado a la actividad código como texto plano. Para ello, seleccionará dentro del campo de texto, o bien un punto en una posición concreta, o bien un rango desde una posición concreta hasta otra, pudiendo ser por lo tanto parcial o completo. Si se ha seleccionado un punto concreto, se pegará el contenido a importar en ese punto; por el contrario, si se seleccionó un rango, el contenido seleccionado se sustituirá por el contenido importado.

- **Precondiciones**: Haber cargado la ficha de datos de una actividad de tipo "*Actividad genérica*" a través del caso de uso "#1 Editar".
- **Poscondición**: El código fuente habrá sido importado.
- **Actores**: *Usuario*.
- **Descripción**: El usuario podrá importar contenido en el campo de texto correspondiente al código fuente asociado a la actividad dada.

#### 4.1.6 – Probando actividades genéricas

La última de las acciones que se pueden realizar dentro de la ficha de datos de una actividad tipo "Actividad genérica" concreta es la de probar la propia actividad. Para ello, el usuario dispone de un botón "Probar" que, tras realizar las validaciones pertinentes, lanzará la actividad si todo es correcto. La aplicación genérica se mostrará en el área de pruebas ubicada debajo del botón "Probar". Se mantendrán en todo momento las proporciones (ancho y alto) originales de la ficha de datos y de la propia área de pruebas. Si esta operación se realizó previamente, la actividad genérica en curso se cancelará inmediatamente y posteriormente se iniciará una nueva actividad genérica sin que puedan causarse interferencias.

- **Precondiciones**: Haber cargado la ficha de datos de una actividad de tipo "*Actividad genérica*" a través del caso de uso "#1 Editar".
- **Poscondición**: La actividad genérica se ejecutará en el área de pruebas.
- **Actores**: *Usuario* y *Controlador de Actividades Genéricas*.
- **Descripción**: El usuario hará que la aplicación genérica se inicie en el área de pruebas si todas las validaciones previas lo permiten.
- **Excepciones**: se informará al usuario de los posibles errores que se pudieran detectar a la hora de validar el código fuente asociado a la actividad genérica dada impidiendo que se inicie.

Puesto que este caso de uso se compone de dos subcasos, se expondrán por separado cada uno de ellos a continuación:

##### 4.1.6.1 – Validando actividades genéricas

Dentro del caso de uso que permite probar una actividad genérica se encuentra la funcionalidad encargada de validar una actividad genérica.

- **Precondiciones**: Las mismas que en el caso padre.
- **Poscondición**: La actividad genérica a ejecutar se considera válida.
- **Actores**: *Controlador de Actividades Genéricas*.
- **Descripción**: El *Controlador de Actividades Genéricas* determinará si el código fuente asociado a la actividad genérica en curso es válido.

- **Excepciones**: si el código fuente no sigue el formato JSON, se emitirá el mensaje “Error: formato incorrecto”; si el código no cumple con la semántica definida, el mensaje que se emitirá será “Error: código inválido”. En ambos casos, se determinará que el código fuente no es válido.

#### 4.1.6.2 – Lanzando actividades genéricas

Finalmente, el *Controlador de Actividades Genéricas* se encargará de mostrar la actividad genérica correspondiente al código fuente asociado a la actividad dada. Para ello, procesará el código de forma que se muestren los elementos soportados en el área de pruebas. El usuario podrá entonces interactuar con la actividad genérica.

- **Precondiciones**: La poscondición del subcaso anterior.
- **Poscondición**: La misma que en el caso padre.
- **Actores**: *Controlador de Actividades Genéricas*.
- **Descripción**: El *Controlador de Actividades Genéricas* mostrará la actividad genérica en el área de pruebas y el usuario podrá interactuar con ella.

### 4.1.7 – Actualizando actividades genéricas

Finalmente, cuando el usuario haya accedido a la ficha de datos de una actividad de tipo “*Actividad genérica*” concreta, habiendo realizado o no alguna tarea, podrá volver a listado de actividades solicitando la actualización de los datos asociados a la misma. El usuario pulsará en el botón “Guardar” y los datos serán enviados al *Servicio de Gestión de Actividades* para que se actualicen en la plataforma.

- **Precondiciones**: Haber cargado la ficha de datos de una actividad de tipo “*Actividad genérica*” a través del caso de uso “#1 Editar”.
- **Poscondición**: Los datos asociados a la actividad dada serán actualizados y el usuario volverá al listado de actividades.
- **Actores**: *Usuario* y *Servicio de Gestión Actividades*.
- **Descripción**: El usuario solicitará la actualización de la actividad dada.
- **Excepciones**: si ocurre algún problema durante la actualización, el usuario será informado e igualmente se le llevará al listado de actividades.

## 4.2 – Arquitectura del sistema

La arquitectura requerida por el módulo de aplicaciones genéricas es la que actualmente dispone la herramienta de teleasistencia. **No se requiere ningún cambio o adaptación que** afecte a servidores o sistemas de comunicación o a los componentes hardware o software que sustentan la aplicación.

Esta arquitectura actual incluye:

- Un **servidor web** con los servicios asociados a los diferentes gestores que facilitan las funcionalidades de persistencia y comunicación entre pares, así como el servicio de registro remoto y página web.
- Líneas de comunicación sobre **Internet**.
- **Dispositivos** para los clientes finales (usuarios, monitores, supervisores, administradores) que pueden ser PC o tablets (de 10 o más pulgadas) o para ciertas operaciones también teléfonos inteligentes (de menos de 10 pulgadas).

A nivel interno, la arquitectura incluye los componentes propios de un esquema **Modelo-Vista-Controlador** (MVC) presente en el resto de partes de la herramienta.

## 4.3 – Clases para el MVC

El diseño de clases se hace encajar dentro del esquema MVC con el objetivo de facilitar la separación de funcionalidades de manera análoga a como se ha hecho con el resto de módulos de la herramienta. Este esquema se compone de tres grupos de funcionalidad que persigue la persistencia (modelo), la visualización (vista) y el procesado (controlador) de los datos que conforman una solución informática.

### 4.3.1 – Modelos

Los modelos corresponden con las clases destinadas a la gestión de la persistencia. Su cometido principal es cargar y guardar información almacenada en estructuras de datos. En la herramienta todos los modelos son objetos JSON que, gracias al *Gestor de Persistencia* preexistente, se pueden enviar y/o recuperar de la base de datos que los almacenará. Desde el punto de vista de este módulo, este gestor funciona como una caja negra[21].

A modo de resumen, este gestor ofrece en la capa de cliente (navegador web) una implementación basada en el estándar web **JSON-RPC**[22] que es capaz de transmitir a un servidor web una petición junto con el contenido íntegro de un objeto JSON optativamente y esperar la respuesta también en formato JSON (acompañada además de los propios códigos de estado HTTP) para indicar el resultado de la operación. En la capa de servidor (servidor **php** en el caso de la herramienta RETEMANCOSI) esa petición web se procesa de manera natural como una consulta POST de HTTP y el resultado se envía en formato JSON. Con este protocolo, se puede tanto enviar desde el cliente un objeto JSON al servidor, como cargar en el cliente un objeto JSON desde el servidor. De esa manera, se consigue la base con la que construir un sistema de persistencia. Sobre esa base, el *Gestor de Persistencia* de la herramienta implementa unos objetos JSON que se recubren con una cabecera compuesta por una localización y un identificador único y seguida por el área de datos correspondientes al objeto en cuestión que se está transmitiendo.

Siguiendo esta misma línea, para todas las aplicaciones diseñadas con la herramienta, la estructura de datos que utilizan se ajusta a este diseño basado en objetos JSON con una cabecera. Para no perder esa línea, las nuevas actividades de tipo "*Actividad genérica*" también mantendrán esa estructura: tanto la localización como el sistema para asignar el identificador único de cada objeto correspondiente a las nuevas actividades se mantiene respecto a como lo hacen las demás actividades.

La única particularidad que tienen las actividades de tipo "*Actividad genérica*" es la existencia del campo que incluye el código fuente asociado a cada actividad. Este campo se denomina "codigo" (sin tilde) dentro de la estructura definida para todos los objetos del modelo y su valor corresponderá con el código fuente asociado a la actividad en cuestión, debidamente escapado[23] según el formato JSON.

#### 4.3.2 – Vistas

Las vistas en el esquema MVC en el contexto de las aplicaciones interactivas están vinculadas a todo aquello que ve el usuario en pantalla. Son, por lo tanto, las clases encargadas de visualizar el contenido asociado a las diferentes funcionalidades.

En el desarrollo web basado en HTML, todas las vistas se deben organizar dentro del DOM[24]. Esta forma de organizar elementos gráficos se podría decir que corresponde con una estructura jerárquica, en forma de árbol, donde aparecen los diferentes elementos HTML (la mayoría de ellos, con un cometido visual, aunque los hay que excepcionalmente no) que el navegador web procesará con el objetivo de mostrarlos adecuadamente sobre el área de visualización. Siguiendo las directrices implementadas por el navegador que vaya a procesar esta estructura, se mostrará el contenido deseado al usuario de cierta manera; gracias a la cada vez mayor estandarización y unificación por parte de los principales navegadores, esas diferencias cada vez son menos apreciables por los consumidores finales de aplicaciones web. Para seguir el código de buenas prácticas del desarrollo HTML, cada elemento de esa estructura DOM se le puede (y debe) asignar un identificador único ("id") que servirá de referencia inequívoca al mismo desde cualquier punto de la página web; generalmente, con el objetivo de consultar o alterar su información. La representación del contenido en pantalla variará al modificar las vistas que conforman este modelo o simplemente su orden. Gracias al motor javascript, los elementos visuales nativos de un navegador web se pueden agrupar en clases mayores que siguen siendo vistas.

De manera análoga a como se hace en el resto de partes de esta aplicación web, también se utilizarán elementos nativos del navegador a la hora de montar la ficha de datos de las actividades de tipo "*Actividad Genérica*": identificador (), nombre (), código fuente (<input type="text">), y botones (<button>).

Por otra parte, el área de pruebas donde se visualizará la actividad genérica se representará como una vista compuesta de varios elementos. Esta vista será el resultado generado por el *Controlador de Aplicaciones Genéricas* que se verá con detalle más adelante. Este resultado se incorporará en el elemento web intermedio ( ) reservado para ese cometido. Será el motor JavaScript el encargado de convertir la respuesta del controlador en una cascada de elementos que se empotrará en el área de pruebas que hay junto al resto del contenido de la ficha de datos de la aplicación que se esté editando en ese momento.

Paralelamente, a cada elemento visual se le puede ajustar un estilo. El estilo corresponde con la apariencia específica que se da a un elemento. En entornos web como en el que se ubica esta solución, la tecnología para definir los estilos es CSS[25]. De esta forma, cambiando la hoja de estilos, es posible modificar atributos como el tamaño, color, borde, etc. de cada elemento. Para facilitar esta tarea, se puede (y se debe) asignar una clase a cada elemento visual. En el caso del módulo de aplicaciones genéricas, se definen las clases asociadas a los nuevos elementos. Estas clases tendrán valores por defecto, pero facilitarán la adaptación de los elementos que las incluyan de cara a un futuro ajuste del aspecto visual de la herramienta.

#### 4.3.3 – Controladores

Finalmente, los controladores en los esquemas MVC se componen por las clases encargadas de dar funcionalidades a las soluciones informáticas que las incorporan. En el caso de este módulo de aplicaciones genéricas solo se define un controlador:

- **CONTROLADOR DE APLICACIONES GENÉRICAS**: será la clase encargada de procesar el código fuente asociado a una actividad de tipo "*Actividad genérica*" y producirá el código HTML a empotrar dentro de la vista correspondiente. Será además la encargada de producir los mensajes asociados a los posibles errores ( "Error: formato incorrecto" o "Error: código inválido") cuando se den las circunstancias que los motiven.

Esta clase se instanciará en el propio objeto global que provee el navegador web siguiendo el patrón de diseño *Singleton*.

### 4.4 – Estructura de datos

Como se comentó en la sección anterior, el propio modelo sirve de guía para definir el esquema que seguirán todos los datos manejados por el módulo. Concretamente, la estructura vinculada a una actividad de tipo "*Actividad genérica*" seguirá la estructura de datos definida por el modelo comentada anteriormente. No se contempla el uso de ninguna estructura de datos concreta durante el proceso de validación o transformación del código fuente asociado a una actividad de este tipo a

código HTML puesto que es una transformación iterativa que incrementa de manera independiente tras cada elemento y no necesita mantener ningún estado previo relevante. Con un uso sencillo de variables nativas básicas es posible ofrecer esta funcionalidad en los términos definidos.

## 4.5 – Interfaces de usuario

El módulo de aplicaciones genéricas añade una interfaz de usuario totalmente nueva a las definidas de serie por la herramienta RETEMANCOSI. Además de ésta, también se modificará la interfaz preexistente de *Creación de Nuevas Actividades* añadiendo nueva información en su desplegable de tipos de actividad disponibles.

Se usarán los conceptos gráficos y la nomenclatura de bootstrap[26] para describir el contenido de las mismas a continuación.

### 4.5.1 – Creación de aplicaciones

En la sección de *Creación de Nuevas Actividades* ya existente en la herramienta de teleasistencia se añadirá en el desplegable un nuevo tipo de actividad. El usuario podrá seleccionar el nuevo tipo de actividad "*Actividad genérica*". El estilo visual del desplegable se mantendrá por defecto. Se establecerá a través de la hoja de estilos base de la herramienta. Igualmente, el elemento que se añadirá deberá incluir un identificador único concreto que permita ajustarle un estilo visual determinado.

```
<div class="tipo_actividades">
    <p>Tipo</p>
    <select class="selector" id="actividad_tipo_valor" name="actividad_tipo_valor">
        <option class="disable" value="" selected="selected" disabled="">Seleccionar el tip
        <option class="opcion_elegida" value="generica" id="generica">Genérica</option>
        <option class="opcion_elegida" value="dado">Dado</option>
        <option class="opcion_elegida" value="contador">Contador</option>
        <option class="opcion_elegida" value="cronometro">Cronometro</option>
        <option class="opcion_elegida" value="texto">Texto</option>
        <option class="opcion_elegida" value="imagen">Imagen compartida</option>
        <option class="opcion_elegida" value="video">Vídeo compartido</option>
        <option class="opcion_elegida" value="audio">Audio compartido</option>
        <option class="disable" value="diferencias" disabled="">Buscar diferencias</option>
        <option class="opcion_elegida" value="emav">EMAV</option>
        <option class="opcion_elegida" value="bads">BADS</option>
        <option class="opcion_elegida" value="d2">D2</option>
        <option class="opcion_elegida" value="5digitos">Cinco dígitos</option>
        <option class="opcion_elegida" value="simbolos">Búsqueda de símbolos</option>
    </select>
</div>
```

*Figura 4.7.* Código de la actividad genérica debidamente identificable ("class" e "id").

#### 4.5.2 – Ficha de datos

Tras crear una nueva actividad de tipo "*Actividad genérica*" o tras seleccionara con el propósito de editarla, se accederá a la ficha de datos asociada a dicha actividad genérica tras realizar correctamente la propia carga de los datos. La ficha de datos de una actividad de este tipo se compone de varias zonas distribuidas de arriba abajo:

- Identificador único.
- Nombre.
- Código fuente asociado.
- Botón de probar.
- Área de Pruebas.
- Botón de guardar.
- Botón de volver. (Común)

### 4.6 – Plan de pruebas

Idealmente, cada componente involucrado en las nuevas funcionalidades dentro del ecosistema generador por el módulo de actividades genéricas contará con diferentes pruebas en la medida de lo posible. En primer lugar, debería haber pruebas **unitarias** que verifiquen, con total cobertura, las salidas de todos los métodos implementados durante este proyecto a partir de una variedad de entradas. Quedan fuera de este ámbito todos elementos delegados; por ejemplo, aquellos componentes gráficos que implementados internamente por el propio navegador (vistas, por ejemplo), pero también aquellos controladores pertenecientes a otros módulos preexistentes (modelos para la persistencia, por ejemplo), no será posible obtener el 100% de cobertura a nivel de pruebas unitarias a nivel de funcional. Se establece por lo tanto este punto exclusivamente para la parte de los controladores creados para este módulo en cuestión. Estas pruebas se enmarcan a partir de las primeras etapas de la fase de IMPLEMENTACIÓN, manteniéndose durante el resto de vida del proyecto.

Posteriormente se realizarán pruebas a nivel de **sistema**. Estas pruebas persiguen comprobar el correcto desempeño cada uno de los casos de uso. Para ello, se definen los pasos de prueba asociados a cada uno. Por último, se ejecutarán estos pasos de

prueba para validar la implementación realizada. Los casos de uso se crean a partir de la definición de los pasos de prueba que corresponden con los puntos presentes en la definición de cada caso de uso.

Finalmente, se realizarán pruebas **reales**. Estas pruebas estarán enmarcadas en la fase de DESPLIEGUE de la aplicación. Para poder realizar estas últimas pruebas se dispondrá de varios escenarios realistas en los que se podrá certificar que la funcionalidad de manera completa.

### 4.6.1 – Especificación común del procedimiento de prueba

Para la realización de todas las pruebas siempre se utilizará como punto de partida la configuración por defecto de la herramienta de teleasistencia. Las pruebas unitarias se ejecutarán a lo largo de todo el ciclo de desarrollo de la aplicación una vez empiecen a estar disponibles. Al no disponer ni requerirse un sistema de integración continua, las pruebas unitarias no estarán automatizadas. De todas formas, éstas se ejecutarán directamente sobre la aplicación al iniciarse o durante los pasos propios del desarrollo (codificación y entrega de versiones) de manera semiautomática.

Aunque no exista un plan de pruebas unitario para los elementos gráficos, se realizarán permanentemente pruebas visuales para comprobar que el estilo de los elementos gráficos es el adecuado. Se valorará tanto el nivel de calidad de la visualización y la ubicación en pantalla de los diferentes elementos, así como de los tiempos de respuesta, aspectos que evalúen si se está consiguiendo un buen rendimiento, y la usabilidad pensando siempre en la diversidad de clientes finales que podrá tener la herramienta (con especial interés en personas de avanzada edad). Al haberse establecido requisitos que fijan los límites admisibles dentro de la herramienta en el ámbito de la usabilidad, se harán pruebas con diferentes dispositivos y configuraciones compatibles. Aún así, como ya se recalcó anteriormente, todas las tareas específicas relativas al diseño gráfico más allá del cumplimiento de los requisitos concretados está fuera del ámbito del proyecto. De todas formas, a un nivel más alto, sí se usarán los diagramas provistos durante la fase de diseño para la validación del maquetado de las interfaces.

En cualquier caso, siempre se podrán aplicar ligeras optimizaciones que pudieran mejorar: la presentación de los elementos gráficos en la pantalla de los usuarios de la herramienta, la información y textos que se muestran, todo aquello que haga a la herramienta más realista, haga más eficiente el trabajo con la aplicación o simplemente facilite las labores a la hora de codificar o mantener el código sin que se produzca una pérdida de la funcionalidad descrita. Por ese motivo, algunas pruebas podrían quedar invalidadas o deberían ajustarse.

El rendimiento de la herramienta, tanto a nivel general como específicamente a nivel del nuevo módulo, se medirán haciendo uso de los perfiladores incluidos de forma nativa en los navegadores modernos que se utilizarán. Se realizarán varias mediciones y se calculará el promedio para reducir el efecto de agentes externos.

Para las pruebas de despliegue se utilizará el siguiente hardware y software:

- **Ordenador**: Ubuntu 20, navegador Firefox en la última versión compatible.
- **Tablet** de 10'': Android 9, navegador Chrome en la última versión compatible.

#### 4.6.2 – Especificación de los pasos de prueba

##### 4.6.2.1 – Unitarias

Para la implementación de las pruebas unitarias, como se ha indicado anteriormente, se buscará una cobertura del 100% en los métodos de las clases que implementen los controladores de la solución. Puesto que en este caso solo existe el *Controlador de Aplicaciones Genéricas*, serán los métodos de este controlador los que se validen por esta familia de pruebas.

La mecánica que se perseguirá a la hora de realizar las pruebas unitarias será la habitual. Se verificará que el valor de salida (o comprobando el objeto alterado si es por referencia) coincide o no con el esperado para cierto valor de entrada que se evaluará. Se establecerán diferentes combinaciones de valores de entrada y valores esperados. Deberán ser lo suficientemente variados, incluyendo cuando sea posible también valores extraños (fuera de los rangos establecidos en los requisitos, por ejemplo), para los cuales se deberá mantener cierta coherencia y estabilidad.

#### 4.6.2.2 – Aceptación

Las pruebas de aceptación se harán siguiendo el objetivo de comprobar el cumplimiento de todos los requisitos acordados. Para ello, puesto que se han definido casos de uso que cubren todos los requisitos funcionales, se establecerán pruebas que ejecuten cada uno de los casos de uso definidos. Se seguirán dos estrategias:

- Se ejecutarán los pasos con valores genéricos/habituales.
- Se ejecutarán los pasos con valores particularmente extraños.

#### 4.6.2.3 – Reales

Para finalizar, la realización de las pruebas reales se llevará a cabo directamente por un profesional que potencialmente utilizá el módulo una vez desplegado. El monitor pondrá a prueba la herramienta con escenarios similares a los esperados en el día a día con los usuarios. Con sus conclusiones se podría ir elaborando ajustes finos en la configuración o estilo gráfico a incorporar durante el MANTENIMIENTO.

# Capítulo 5:

# Implementación

*Codificar las diferentes funciones y estructuras que componen la solución a desarrollar.*

## 5.1 – Base tecnológica

Como se ha visto en capítulos anteriores, dado que este proyecto trata de la construcción de un nuevo módulo para una herramienta preexistente basada en las tecnologías web habituales, el uso del tridente **HTML/JS/CSS** resulta inevitable. Por fortuna, estas tecnologías son ampliamente usadas a día de hoy y disponen de una enorme comunidad de desarrolladores dispuestos a compartir código, metodologías y ayuda de manera desinteresada. Además, estas tecnologías son libres, lo que supone un importante ahorro económico en lo relativo al pago por licencias. Y no menos importante, la mayoría de elementos están estandarizados y gozan más que nunca de un amplísimo soporte y fidelidad en la mayor parte de navegadores modernos.

Las tecnologías a nivel de aplicación que se usarán serán:

- El núcleo del módulo estará construido con **JAVASCRIPT**[27] puro, en la versión que se denomina **ECMAScript 2016**, que es la versión mayormente adoptada en la actualidad. Se aprovecharán todos los gestores, controladores y módulos ya incorporados de serie en la herramienta de teleasistencia.
- Para la composición visual del módulo se usará **HTML** en su versión **5**.
- Para la aplicación de estilos gráficos se usará **CSS** en su versión **3**.

## 5.2 – Metodología de desarrollo

Previo al inicio de la etapa de codificación del módulo, se necesita establecer ciertos conceptos íntimamente relacionados con el proceso de creación del software.

### 5.2.1 – Entorno de desarrollo

La primera elección a la hora de abordar la tarea de escritura del código es establecer el entorno donde se realizará esa acción. Al final de cuentas, un desarrollador pasa gran parte del ciclo de vida de una solución informática basada en

software rodeado de código. La codificación, revisión, depuración, etc. del código son tareas con ciertos pasos mecánicos que se pueden automatizar. También hay numerosas facilidades que pueden ayudar al desarrollador a alcanzar de una manera mucho más satisfactoria y eficiente los objetivos relativos a estas tareas. El entorno de desarrollo se puede convertir por lo tanto en un aliado casi perfecto durante esta fase. Es por ello que su elección es un paso inicial que esconde cierta relevancia; ¡se vuelve un paso de vital importancia cuando se persigue el éxito del proyecto!

Hay muchos entornos disponibles para el desarrollo web, especialmente usando JAVASCRIPT. Por las características intrínsecas de este encargo, el entorno de desarrollo escogido ha sido **NOTEPAD++**[28]. Esta herramienta se conoce más por ser un "editor avanzado" de texto plano que un IDE[29]. Pero el hecho de incluir el resaltado de sintaxis cuando el texto corresponde a ciertos lenguajes soportados lo vuelve capaz de trabajar con código fuente de manera básica. Pero no se queda ahí: esta herramienta puede ampliar sus funcionalidades añadiendo ciertos complementos. Con la adecuada configuración, además del resaltado de sintaxis, es posible obtener ayudas al programador a través de atajos de teclado. Algunas de las ayudas que ofrece es la generación automática de código, búsqueda de referencias a varios niveles y agrupación en proyectos y navegación rápida entre diferentes ficheros. Además, permite la visualización de objetos agrupados por tipo (funciones, variables, clases...), la conexión con el control de cambios, de calidad, etc. a través de distintos *plugins*.

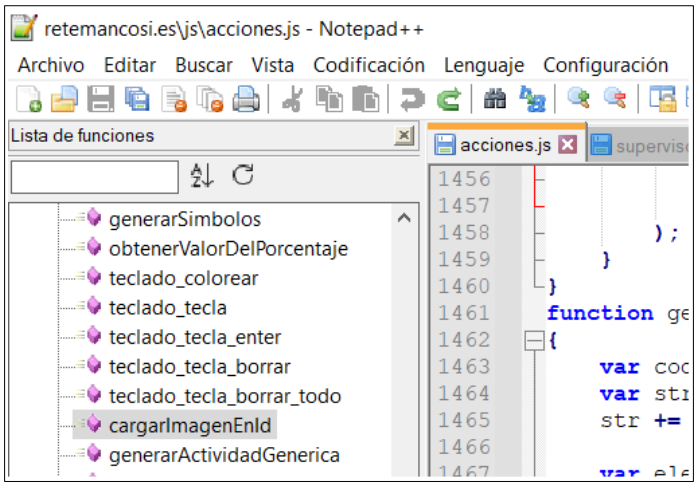


*Figura 5.8.* Una "lista de funciones" en NOTEPAD++.

Debido a que el entorno de desarrollo elegido no integra completamente todos los elementos involucrados en las taras de desarrollo, de manera complementaria al uso de este potente editor, se han utilizado las **Herramientas para desarrolladores** que incluye el navegador FIREFOX nativamente para la revisión, análisis del rendimiento y depuración de la solución en general y del nuevo módulo en particular.

### 5.2.2 – Control de cambios

El control de cambios es otra de las características a tener en cuenta a la hora de definir el plan de trabajo relativo al proceso de desarrollo de un proyecto. Como es de sobra conocido, llevar un histórico de las versiones de los diferentes ficheros que van generándose durante la construcción de una solución informática basada en software permite la trazabilidad del proceso de codificación y ofrece una buena protección ante posibles problemas de gestión de la configuración[30], además de facilitar la evolución (incluso en paralelo) del mismo.

Existen varios sistemas de control de cambios con diferentes objetivos. Para este desarrollo se optó por el uso de **subversion** (SVN) por cuatro motivos:

- Es un sistema de control de versiones sencillo y que se integra fácilmente tanto en los sistemas de despliegue como en el entorno de desarrollo elegido.
- El desarrollo se iba a realizar por una persona y sería lineal. SVN es un sistema que, aunque permite el trabajo en ramas, no está especialmente pensado para manejar bifurcaciones e integraciones, por lo que no supone una limitación.
- El código de la herramienta ya estaba bajo ese tipo de control de versiones.
- Dispone de herramientas que permiten extraer datos estadísticos de uso.

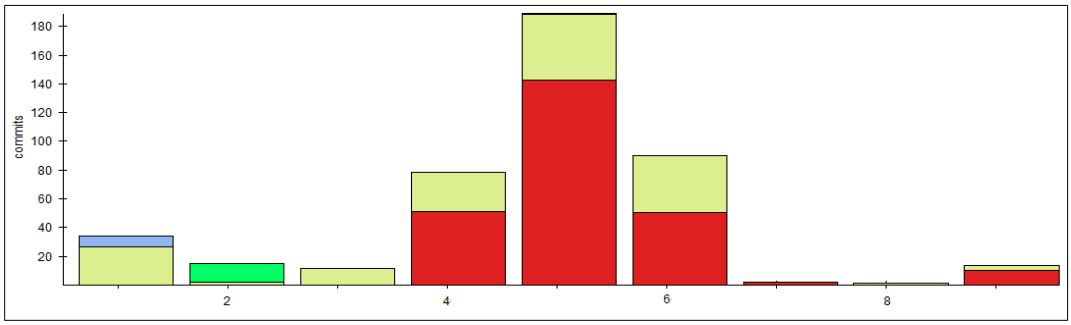


*Figura 5.9.* Estadísticas de entregas en SVN (en rojo las relativas al módulo).

### 5.2.3 – Manual de estilo

El tercer punto a tener en cuenta antes de abordar el proceso de codificación del código de una solución informática basada en software es fijar el estilo de programación que se deberá seguir. El manual de estilo que se ha seguido para la codificación del código fuente y los demás recursos o conceptos relacionados con el código que se incluyen en este trabajo sigue las directrices marcadas por la Guía de Estilo que aparece en el anexo [10.2]. Es la guía vigente tanto en la herramienta usada en RETEMANCOSI como en los demás desarrollos de ESTUDIOS PAWI SL.

En esta línea, como valor añadido incorporado durante este trabajo en lo relativo al desarrollo de la herramienta de teleasistencia, se ha configurado como complemento dentro del entorno de desarrollo elegido una aplicación llamada ASTYLE[31] con el objetivo de facilitar el cumplimiento de la guía de estilo en la mayor medida posible. Para ello, esta aplicación comprobará tras cada "guardado" del código editado en el entorno de desarrollo si éste cumplen cada uno de los puntos que se definen en la guía de estilo. Por suerte, la herramienta ofrece por defecto el estilo "*Allman style/bsd*". Este estilo es prácticamente el estilo de programación que se define la guía de estilo vinculada al trabajo. Por ello, no fue necesario modificar más que unas pocas opciones de personalización que hay en la herramienta.

## 5.3 – Codificación de la funcionalidad en paquetes

Debido a que desde la implementación global de la herramienta de teleasistencia se establece con precisión la estructura que deben seguir los módulos, el contenido de los diferentes paquetes que conformarán el módulo está prefijado. Básicamente un módulo compatible con la herramienta se compone de los siguientes paquetes:

- **Plantillas HTML**: se trata de fragmentos de código HTML que definen vistas específicas propias del módulo. Aunque no es el caso, pueden sobrescribir otras vistas previamente definidas, completándolas o sustituyéndolas totalmente. Las plantillas pueden ser estáticas si solo contienen código HTML original, pero pueden ser dinámicas si incluyen ciertos elementos que serán cambiados por determinados valores en tiempo de ejecución antes de verse.

- **Estilo CSS**: se trata de ficheros con reglas de estilo que, por la mera construcción que tiene el propio lenguaje CSS, añaden o sobrescriben los comportamientos visuales que pudieran tener ciertos elementos visuales.
- **Código JS**: un módulo podrá incorporar las funciones propias de los controladores que implemente en ficheros de código javascript.
- **Otros ficheros de recursos**: se pueden incluir otros paquetes para incorporar imágenes, sonidos, iconos y demás recursos complementarios. De esa forma, el módulo los podrá explotar directa o indirectamente. También es posible sobrescribir algunos de los recursos preexistentes.

### 5.4 – Construcción del sistema real

Con toda la información indicada hasta este punto, se pudo iniciar el proceso de implementación del módulo. La división en paquetes facilita la organización de los diferentes ficheros que se deberían crear. Disponer de un DISEÑO claro fruto de una fase de ANÁLISIS adecuada no hizo más que facilitar esta fase.

Lo primero que se realizó fue la creación de los cuatro paquetes definidos en carpetas e incluir en cada una de ellas los ficheros necesarios. Para el paquete "html" se incluyó la plantilla con el selector del nuevo tipo de aplicación y la vista dinámica relativa a la *Ficha de datos* utilizada a la hora de editar una aplicación de tipo "*Aplicación genérica*". Para el paquete "css" se incluyó el fichero con el estilo genérico para las nuevas clases gráficas definidas en el módulo. Para el paquete "js" se incluyó el código funcional orientado a objetos; incluyendo las estructuras de programación (clases, funciones, variables…) correspondientes con el diseño elegido. Por último, en una carpeta "img" (recursos de imagen) para el paquete de recursos gráficos se incluyeron imágenes que serán consumidas por ciertas actividades genéricas planteadas.

Finalmente, las carpetas de los cuatro paquetes fueron comprimidas en un fichero .zip según las directrices relativas al formato de los módulos y su distribución normalizada que están marcadas por la propia herramienta de teleasistencia. Este fichero será el utilizado en la fase de DESPLIEGUE para instalar el módulo.

### 5.4.1 – Particularidades detectadas durante la codificación

Durante los avances realizados en esta fase, y como estaba estipulado, se ejecutaron las pruebas unitarias una vez fueron creadas. También se hizo todo lo posible por mantener el código generado alineado con las normas indicadas en la guía de estilo. Esta segunda cuestión resultó sencilla de alcanzar gracias al complemento de supervisión comentado anteriormente.

El uso del control de versiones, como se puede constatar en la figura 5.6, siguió una distribución normal, produciéndose el pico de máxima actividad en la segunda semana desde el inicio de la fase de IMPLEMENTACIÓN.

Por otra parte, disponer del resto de controladores y facilidades que conforman la herramienta de teleasistencia original usada en RETEMANCOSI, unido a un buen conocimiento de todas sus entrañas, facilitó enormemente el trabajo. Está claro que el proceso de desarrollo partiendo de una hoja en blanco (no habiendo un análisis/diseño previo, pero también a veces habiéndolo) es muy complicado. Por suerte, en este caso, tanto las características a bajo nivel como el grueso de las partes funcionales de la herramienta estaban ya hechas, lo que solo facilitó la implementación la funcionalidad propia del nuevo módulo.

# Capítulo 6:

# Pruebas

*Ejecutar el plan de pruebas acordado y verificar el cumplimiento de todos los requisitos que se han fijado.*

## 6.1 – Pruebas unitarias

Las pruebas unitarias son las pruebas más importantes de toda la cadena de montaje del software. Debido a su importancia, deben ser elaboradas adecuadamente. Los resultados arrojados por estas son un claro indicativo de la calidad del software que validan. Las pruebas unitarias para este proyecto, que ha seguido para su implementación el esquema MVC, buscan cubrir la mayor parte del código funcional (controladores) ya que la comprobación a nivel de vista o de modelo se sale del ámbito de actuación de este trabajo.

Además de una correcta elaboración que garantice una validación exquisita de los componentes funcionales de manera unitaria, es necesario que las pruebas unitarias sean ejecutadas con mucha frecuencia. A falta de un entorno de integración continua, como se indicó anteriormente, la ejecución de las mismas se encuentra vinculada con el guardado del código fuente. De esta forma, cada vez que se modificara o se ampliara cierta funcionalidad, la batería de test unitarios se debería ejecutar completamente; en caso de detectarse un error, se deberá emitir un aviso al desarrollador para que compruebe los cambios. Los test unitarios también se deberán lanzar al realizar las entregas en el control de versiones, garantizando la estabilidad a bajo nivel de los diferentes componentes que conforma la solución en curso.

Aunque el enfoque del proyecto en relación al calendario de fases estableciera anteriormente que las pruebas se deberían ejecutar tras finalizarse la fase de IMPLEMENTACIÓN, en el caso de las unitarias, su codificación y ejecución se realiza en paralelo con la construcción del módulo. Sin llegar a realizarse una metodología basada en pruebas, cada vez que una pequeña funcionalidad sea completada, se deberán dar de alta en el sistema de pruebas unitarias las pertinentes comprobaciones.

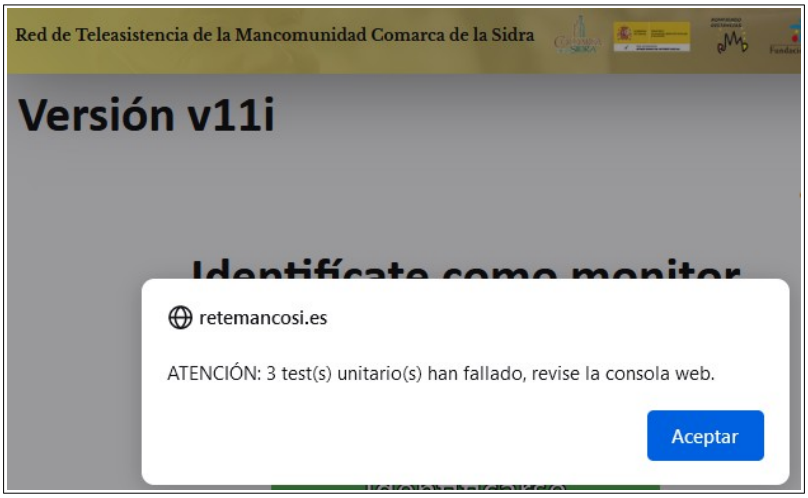


*Figura 6.10.* Reporte erróneo tras ejecutar los test unitarios.

Una vez configurado, el sistema de validación basado en pruebas unitarias se mantendrá operativo continuamente. De esta forma, es sencillo adelantarse en la detección de posibles errores al ejecutar la herramienta (supuestamente estable en cuanto no aparecen fallos) en nuevos dispositivos. Debido a que el flujo de ejecución de la funcionalidad recogida en este módulo es altamente determinista (esto es, que los resultados verificados por los test unitarios no dependen de elementos aleatorios o que pudieran estar fuera del control del propio entorno de ejecución de la prueba), cualquier fallo reportado que ocurra sistemáticamente, pero de manera aislada, estará justificado, o bien por un fallo en la implementación del mismo o un problema en el algoritmo implementado que incumple las reglas de estilo que buscan la estanqueidad de la implementación con situaciones externas, o bien una potencial situación no controlada correctamente en el entorno de ejecución donde se produzca.

## 6.2 – Pruebas de aceptación

Dada la particularidad de las pruebas unitarias (se ejecutaron en paralelo con la fase de IMPLEMENTACIÓN), las pruebas de aceptación constituyen la primera capa de validación que se ejecuta en esta fase. El objetivo de estas pruebas es comprobar con diferentes valores (genéricos y extraños) que todos los casos de uso asociados a los requisitos funcionales del proyecto se ejecutan conforme a los adecuadamente establecidos en el plan de pruebas.

Todas las pruebas definidas se ejecutaron siguiendo los pasos de prueba definidos. Los resultados arrojados por las pruebas fueron satisfactorios, por lo que se garantizó que los requisitos funcionales establecidos habían sido correctamente implementados.

Paralelamente a la verificación de los casos de uso, también se validaron en las pruebas de aceptación aspectos relativos al resto de requisitos. En el marco de los requisitos tecnológicos, se realizó la validación del formato web los ficheros HTML generados. Debido a la construcción dinámica que realiza en ocasiones la herramienta (ciertos elementos estáticos están presentes ya durante la carga de la página web, pero otros se generan dinámicamente y no se pueden validar hasta que el usuario ejecuta ciertas acciones), para la validación se han usado dos técnicas diferentes.

Por una parte, se pasó un validador gratuito de HTML5 que ofrece la W3C, dando como resultado la validación total del documento.

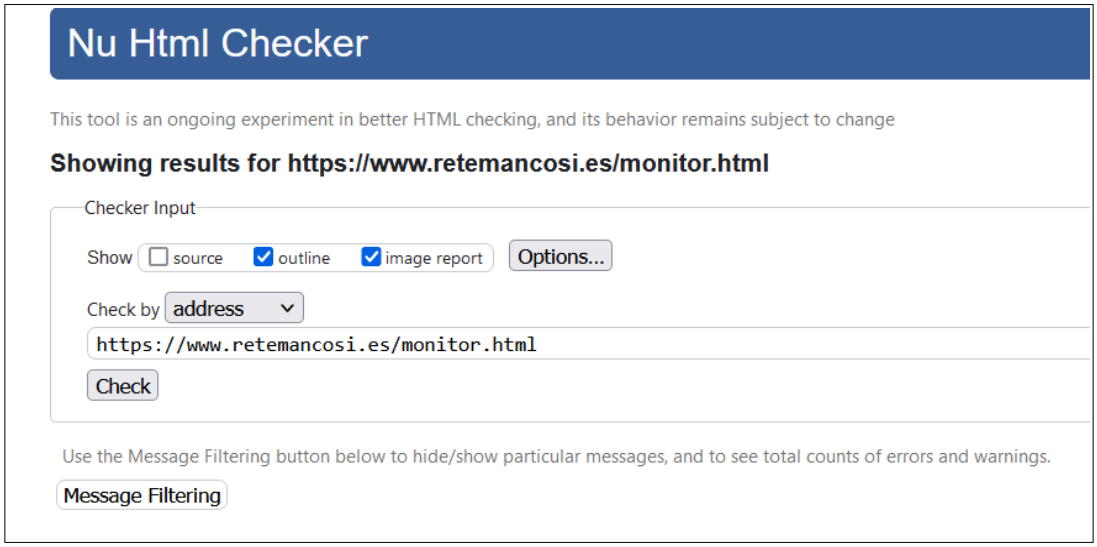


*Figura 6.11.* Validación online de la página principal de monitores.

Este resultado muy probablemente esté vinculado al hecho de que, a través del plugin del navegador "HTML Validator"[32] instalado como parte del entorno de desarrollo, se realiza de manera automática la validación con el W3C. En caso de haber alguna violación del estándar, se reporta gráficamente en el navegador. Pasa a ser entonces responsabilidad del desarrollador arreglar o no el error; por lo general, cuando se parte de cero errores y aparece uno, lo más sensato es arreglarlo sin más.

Por otra parte, fue necesario validar el código HTML generado dinámicamente dentro de la herramienta a partir de las diferentes acciones que incorpora este módulo. Por suerte, este mismo plugin pudo realizar esas comprobaciones durante la ejecución del resto de pruebas de aceptación. El resultado reportado fue sobresaliente durante todas las pruebas, ya que no se reportaron errores.

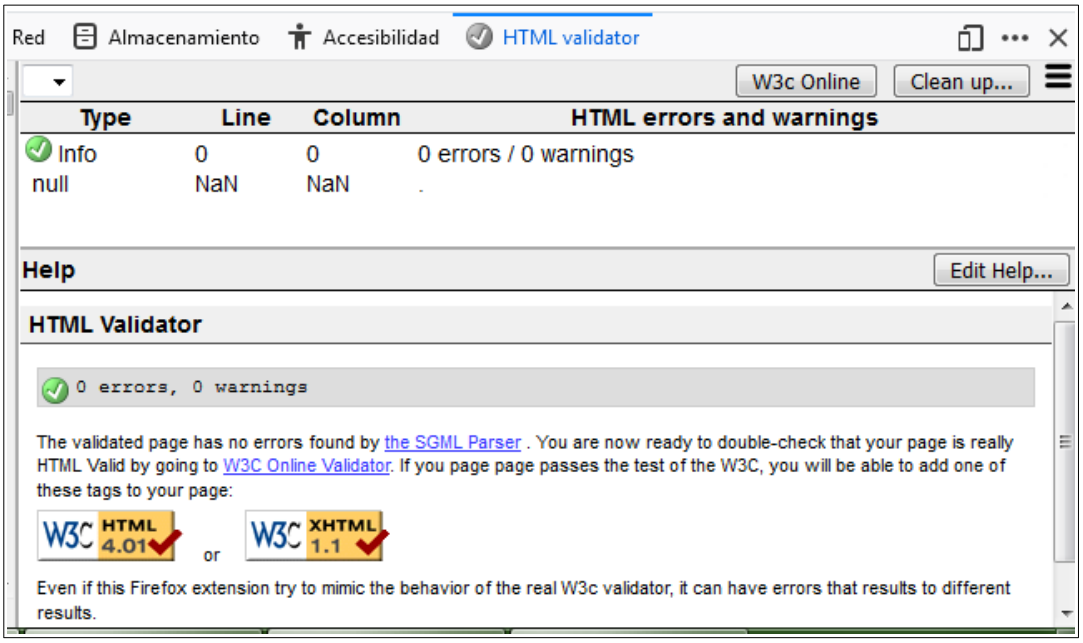


*Figura 6.12.* Validación en tiempo real dentro del navegador.

Por otra parte, también se realizaron pruebas de aceptación referentes a los requisitos de usabilidad. Estos requisitos establecen que la funcionalidad debe poderse realizar tanto desde PC como tablet. Para ello, una vez validados las pruebas de aceptación en PC, se repitieron de nuevo para tablets. Debido a lo complicado que resulta en ocasiones colocar elementos visuales en áreas pequeñas (pese a que el tamaño mínimo estuviera fijado en 10 pulgadas), se detectaron algunos problemas de usabilidad en ese tipo de dispositivos; pero fueron sencillos de resolver.

Puesto que fue necesario realizar algún cambio, se realizaron pruebas de regresión de nuevo en PC para comprobar que no se hubieran introducido defectos.

Finalmente, todas las pruebas de aceptación se ejecutaron satisfactoriamente tanto en PC como en las tablets compatibles.

## 6.3 – Pruebas reales durante el despliegue de la aplicación

Tal y como se definió en fases anteriores, las pruebas reales se realizaron por profesionales debidamente instruidos tras haber recibido la formación pertinente. Se crearon actividades genéricas utilizando los ejemplos suministrados e incluso se crearon actividades genéricas desde cero aplicando las construcciones presentes en el manual de uso. Los profesionales pudieron ejecutar satisfactoriamente durante sus talleres con usuarios reales las actividades y obtener resultados de alto valor añadido.

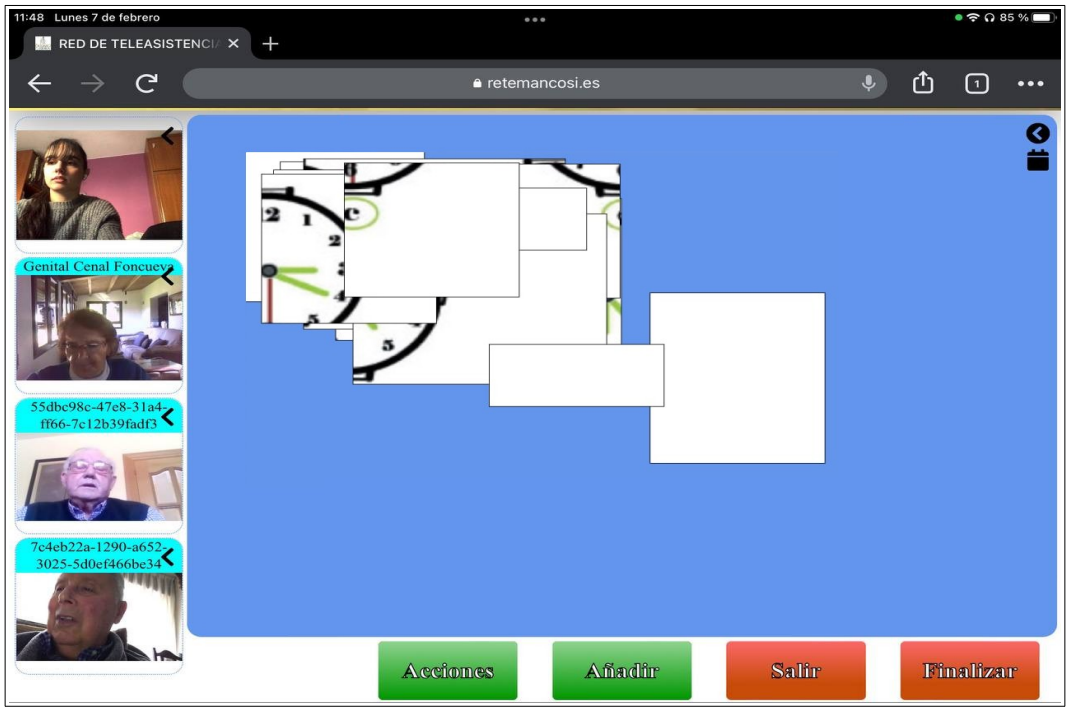


*Figura 6.13.* Actividad genérica “reloj puzle” durante un taller.

# Capítulo 7:

# Despliegue

*Instalar la solución en el sistema informático de la Mancomunidad "Comarca de la Sidra" y dar formación a los monitores.*

## 7.1 – Redacción del manual de usuario

Este módulo, así como la herramienta en su conjunto, son soluciones informáticas que no están necesariamente destinadas a usuarios finales con un perfil técnico. Si bien se requiere un poco de práctica para lograr la construcción de actividades genéricas con cierta relevancia, cualquier persona debería estar en condiciones de utilizar el módulo para digitalizar una actividad genérica previamente pensada sobre el papel. Debido a la novedad que alguno de los conceptos con los que se trabaja en este nuevo módulo supone cierta novedad para el día a día de la mayoría de profesionales que van a usar esta capacidad, se ha elaborado un *Manual de Usuario* para este módulo. En este manual se detallan todas las posibilidades que ofrece esta implementación a los profesionales, así como ejemplos de actividades que podrán ser ejecutadas por los participantes de un taller. Además del contenido textual explicando las diferentes operaciones que se pueden realizar, todas ellas van acompañadas de capturas de pantalla para facilitar una mayor comprensión de los conceptos presentados en ese documento. También se usa un lenguaje más informal con ese objetivo.

El manual se compone de seis bloques:

- Creación de una actividad genérica.
- Editando una actividad genérica.
- Construyendo una actividad genérica estática.
- Construyendo una actividad genérica dinámica.
- Ejecución de una actividad genérica en un taller.
- Procesado de la información recogida en una actividad genérica.

En el apéndice se incluye el *Manual de Usuario* de manera íntegra, puesto que sirve también como demostración de las posibilidades que ofrece este desarrollo.

## 7.2 – Formación

Como complemento al manual, en la oferta se incluye una **sesión de formación** impartida por ESTUDIOS PAWI SL de 5 horas a los profesionales de RETEMANCOSI. Esta sesión de formación consistirá en la lectura conjunta del Manual, explicación de los ejemplos proveídos y aclaración de las dudas que pudieran surgir por parte de los participantes de la sesión formativa.

La sesión se realizó a través del canal de Telegram que ya existía debido a la prestación del servicio de MANTENIMIENTO. Para poderse reutilizar en el futuro, la sesión fue guardada.

## 7.3 – Instalación del módulo en RETEMANCOSI

Todo el desarrollo del módulo se hizo en un entorno controlado para pruebas (sandbox)[33]. Este entorno consistía en una instalación limpia de la herramienta de teleasistencia PAWI CONECTA con una colección de datos predefinida generada a partir de datos pseudoaletorios. De esta manera, cualquier potencial error y todos los recursos generados no entrarían en conflicto con los datos en producción dentro de RETEMANCOSI, la versión de demostración abierta en ESTUDIOS PAWI u otros despliegues.

Llegado el momento acordado y tras superar todas las validaciones, se procedió al despliegue del módulo desarrollado en la herramienta de la Red de Teleasistencia de la Mancomunidad “Comarca de la Sidra”. El proceso documentado en el Manual de Instalación sigue el procedimiento general de instalación de módulos que tiene la herramienta de teleasistencia.

Básicamente una vez generado de manera adecuada el fichero .zip con el contenido del módulo, este debe adjuntarse en el apartado de administración de módulos dentro de la sección de administración de la herramienta. Para eso es necesario autentificarse con un usuario que disponga del rol administrador.

El proceso internamente desempaqueta el fichero .zip de manera ordenada y da de alta el módulo en el sistema para que se pueda consumir. De manera automática, las funcionalidades aparecen en la herramienta actualizándola o volviendo a acceder.

### 7.4 – Garantía, soporte y mantenimiento

Una vez hecho el despliegue del nuevo módulo en RETEMANCOSI y realizada la transferencia de conocimiento a través de formativas, se da por finalizado el desarrollo del proyecto. ESTUDIOS PAWI SL ofrece un servicio de garantía a todos los usuarios que soliciten un desarrollo a medida. Esta garantía establece que la empresa se hará cargo de cualquier defecto en el funcionamiento que pueda aparecer en el sistema, solucionándolo de forma totalmente gratuita, si es reportado en un plazo de 6 meses desde la entrega del trabajo. La garantía no cubre modificaciones del alcance, de los requisitos establecidos, omisiones o interpretaciones que no se hayan aclarado durante el desarrollo del proyecto.

Por otro lado, la empresa ofrece el servicio de soporte y mantenimiento. En este caso, la empresa podrá realizar modificaciones bajo demanda relacionadas con el ajuste de ciertas características, pequeñas alteraciones a nivel gráfico, sesiones formativas e instalación de actualizaciones que pudieran darse en la herramienta o módulos incorporados. Un ejemplo de actividades relacionadas con el soporte son las intervenciones realizadas en un grupo de Telegram con todos los profesionales que usan la herramienta en RETEMANCOSI donde se aclaran las dudas que se exponen y se reportan errores puntuales con la plataforma.

# Capítulo 8:

# Cronograma y costes

## 8.1 – Planificación temporal

Las tareas ejecutadas fueron realizadas por un equipo de trabajo formado por una única persona con dedicación exclusiva al proyecto en horario laborable, de lunes a viernes, de 09:00 a 14:00 (5 horas), un total de **77 días**, que corresponden con un total de **385 horas** de trabajo efectivo.

En la siguiente tabla se muestra la dedicación al **proyecto**, en horas y días, así como las fechas de inicio y fin y el esfuerzo correspondiente a cada fase y la suma.

| Fase | Fecha Inicio | Duración (días) | Acumulado (días) | Duración (horas) | Acumulado (horas) | Fecha Fin | Porcentaje | Suma porcentaje |
|---|---|---|---|---|---|---|---|---|
| **Formalización** | 10/01/2022 | 10 | **10** | 50 | **50** | 21/01/2022 | 13% | 13% |
| **Análisis** | 07/02/2022 | 13 | **23** | 65 | **115** | 23/02/2022 | 17% | 30% |
| **Diseño** | 07/03/2022 | 18 | **41** | 90 | **205** | 30/03/2022 | 23% | 53% |
| **Implementación** | 31/03/2022 | 22 | **63** | 110 | **315** | 06/05/2022 | 28% | 81% |
| **Pruebas** | 18/04/2022 | 9 | **72** | 45 | **360** | 20/05/2022 | 12% | 93% |
| **Despliegue** | 20/04/2022 | 5 | **77** | 25 | **385** | 20/05/2022 | 7% | 100% |
| **Total** | 10/01/2022 | **77** (días) | | **385** (horas) | | 20/05/2022 | **100%** | |

*Tabla 8.7.* Reparto del tiempo de esfuerzo.

Ténganse en cuenta para el cálculo del tiempo de cada fase el marcado en las columnas "duración" en vez del lapso entre la fecha de inicio y la de fin, puesto que algunas semanas incluyen días festivos o periodos en los que las tareas se pausaron.

Existe un lapso de tiempo significativo entre la primera fase de FORMALIZACIÓN y la siguiente de ANÁLISIS. Esto se debió a que hubo que cuadrar horarios y demás disponibilidades con el cliente con el fin de realizar profesionalmente las entrevistas y reuniones donde se expusieron y definieron los requisitos de la solución.

A nivel global, en lo relativo al cómputo total de horas propias del **trabajo**, se dedicaron 50 horas (~11% del total) a la fase de formalización, 65 horas (~15% del total) a la fase de análisis, 90 horas (20% del total) al diseño, 110 horas (~24% del total) a la implementación, 48 horas (~10% del total) a la realización de pruebas, y 25 horas (~5% del total) al despliegue. La gestación de la idea, obtención de conclusiones, la creación, revisión y adecuación de toda la documentación que acompaña a este trabajo, y la propia elaboración final de esta Memoria, así como las reuniones con el tutor y cotutor sumaron el resto de las 65 horas (~15% del total) para alcanzar las 450 horas correspondientes a los 18 ETCS.

Dado que la fase de PRUEBAS se realizó de manera intermitente entre las fases IMPLEMENTACIÓN y DESPLIEGUE, probablemente se visualice de manera más clara el cronograma del trabajo en el siguiente diagrama estilo Gantt. Los huecos verticales sin asignación representan los días festivos o de descanso.

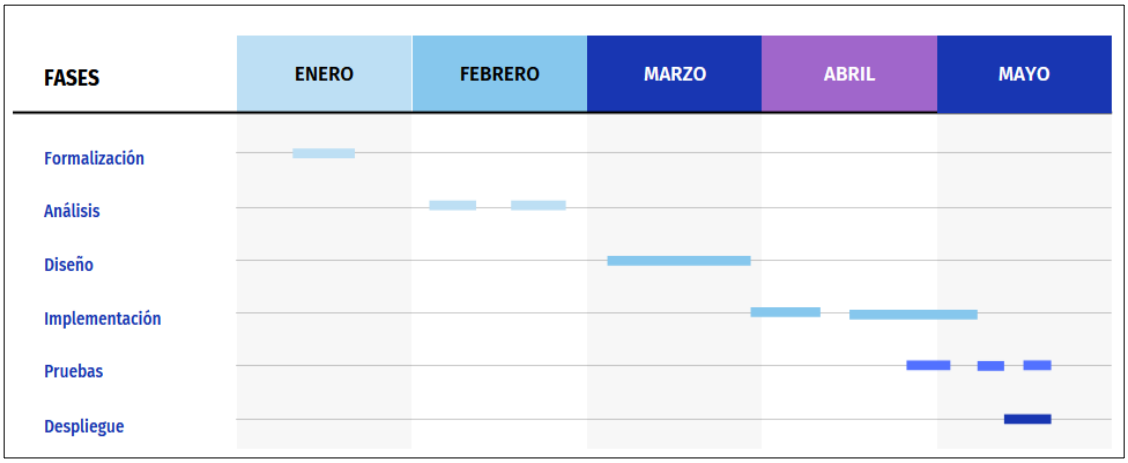


*Figura 8.14.* Distribución de las fases a lo largo del trabajo.

## 8.2 – Costes asociados al proyecto

De manera complementaria a la gestión de los tiempos en el trabajo, se elabora una estimación del coste real asociado a la construcción de este proyecto. El coste total está desglosado en diferentes apartados. En la primera fila aparecen anotados los gastos de personal, que corresponden al coste bruto de un programador según convenio[34]. El siguiente bloque corresponde al apartado de coste del material requerido; se incluyen los costes aplicando una amortización del 40% ajustada al tiempo de vida del proyecto. Después están los conceptos relativos a la adquisición de licencias para el desarrollo adecuado de la aplicación. Luego está el apartado de gastos correspondientes al pago de servicios necesarios para el correcto desarrollo del proyecto. Todos estos costes incluyen los impuestos de aplicación por la legislación vigente. Por último, del subtotal de gastos, se añade un 14% extra para otros gastos: amortización del alquiler, gastos empresariales, beneficios, márgenes, etc.

| Apartado | Concepto | Cantidad | Coste unitario | Coste total |
|---|---|---|---|---|
| Mano de obra | Salario programador | 385 horas | 22 euros / hora | 8.470 euros |
| Material (amortización del 40%) | PC | 1 unidad | 180 euros | 180 euros |
| | Tablet prototipos | 2 unidades | 48 euros | 96 euros |
| Licencias | Ubuntu 24 | 1 unidad | Gratuito | Gratuito |
| | Firefox y plugins | 1 unidad | Gratuito | Gratuito |
| | NotePad++ | 1 unidad | Gratuito | Gratuito |
| | LibreOffice | 1 unidad | Gratuito | Gratuito |
| Servicios | Alojamiento web | 5 meses | 16 euros / mes | 192 euros |
| | Registro dominio web | 1 año | 12 euros | 12 euros |
| | Certificado SSL | 2 trimestres | Gratuito | Gratuito |
| Subtotal | | | | 8.950 euros |
| Otros gastos | Calculado como el 14% del subtotal | | | 1.253 euros |
| Total | | | | 10.203 euros |

*Tabla 8.8.* Tabla de costes.

# Capítulo 9:

# Reflexión

## 9.1 – Sobre los resultados experimentales

El módulo creado durante este proyecto para una herramienta del calibre de la utilizada en RETEMANCOSI ha sido una experiencia única. A nivel técnico ha supuesto una mejora sustancial de un servicio público de especial interés en nuestra región. El hecho de recibir por parte de los usuarios, profesionales y del propio cliente una valoración positiva y un agradecimiento por las tareas realizadas no hace más que evidenciar un trabajo realizado a la altura de las circunstancias. Si bien es cierto que en mi autocrítica aparecen aspectos a mejorar, los resultados son positivos.

La realización de las pruebas finales por parte de los profesionales con usuarios reales fueron satisfactorias. Se cubrieron todos los requisitos demandados. Incluso se fue más allá en algunas ideas planteadas originalmente antes de fijar el alcance de la oferta. El resultado entregado cumplió con los códigos de buenas prácticas establecidos en las condiciones de uso de la herramienta y las directrices que exige su licencia. Los diseños elaborados tras el análisis de las necesidades del cliente facilitaron y agilizaron en cierta medida la construcción de la solución.

En resumidas cuentas, el proyecto consiguió cumplir los objetivos establecidos sin mayores contratiempos y el resultado ya se puede utilizar en la página web del proyecto RETEMANCOSI.

## 9.2 – Valoración empresarial

Como se puede observar, el coste del desarrollo de este proyecto es sustancialmente superior al presupuesto enviado al cliente: los costes multiplican por cinco el importe del presupuesto antes de impuestos.

Ciertos razonamientos podrían achacar esta situación al uso del software libre. Pero nada más lejos de la realidad; la ausencia del cobro por la licencia de uso (en la parte del presupuesto al cliente) va en consonancia con el ahorro en costes por uso de

licencias de terceros durante el proceso productivo. Hacer un uso interesado del software libre es cuanto menos cuestionable a nivel de comunidad. Si a eso le añadimos que, particularmente en este caso, el desarrollo ha sido financiado en la totalidad por fondos públicos, cualquier otro comportamiento que no implique la liberación del código realizado rozaría la inmoralidad, al menos con una visión comprometida con la sociedad. Al margen de lo que pueda pensar cada individuo, es importante incorporar en las actuaciones relativas a la **Responsabilidad Social Corporativa**[35] este tipo de ideales.

Pero si algo está claro es que una empresa tampoco se crea para perder dinero. Si bien el resultado por separado se podría catalogar dentro de ese conjunto de contratos que nunca se debían haber aceptado, en este caso hay que entenderlo en un contexto mucho mayor. Por una parte, la vinculación previa de la empresa con la Mancomunidad sí ha generado un *colchón financiero* que puede sufragar parte de los gastos de este desarrollo que no se compensan con los ingresos percibidos directamente a través de la facturación del propio trabajo. Por otra parte, el modelo de negocio de la empresa, que dentro del servicio también ofrece el mantenimiento de la herramienta, permite cubrir parte de esos gastos adicionales en el corto plazo.

Tras este trabajo lo que queda de manifiesto es que la condición de software libre es una cualidad que, si no logra considerarse beneficiosa, al menos sí actúa como neutral a la hora de implantarse en una empresa en el peor de los casos. Personalmente opino que la balanza se inclina hacia el lado bueno: los costes fijos que implica el uso de aplicaciones con licencias de pago o las dificultades de usar herramientas sin la posibilidad ni tan siquiera de visualizar cómo están hechas por dentro suponen ¡incluso! un riesgo tanto para el proceso productivo de una empresa basada en software como para su propia gestión. En entornos de colaboración con las Administraciones Públicas es un factor que adquiere más importancia si cabe. A falta de una legislación que obligue a la liberación de todo el código usado y subvencionado por el sector público, predicar con el ejemplo parece lo mínimo que las empresas deberían hacer. Desde ESTUDIOS PAWI SL se fomentará el desarrollo de código bajo licencias libres como factor que añade valor añadido a los proyectos.

## 9.3 – Mejoras y posibles evolutivos

Quizás la mayor ausencia en todo el desarrollo ha sido abordar el asunto de la programación gráfica dentro de la misma herramienta, pero ha sido la funcionalidad a sacrificar para que aspectos como el coste económico y temporal del proyecto no comprometieran la viabilidad del proyecto. Probablemente más adelante se puedan incorporar mejoras en esta línea, tanto de manera interna a través de un módulo análogo al desarrollado durante este proyecto, como a través de una nueva herramienta complementaria dentro del paquete de PAWI CONECTA o de un tercero. Incluso, ¿por qué no?, como parte de otro trabajo fin de estudios.

En lo referente al propio módulo, probablemente añadir el soporte a nuevos elementos (más figuras geométricas básicas o también complejas), nuevos efectos sobre los textos, animaciones, soporte para recursos de audio o vídeo podrían ampliar sustancialmente las posibilidades de las actividades de tipo "*Actividad genérica*". En esta línea el mayor reto lo acapara la problemática que hay a la hora de definir cómo se amplía el soporte a más y más enrevesadas características teniendo en cuenta, por un lado, las posibles limitaciones que pudiera tener el formato actual del propio código, y por otro lado no menos importante, la dificultad extra que podría suponer a los profesionales que no tienen suficiente experiencia técnica en la programación ir añadiendo más y más complejidad.

Sea lo que sea, estoy convencido de que la herramienta seguirá creciendo tanto en usuarios como en funcionalidades, y que todas las mejoras abrirán nuevas posibilidades de gran interés para todas las partes interesadas.

## 9.4 – Conclusiones generales del trabajo

El desarrollo de un TFM como parte de un proyecto real, llevado a cabo a través de una empresa de reciente creación de la que uno es socio fundador, es sin duda alguna un viaje repleto de retos y muchísimo aprendizaje.

Las experiencias vividas tras las diferentes etapas formativas y laborales por las que he tenido que pasar desde mi primer contacto con la informática me han provisto de conocimientos y técnicas suficientes para poderme enfrentar a retos cada vez más

difíciles. Específicamente, considero que muchos de los conocimientos adquiridos o reforzados durante la docencia del Máster Universitario me han ayudado a superar satisfactoriamente este trabajo. Particularmente considero que el desarrollo realizado en este proyecto no ha sido la tarea más ardua que he abordado en el ámbito de la informática, pero sí supone una labor que va más allá de tirar líneas de código.

Si bien es cierto que el principal producto de este proyecto son líneas de código que hacen posible la funcionalidad demandada por el cliente, todo el trabajo previo y posterior a la codificación, que va incluso más allá de las fases específicamente técnicas, ha supuesto para mí un reto en mayor o medida. Probablemente el desarrollo siguiendo una metodología a mi juicio adecuada y con una planificación realista sin agobios ha facilitado enormemente la consecución de los objetivos planteados de una manera eficiente. Si a eso se le une el marco universitario en el que ha sido desarrollado, con la colaboración en todo momento del tutor y cotutor, y el ámbito ampliado que me ha ayudado a poner en práctica aspectos de la gestión de proyectos, considero que el resultado global es positivo.

Por último, y no menos importante, me parece oportuno resaltar que el trabajo se realizó en colaboración con un equipo extremadamente multidisciplinar. A nivel personal, me supuso un gran reto (muy enriquecedor, por otra parte) el trabajo con las personas mayores, siempre muy participativas y espontáneas. También tuve que dar soporte y formación a personas del ámbito de los servicios sociales (psicólogos, alumnos en prácticas, animadores socioculturales…), perfiles que habitualmente no tienen unos conocimientos profundos en nuevas tecnologías. Considero que este trabajo podría favorecer futuras colaboraciones entre Departamentos y profesionales que históricamente han estado alejados del ámbito las ingenierías.

Quizás el desarrollo realizado en este trabajo pueda suponer un paso con cierta trascendencia en el camino que, poco a poco, introduce en el área psicosocial las ventajas de la Ingeniería Informática. ¡Todo se verá!

# Capítulo 10:

# Apéndice

## 10.1 – Anexos

Los documentos anexos o relacionados con esta Memoria se encuentran disponibles en el repositorio público del proyecto RETEMANCOSI. Con el objetivo de mantener autocontenido el grueso de la documentación, a continuación se incorporan directamente algunos documentos de especial relevancia.

## 10.2 – Guía de estilo

La guía de estilo utilizada para el desarrollo de código de este proyecto es un conjunto de directrices presentes en variedad de guías de estilo ampliamente utilizadas por multitud de empresas o comunidades de desarrolladores de software:

- Google JavaScript Style Guide.
- jQuery Core Style Guidelines.
- Principles of Writing Consistent, Idiomatic JavaScript.

El objetivo que se persigue con esas directrices es la de realizar un código homogéneo, de fácil lectura y mantenimiento por otros profesionales. Generalmente tienen más sentido en ambientes donde conviven más de un programador, pero incluso para realizar trabajos individuales resulta conveniente tener una guía que seguir. Aunque las guías de estilo suelen son más extensas, se han ignorado aquellos aparatados que no se tuvieron que aplicar.

| **Directriz** | **Explicación** | **Ejemplo** |
|---|---|---|
| D-0 | Usa `const` para todas tus referencias constantes; evita usar solo `var`. | `const valor_constante = 1;`<br>`var variable = valor_constante + funcion();` |

| | | |
|---|---|---|
| D-1 | Crear los objetos con llaves. | `var objeto = {};` |
| D-2 | Crear las matrices con corchetes. | `var matriz = [];` |
| D-3 | Usar el método `Array.push`, en vez de asignación directa, para agregar elementos a una martiz. | `matriz.push(nuevo_elemento);` |
| D-4 | Usar JSON para copiar matrices siempre que no incluyan objetos no serializables. | `var copia = JSON.parse(JSON.stringify(original));` |
| D-5 | Usar comillas simples `''` para las cadenas de texto | `const texto = 'Hola Epi';` |
| D-6 | Dejar un salto de línea antes y después de cada llave de bloque. | `function negarlo_todo()`<br>`{`<br>`  return false;`<br>`}` |
| D-7 | Usar llaves con todos los bloques, tanto de una como de múltiples líneas. | `if(test)`<br>`{`<br>`  funcion(1);`<br>`}` |
| D-8 | Evitar los espacios antes de los paréntesis de apertura en las sentencias de control (if, while, etc.) ni en las invocaciones y declaraciones de funciones. | `var fuerza = 10;`<br><br>`while(0 < (--fuerza))`<br>`{`<br>`  acum += funcion(1, 2);`<br>`}` |
| D-9 | Separar a los operadores y los argumentos con espacios. | `var fuerza = exp(3, 5) + 1;` |
| D-10 | Colocar las comas al final de las líneas cuando sea necesario. | `var yo = {`<br>`  'nombre': 'lolo',`<br>`  'edad': 31,`<br>`  'asiste': false`<br>`};` |

| | | |
|---|---|---|
| D-11 | Colocar los puntos y coma al final de todas las instrucciones aunque no fuera necesario. | `if(0 < valor)`<br>`{`<br>`  repetir();`<br>`}` |
| D-12 | Usar `undescore_case` al nombrar objetos, funciones e instancias. | `const valor_constante = 1;`<br>`funcion_especial();` |
| D-13 | Identar el código anidado con tabulaciones respetando el sangrado. | `if(0 < valor)`<br>`{`<br>`\tabreturn false;`<br>`}` |
| D-14 | Anteponer las constantes a las variables en las comparaciones. | `if(false == valor)`<br>`{`<br>`  funcion(1);`<br>`}` |
| D-15 | Usar el operador de preincremento o predecremento. | `var a;`<br>`for(a = 0; a < 10; ++a)`<br>`{`<br>`  funcion(a);`<br>`}` |

*Tabla 10.9.* Guía de estilo.

## 10.2 – Material utilizado

A continuación, se muestran fotografías del material utilizado durante el proyecto.

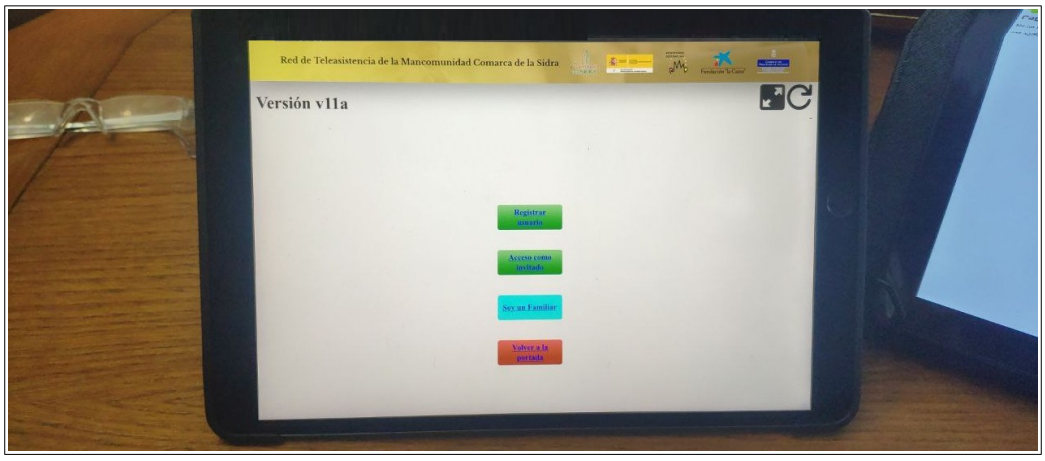


*Figura 10.15.* Tablet de 10 pulgadas enviadas a los profesionales.

*Figura 10.16.* Tablet de 11 pulgadas para pruebas en dispositivos portátiles.

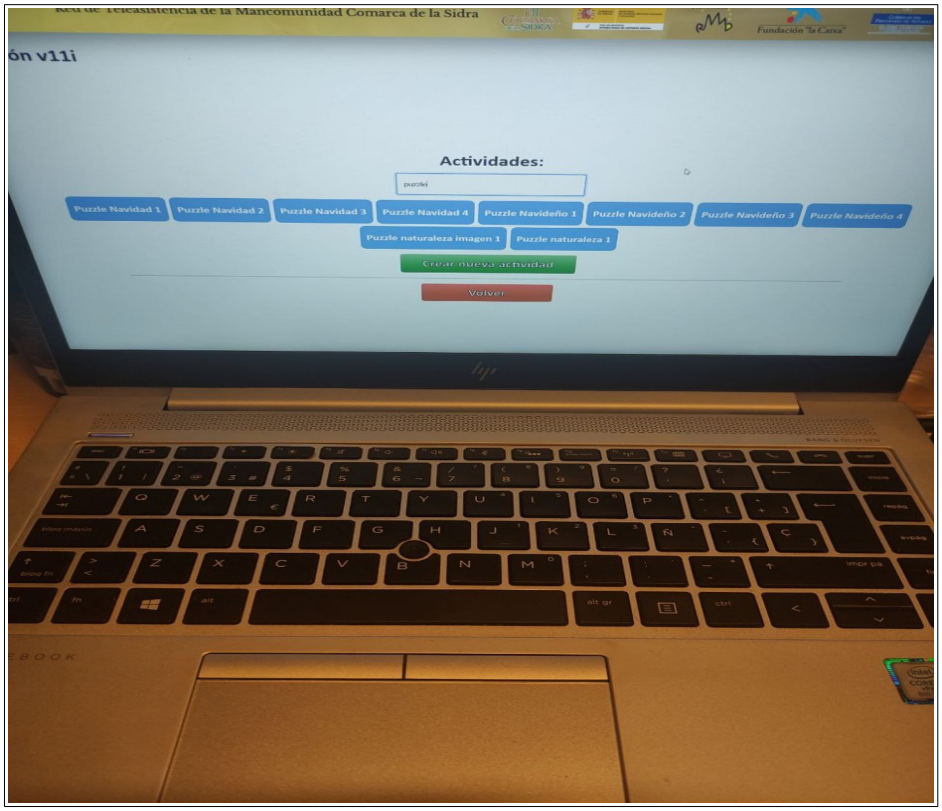


*Figura 10.17.* Equipo de escritorio con teclado y ratón para diseñar actividades genéricas.

## 10.3 – Código fuente

El código fuente del módulo desarrollado, así como el resto del código de la herramienta de teleasistencia, está liberado. Como se explicaba en la fase de IMPLEMENTACIÓN, el código fuente que conforma un módulo está separado en paquetes. A continuación se mostrará el nombre de cada paquete. Junto al nombre aparece el tamaño del paquete en bytes y el resultado de calcular su huella digital con el algoritmo **CRC**[36].

- `/html` (`E25B81E0 / 91.687 btyes`)
    - Aquí se incluyen los ficheros de plantillas que corresponden con los elementos HTML que incorpora el módulo.
- /`css` (`9741757A / 88.722 btyes`)
    - Aquí se incluyen los estilos que aplicarán a los elementos del módulo.
- `/js` (`A02A08ED / 566.588 btyes`)
    - Aquí se incluyen las funciones en JavaScript que conforman el módulo.
- `/img` (`25BF0196 / 920.960 btyes`)
    - Aquí se incluyen los recursos gráficos consumidos desde el módulo.

El acceso al código se realiza a través del enlace al repositorio disponible en los recursos de RETEMANCOSI que redirige directamente al control de versiones SVN de la herramienta de teleasistencia en producción. La versión cabecera (*HEAD rev*) en el momento de la finalización de esta memoria es la número `780`.

## 10.4 – Manual de usuario del módulo de aplicaciones genéricas

El módulo de aplicaciones genéricas es un complemento para la herramienta de teleasistencia PAWI CONECTA y sucedáneos. Una vez instalado, podrás crear y editar actividades de tipo "*Actividad genérica*". Estas actividades las podrás ejecutar en tus talleres de igual manera que usas las demás actividades de la herramienta. Gracias a las actividades genéricas podrás elaborar multitud de ejercicios con un nivel de personalización superior a que habitualmente se consigue con las actividades por defecto. Además, podrás incluir cierto dinamismo a tus ejercicios.


*Figura 10.18.* Página de inicio previa al inicio con el rol de monitor.

Inicia la herramienta con tu rol de monitor. En los próximos capítulos veremos cómo sacar partido a este módulo.

### 10.4.1 – Creación de una actividad genérica

Para crear una actividad genérica tienes que seguir casi los mismos pasos que para el resto de actividades. Primero, comprueba que has accedido a la herramienta con cuenta de monitor, y que estás en la página de inicio.


*Figura 10.19.* Página de inicio vista por con el rol de monitor.

Posteriormente, accede al gestor de actividades pulsando en el botón con fondo verde que pone “Gestionar actividades”. Verás el listado de todas las actividades presentes en la herramienta. En la parte superior tendrás disponible un filtro instantáneo. Con el filtro podrás buscar un subconjunto de las actividades: escribe parte del título de la actividad o actividades que desees localizar (no distingue mayúsculas o tildes) y de manera inmediata se mostrarán solo las que coincidan con el filtro que esté escrito en ese momento.


*Figura 10.20.* Página con el listado de actividades.

En la parte inferior, justo debajo del listado de actividades mostrado, hay un botón también con fondo verde que pone “Crear nueva actividad”. ¡Púlsalo! Inmediatamente verás el menú de creación de aplicaciones . Si quieres volver atrás, simplemente pulsa el botón con fondo rojo que pone “Volver”; abandonarás el listado de actividades creadas y regresarás a la página de inicio.

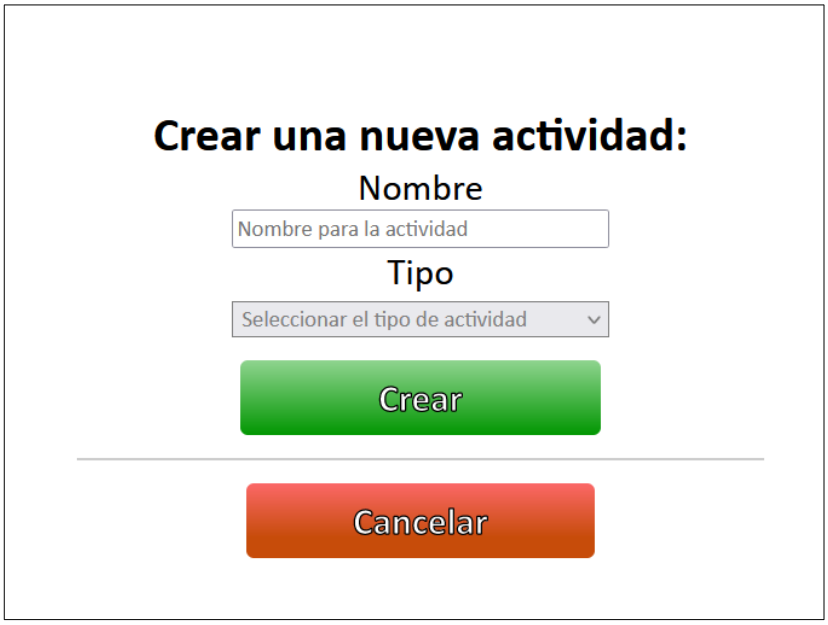


*Figura 10.21.* Página con el listado de actividades.

Ahora solamente tienes que rellenar el nombre que deseas que tenga tu actividad. ¡Recuerda que debe ser un nombre único! Una vez lo hayas escrito, presta atención al desplegable que está justo debajo de la palabra "Tipo". Tienes que seleccionar el tipo de actividad "Genérica". Es la primera opción, arriba del todo.

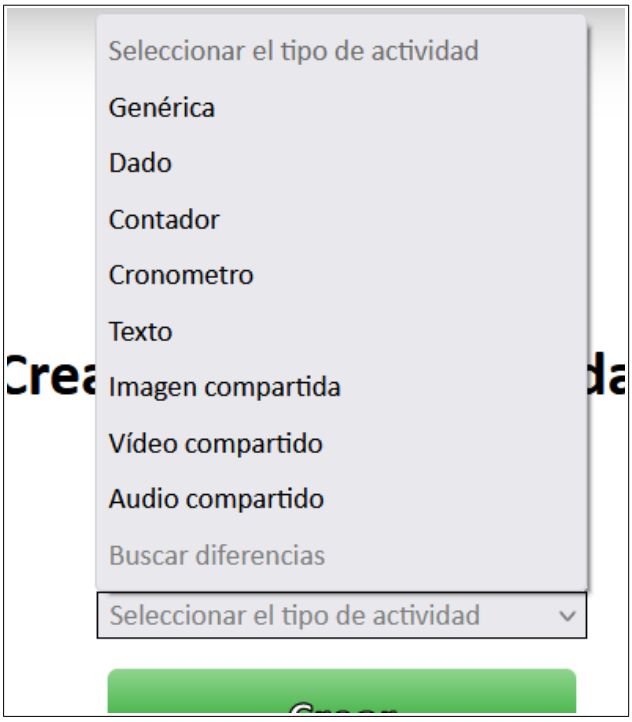


*Figura 10.22.* Diferentes tipos para las actividades.

Comprueba que ha quedado seleccionada ese tipo de actividad y revisa el nombre. Si todo es correcto, pulsa en el botón sobre fondo verde que pone “Crear”. Pasados unos instantes, si no hay ningún error, verás la ficha de datos de la actividad que acabas de crear. Por el contrario, si por alguna razón hubiera ido algo mal, te aparecerá un error. En caso de que no sepas a qué se debe el error o cómo solucionarlo, ponte en contacto con el supervisor de la herramienta para informarle de lo que ha ocurrido.

### 10.4.2 – Editando una actividad genérica

Siempre que crees correctamente o pretendas editar un actividad genérica seleccionándola del listado de actividades, te aparecerá la ficha de datos de esa actividad.

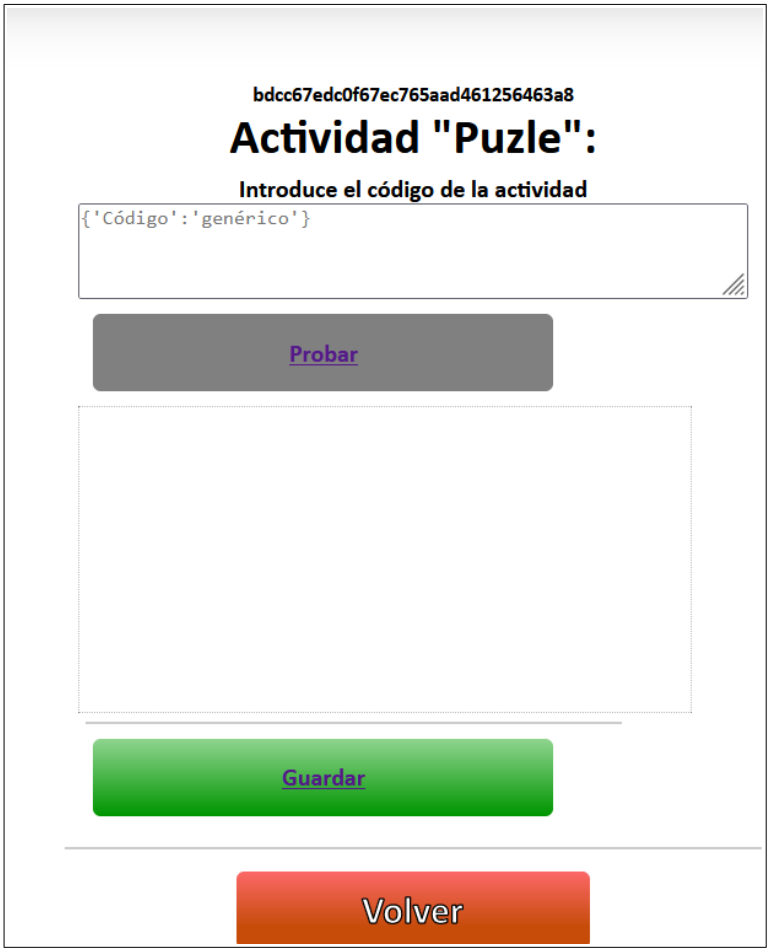


*Figura 10.23.* Ficha de datos de una actividad genérica sin código asociado.

Fíjate de arriba abajo. En la parte superior, verás un código compuesto de números y las letras de la “a” a la “f”. Ese código es el identificador único de la actividad. Todas las actividades, sean genéricas o no, tienen ese atributo. ¿Recuerdas?

Una fila más abajo aparece en grande el nombre de la actividad entre comillas. Esto también ocurre en el resto de actividades; así que ninguna novedad.

Lo nuevo empieza ahora… ¡fíjate más abajo! Debajo de la frase “Introduce el código de la actividad:” hay un campo de texto. Si la aplicación la acabas de crear aparecerá vacío. ¡Sí!, está vacío aunque aparezca con letras grises la enigmática frase “{'Código':'genérico'}”; no te preocupes, esto simplemente es una indicación para que sepas que ahí irá el código genérico de tu aplicación genérica. Mientras no añadas nada en ese campo, la actividad mantendrá ese atributo vacío. Si por el contrario has modificado una actividad para incluir algún código genérico, su código aparecerá en ese campo de texto con letras negras.

Dicho esto, ahora toca el turno de poner ahí algo. Si quieres, puedes dar rienda suelta a tu creatividad y escribir en ese campo de texto directamente. Para ello, selecciona el campo de texto y escribe usando el teclado. Presta atención a las reglas que hay que seguir para construir una actividad genérica; tendrás más información más adelante.

Por el contrario, si no quieres complicarte, puedes importar un ejemplo de actividad desde un modelo que tengas por ahí guardado. Si no tienes, más adelante verás alguno. Para ello, selecciona el código en el origen, cópialo en el portapapeles (igual puedes usar el atajo de teclado CONTROL+C) y dirígete de nuevo a la herramienta. Selecciona ese campo de texto de entrada y presiona CONTROL+V para pegar el contenido; también lo puedes pegar haciendo clic con el botón secundario de tu ratón sobre el campo de texto y seleccionando la opción de “Pegar”.

Una vez tengas el código genérico en el campo de texto, puedes pulsar en el botón sobre fondo gris que pone “Probar”. Si todo es correcto, aparecerá tu actividad genérica en el recuadro justo debajo del botón que acabas de pulsar. Haz alguna prueba con una actividad sencilla para comprobar que todo está bien. Ten presente que las proporciones se mantendrán independientemente del tamaño de la pantalla.

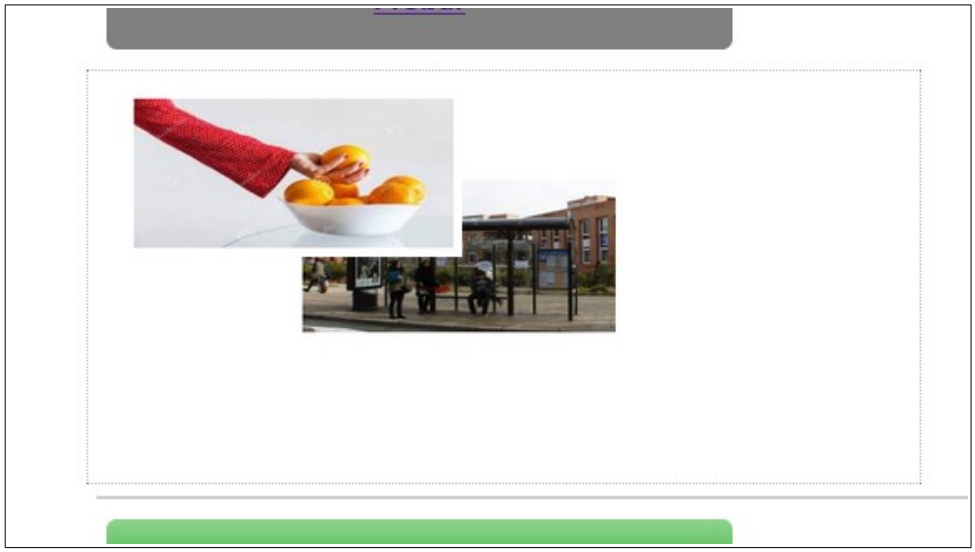

*Figura 10.24.* Área de pruebas mostrando una actividad genérica con imágenes superpuestas.

Si por alguna razón no aparece tu actividad genérica y, en cambio, ves un mensaje de error… intenta comprenderlo. Si no has practicado aún mucho con este módulo, es posible que hayas cometido algún fallo en el proceso de redacción del código o durante su importación. ¿Has duplicado alguna parte? ¿Falta algo? Si no ves el error, ponte en contacto con el supervisor de la herramienta para que te pueda ayudar.

Finalmente, si tras las pruebas ves todo correcto, pulsa en el botón de fondo verde que pone “Guardar” para que se actualice la actividad con ese código genérico nuevo. Al finalizar el guardado, volverás de nuevo el listado de actividades. Si hubo algún fallo, verás un mensaje antes de regresar al listado de actividades. Lo mismo… si no entiendes qué ha pasado, ponte en contacto con el supervisor de la herramienta.

Si por el contrario te arrepientes de los cambios que has ido haciendo y no los quieres guardar, basta con que pulses sobre el botón de color rojo con el texto “Volver” y todos los cambios se descartarán. Volverás al listado de actividades.

Por último, un consejo: aunque puedes confiar en la herramienta para guardar diferentes actividades genéricas con su correspondiente código, es recomendable que te hagas una copia local de las más interesantes. Para ello, selecciona el código, cópialo, y pégalo en un documento de texto en tu PC. Lo podrás importar más adelante siguiendo los pasos anteriormente descritos en caso de que una actividad se borre o se corrompa.

### 10.4.3 – Construyendo una actividad genérica estática

La construcción del código para las actividades genéricas es todo un arte. No es complicado, pero requiere cierta práctica y unas pequeñas nociones básicas sobre codificación. Aunque las actividades genéricas pueden construirse con muchas pretensiones, vamos a empezar con las actividades genéricas estáticas. Esta clasificación se hace debido a que las estáticas son actividades genéricas muy básicas.

El rasgo común en estas actividades es que están compuestas exclusivamente de elementos que se mantienen siempre en la misma posición, de manera permanente, durante toda la ejecución de la actividad genérica. El cometido práctico de este tipo de actividad genérica es colocar en ciertas posiciones dentro de la zona de actividades diferentes elementos gráficos. ¡Como si fuera un *collage*!

Podemos incluir en la zona de actividades figuras geométricas, texto o imágenes correspondientes a actividades de tipo "imagen". Para ubicar a cada elemento se tendrán que indicar las coordenadas cartesianas ("x" para la posición horizontal, "y" para la posición vertical) como una posición relativa dentro de la zona de actividades. De forma análoga, para determinar la superficie que ocupará ese elemento sobre la zona de actividades, se indicará el "ancho" y el "alto" relativo a la propia zona de actividades a través los atributos homónimos.

Vamos a explicarlo mejor con un ejemplo. Para simplificarlo al máximo, esta actividad genérica estará compuesta exclusivamente por dos figuras geométricas que se ubicarán en ciertas posiciones de la pantalla. El resultado esperado sería algo así:

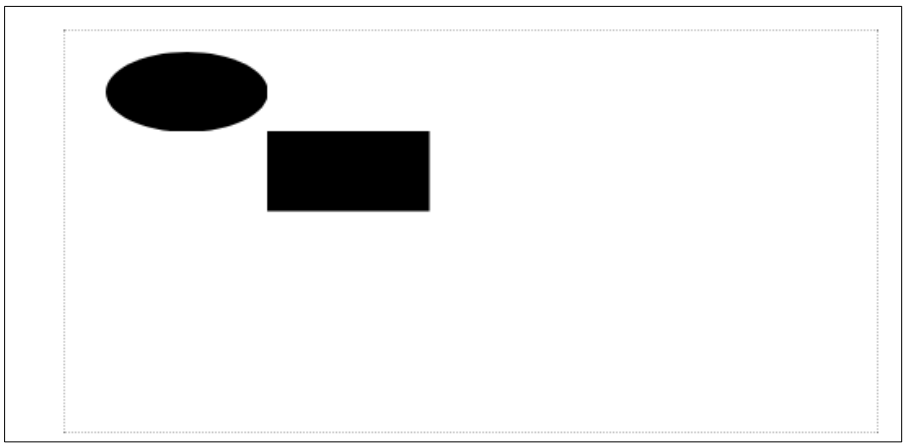

*Figura 10.25.* Resultado del código de ejemplo con dos elementos.

¿Y cómo se plasma eso en el código? Básicamente habrá que transcribir nuestro deseo de cierta forma. Para este ejemplo el código tendrá la siguiente pinta:

```
{
	"elementos":
	[
		{
			"id": "rectangulo1",
			"figura": "rectangulo",
			"x": "35%",
			"y": "5%",
			"ancho": "50%",
			"alto": "20%"
		},
		{
			"id": "circulo1",
			"figura": "circulo",
			"x": "5%",
			"y": "5%",
			"ancho": "20%",
			"alto": "40%"
		}
	]
}
```

Lo primero que hay que hacer es prestar atención a la estructura del código. Presta atención a los diferentes símbolos que hay: llaves (“{“ y “}”), corchetes (“[“ y “]”), dos puntos (“:”), comas (“,”), y a las comillas dobles. Estos elementos conforman la “sintaxis” del código, es decir, la estructura que se debe seguir para definir el conjunto de elementos que hay. Internamente el código tiene mucha más miga.

¿Y lo demás? ¿Tiene sentido, no? En el código se definen dos elementos, con identificador “circulo1” y “rectangulo1”. El atributo “figura” es “circulo” (sin tilde) y “rectangulo” (también sin tilde) respectivamente. Tal y como se explicó, los pares de atributos “x” e “y” y “ancho” y “alto” determinan la posición y superficie de cada

elemento respectivamente. Estos atributos conforman lo que se denomina "semántica" del código.

Una vez comprendido ese código, vamos a darle una vuelta de tuerca. ¡Demos una capa de pintura! Las figuras se pueden pintar con dos colores: uno para el borde (perímetro) y otro para el relleno. Mira el código de ejemplo "Rectángulos y círculos" y busca las diferencias respecto al código anterior.

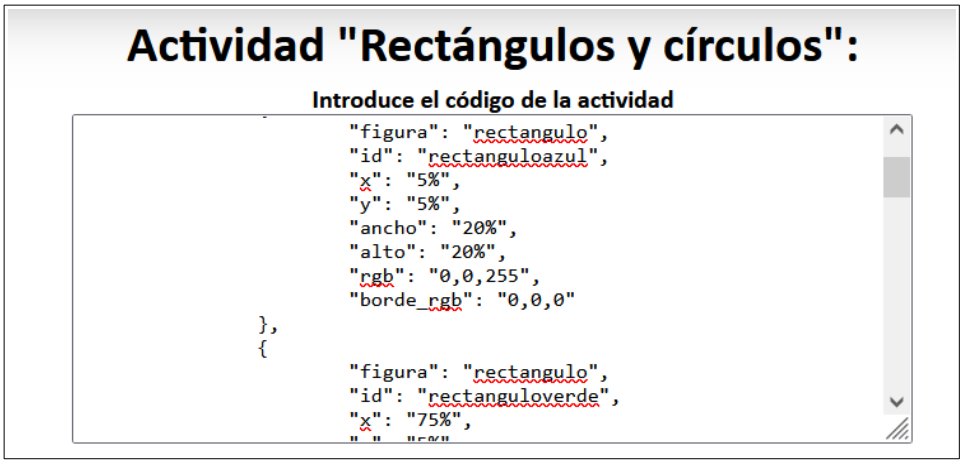


*Figura 10.26.* Porción del código de la actividad "Solo rectángulos y círculos".

Si te fijas bien, verás de primeras que se han añadido más elementos. Ahora hay seis figuras entre rectángulos (4) y círculos (2). Pero, además, en cada elemento, se han añadido dos atributos nuevos: "rgb" y "borde_rgb". Eso de RGB es una abreviatura del inglés "red", "green" y "blue", es decir, "rojo", "verde" y "azul". Básicamente lo que representa este atributo es un color definido como la suma de los valores para esas tres componentes. Revisa cada uno de los elementos del código para entender mejor cómo funciona este atributo.

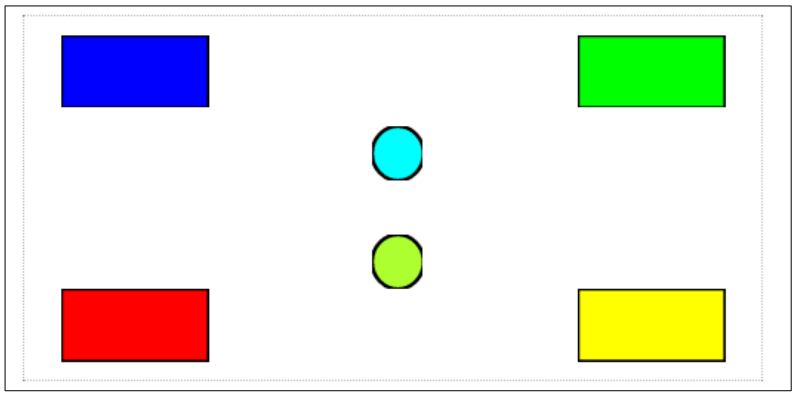

*Figura 10.27.* Resultado de la actividad "Rectángulos y círculos".

Una vez hayas llegado a este punto solo quedará por resolver el misterio de cómo se pueden añadir imágenes en vez de formas geométricas. ¡Pues muy fácil! Una vez hayas creado una actividad de tipo “imagen” con su correspondiente fichero, apunta su identificador único. Ese valor es el que se necesita para indicar que un elemento debe cargar esa imagen y no otra.

```
{
	"figura": "imagen",
	"ref": "ad8be923620fe8839bbfa8c8381ee9bd",
	"x": "5%",
	"y": "5%",
	"id": "prueba",
	"ancho": "20%",
	"alto": "20%"
}
```

Esta definición de un elemento es muy similar a la de una figura geométrica. La única diferencia es que el valor que acompaña al atributo “figura” es “imagen” y que existe un nuevo atributo “ref” cuyo valor deberá corresponder con el identificador único de la actividad de tipo “imagen” que se va a visualizar en la posición indicada por los atributos “x” e “y” cubriendo la superficie definida por “ancho” y “alto”.

*Figura 10.28.* Resultado con figura de tipo imagen.

### 10.4.4 – Construyendo una actividad genérica dinámica

Una vez que sabes cómo crear una actividad genérica estática, verás que crear las dinámicas no es algo muy complejo y, por el contrario, te facilitará nuevos mecanismos para que consigas hacer que tus actividades genéricas sean mucho más interesantes para los participantes de tus talleres. Las actividades genéricas dinámicas se dividen en tres grupos.

- Por un lado, tenemos las que son dinámicas por ubicar los elementos de manera programática. Es decir, la posición que se le asignará a un elemento no será un valor fijo, sino que se calculará a través de una operación. Podrá ser más o menos variable en función de qué operación la vaya a generar.

  Para construir este tipo de actividades genéricas lo que se hará será utilizar los atributos especiales “xxx” e “yyy” en vez de “x” e “y” para determinar de manera programática la posición de cada elemento. Una posibilidad es utilizar un rango aleatorio (entre “min” y “max” usando como unidad el “%”).

```
"id": "pieza11",
"xxx": {"min": 1, "max": 14, "u": "%"},
"yyy": {"min": 1, "max": 14, "u": "%"},
"ancho": "21%"
```

*Figura 10.29. C*ódigo con posiciones dinámicas .

- Por otro lado, tenemos las que son dinámicas porque en ellas se permite (generalmente por parte de los participantes) cambiar la posición de un elemento. Básicamente lo que se añade es la posibilidad de arrastrar con el dedo un elemento. Adicionalmente, se puede detectar si un elemento se ha arrastrado dentro, sobre o fuera de algún otro elemento.

  Para conseguirlo, puedes definir el atributos “rol” con “origen” o “destino”. Al ponerle a un elemento el rol “origen”, se podrá mover libremente. Al poner un elemento el rol “destino”, la actividad emitirá en evento de “colocar” el origen sobre el destino. Revisa la actividad “Mueve rectángulos y círculos” fijándote en esos atributos; ejecútala y prueba a desplazar un círculo sobre un rectángulo cualquiera para recibir el evento.
- Finalmente, tenemos las que son dinámicas porque se configuran para que ciertos eventos desencadenen acciones más allá del movimiento de elementos por encima del lienzo. Básicamente, al igual que los “elementos”, se define un listado de “eventos”. En ellos se definen las acciones que se ejecutarán cuando suceda un evento concreto. Mira los ejemplos avanzados para entender mejor cómo se consigue hacer todo esto.

### 10.4.5 – Controlar de una actividad genérica en un taller

Si quieres ejecutar una actividad genérica en un taller, no te comas la cabeza. Se hace exactamente igual que con el resto de actividades. Evidentemente, necesitarás estar en un taller. Así que si estás dentro, podemos empezar.

Como ya sabrás, lo primero que hay que hacer para poder iniciar una actividad (con el objetivo de que se vea bien) es activar una de las dos distribuciones de contenido que hay en la herramienta. Dirígete al menú de acciones y selecciona una distribución. Puedes usar “Horizontales con actividad” o “Verticales con actividad”.


*Figura 10.30.* Desplegable con las distribuciones disponibles.

Una vez seleccionada una de esas dos distribuciones ya podrás iniciar una actividad. Para ello, en ese mismo menú de acciones, en la parte superior, verás el listado de actividades. Selecciona la actividad genérica que quieras ejecutar en el taller en curso y dale al botón verde con el texto “Iniciar”.

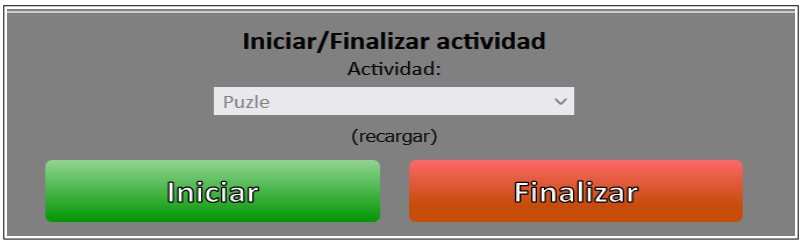


*Figura 10.31.* Desplegable y botones para controlar la actividad en curso.

En ese momento, la actividad empezará a cargarse en todos los participantes y finalmente se mostrará en el lienzo de la actividad del taller en curso. Como monitor, se te informará a medida que los participantes empiecen a cargar la actividad y de nuevo cuando cada una de esas cargas iniciadas hayan finalizado.

Si quieres finalizar una actividad en curso, simplemente regresa al menú de acciones y, esta vez, pulsa en el botón con fondo rojo con el texto “Finalizar”. La actividad en curso finalizará para todos los participantes.

### 10.4.6 – Procesado de la información recogida en una actividad genérica

Al igual que el resto de actividades, es posible que se genere información durante su ejecución. ¡Eso ocurrirá especialmente en las actividades genéricas que entre en el grupo de dinámicas! Esa información se visualizará en la ventana de registro durante el taller. Como en el resto de registros, podrás limpiar completamente esa ventana desde el menú de acciones. También podrás limpiar del listado un registro concreto pulsando sobre él.

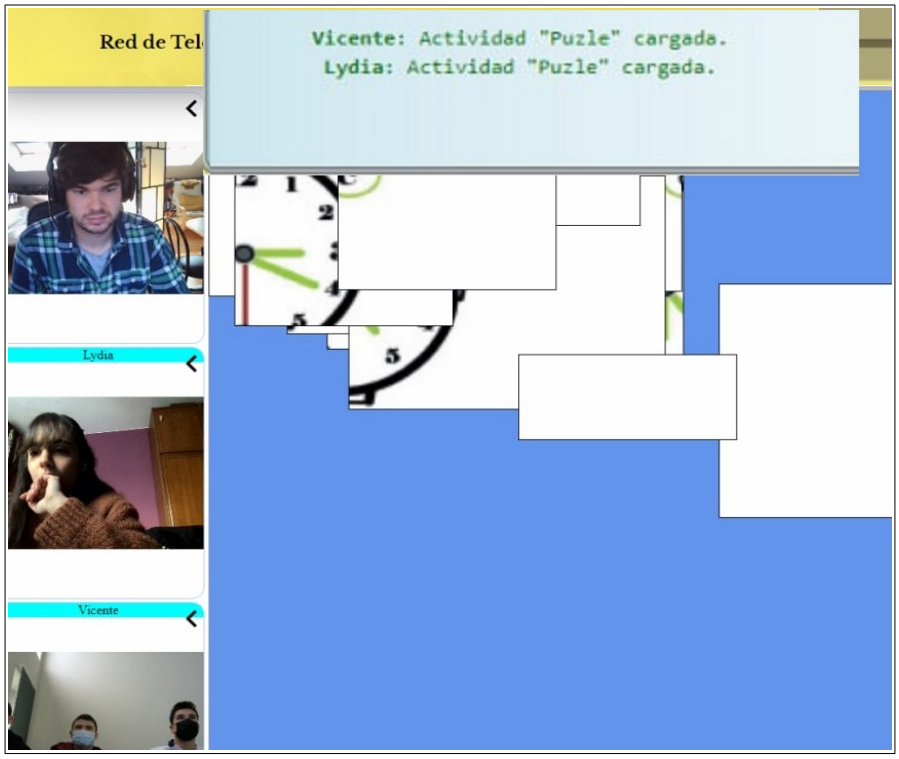


*Figura 10.32.* Visualización de los eventos recogidos en una actividad en curso.

## 10.5 – Vita

**Guillermo Álvarez** nació en 1988 en Gijón (Asturias). Inició sus estudios de música con 8 años en el *Conservatorio Profesional de Música de Gijón* bajo la modalidad de *Trombón*. Un año después, en 1997, tuvo su primer ordenador, un *Pentium 166 Mhz*. Con el cambio de milenio hizo su primera incursión en la informática creando programas básicos en *WinLogo* a raíz de un trabajo para la asignatura de Tecnología. Construye y publica su primera página web utilizando la tecnología *Flash* en el año 2003. La página implementaba una especie de red social primitiva donde se compartían contenidos entre compañeros del instituto. Comenzó sus estudios de *Ingeniería Técnica Informática* en *Inforg* del *Campus de Gijón* en el 2006. Finaliza los estudios musicales con el *Grado Profesional de Música* en el año 2010 y los estudios de *Grado Universitario en Informática* en el 2019.

Durante sus años formativos, compagina sus estudios con diferentes trabajos. Primero, en la Joven Orquesta Sinfónica del Principado de Asturias, como informático en condición de becario; más tarde en las oficinas del SGSI del Principado de Asturias; luego, como desarrollador de un variado surtido de módulos para Aplicaciones de Escritorio en C/C++ en diferentes empresas nacionales e internacionales. Todo ello sin dejar de lado la investigación en las áreas de las Aplicaciones y Servicios digitales. En el año 2016 ofrece en la EPI la conferencia "*Aplicaciones informáticas relacionadas con drones*" donde se presenta la librería EWADRONES y se realizan algunas demostraciones en directo.

Actualmente centra sus energías en reciclarse, vivir y, en cierta medida, abordar algunas de las líneas de investigación en el marco de su proyecto personal, el PROYECTO EWA, que quedaron en el tintero años atrás; y es que el tiempo no es capaz de hacer que muchas de ellas pierdan ese interés especial que algún día le mostraron. Paralelamente, con la creación de ESTUDIOS PAWI, ha colaborado con la implantación de diferentes ideas de negocio, tales como la incluida en este trabajo, en diferentes clientes que apostaron por una forma diferente de llevar la tecnología al mundo real con el objetivo de hacer la vida más fácil y alegre a la gente.

[1] Memoria de las Prácticas de Empresa
https://www.retemancosi.es/recursos/uo190887_memoria_prácticas_empresa.pdf

[2] LibreOffice
https://es.libreoffice.org/descarga/libreoffice/

[3] Reglamento TFM de la EPI
https://masterinformatica.uniovi.es/wp-content/uploads/2017/04/Reglamento_TFM_EPI-marzo-2017.pdf

[4] Normas APA actualizadas
http://normasapa.com/

[5]Repositorio del proyecto
https://www.retemancosi.es/recursos/

[6] Página web del proyecto RETEMANCOSI
https://www.retemancosi.es/

[7] Página web de ESTUDIOS PAWI SL
https://www.estudiospawi.com/

[8] Plataforma de teleasistencia utilizada por el Ayuntamiento de Zaragoza.
https://teleasistencia.es/es/

[9] Tunstall despliega soluciones tecnológicas de teleasistencia y salud.
https://www.televida.es/nuestras-soluciones/cuidados-conectados/teleasistencia/

[10] Programación visual
https://es.wikipedia.org/wiki/Programaci%C3%B3n_visual

[11] Da Rocha Padilla, J. A. (2019)
"Metodología y tecnología docente para la enseñanza de programación".

[12] Tynker, coding for kids [EN]
http://www.tynker.com/

[13] Visualino, Visual programming environment for Arduino [EN/ES]
http://www.visualino.net/ / http://www.visualino.net/index.es.html

[14] Arduino [EN]
https://www.arduino.cc/

[15] A JavaScript library for building visual programming editors [EN]
https://developers.google.com/blockly/

[16] Plan de Estudios del Máster Universitario en Ingeniería Informática
https://sies.uniovi.es/ofe-pod-jsf/web/oferta/seccion-5.faces

[17] Concepto de programación
https://es.wikipedia.org/wiki/Programaci%C3%B3n

[18] RGB, ***r**ed, **g**reen, **b**lue* (*rojo, verde, azul*)
https://es.wikipedia.org/wiki/RGB

[19] World Wide Web Consortium [EN]
https://www.w3.org/

[20] Defining Out of Scope [EN]
https://devmanagement.wordpress.com/2011/03/11/requirements-defining-out-of-scope-how-important-is-it-in-your-statement-of-work/

[21] Definición de una Caja Negra en el contexto del diseño de software.
https://es.wikipedia.org/wiki/Caja_negra_(sistemas)

[22] A light weight remote procedure call protocol. It is designed to be simple! [EN]
https://www.jsonrpc.org/

[23] JSON Escape / Unescape [EN]
https://www.freeformatter.com/json-escape.html

[24] Modelo de Objetos del Documento (DOM)
https://developer.mozilla.org/es/docs/Glossary/DOM

[25] Hojas de estilo en cascada (CSS)
https://es.wikipedia.org/wiki/CSS

[26] The most popular HTML, CSS, and JavaScript framework for developing responsive, mobile first projects on the web [EN]
https://getbootstrap.com/

[27] JavaScript (JS) es un lenguaje ligero e interpretado
https://developer.mozilla.org/es/docs/Web/JavaScript

[28] Notepad++ is a source code editor that supports several languages [EN]
https://notepad-plus-plus.org/

[29] IDE**,** *Integrated Development Environment* (Entorno de Desarrollo Integrado)
https://es.wikipedia.org/wiki/Entorno_de_desarrollo_integrado

[30] Gestión de la configuración
https://es.wikipedia.org/wiki/Gesti%C3%B3n_de_la_configuraci%C3%B3n

[31] Artistic Style 3.1 [EN]
http://astyle.sourceforge.net/astyle.html

[32]HTML Validator is a Mozilla extension that adds HTML validation inside Firefox [EN]
https://addons.mozilla.org/es/firefox/addon/html-validator/

[33]Entorno de pruebas (sandbox)
https://es.wikipedia.org/wiki/Entorno_de_pruebas_(inform%C3%A1tica)

[34] Convenio colectivo nacional de empresas de Ingeniería
https://www.boe.es/diario_boe/txt.php?id=BOE-A-2018-3156

[35] Responsabilidad Social Corporativa
https://es.wikipedia.org/wiki/Responsabilidad_social_corporativa

[36]Verificación de redundancia cíclica (CRC)
https://es.wikipedia.org/wiki/Verificaci%C3%B3n_de_redundancia_c%C3%ADclica