UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA DE SISTEMAS INFORMÁTICOS

GRADO EN INGENIERÍA DE COMPUTADORES

PROYECTO FIN DE GRADO

CERTIFICACIÓN DE PROYECTOS SOFTWARE EN ENTORNOS FERROVIARIOS SEGÚN LA NORMA CENELEC 50128

MARIO CASTELLANOS

CURSO ACADEMICO 2018/19

PROYECTO FIN DE GRADO

TÍTULO: Certificación de proyectos software en entornos ferroviarios según la norma CENELEC 50128

AUTOR: Mario Castellanos

DIRECTOR: Jesús Sánchez López

Escuela Técnica Superior de Ingeniería de Sistemas Informáticos

TRIBUNAL:

PRESIDENTE:

VOCAL:

SECRETARIO:

FECHA DE LECTURA:

CALIFICACIÓN:

Índice

Índice de tablas

Índice de ilustraciones

1. Introducción

Uno de los primeros planteamientos que nos debemos hacer a la hora de realizar un desarrollo software, es si este va a afectar a la seguridad de las personas. Si se da el caso de que la seguridad es un elemento importante (y en el ámbito ferroviario junto con la disponibilidad es el pilar del desarrollo) no podemos dejar la administración del proyecto a como se hace habitualmente, pocas pruebas que avalen la funcionalidad del sistema y un desarrollo en general poco planificado. Debido a esto a veces pasado el tiempo aparecen errores o fallos de elementos que se usan poco o que se dan con poca frecuencia.

Para evitar todos estos problemas distintos comités de normalización (IEC y CENELEC) han realizado diferentes normas con el objetivo de reducir los riesgos que se puedan dar durante la explotación comercial y/o afecten a la seguridad de operadores y usuarios del sistema.

El enfoque de estas normas siempre es el mismo, redactar unos requisitos de seguridad basado en un estudio de riesgos y trazar un camino desde esos requisitos hasta el resultado final mediante un diseño y unas pruebas. De esta forma para cada riesgo se ha diseñado y probado un elemento que cumple con el objetivo de reducir la probabilidad de que ocurre y el impacto que pudiese tener este.

1.1. Objetivos

Con este proyecto he querido crear una serie de guías sobre cómo realizar la documentación necesaria para el desarrollo de un software en un entorno ferroviario conforme a la norma CENELEC UNE 50128-2012 (EN 50158).

Mi motivación para realizar este proyecto es el haber trabajado en un proyecto en el cual se debía certificar un software conforme a la norma. Me encontré con que existían dudas que ni la propia consultora encargada de ayudar en la documentación tenía claro que hacer, con la experiencia de este proyecto quise sacar adelante un documento que sirviese para futuros proyectos.

El objetivo es marcar un camino a seguir cuando se desee realizar un desarrollo y se quiera decir que el software elaborado es seguro. Para ello se detallan una serie de procesos, planes e informes y guías generales que permiten ahorrar tiempo a la hora de estudiar los requisitos que pide la norma CENELEC UNE 50128-2012 a la hora de diseñar, desarrollar y probar un software.

También estas guías sirven para cualquier proyecto de software debido a que la estructuración del proyecto (fases, pruebas, criterios…) que se detalla en la norma CENELEC UNE 50128-2012 es similar a las metodologías estudiadas en diferentes cursos de ingeniería del software.

Otro objetivo es dar una visión global de cómo es y que fases tiene un proyecto ya aplicado y para ello especificare un caso práctico en líneas generales.

Por último, quiero añadir al proyecto una serie de guías generales y recomendaciones como resultado de la experiencia obtenida del trabajo en un desarrollo de software ferroviario aplicando la norma CENELEC UNE 50128-2012. Con el fin de facilitar el

trabajo no solo a la hora de realizar los documentos, sino a la hora de planificar y llevar a cabo todo el proyecto.

1.2. Justificación

El software es un elemento complejo donde muchas veces ni las propias personas que lo crean son capaces de defender con total seguridad cual es el comportamiento final de cada rama del código, creando elementos de incertidumbre y comportamientos que no son lo esperado. Para acotar esta incertidumbre se crean procedimientos de revisión y pruebas, se sirven métodos de fragmentación para que resulte más sencillo encontrar los errores y se crea documentación que permita ver qué modificaciones se han realizado. Con el tiempo estos procedimientos se refinan, estandarizan y se termina creando una norma oficial. Aunque el objetivo de las normas es siempre es accesible para todos los que participan en un proyecto donde debe ser aplicado, la realidad es distinta y estas normas son complejas e interpretables. Con todo esto en mente veo la necesidad de crear una guía para entender la norma y como aplicarla.

1.3. Resumen del proyecto

Con el objetivo de ayudar en el uso a la hora de usar como apoyo esta documentación este proyecto está estructurado en tres partes:

-Introducción: Aquí se detalla los objetivos, el ámbito del proyecto y una serie de elementos que son necesario para entender el resto del proyecto.

-Estado del arte: donde se explican los organismos externos que afectan a un proyecto de estas características y las normas relacionadas con el proyecto.

-Mapa conceptual: un referente para entender la propuesta de aplicación metodológica, que actores participan, su relación y que debe realizar cada uno.

-Información general: en este apartado el objetivo es tener una serie de guías que ayuden a la hora de plantear el proyecto. Entender algunos conceptos generales que se dan en la norma CENELEC UNE 50128 como la diferencia entre desarrollar un software e implementar un algoritmo en un software ya certificado o los grupos de trabajo que se necesitaran para las necesidades de independencia.

-Guías de documentación: Cada documento que se necesita para la norma tendrá asociado una guía de documentación donde se explicara los requisitos y el contenido que tendrá que incluir ese documento para que el resultado de una posterior certificación sea satisfactorio. Es importante que decir que no todos los documentos se deben realizar, por ejemplo, si estamos en el caso del desarrollo de un software no son necesarios ninguno de los documentos relativos al desarrollo de aplicaciones con software ya certificado.

-Caso Práctico: Un pequeño resumen de cómo sería llevar un proyecto SIL 3. Ver apartado de guía para entender los niveles SIL.

2. Estado del arte

2.1.Marco legal

Todo este proyecto está enfocado a seguir el cumplimiento de una norma desarrollada por una instancia independiente que permita asegurar a otra tercera que el producto que estamos desarrollando cumple con unos criterios de calidad y seguridad.

Para entender esto, es necesario comprender quién es cada actor y cuál es su función.

2.1.1. Ámbito legal

Todo lo descrito para este trabajo aplica a proyectos que quieran realizar un desarrollo seguro conforme a la normativa español que a su vez al tratarse de una norma "–EN" es conforme a la normativa europea. ("European Standards", 2019) Es importante tener en cuenta que en otros países donde la legislación no está desarrollada puede fijarse en estas normativas para aplicar.

El equivalente de esta norma (UNE-EN 50128) a nivel mundial es la norma IEC 62279, ambas tratan sobre la administración de desarrollos ferroviarios.

2.1.2. Organismo regulador

El organismo regulador es el encargado de crear una serie reglas que conformen una norma que persigue explicar cómo se debe realizar el proyecto para cumplir con la normativa vigente [Normas, 2015].

2.1.3. Organismo certificador

El certificador se trata de una empresa independiente al proveedor de servicios e instalaciones. Esta empresa debe estar acreditada por parte del organismo acreditador en este caso y para España AENOR. Esta empresa es externa al proyecto, su valoración será independiente de todos los actores del proyecto incluido el cliente final. Esta empresa tendrá el contrato de trabajo con el cliente final y no con la empresa de desarrollo.

2.2.Normas de aplicación

2.2.1. Aplicaciones ferroviarias: Sistemas de comunicación, señalización y procesamiento. Software para sistemas de control y protección del ferrocarril.

La norma en la que se centra este proyecto fue escrita por el subcomité SC 9XA, sistemas de comunicación, señalización y procesamiento y el comité técnico TC 9X, Aplicaciones eléctricas y electrónicas para ferrocarriles de CENELEC y aprobada por CENELEC el 25-04-2015. El nombre oficial de la norma es CENELEC UNE-EN 50128:2012.

El objetivo de esta norma es definir una serie de procesos y métodos que con el fin de mejorar la calidad y aplicar soluciones que buscan reducir los fallos que se puedan dar.

Pese a ello, hoy en día es imposible crear sistemas donde se garantice y demuestre de forma inequívoca la seguridad de un sistema.

La norma CENELEC UNE-EN 50128:2012 define una serie de fases y procesos a seguir muy similar a las que se sigue en la metodología de desarrollo por etapas. La norma propone una serie de fases y una verificación de cada una de las fases, con la peculiaridad de que añade una lista de procesos para cuando en lugar de crear un software se configura uno preexistente cumpliendo unos requisitos adicionales. Este desarrollo no aplica solo al desarrollo de alto nivel, sino que incluye desde el de bajo nivel hasta las herramientas de soporte y sistemas operativos. En un entorno ferroviario y en general en cualquier entorno seguro, no se pueden admitir errores que pongan en peligro la vida de las personas vengan de errores en el propio software o en el sistema operativo que lo ejecuta.

Los principales procesos que se plantean son: garantía, requisitos, diseño, diseño de componentes, implementación y ensayos, integración, validación, implantación y validación. Además, para cada proceso se añade la verificación del mismo.

Para definir el grado de integridad del sistema la norma define cinco niveles donde cero es el menos íntegro. Estos niveles se alcanzan siguiendo una serie de procesos y técnicas que en función de su combinación y uso definirá cuan de integro es nuestro producto.

La norma define también otros elementos relativos al ámbito del proyecto como los roles: las restricciones y las responsabilidades que tendrá cada uno de ellos, con el fin de asegurar unos niveles de independencia óptimos.

Norma	Descripción
Fases y procesos	Todas las fases y los procesos necesarios para desarrollar el software, también la interacción entre los procesos y las técnicas a usar.
Metodología	Modelo en V para definir las fases del proyecto.
Independencia	Definir roles y sus niveles de independencia que eviten viciar el proceso de creación y pruebas.
Trazabilidad	Permitir siempre saber dónde se origina la funcionalidad, que pruebas le afecta o en casos de cambios que es afectado.
Seguridad	Definir procesos, estratos y niveles con el objetivo poder ajustar que niveles de reducción de riesgos es necesario.
Demostración	Descripciones completas de la documentación que se debe realizar para demostrar que el proceso se ha realizado correctamente.
Técnicas y medidas	Una colección de todas las técnicas y medidas que se deben usar para hacer las distintas tareas necesarias para el proyecto en función del nivel de seguridad.

Tabla 1 Principales elementos de la norma

En resumen, la norma intenta guiar sobre cómo realizar un proyecto de forma segura y donde se pueda verificar que se han seguido los procedimientos para evitar errores inesperados o carencias en la funcionalidad segura.

2.2.2. Metodología seguida.

En la norma 50128 se sigue la metodología en "V", una versión de la metodología en cascada [Balaji, 2012], Esta metodología se basa en representar las fases gráficamente las fases del ciclo de vida del proyecto como una secuencia en forma de V. Además, en esta representación en el brazo izquierdo se descomponen las fases principales de adquisición de requisitos, desarrollo de especificaciones y diseño, mientras que en el brazo derecho se encuentran las fases de integración, pruebas y verificación. Por último, la implementación es el punto central.

Al igual que en la metodología de cascada se basa en la rigidez de las fases y las iteraciones. Pero a diferencia de esta la verificación se hace al finalizar una fase y su salida da inicio a la siguiente fase. Como ventaja esta su facilidad de uso y la integración del equipo de pruebas en las fases iniciales, sin embargo, las fases son muy rígidas y requiere muchas revisiones [Balaji, 2012].

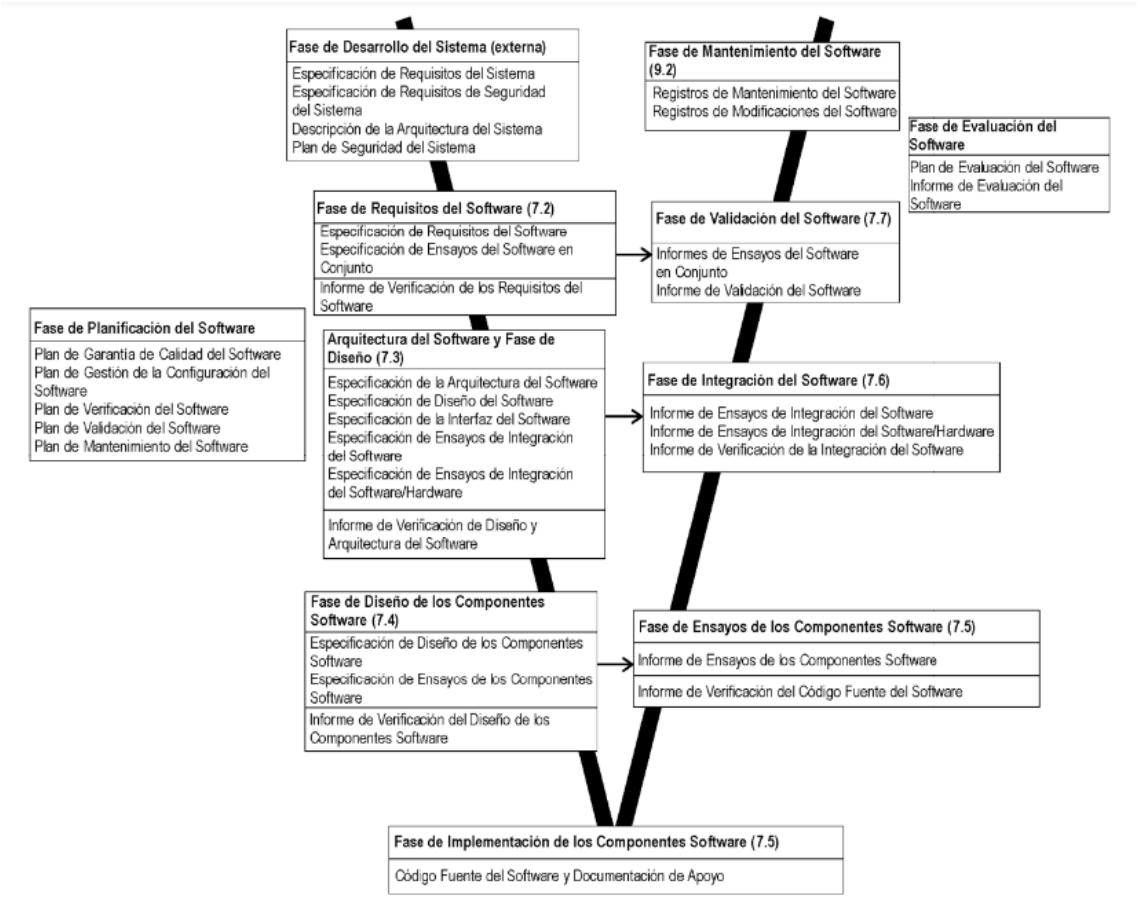

Ilustración 1 Relación fase-documento. EN 50128 figura

2.2.3. Otras normas de interés: UNE-EN 50126-1:2005 Y UNE-EN 50129:200

La norma UNE-EN 50128 (EN 50128) no es la única a tener en cuenta a la hora de desarrollar un software seguro en entornos ferroviarios. Junto a esta se debe tener en cuenta las normas UNE-EN 50126 (EN 50126) y UNE-EN 50129 (EN 50129).

-Aplicaciones ferroviarias. Especificación y demostración de la fiabilidad, de la mantenibilidad, de la disponibilidad y de la seguridad (RAMS). Parte 1: requisitos básicos y procesos genéricos:

La norma UNE-EN 50126 es una norma más general que la UNE-EN 50128 y habla de los sistemas de forma más amplia y no solo del software.

El objetivo de esta norma es detallar los elementos y procesos sirvan para reducir los riesgos relativos a las RAMS (fiabilidad, mantenibilidad, seguridad y disponibilidad por sus siglas en inglés) dentro del proyecto. Para ello detalla un ciclo de vida general y no solo del software.

Las normas definen una serie de requisitos a seguir para reducir los riesgos que se dan relativos a los elementos RAMS. Habitualmente al inicio de cualquier proyecto ferroviario se analizan los posibles riesgos que se pueden dar en los cuatro ámbitos que trata la norma (fiabilidad, mantenibilidad, seguridad y disponibilidad).

-Aplicaciones ferroviarias. Sistemas de comunicación, señalización y procesamiento. Sistemas electrónicos relacionados con la seguridad para la señalización:

Esta norma se centra en los aspectos de seguridad relativos a las señales, tanto hardware como software. Se especifican una lista de requisitos de evidencias que se deben presentar para asegurar que el sistema es seguro. También se añaden una serie de requisitos adicionales si se usa lógica programable relativa a la norma 50128.

Realizar un proyecto acorde a la norma europea 50129 generará una serie de documentación adicional entre la que se incluye principalmente:

- Evidencias de calidad
- Evidencias de seguridad
- Evidencias de funcionalidad

Esta norma también da una guía para crear los requisitos de seguridad basados en los análisis de riesgos y el control de errores.

3. Propuesta de aplicación metodológica

3.1.Mapa conceptual de los actores del proyecto

En el siguiente mapa conceptual se pueden ver las relaciones entre los distintos grupos, los rombos representan los diferentes roles, descritos en el apartado de roles. Las interfaces cliente y certificación representan los agentes externos al proyecto. La documentación que genera cada grupo está repartida en el anexo de documentos y autores.

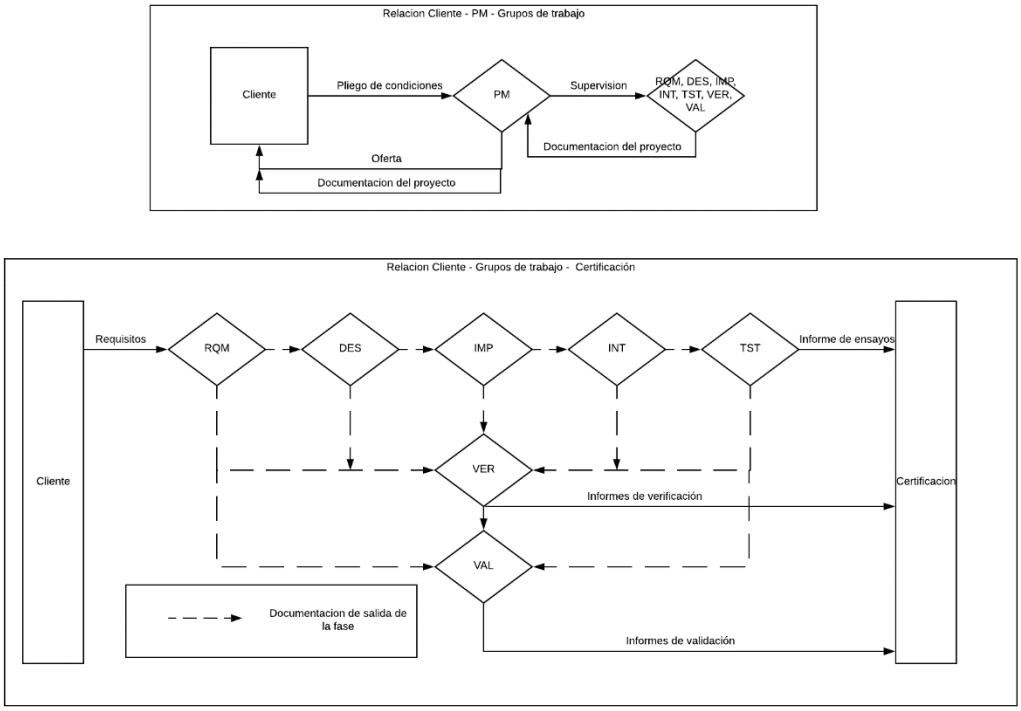

Ilustración 2 Esquema de relaciones. (Elaboración propia)

3.1.1. Orden de consulta

Cuando un jefe de proyecto empiece a planificar el proyecto le recomiendo consultar todo lo relacionado con el apartado de estado del arte (cap. 2), información general (ap. 3.2) y el anexo (1) de documento y autores.

Cuando una persona tenga que elaborar un documento y quiera consultar esta documentación, tendrá que leer previamente una serie de apartados que dan un contexto para entender el propio apartado de la guía del documento. El orden será el siguiente:

- Contenido general de la documentación (ap. 3.2.3): donde tendrá la información general que tienen que contener todos los documentos.

- Guía para entender los niveles SIL (ap. 3.2.6): de forma que sepa cómo se debe ajustar el documento.

-Apartado con el nombre del documento que debe elaborar

-Luego para redactar el propio documento del proyecto, se debe consultar la documentación de entrada que se indica.

3.2. Información general

En los próximos apartados se introducirán una serie de guías generales y de documentación. A todas estas guías se les ha tratado de dar ejemplos y simplificar en la medida del posible debido a que el todo el personal del proyecto puede encontrar utilidad en estas guías.

Los subsiguientes apartados explican elementos concretos de la norma que no atañen a un documento especifico como pueden ser los roles o los niveles SIL. El objetivo de estos apartados es aclarar términos complejos para todas las personas que participan en el proyecto y por tanto debería ser leído por todos.

En las guías de documentación cada apartado pertenece a la guía de un documento y debe ser leído por el responsable de realizar dicho documento y el verificador. Existen documentos que son muy similares y se han agrupado como pueden ser los distintos informes.

En total son seis guías generales y cincuenta y tres de documentación.

3.2.1. Roles

Para asegurar los niveles de independencia que requiere la norma es importante dividir al personal del proyecto en diferentes equipos y en para ciertas labores la independencia se debe llevar hasta que el grupo debe pertenecer a otra empresa.

Los roles principales que se necesitaran en el proyecto son:

-Jefe de proyecto (PM): Su función es asegurar que se cumplen los estándares de calidad indicados y la independencia de roles dentro del proyecto. Es su deber y responsabilidad asignar los recursos necesarios a los demás grupos para que puedan realizar su trabajo de forma óptima.

-Gestor de requisitos (RQM): Debe ser el encargado de realizar, controlar y mantener toda la documentación relacionada con los requisitos, así como llevar la trazabilidad desde los requisitos del sistema a los del software.

-Diseñador (DES): Es el encargado de transformar los requisitos relativos al software en una arquitectura y un conjunto de soluciones que cumplan con los estándares descritos en la norma.

-Implementador (IMP): Lleva a cabo la transformación del diseño a un software final. Para ello sigue las guías de codificación y diseño que se hayan creado para el proyecto.

-Ensayos (TST): Se debe encargar de realizar la especificación de ensayos y la planificación de pruebas. También debe definir el entorno de pruebas.

-Verificador (VER): Realiza el plan de verificación y revisa el resto de la documentación para realizar los pertinentes informes de verificación, los principales puntos son la que los documentos revisados son completos, coherentes, correctos y trazados.

-Integrador (INT): Es el encargado de gestionar la integración del software y los ensayos de integración.

-Validador (VAL): Realiza el plan e informe de validación definiendo las tareas para realizar la validación. Debe evaluar si el software y los procesos de realización cumplen con el nivel SIL elegido. Debe evaluar si la verificación ha sido completa.

-Evaluador (ASR): Debe evaluar si el software y el proceso del software, junto con el resto de los procesos como el de verificación y validación se han desarrollado conforme a los criterios del plan de evaluación. También debe realizar auditorías de seguridad a lo largo del proyecto.

-Gestor de la configuración (CM): Es el responsable de mantener el sistema de gestión de la configuración actualizado y realizar las Notas de versión publicadas.

En función del nivel de integridad que se quiera obtener es necesario diferentes niveles de independencia. Para ellos y extraído de la norma UNE-EN 50128 tenemos:

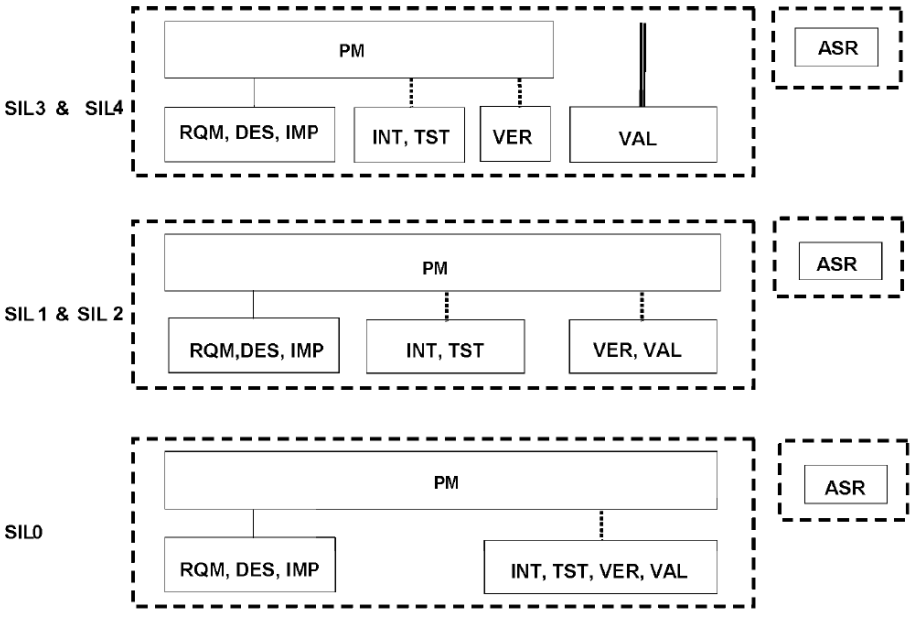

Ilustración 3 Distribución de roles. EN 50128 figura 2

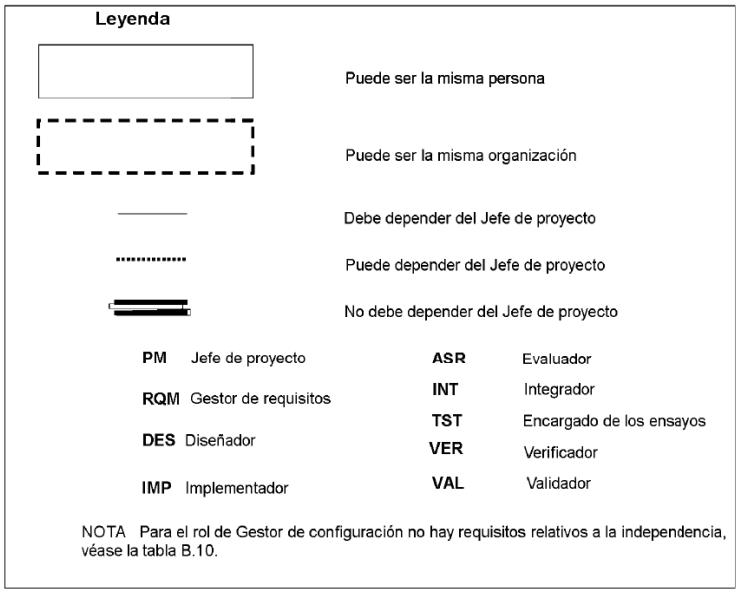

Ilustración 4 Leyenda de roles

Conforme a los descrito anteriormente y en relación a las fases de un proyecto de este tipo se puede extraer la información para completar la siguiente matriz RACI que asigna a cada rol la tarea de realizar (R), vigilar (A), ser informado (I) o ser consultado (C).

RACI	PM	RQM	DES	IMP	TST	VER	INT	VAL	ASR
Requisitos	AI	R	-	-	I	I	-	I	-
Diseño	AI	C	R	I	-	I	-	I	-
Desarrollo	AI	-	C	R	-	I	-	I	-
Testing	AI	C	-	-	R	I	R	I	-
Verificación	AI	C	C	C	C	R	C	IC	C
Validación	AI	C	-	-	C	I	C	R	I
Evaluación	AI	-	-	-	-	I	-	I	R

Tabla 2 Matriz RACI

En esta tabla no se incluye el gestor de la configuración (CM) debido a que su labor es secundaria y su trabajo es según sale versiones de software.

Este tipo de matrices son comunes en los ámbitos de calidad y se usan para repartir las tareas a cada persona o grupo que participa en un proyecto. [RACI, 2016]

3.2.2. Subcontratación

En un desarrollo de esta magnitud es habitual, al margen de los requisitos de independencia, contratar a una empresa especializada en hacer una parte del proyecto (seguridad, diseño, validación…) es importante tener en cuenta que esta empresa no puede ser una caja negra a ojos del jefe de proyecto. Debe entender y estar informada de

las otras partes del proyecto, usar las mismas herramientas y formatos que se usen en el resto del proyecto. También hay que tener en cuenta que todos los documentos que se realicen tienen que estar firmados, si una subcontrata abandona el proyecto antes de su finalización y entrega, estos documentos quedarán huérfanos y tendrán que replantearse dando trabajo adicional.

Por la experiencia obtenida en proyectos de este ámbito desaconsejo subcontratar una parte vital como la de los ensayos, no controlar la planificación de pruebas y realización de pruebas puede provocar serios retrasos a la hora de realizar un proyecto.

3.2.3. Contenido general de la documentación

Todos los documentos que no sean anexos y que sean relativos a la certificación del producto deben de tener una serie de características generales. Estas características son una guía, la información debe mostrarse, pero el formato puede diferir a lo expuesto aquí.

Portada: La portada debe incluir toda la información referente a la gestión de la documentación y un cajetín de cambios. Por ejemplo:

VERSIÓN	DESCRIPCIÓN DE LA REVISIÓN	FECHA	NOMBRE REV.	NOMBRE APROB.	FIRMA APROB.
CAJETÍN DE REVISIONES					

Tabla 3 Cajetín de revisiones

Nombre Proyecto				
CÓDIGO DE DOCUMENTO:				
DENOMINACIÓN DEL DOCUMENTO:				
NOMBRE DEL CONTRATO:		ARCHIVO INFORMÁTICO:		
Hoja:		Fecha	Nombre	Firma
1/19	Redactado			
	Revisado			
	Aprobado			

Tabla 4 Portada

Introducción: Todo documento debe incluir un alcance donde se explique el propósito general del proyecto y los objetivos más específicos del documento y la audiencia a la que va dirigida.

Gestión de cambios: A parte de lo incluido en la portada el documento debe contener un apartado de evolución del documento que incluya los cambios que se han ido realizando desde su concepción, esto es una versión más extendida y en profundidad que lo expuesto en la portada.

Una lista de acrónimos y definiciones que se usen en este documento. Puede existir uno general a nivel de proyecto.

Guías y referencias: Este es un apartado importante, ya que muestra la trazabilidad entre documentos. La forma en la que se referencian los documentos tiene que estar bien realizada, si se referencia la versión de un documento y este cambia, esto implica que se tiene que actualizar este documento. Por ello recomiendo que se separe en tres apartados:

-Guías y referencias: donde la versión no es importante, por ejemplo, si estamos en el plan de verificación y referenciamos el informe de verificación, no es necesario saber la versión del informe porque es un documento que se realizará en el futuro.

-Referencias de entrada: donde la versión es importante, por ejemplo, si estamos en el informe de verificación y cambia el plan de verificación es obligatorio revisar el informe porque depende del plan para realizarse.

Todos los informes de verificación deberán contener la siguiente información que se describe en los próximos puntos.

Pese a que en este documento solo se necesita exponer los resultados de la verificación, siempre que sea posible añadir la evidencia de por qué se ha llegado a ese resultado facilitará la labor de validación del sistema.

Para cada objetivo a verificar habitualmente se crea una tabla del siguiente estilo:

Documento	Código	Versión	Objetivo	Conforme	Comentarios

Tabla 5 Verificación

En este informe se debe reflejar que los documentos a verificar cumplen con los objetivos generales descritos a continuación:

-Que el documento está trazado mediante un número de referencia único (lo más sencillo posible).

-Que los términos y acrónimos usados en el documento son acordes a los usados en el resto de la documentación. También los elementos o conceptos deben ser comunes a todos los documentos.

-Que aplica todas las relaciones y condiciones que impone el documento que le precede jerárquicamente. Además, el documento no contradice a dicho documento.

-Se debe atestiguar que los requisitos están siendo correctamente trazados debido a su importancia en todo el ciclo de vida. Para ello debe seguir la siguiente cadena:

-Requisitos respecto a al diseño u otros objetos que los sustituyan

-Objetos de diseño a objetos de implementación

-Requisitos y diseño en relación a ensayos.

-Si un elemento no puede trazarse debe quedar constancia que no afecta a la seguridad, siendo la labor del verificador repasar dichas argumentaciones.

-La coherencia interna del documento.

El informe debe contener un apartado donde se deje expuestas las observaciones, incidencias y evaluación de los resultados de la evaluación de la verificación de este informe. Todo esto formaría las conclusiones generales del informe.

En las conclusiones se debe hacer referencia a que se ha hecho la verificación y se ha obtenido los resultados apoyándose en la norma 50128.

Todos los informes de ensayos deberán contener la siguiente información que se describe en los próximos puntos.

Los informes deben contener la siguiente información:

-Objetivos del informe. Este campo es similar al que se encuentra en la especificación de ensayo correspondiente.

-Casos de ensayos y de qué tipo son los que se han realizado en el informe. También los datos necesarios para realizarse y los resultados que se esperan

-Entorno de pruebas. Se trata de una descripción de los elementos necesarios para realizar los ensayos.

-Requisitos cubiertos con el ensayo

-Nombre de los encargados del ensayo

Para cada ensayo se debe añadir quién es el responsable del ensayo. El responsable debe incluir el resultado, si se han cumplido los objetivos y si se han seguido los criterios de ensayos en la especificación, en caso de alguna desviación se debe dejar escrita con lo sucedido y cómo ha afectado al ensayo.

3.2.4. Desarrollo del software contra desarrollo de un algoritmo y/o datos de un software previamente certificado

Dado que el ámbito del proyecto no es un entorno informático y por tanto el personal no está familiarizado con el software, en las primeras fases del proyecto se debe tener en cuenta y aclarar sobre la documentación si se está desarrollando un software o usando uno previamente certificado y solamente configurando un algoritmo y/o datos de entradas.

Como ejemplo de esto segundo sería realizar el proyecto con PLCs de Siemens y la aplicación SIMATIC S7 (Beresford, 2011). Esta aplicación requerirá que se cumplan unas condiciones previas dadas por el fabricante.

En el caso de desarrollar el software no se tendrá que aplicar el punto 8 (desarrollo de los datos o algoritmos de aplicación: sistemas configurados mediante datos o algoritmo de aplicación) de la norma CENELEC 50128 y si se usa uno certificado no se aplican los puntos 6 y 7 (garantía del software y desarrollo de software genérico). Adicionalmente,

que no se requiere un plan de calidad del software, no implica que el proyecto no disponga de un plan de calidad general conforme a la CENELEC 50126.

Las ventajas e inconvenientes que nos ofrece realizar el proyecto basado en una arquitectura de PLCs o un software ya certificado son las siguientes:

Ventajas	Inconvenientes
El certificado de una gran multinacional aporta seguridad al cliente	Menor libertad de actuación
Menor documentación	Poca diferenciación de la competencia
Gastos menores	
Facilidad para encontrar desarrolladores además de ser más fácil sustituirlos al ser un programa común y existen cursos de bajo coste.	

Tabla 6 Ventajas e inconvenientes de los PLCs

3.2.5. Documentación del proyecto y documentación del software

Es importante entender la diferencia entre un documento que hable del software y uno que hable de todo el proyecto para evitar malentendido y malas planificaciones.

Pueden existir dos documentos que sirvan para lo mismo, uno a nivel de proyecto y otro a nivel de software. Por ejemplo, el ciclo de vida: existe un ciclo de vida del proyecto y otro del software, es posible que el segundo ciclo completo sea una fase del primero.

También es posible que información de software esté contenida dentro de documentación del sistema. Por ejemplo, los requisitos: los requisitos del software se pueden incluir en los requisitos del sistema, pero deben estar bien diferenciados del resto de requisitos. En otros ámbitos no se puede mezclar, el plan de garantía puede hablar del plan de calidad del software, pero este debe existir y ser específico.

Por último, en los próximos puntos se hablará de guías de documentación, en todo momento se habla de guías de documentación de software.

3.2.6. Guía para entender los niveles SIL

Los niveles SIL (Nivel de Integridad de Seguridad o Safety Integrity Level). Van desde 0 (nivel inferior) hasta el 4 (nivel superior).

Antes de empezar el proyecto si no es un requisito del cliente se debe evaluar los riesgos, integridad y seguridad del sistema para elegir el nivel SIL correcto.

A lo largo de la norma, se describen una serie de técnicas y medidas y se les asignan unos niveles de recomendación según el nivel SIL, estos niveles de recomendación son los siguientes:

- Mandatory (M): La técnica es obligatoria
- Highly Recommended (HR): Si no se usa una técnica o medida considerada altamente recomendada para un nivel de seguridad SIL se debe proporcionar una justificación detallada del motivo por el que se han elegido técnicas o medidas alternativas. Esto debe quedar reflejado en el plan de garantía del software o en otro documento que lo referencie.
- Recommended(R): Se trata de un nivel inferior a HR y se puede combinar con estos para crear un paquete.
- N/S (-): No existen recomendaciones a favor ni en contra de la técnica o medida.
- No Recommended (NR): Al contrario de HR se debe justificar el uso de esta medida de la forma que en un HR.

3.3. Guías de documentación

3.3.1. Planes

3.3.1.1. Plan de evaluación del software. PL10

En este documento se deben detallar los objetivos del plan y de los procedimientos para realizar la evaluación del proyecto.

El objetivo de la evaluación es saber si los procesos del ciclo de vida han sido los correctos y si sus salidas son las correctas para el nivel SIL elegido y su uso. En este plan se deben detallar los objetivos, procedimientos y formas en las que se realiza la evaluación.

Si existen informes de evaluación previos el evaluador debe revisar si se han cumplido los requisitos de independencia en cuyo caso no es necesario emitir nuevos informes de evaluación del software preexistente.

En el plan se deben detallar la formación necesaria que debe requerirse para realizar las tareas de evaluación. En caso de que exista a nivel de empresa esta figura y se detalle estos requisitos se podrá usar ese documento como especificación de requisitos del evaluador.

Para cumplir con los requisitos de independencia el evaluador debe ser independiente al proyecto y debe recibir autorización por parte del jefe de proyecto de realizar la evaluación conforme a este plan de evaluación y emitir un informe de evaluación.

El plan debe definir los aspectos que trate la evaluación. Esto se puede dividir en dos aspectos:

- Aspectos formales: Si el ciclo de vida, fases y procesos han sido los adecuados y se han seguido correctamente o por contrario ha sido necesario realizar ajustes y el motivo de los mismos.
- Características: La solución final que ofrece el software se ajusta y cumple las expectativas sobre las que se elaboró. Un software puede cumplir con todos los requisitos de seguridad, pero el resultado no ser óptimo o se ajusta a lo que espera el cliente.

También se debe definir las actividades que se realizarán (revisión de plazos, resultados obtenidos, comparación entre los planes y los informes…) y sus plazos dentro de la organización temporal del proyecto.
Se deben definir los criterios de aceptación y la forma de trazar los casos de no conformidad y aceptación/rechazo. Aunque la norma no especifica esto, se sugiere la siguiente guía de modelo:

- No Conformidad: Se recomienda un formato simple con el evaluador, elemento y marco que se ha evaluado y motivo de la no conformidad. Si es posible que el motivo sea acotado al incumplimiento de un criterio de aceptación definido en este plan o al incumplimiento de un punto de la norma, es recomendable que se especifique estos puntos.
- Aceptación/rechazo: Para cada elemento evaluado se debe dar el nombre del evaluador, el veredicto y motivo de la evaluación. También es recomendable mostrar las evidencias si existen del cumplimiento de los criterios de evaluación.

3.3.1.2. Plan de Garantía de calidad del software. PL20

Para comenzar el plan de garantía de calidad del software se deberá referenciar al plan de calidad del proyecto donde vendrá definido el análisis del proyecto.

Posteriormente si se decide dividir el plan de garantía en varios documentos como recomienda la norma se deberá referenciar a todos estos apartados.

El plan debe contener una lista de todos los documentos a verificar y un organigrama del grupo de verificación o referenciar al documento que contenga esta información. También debe añadir un apartado explicando la necesidad de evaluar los resultados de verificación e indicando los objetivos que busca la evaluación.

Entre todos los informes que define la norma es importante hacer un especial seguimiento de los requisitos de seguridad

Se puede seguir la ISO 9001 o tomar como guía para la redacción de este documento, a pesar de que se centra más en los procesos de servicios se pueden encontrar similitudes y adaptaciones al mundo del desarrollo software. [Coallier, 1994]

Anexo 1: ciclo de vida

En este documento se debe exponer las diferentes fases que debe atravesar el proyecto. Para cada fase del software se debe definir los siguientes elementos:

Elementos de una fase
Documentación de entrada
Documentación de salida
Actividades y su relación con la documentación de sistema
Actividades de calidad
Principales actividades

Responsable de cada actividad

Tabla 7 Elementos de una fase

Aunque las fases dependen directamente del tipo de proyecto un ejemplo de fases de un proyecto podría ser:

Número	Fase	Descripción
0	Análisis de riesgos	Debido a la necesidad de seguridad que requiere el proyecto es necesario prever con antelación cuáles serán los riesgos del proyecto
1	Especificación de requisitos	A partir del pliego del proyecto se deben realizar unos requisitos de software.
1.1	Especificación de requisitos de seguridad	Para todos los riesgos encontrados se deben definir unos requisitos de seguridad que su funcionalidad anule dicho riesgo
2	Especificación funcional	Esta actividad trata de ampliar y definir todos los aspectos funcionales de los requisitos
3.0	Especificación de arquitectura	Definición de la arquitectura hardware que necesita nuestro software, así como el funcionamiento de seguridad entre el hardware y el software
3.1	Diseño del software	Se debe diseñar todas las funcionalidades descritas en la especificación funcional
3.2	Diseño de interfaces	Si nuestro sistema tiene que interactuar con otros se recomienda tratar estos diseños en una fase independiente
3.3	Diseño de componentes	En función del SIL se deben diseñar unos componentes que satisfacen la especificación de diseño
4.0	Ensayo de componentes	Pruebas para corroborar que cada componente resuelve los requisitos asociados tal y como dice el diseño de componentes
4.1	Ensayos de integración	En estas pruebas se debe demostrar que cada interface cumple con el cometido para el que se ha realizado.
4.2	Ensayos de integración HW/SW	Se debe probar que el software funciona de forma adecuada en el hardware diseñado
5	Validación del sistema	El validador debe realizar un subconjunto de las pruebas anteriores como parte de la actividad de validación
6	Aceptación del sistema	El validador debe realizar un subconjunto de las pruebas anteriores como parte de la aceptación del proyecto
7	Evaluación	En esta fase se debe evaluar la evolución del proyecto y sus diferentes fases

Tabla 8 Fases del software

Pese a que la verificación no es una fase en sí misma, para cada otra fase excepto para la fase de aceptación del sistema se debe verificar las actividades de dicha fase.

Anexo 2: Árbol documental

Dado la importancia de tener clara la documentación necesaria para el proyecto se ha creado una guía propia para estos documentos: plan documental y árbol de dependencias.

Anexo 3: Herramientas, técnicas y medidas

La norma 50128 define que el plan de garantía de calidad del software debe especificar las herramientas, técnicas y medidas acorde a su nivel de integridad. Dado que en el plan de verificación se han de tener también las técnicas para verificar se han aunado ambos ficheros en uno y se ha creado una guía para tal efecto: Herramientas, técnicas y medidas.

Anexo 4: Seguimientos e incidencias

Pese a todos los controles que da la norma 50128, adicionalmente es necesario crear un seguimiento periódico de diferentes actividades. Esto es debido a que habitualmente los proyectos son entes vivos que intentan cambiar (a peor mayormente) durante el transcurso de las fases. Por ello recomiendo el siguiente seguimiento periódico:

Control	Periodicidad
Actividades de calidad	Trimestral
Actividades de pruebas	Semanal
Actividades de verificación	-
Desarrollo (interno del grupo de trabajo)	Diario/Semanal
Desarrollo (externo)	Mensual
Otras actividades documentales	Quincenal
Diseño (interno del grupo de trabajo)	Semanal
Diseño (externo)	Quincenal

Tabla 9 Control de seguimientos

Anexo 5: Control de la documentación

Es necesario dejar detallado un procedimiento de control de la documentación, para ello se define una serie de elementos que deben encontrarse en cada documento. La norma 50128 define tres aspectos que se deben dar en los documentos para realizar el control de la documentación: roles de los implicados en la redacción, aprobación y revisión del documento; campo de aplicación y archivo. Pese a que la norma no identifica más campos es recomendable ampliar esto con lo que se expone en otras normas como la ISO 9001, por ejemplo, un cajetín de cambios y versión del documento.

El objetivo de estos anexos es que todas las personas que participan en el proyecto tengan acceso a los procedimientos que deben seguir para elaborar o modificar un documento o registrar una acción que se da en el proyecto.

Todos los planes se realizan al principio del proyecto y se van actualización a lo largo del proyecto por ello es necesario realizar un estudio de la periodicidad con la que se deben actualizar los planes.

Al margen del control que indica la norma, es imprescindible elegir un sistema de control de versiones documentación. Las características que se requieren son simpleza, agilidad y robustez a la hora de almacenar los documentos. Aunque existen una variedad de opciones en el mercado, se podría usar alguna de las siguientes opciones:

- Git: Simple y ampliamente usado. Permite almacenar en un repositorio descentralizado versiones de del código software. Aunque no es su función principal usarlo como gestor de documentación, permite llevar el control de cambios de la documentación. Hay que tener en cuenta que si la documentación no está en ficheros basados en texto (como un ".docx") no se podrá usar la herramienta de ver diferencias.
- Alfresco: Aunque a priori es un programa que nos brinda todas las características que necesitamos, con el tiempo y una falta de control y mantenimiento el programa se vuelve lento y tedioso.
- Google App: Suite de trabajo de sus herramientas gratuitas, permite almacenar en la nube la documentación. Permite la edición por parte de varias personas de documento al mismo tiempo, además es simple y fácil de usar.

Para el programa elegido se debe dejar a disposición de todos los usuarios un manual o guía de uso.

Anexo 6: Gestión de la configuración

En este anexo debe incluirse una descripción de la estructura y responsabilidades de los roles relativos a la gestión de la configuración, por ejemplo, el gestor de la configuración.

Otro elemento es una lista y descripción de actividades para la gestión. Elegir bien los elementos de la configuración es necesario para ajustar los procesos correctamente, elegir los elementos COTS que se deben registrar, bien por motivos de documentación o por software.

Se debe elegir y registrar un sistema de gestión, detallando todas las herramientas que se usarán y los manuales de uso recomendados, definiendo también los permisos que dispondrá cada persona que pertenezca al proyecto.

3.3.1.3. Plan de implantación y versión de software publicado. PL30

El objetivo de este plan es tener un manual que describa el procedimiento a seguir para instalar, actualizar o reponer una versión anterior del software publicado. También explicar cómo localizar e identificar la versión del software y la configuración asociada.

Requisitos para la implantación del software

El plan debe dejar descrito los requisitos para implantar el software:

Identificación de la versión y la configuración: Una vez implantado el software, se debe poder identificar la versión del software. Para ello es recomendable implementar un sistema de identificación único que identifique si existen cambios no autorizados con respecto a la última versión entregada por la empresa, para ello es recomendado usar funciones hash que analicen todo el código. Lo mismo se debe implementar para los elementos de configuración del software.

Rectificación de la implantación: En caso de que se encuentre un error o se den problemas en la implantación, se debe poder volver a la versión anterior del software instalada. Para ello en el plan debe quedar descrito el procedimiento para hacer la regresión del software.

Compatibilidad de los componentes software y software/hardware: en el plan se debe dejar descrito las restricciones con las que cuenta el software con otros componentes software (interfaces con otros softwares), dejando descrito la forma de comprobar que no se dan las incompatibilidades o la lista de pruebas que se pueden realizar para verificar que no se dan. Se debe listar los elementos hardware con los que el software es compatible y/o incompatible.

Para cada versión de software se debe crear un documento que contenga, la versión y el identificador de versión único, la lista de compatibilidades con los componentes y hardware y por ultimo las condiciones de la aplicación.

Manual de implantación

El plan debe incluir un manual de implantación donde es recomendado fraccionarlo en:

Instalación del software: manual de cómo instalar una versión concreta de software, con todas las dependencias e incompatibilidades específicas de la versión. Si se requiere alguna actualización de firmware o instalación de hardware adicional, un manual completo de cómo realizar estas tareas. Es importante que el manual sea completo, detallando desde cero como realizarlo y sea entendible por los miembros que vayan a realizar el proceso. En caso de que existan indicadores de errores durante la implantación es importante describirlos para que el implantador sea capaz de parar y revertir la instalación.

Regresión: manual sobre como revertir a la versión anterior del software en caso de que se encuentren malfuncionamientos.

Identificación de versión: manual sobre cómo identificar la versión de un programa previo a implantada y la versión ya implantada.

Registro de implantación

Con la primera implantación del software o siempre que se realicen actualizaciones se deben registrar los cambios, bajo la responsabilidad del implantador o del equipo de pruebas si realiza esta labor. Con la fecha, pruebas realizadas, si los requisitos de

implantación se han cumplido, versión e identificador de versión y configuración instalada.

Este registro se debe guardar con el resto de la documentación y entregar junto con la verificación del proyecto.

3.3.1.4. Plan de mantenimiento del software. PL40

En este documento se deben detallar los objetivos del plan y de los procedimientos para realizar el mantenimiento a nivel de software del proyecto.

Se considera que la entrada de este documento son todos los documentos de diseño y con esta información se debe detallar el proceso de mantenimiento y cambios del software después de la entrega del proyecto.

Procedimientos de control

El objetivo de este plan es establecer los mecanismos de control de los errores, autorizaciones, cambio, configuración de hardware/software y las técnicas y medidas a seguir durante la fase de mantenimiento del proyecto.

En el plan de deben definir los roles de mantenimiento y la autoridad que homologa los cambios. Es recomendable que cualquier persona en contacto con el software pueda reportar errores, aunque tras su análisis se puedan desestimar por no ser tales. Se debe definir un verificador y validador que debe seguir los requisitos de objetividad e independencia que se requiere para el nivel de integridad elegido para el proyecto.

Mientras que se puede realizar de forma manual, es recomendado realizar con algún programa de seguimiento de errores (BTS por sus siglas en inglés) como Mantis, donde los usuarios reportan los errores y la autoridad asignará un encargado de supervisión y control de los mismos.

Estas mismas herramientas se pueden usar para realizar el control de la autorización, la evaluación del impacto y el registro de los cambios en los errores (solucionado, en espera…).

En el plan de mantenimiento se debe dejar detallado la herramienta o método se usará, los roles que desempeña cada parte del mantenimiento y el procedimiento que se debe seguir desde reportar un error hasta su solución. El orden recomendado para este procedimiento sería:

-Cualquier usuario del sistema reporta un error: descripción del mismo, entorno donde se ha dado, criticidad y posibilidad de que ocurra.

-El proveedor hace el análisis: análisis del error y solución, que componentes afecta, si es un cambio localizado o general. La versión del software y configuración sobre la que se aplicará el error y posibles errores o problemas que puede desencadenar.

-Autorización de la entidad competente: se debe describir el procedimiento por el cual se autoriza un cambio en el software en mantenimiento.

-Pruebas: conforme a lo expuesto en el análisis del error se deben de repetir o hacer regresión de todas las pruebas a asociadas a los componentes afectos y adicionalmente si en los posibles errores que puede desencadenar hacer un estudio y añadir las pruebas que sean necesarias.

-Verificación y Validación: Siempre que se trate de un cambio localizado se puede realizar la verificación y validación exclusiva de ese cambio siguiendo el plan de verificación y validación.

3.3.1.5. Plan de preparación de la aplicación. PL50

Este documento solo se debe realizar si se está usando una aplicación ya homologada y se está configurando un conjunto de datos y/o algoritmos que adapta una aplicación genérica para el uso específico del proyecto. El documento debe ser redactado bajo la autoridad del diseñador.

Descripción del desarrollo

El plan debe incluir una descripción de las actividades de desarrollo, la estructura de documentos de entrada y salida para los procesos descritos en el plan. La norma no especifica qué actividades se deben dar en este tipo de desarrollo, por ello la mejor opción es adaptar lo que dicta en un desarrollo nuevo. Una fase de análisis de riesgo, diseño, implementación, pruebas, verificación y validación.

Se debe incluir los roles de los miembros del grupo y los requisitos de independencia entre los distintos grupos, siendo estos principalmente diseño, desarrollo, pruebas, verificación y validación.

Análisis de riesgos

Se deben incluir un análisis de los riesgos de las herramientas usadas y de los procesos usados.

Se deben definir unos criterios de aceptación de riesgos, para ello se define una tabla con los niveles de severidad y otra con niveles de frecuencia. Con estos dos datos se define el nivel de riesgo. Un ejemplo sería:

Frecuencia de Ocurrencia	Niveles de Riesgo			
Frecuente	No deseable	Intolerable	Intolerable	Intolerable
Probable	Tolerable	No deseable	Intolerable	Intolerable
Ocasional	Tolerable	No deseable	No deseable	Intolerable
Remota	Despreciable	Tolerable	No deseable	No deseable

Improbable	Despreciable	Despreciable	Tolerable	Tolerable
Increíble	Despreciable	Despreciable	Despreciable	Despreciable
	Insignificante	Marginal	Crítico	Catastrófico
	Niveles de gravedad de la consecuencia de la amenaza			

Tabla 10 Clasificación de riesgos

Después se define una lista con toda la lista de riesgos, donde para cada riesgo se define unos datos:

Id	Identificador
Situación	Descripción del peligro potencial
Amenaza	Posibles Causas de la situación anterior
Riesgo inicial	Nivel de riesgo conforme a la tabla anterior
Alcance	Elementos internos o externos que pueden ser afectados
Requisito	Requisito de seguridad que debe cumplir la herramienta o proceso para evitar que suceda, minimizar o prevenir este riesgo

Tabla 11 Características de riesgos

Análisis de herramientas, técnicas y medidas

Se debe incluir un análisis de las herramientas para el desarrollo de los datos y/o algoritmo y una justificación de las técnicas y medida.

Por último, el plan debe describir el procedimiento para verificar y validar las herramientas usadas en el desarrollo, su idoneidad y su compatibilidad con la aplicación genérica previamente homologada.

Verificación de los datos y/o algoritmo

El plan debe detallar las actividades de verificación y procedimientos para llevar a cabo las actividades de desarrollo.

Los procedimientos de verificación suelen basarse en la comparación de un conjunto de datos (entradas, alarmas, conexiones…) definidas durante el diseño con lo implementado en el software para comparar y asegurar que se ha realizado lo que se definió durante la fase de diseño. Estas actividades se pueden sustituir por informes generados por la propia herramienta si lo permite el programa.

También se debe incluir en la verificación si el conjunto de datos es compatible con la aplicación y cumple con los requisitos o condiciones de la aplicación genérica.

3.3.1.6. Plan de validación. PL60

El objetivo de la validación es que las especificaciones se ajustan y cubren los requisitos de seguridad y que los ensayos cubren los requisitos y se han realizado correctamente.

El plan de validación junto al informe de validación son dos elementos críticos de la documentación del proyecto, ambos deben ser escritos bajo la autoridad del validador y este debe ser completamente independiente de los otros roles.

Descripción del proceso de validación

El plan debe contener una descripción de la actividad de validación. Para realizar la validación habitualmente se realiza siguiendo o creando una trazabilidad entre las distintas etapas entre los requisitos y los resultados de las pruebas. Evaluando si todos los requisitos están cubiertos por las especificaciones y las pruebas definidas y posteriormente realizadas. Los pasos a seguir para realizar la validación son:

1. Evaluar si todos los requisitos del software cubren todos los requisitos de seguridad.
2. Evaluar si todos los requisitos se han implementado o se ha diseñado funcionalidad (no aplica para requisitos informativos o no funcionales).
3. Evaluar si toda la funcionalidad queda probada mediante ensayos.
4. Evaluar si los ensayos se han realizado correctamente y los resultados son positivos.

Si el resultado es positivo en los cuatro puntos, se puede afirmar que todos los requisitos de seguridad quedan cubiertos por los resultados de los ensayos realizados.

Por último, es importante recalcar que el objetivo la validación no es el objetivo del plan, el de este es explicar cómo se va a realizar este proceso, no realizar o evidenciar este proceso.

Aunque el proceso de validación viene especificado en la norma, existen diferentes métodos de implementar lo especificado y por ello es recomendable usar plantillas y modelos que hayan demostrado su utilidad como los expuestos en el libro “Software Verification and Validation for Practitioners and Managers”. [Steven, 2001]

Descripción de las pruebas

El plan debe describir y justificar los tipos de pruebas que se van a realizar. En caso de que se realicen pruebas en entornos simulados, debe analizarse la idoneidad de estos entornos.

3.3.1.7. Plan de verificación. PL70

En este documento se deben detallar los objetivos del plan y de los procedimientos para realizar la verificación de los documentos y las actividades del proyecto.

El documento debe detallar el organigrama, funciones y responsabilidades del personal relacionado con el proyecto y en especial el grupo encargado de la verificación del proyecto.

En el plan se deben de detallar las técnicas y medidas que se usarán en la verificación. Se debe justificar el uso de técnicas y medidas desaconsejadas o el no uso de aconsejadas. En caso de que esta información se recoja en algún documento del proyecto debe de referenciarse en este plan. Las técnicas vienen dadas por la tabla A.5 de la norma 50128

Se debe incluir la lista de herramientas usadas para la verificación y las usadas para el proyecto. Estas herramientas según la norma 50128 se clasifican en T1 (no influyen en el código), T2 (verificación y pruebas, un error en esta herramienta no muestra errores en el código, pero no los crea) y T3 (su salida contribuye al código).

Para todas las herramientas de tipo T2 y T3 se debe justificar su uso, o bien mediante su certificación previa o realizando un estudio con los siguientes puntos:

-Análisis de fallos potenciales y las medidas para evitar dichos fallos.

-Especificación o manual de uso y las restricciones de la herramienta.

Además, para herramientas T3 se debe añadir:

-Pruebas que acrediten que la salida es la correcta o que se detectan los fallos.

-Validación de la herramienta para el uso que se le debe dar.

Para cada herramienta se debe dar una información resumida en este plan que contenga una descripción, el uso que se le pretende dar y una justificación. Para las herramientas T2 y T3 de debe añadir la versión sobre la que se está usando.

Se deben dejar redactados tipos de procedimientos de verificación: documentación, administrativos y procedimientos.

También se debe verificar el software, pero ese contará con su propia guía.

Verificación de la documentación

Este procedimiento de incluir una lista de toda la documentación a verificar y una lista de los principales objetivos de una documentación verificada: que cumple con los requisitos de elegibilidad y trazabilidad, así como la coherencia interna del documento. No se debe de olvidar que cada documento contendrá otros elementos de verificación que se especificarán en el informe.

Todo este proceso quedará recogido en el Informe de verificación.

Verificación de las fases.

En este procedimiento se definirán las fases y los objetivos a cumplir en cada una de ellas. No solo las fases de desarrollo del software (asociadas a la norma 50128), sino también

aquellas asociadas al proyecto general (asociadas a la norma 50126). Se puede realizar un informe para cada fase o quedar todas recogidas en un único informe.

Todo este proceso queda recogido en el Informe de fin de fase.

Verificación del software o verificación de los datos/algoritmo

Tanto desarrollando una aplicación o usando una aplicación ya certificada, se deberá realizar una verificación donde se confirme que el programa está haciendo aquello para lo que se ha definido. Esto se realiza mediante inspección del código o el análisis de las pruebas de componentes.

También se debe verificar que se ha cumplido la guía de codificación del software presentando las evidencias pertinentes. En caso de que se trate de una verificación de una aplicación ya certificada, se deben verificar que las condiciones de aplicación se han cumplido presentando las evidencias pertinentes.

El resultado queda plasmado en el informe de verificación del software.

3.3.1.8. Plan documental. PL80

El objetivo de este documento es tener un registro de los documentos del proyecto. Además, servirá para otras funciones como usarlo de lista de referencias y últimas versiones.

También se podrá almacenar otra información de interés para el jefe de proyecto como si el documento está cerrado o se esperan nuevas versiones.

Columna	Descripción
Sistema	Si pertenece a los subsistemas
Referencia	Número de referencia al documento
Código	Código del documento
Nombre	-
Versión	Última versión liberada
Empresa Responsable	-
Autor: Rol	Rol de la persona encargada
Revisor: Rol	Rol de la persona encargada
Aprobador: Rol	Rol de la persona encargada
Autor: Nombre	Nombre de la persona encargada
Revisor: Nombre	Nombre de la persona encargada
Aprobador: Nombre	Nombre de la persona encargada
Fase	Fase de realización del documento
Certificación	Si el documento pertenece a la certificación o es documentación de apoyo
Estado	Estado del documento

Tabla 12 Características documentación

3.3.2. Especificaciones

3.3.2.1. Especificación de diseño del software. ES10

En este documento se encuentra la información general de los componentes del software. Debe ser realizado por el diseñador y tener en cuenta los documentos de requisitos, arquitectura e interfaces del software.

Debido al momento temprano en el que se realiza este documento, es posible que los otros documentos de entrada no se encuentren finalizados, aunque sí en un estado de desarrollo avanzado.

Este documento sirve como enlace o índice de los documentos de especificación de componentes y especificación de ensayo de componentes que debe existir para cada componente

Listado de componentes

Se debe detalla para cada componente la siguiente información:

-Nombre y descripción: se debe describir la funcionalidad que da el componente y se debe detallar las secuencias y algoritmos que realiza.

-Nivel de integridad y trazabilidad en la arquitectura del software.

-Interfaces: descripción con los interfaces externos que tenga el componente (intercambio de datos con otros programas, entradas o salidas…) y una lista de interfaces con otros componentes. También es recomienda añadir una descripción de cómo se comunica con otros elementos o qué conjunto de datos necesitará.

-Estructura de datos: Si el componente tiene un conjunto de datos asociados se debe dar una descripción del mismo. Si existen algún dato o variable cuyo origen es complejo o dispone de especial importancia se podrá añadir a este punto con el mismo requisito de descripción.

-Trazabilidad del componente: se debe incluir una lista de requisitos del software que este componente satisfaga o implemente lo descrito en el requisito.

-Errores: Si el componente informa de los posibles errores que se puedan dar se debe describir todos los errores y cómo repercutirán en otros interfaces.

3.3.2.2. Especificación de ensayos de integración del software. ES20

Se deben especificar un conjunto de ensayos donde se pruebe que todos los interfaces funcionan correctamente, para ello se debe ejecutar el software en conjunto y observar si todos los interfaces y por tanto el software funciona correctamente.

También se debe especificar un ensayo donde se pruebe que sucede cuando un interfaz externo arroja una entrada al software que no está especificada (siendo lo más habitual que el sistema se vaya a modo seguro).

Por último, todos los ensayos de componentes se deben poder reutilizar en el software en conjunto. Esto significa que todos los ensayos de componentes se pueden aplicar al software en conjunto. Si se prueba el movimiento de un cambiavía en un componente, se puede probar con todo el software.

3.3.2.3. Especificación de ensayos de integración del software/Hardware. ES30

Se considera que estos ensayos son los más completos que se realizan por parte del integrador. Este documento sirve para antes de comenzar el desarrollo y ensayo de todo el software tener en cuenta las necesidades que se requerirán para realizar las pruebas.

Se debe definir los casos y datos de ensayo, una definición de la prueba a realizar y que datos de entrada se usarán. También se debe definir los criterios de aceptación del ensayo.

En estas pruebas se deben poder reutilizar los ensayos de componentes y de software en completo, todo lo que se pruebe en el software se debe probar con el hardware y en campo. Además de las pruebas anteriores se debe someter a ensayo que el software funciona correctamente a través de los interfaces hardware que disponga.

Es importante que se pruebe si el software se detecta y gestiona todos los fallos del hardware y cumple con los requisitos de temporización y prestaciones que se le requiere.

Entorno de pruebas

Se debe definir el entorno de prueba. Para cada ensayo que se defina se debe añadir si la prueba se puede realizar en las instalaciones de laboratorio o se deben hacer en las instalaciones del usuario finales.

Todas las herramientas que se vayan a usar para realizar los ensayos deben quedar descritas en este documento, con el uso que se le va a dar, también para el software de apoyo que se necesite y la descripción de la configuración.

3.3.2.4. Especificación de ensayos de la aplicación. ES40

Dado que la aplicación ya se encuentra previamente certificada, solo se ha de probar los elementos que varían con respecto a otro proyecto. Fuera del ámbito del software esto significa probar todos los elementos instalados y que las conexiones son las correctas. En el ámbito del software esto significa que se debe someter a ensayos de forma que se garantiza que los datos son correctos y el algoritmo está correctamente implementado en relación a la aplicación y a la arquitectura elegida.

3.3.2.5. Especificación de ensayos del software en conjunto. ES50

En este documento se debe redactar una serie de ensayos que sirvan para probar la funcionalidad de cada componente. Para ello se deben realizar dos tipos de ensayos en cada componente:

-Ensayos de caja negra: donde no se analiza el interior del componente, se prueba que este cumpla con la funcionalidad indicada. Para el ensayo el componente es una caja donde no se ve el interior y solo ve que se dan unas entradas y se obtienen unas salidas, no se analiza el cómo se hace.

-Ensayos de caja blanca: Se prueba cómo interactúan las partes del componente internas de forma que se puede ver como se ha implementado el componente. Se debe someter a ensayo todas las partes del componente de forma que no quede un elemento sin ejecutar.

3.3.2.6. Especificación de ensayos del software en conjunto. ES60

Este documento debe incluir una descripción completa de todas las pruebas que se deben realizar sobre un software terminado. Debe existir una prueba o más pruebas para cada funcionalidad y que queden probado todos los aspectos de una funcionalidad.

En resumen, un ensayo consiste en la descripción de la prueba a realizar, qué funcionalidad se va a probar, requisitos de señales de entrada, secuencias y valores, qué valores o secuencias en las señales de salida se espera y por último qué criterios de éxito se requieren.

Por ejemplo, si se quiere probar el accionar un cambiavía para moverlo manualmente (funcionalidad) se debe describir el proceso o número de pasos para realizar la acción y las señales afectadas y que entrada deben tener. Los resultados serán el conjunto de señales de salida (cambiavías encendido, cambiavías posición…) y se especificará qué conjunto de valores de salida se consideran un éxito.

Un ensayo puede ser válido para probar múltiples funcionalidades si el sistema está correctamente modelado (equivalencias). En el ejemplo anterior se puede crear un único ensayo para probar el movimiento en todos los cambiavías si la entrada de cada elemento es, por ejemplo, cambiavías_X y posición_CV_X donde X es la nomenclatura del cambiavía. En este caso se debe indicar para que elementos aplica el ensayo.

Probar el movimiento de un cambiavía para ver si el software realiza la acción bien es un ensayo de software, en este caso no es necesario que exista el elemento de vía y puede ser sustituido por un simulador si este está correctamente modelado y es capaz de detectar errores. Si se realiza la misma prueba en cada cambiavía sabiendo que nuestro software funciona correctamente se trata de una prueba de vía con el objetivo que todos los cambiavías estén correctamente instalados y funcionando.

Para diseñar las pruebas es importante ver los requisitos y diseñar un conjunto de pruebas que abarquen a todos los requisitos, un ensayo puede desarrollar y probar varias funcionalidades o requisitos. Esto puede resultar difícil de hacer ya que elegir un criterio a la hora de elegir cómo plantear los ensayos puede diferir de la forma en que lo evalúa el equipo de validación, por ello es recomendado formarse antes. [Hong Zhu, 1997]

3.3.2.7. Especificación de la arquitectura del software. ES70

Una parte vital de toda la documentación de un proyecto es la descripción de la arquitectura del software, esto acorde a la norma UNE 50128 consta de varias partes: descripciones generales, componentes y software preexistente.

Aunque no es necesario reflejar en este documento es recomendable antes de realizar todo el diseño de la arquitectura evaluar la viabilidad de los requisitos, es posible que el cliente pida funcionalidades que por motivos lógicos o de seguridad no sea posible cumplimentar

Arquitectura del software

Se debe dar una descripción extensa y clara de la arquitectura elegida para el desarrollo del software, motivos por los que se ha elegido esta descripción de cada elemento y funcionalidades.

También se debe incluir el nivel de integridad elegido, los motivos y su viabilidad. Es importante que el nivel de integridad del software lo da el nivel del menor de sus componentes a menos el componente sea independiente de los demás.

En la arquitectura del software se debe incluir como interactúa el hardware con el software. Esto consta de qué restricciones presenta cada uno y cómo se van a cumplir y un análisis de qué errores se puede encontrar de la integración del software en el hardware.

A lo largo del proyecto y debidos resultados de pruebas, cambios en el proyecto o investigación de elementos es posible que se necesite cambiar la arquitectura del software. Debido a que se trata de algo desaconsejado es muy recomendable que se evalúe a fondo el cambio y cómo afectará al resto del proyecto y en especial a los componentes del software. En caso de que se haya realizado las pruebas, la verificación y/o validación de los componentes y otros elementos que dependen de la arquitectura se deberá repetir en todos los elementos a los que afecte.

Componentes del software

Para componente del software que se incluye se debe incluir en la misma o posteriormente en una tabla los siguientes elementos:

-Nombre del componente.
-Existencia previa: si el componente no es nuevo se debe incluir si el componente se ha validado previamente.
-Nivel de integridad del componente.
-Requisitos: los requisitos que cubre el componente, es importante que esto sea bien identificado.

Software preexistente

En caso de que exista software preexistente en el proyecto se deben cumplir los siguientes requisitos: Descripción del software, validación del mismo, requisitos del software

preexistente (condiciones que debe cumplir nuestro proyecto para que ese software mantenga la calificación de seguridad), interfaces del software y los requisitos que va a satisfacer este software.

Para niveles SIL 3 y SIL 4 se deben añadir unos requisitos adicionales para el software preexistente. Se debe realizar un análisis de los posibles errores que genere y sus consecuencias, así como la forma de detectarlos y cómo se protegerá el sistema. En general eso consiste en realizar un análisis de riesgos y evaluar cómo se cubrirá el nuestro software y como avisaremos al equipo pertinente de que se ha dado un fallo previo. Es posible que el propio software preexistente incluya un análisis de estos datos. Por último, se debe detallar en la verificación y validación del producto que el software preexistente cumple con los requisitos asignados (así como el resto del software) y en la fase de pruebas se ha observado el comportamiento de detección de fallos del software preexistente y es el correcto.

3.3.2.8. Especificación de la interfaz del software. ES80

Para este entorno entendemos como interfaz una parte del software o un componente que sirve para comunicarse con otro componente, software o elemento de campo a través de las señales. En concreto como entre componentes se trata de la parte de uno que sirve para comunicarse con otros o en caso de que la funcionalidad del componente sea servir de medio de comunicación entre componentes todo el componente se trata de una interfaz. También los componentes que tratan de la comunicación con elementos de campo (vías, cambiavías, señales…) se tratan de interfaces.

3.3.2.9. Árbol de interfaces

Aunque no es parte de la norma, para facilitar la comprensión del proyecto y en especial las fases de ensayos y verificación es recomendable hacer un esquema de los elementos de campo. Por ejemplo:

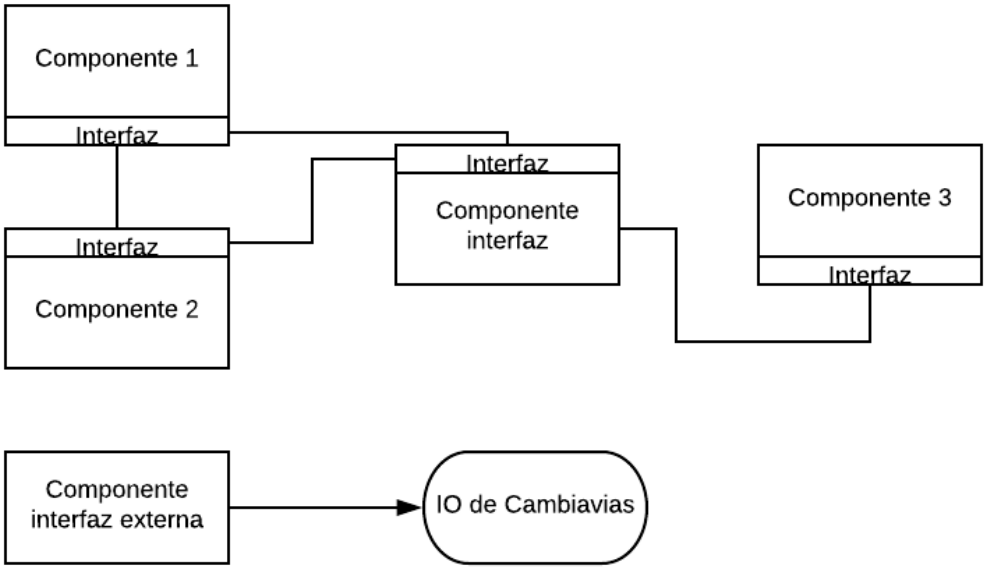

Ilustración 5 Árbol de interfaces

Con esto es fácil obtener una visión general de la estructura del proyecto sin necesidad de revelar demasiada información del código.

3.3.2.10. Interfaz

Para los interfaces se debe crear un análisis con el objetivo de evaluar los posibles errores y desbordamientos de forma que se sepa las posibles causas y cubra los posibles problemas que generen, así como se espera que reaccione el sistema.

Lo primero será evaluar los valores límites. Definir y describir los valores límites, esto significa que si una variable admite los valores [0,10] que pasa si se le asigna un número superior a 10 o si se opera con él y el resultado es un número superior al rango. También se debe definir el comportamiento que debe tener el sistema cuando se sobrepasa un valor límite, por ejemplo, si existe un desbordamiento y no se puede asegurar el resultado se enviará el sistema a modo seguro (aunque lo ideal es que el software está protegido para evitarlo).

A continuación, se debe realizar el mismo análisis y modelado de comportamiento para los buffer de memoria y para las IO donde el tiempo es un elemento crítico. En este último caso se debe añadir la restricción de tiempo y la gestión de excepciones cuando se sobrepasa ese tiempo, es decir, si el tiempo que espera una función para recibir los datos de otra se supera como debe reaccionar esta función. La sincronización entre funciones está incluida en este apartado y debe tratarse como si se tratase de una restricción de tiempo.

Con el objetivo de reducir la cantidad de ensayos que se realiza sobre una interfaz es posible modelar datos de equivalencia, esto significa que para distintos conjuntos de datos que se comportan de forma similar o de tienen elementos comunes (como por ejemplo la

misma situación de valor límite). Para estos datos se debe describir en qué consiste la equivalencia.

Puede suceder que para datos críticos se defina equivalencias prohibidas, que no se pueden realizar.

3.3.2.11. Especificación de requisitos de la aplicación. ES90

Para este documento se puede tomar como referencia lo expuesto en la especificación de requisitos del software.

Es necesario incluir los requisitos específicos de la instalación de la aplicación teóricos. Esto significa que una serie de elementos físicos afectan a la configuración de la aplicación, por ejemplo, el trazado de un itinerario que variará de un proyecto a otro se debe incluir en los requisitos.

Se debe añadir un resumen o trazar a las condiciones de la aplicación y las herramientas de la aplicación.

Por último, se deben añadir los requisitos relativos a los datos o algoritmos de aplicación que vaya a usar el software genérico de la aplicación.

3.3.2.12. Especificación de requisitos del software para el software genérico. ES100

En el ámbito de un proyecto en el que se usa un software previamente certificado (por ejemplo, un sistema de PLCs y la aplicación) es necesario documentar los requisitos de dicha aplicación.

Los requisitos descritos en esta guía deben seguir las mismas características que las descritas en la especificación de requisitos del software.

Este documento debe contener un resumen o referencia a las condiciones de la aplicación genérica, las herramientas y las normas a seguir.

Los requisitos de la aplicación genérica tienen que cubrir tres campos: los necesarios para la instalación en estudio, los relativos a datos y algoritmos que procese la aplicación genérica.

3.3.2.13. Especificación de requisitos del software. ES110

El objetivo de este documento es aunar las propiedades requeridas del software, estas propiedades deben incluir: la robustez, la seguridad, la eficiencia, la usabilidad y la portabilidad.

Debe incluir el nivel de integridad conforme al capítulo 4 de la norma CENELEC 50128.

El documento debe trazar los requisitos del software a los requisitos del sistema. El documento debe ser redactado de forma que sea comprensible por todo el personal del proyecto.

Deben existir requisitos que definen los siguientes puntos:
-Autocomprobación del software y el hardware (la detección y aviso de fallos).
-Las existencias de restricciones entre hardware y software.
-Los modos de operación y los modos de comportamiento para los sistemas electrónicos programables, en particular en el modo fallo.
-Las interfaces con otros sistemas.

Los requisitos deben estar expresados de forma que permitan el ensayo de todas las funcionalidades.

Si existe alguna funcionalidad no relacionada con la seguridad debe estar claramente especificado.

Es interesante a la hora de realizar los requisitos revisar la guía del Informe de verificación de los requisitos del software, para a la hora de realizar los requisitos tener en cuenta que aspectos nos va a requerir el verificador.

Una sugerencia de distribución:

Requisitos
Seguridad
Funcionalidad
Restricciones
Interfaces
Modos de funcionamiento
Modos de operación

Tabla 13 Tipos de requisitos

Formato

Para cumplir lo anteriormente descrito se recomienda una tabla donde para cada requisito se exponga los siguientes datos:

Nombre	Descripción
Id	Identificador único del requisito, donde quede expuesto si es normal o de seguridad, el subgrupo al que pertenece el ámbito y el número (Ej.: RS_HMI_1, R_VIA_2)
Título	Título a modo de resumen del requisito
Descripción	Debe cumplir lo dictado en el punto anterior
Tipo	Funcional/ no funcional/ información
Ámbito	Software, Hardware o Software/Hardware
Origen	Cliente, Análisis de riesgos …

Trazabilidad	A qué requisito del sistema se traza el requisito del software

Tabla 14 Características de los requisitos

Software propio y software de terceros

Es importante diferenciar en los requisitos de software (y en general en todo el proyecto) aquellos que pertenecen al software que se está desarrollando y los que hacen referencia a software de terceros. Por ejemplo, si en el proyecto se usa un SCADA comercial no se debe confundir con los requisitos del software desarrollado, estos requisitos no se podrán trazar a bloques de diseño a diferencia de los del software desarrollado.

3.3.2.14. Especificación del diseño de los componentes software. ES120

En este documento se debe desarrollar una ficha con información para cada componente.

Recomendaciones en el diseño

A la hora de diseñar los componentes es importante tener en cuenta los siguientes aspectos:

-Todos los componentes deben poder someterse a ensayos.

-Los componentes deben de ser equilibrados en complejidad, no puede haber un componente que represente el 90% de la funcionalidad y otros diez que solo representan un 1% cada uno.

-Se debe tener en cuenta la manejabilidad de los componentes, ser posible modificarlos en el futuro. El uso de guías de estilo, código comentado, uso de nombres de variables representativos...

Ficha de diseño de componentes

Autor: Miembro/s de desarrollo encargado del componente.

Versión: Identificador único de la versión del componente.

Historia de la configuración: versión, fecha y descripciones de las modificaciones

Descripción: Descripción breve de la funcionalidad definida para el componente.

Subelementos: lista de subelementos que forman el componente. Solo puede contener subelementos del mismo nivel de integridad que el del componente.

Interfaces: lista de interfaces con elementos externos y con componentes de comunicación.

Datos: lista de estructura de datos usados.

Ficha de diseño de datos

Identificador: Nombre que represente la estructura.

Estructura de datos: descripción y requisitos que debe cumplir la estructura de datos.

Ficha de diseño de subelementos

Identificador: Nombre que represente la subelemento.

Descripción: Descripción breve de la funcionalidad definida para la subelemento.

Tipo: Si el elemento se trata de una subrutina, un algoritmo, método, procedimiento…).

3.3.2.15. Arquitectura y diseño de la aplicación. ES130

Aunque no es necesario que exista este documento específicamente, se debe detallar aparte o contenido en otro documento algunos datos relativos a la arquitectura y el diseño de la aplicación.

Al igual que al desarrollar un software, pero de forma más sencilla se debe redactar una lista con la cantidad y tipos de componentes hardware y software genéricos que se van a usar. Se debe definir dónde se ubicarán estos componentes y los datos y algoritmos.

También se debe describir el diseño del algoritmo que se realice para el proyecto.

3.3.2.16. Herramientas, técnicas y medidas. ES140

Pese a que no es un documento específico de la norma (o suele formar parte del plan de garantía de calidad del software), la norma pide dejar reflejada en diferentes documentos el uso de técnicas en diferentes momentos. Este documento aúna todos esos puntos y los simplifica en un solo fichero donde es fácil consultar qué técnicas se han usado durante la realización del proyecto.

Este documento debe referenciar al plan de garantía de calidad del software.

Herramientas

Todas las herramientas de desarrollo o apoyo usadas en el proyecto deben quedar reflejada en este apartado, con el siguiente formato recomendado para cada una de ellas:

Nombre	Descripción	Uso	Versión	Tipo

Nombre	Nombre comercial de la aplicación, si se trata de un paquete con varias aplicaciones solo el nombre del paquete.

Descripción	Resumen
Uso	En qué fases y para que se está usando la aplicación
Versión	Versión por la que se le identifica
Tipo	Clasificación

Tabla 15 Plantilla y característica de las herramientas

Clasificación de herramientas

La norma 50128 define una clasificación de herramientas y una serie de factores a tener en cuenta:

Clase	Descripción	Apartados aplicables
T1	No generan salidas que afectan al código (editores de texto, repositorios…)	6.7.4.1
T2	Permiten el ensayo o verificación, un error en esta herramienta no supone un error en el código, pero sí que no se revele fallos en el código (herramientas de análisis…)	6.7.4.1, 6.7.4.2, 6.7.4.3, 6.7.4.10 y 6.7.4.11
T3	Herramientas que generan salidas que afectan al código o los datos (compiladores)	6.7.4.1, 6.7.4.2, 6.7.4.3, 6.7.4.4, 6.7.4.5 o 6.7.4.6, 6.7.4.7, 6.7.4.8, 6.7.4.9, 6.7.4.10 y 6.7.4.11

Tabla 16 Tipos de herramientas

Si una de las herramientas usadas en el proyecto está previamente certificada se puede adjuntar dicho certificado en lugar de realizar el análisis de las herramientas.

Justificación de herramientas T2 y T3

Para cada herramienta T2 y T3 se deben justificar los apartados pertinentes y resumidos en el apartado de resumen de la norma. Detallando las actividades que se realizarán para realizar el informe de verificación de las herramientas.

Técnicas y medidas

Durante todas las fases y en la mayoría de las actividades la norma define unas técnicas recomendadas o que deben seguirse para realizar las actividades de forma correcta e íntegra.

En este apartado se deben reflejar todas las medidas usadas en el proyecto, y en la fase, documento o actividad se están empleando. Se debe dejar evidencia o explicación de su uso.

Las diferentes técnicas están clasificadas en función de su necesidad y dependen del nivel SIL elegido:

-"-": No existe recomendación ni a favor ni en contra de su uso

-"M": Su uso es obligatorio

-"HR": su uso es altamente recomendado, en caso de no utilizar esta técnica se debe especificar el motivo y cuál es la alternativa elegida.

-"R": Su uso es recomendable

-"NR": su uso no es recomendado, en caso de utilizar esta técnica se debe especificar el motivo.

Se recomienda el siguiente formato, donde se incluirán todas las técnicas M y HR que impone la norma, también las R que se usen:

Técnica/Medida	Referencia	Actividad	Fase	SILX	Uso	Evidencia

Tabla 17 Plantilla de técnicas y medidas

En la casilla SILX se pondrá si para el SIL elegido la técnica es M, HR o R.
En la casilla Uso se dirá si se usa o no.

3.3.3. Informes

3.3.3.1. Informe de ensayos del software.IN10 - IN60

Este informe está compuesto a su vez por varios informes que se realizarán a lo largo de las distintas fases del proyecto. El objetivo y alcance de cada informe viene en su plan correspondiente. Todos los informes contendrán la siguiente parte común:

Se deben detallar todas las herramientas y/o software de prueba auxiliares usados y estos deben de ser reproducibles en su uso, otro grupo podrá usarlos para repetir las pruebas. También debe incluir el estado del arte tanto hardware como software para permitir que las pruebas sean reproducibles.

La parte principal del informe será una lista con cada prueba realizada y su resultado. La información recomendada a añadir será: Código de la prueba, descripción, versión del software o Hardware, fecha de realización, resultado, observaciones, nombre y firma del responsable.

Por último, el informe debe incluir cualquier anomalía o desviación del plan de pruebas y el motivo. Sin ser necesario, pero habitualmente de gran utilidad es recomendable añadir opiniones relacionadas del personal encargado de la prueba, esto permite mejorar a través de la experiencia si se deben repetir las pruebas.

Informe de ensayos del software en conjunto

Para las pruebas realizadas en este informe se pueden usar señales simuladas (recomendado) siempre que no difieran de las reales y se debe probar tanto el funcionamiento habitual como los casos atípicos y los no deseados.

Informe de ensayos de integración software

Además de lo descrito anteriormente este informe debe incluir una demostración de la aplicación de las técnicas correctamente (pruebas de caja negra). Para ello, lo mejor es detallar una explicación exhaustiva del proceso realizado y de cómo se han realizado las pruebas

Informe de ensayos de integración software/Hardware

Se trata del mismo proceso del informe de ensayos de integración software, pero ya corriendo en el Hardware final o en un modelo igual al instalado en campo.

Informe de ensayos de los componentes software

Junto con las pruebas realizadas, se debe añadir a modo de resumen si cada componente ha superado todas las pruebas. Además, se debe justificar que el componente queda completamente probado. Para ello se puede realizar una tabla de relación de los requisitos que cubre el componente y las pruebas que buscan verificar que el componente cumple dicho requisito.

Informe de ensayos de la aplicación

Al igual que en otras fases del proyecto cuando se usa una aplicación genérica previamente certificada, no es necesario probar ciertos elementos y comprobar otros respectos a la configuración. Por ello, este informe es sustitutivo de los anteriores cuando estemos en el escenario descrito.

3.3.3.2. Informe de fin de fase. IN70

Pese a que no es un documento específico de la norma (aunque sí se necesita para la norma UNE 50126) es recomendado para llevar un control del proyecto, sirve para realizar una evaluación de cada una de las fases del proyecto y permite corregir errores y evitar futuros.

Este informe, si se crea debe ser incluido en el plan de verificación.

En el informe se tendrán en cuenta los siguientes puntos que evaluarán cada fase del proyecto:

- ¿Se ha realizado la verificación de la todos los elementos de la fase? ¿Ha resultado positiva?

- ¿Se ha seguido las guías de cambios, trazabilidad y gestión de la documentación durante la fase?

- ¿Se ha respetado el orden de creación de documentos?

- ¿Han existido desviaciones de lo expuesto al principio de la fase?

- ¿Se han actualizado los documentos de seguridad con las desviaciones y cambios que se han dado durante la fase?

-Evaluación de los puntos anteriores y cambios a realizar para evitar la repetición de los errores o desviaciones que se hayan contemplado.

Por último, el informe debe de contar con una conclusión para cada fase o para para el total de las fases con los resultados de la verificación de las fases.

3.3.3.3. Informe de la implantación. IN80

La base para realizar este documento es: todos los documentos de diseño, desarrollo e implantación que sean necesarios.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento que deben quedar constancia son:

-El manual de implantación define la forma de instalar e identificar una versión de software

-Existe un registro de implantación con las cargas de software realizadas correctamente cumplimentado.

-La nota de versión publicada contiene las condiciones de la aplicación, compatibilidad y restricciones de la versión software.

3.3.3.4. Informe de la verificación de la integración software. IN90

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Los documentos de entrada de este informe son: la especificación de ensayos de integración y del software/hardware y los informes de ensayos correspondientes.

Objetivos de la verificación específicos

Los objetivos específicos de este documento que deben quedar constancia son:

-El informe de ensayos de integración de software y hardware/software es acorde a requisitos genéricos de los informes de ensayos.

-Existe una coherencia entre el informe y la especificación de ensayos de integración del software. Esto significa que el informe es un registro de las pruebas realizadas contra la especificación, no existe una prueba en el informe que no esté descrita en la especificación y se han realizado todas las pruebas pertinentes que se detallen en la especificación. Se debe realizar lo mismo para el informe y la especificación de ensayos de integración software/hardware.

-El informe de ensayos de integración software y software/hardware contiene los resultados de los ensayos. No es necesario el registro de todos los casos de pruebas, si existe un registro independiente de los casos de pruebas. Es importante dejar constancia de los errores que se den y toda la información relativa al fallo.

-Existe un registro de los casos de ensayos y su resultado. Esto significa que para cada prueba existe una ficha donde se encuentra quien lo ha realizado los pasos que ha seguido y el resultado de cada uno y si cumple el objetivo marcado.

-El informe de integración software registra la identidad y configuración de los componentes envueltos en cada prueba.

3.3.3.5. Informe de mantenimiento del software. IN100

Este informe debe incluir los puntos tratados en el manual de mantenimiento. Para ello se incluirá una lista con todos los registros de modificaciones del software, el historial de configuración del software y el conjunto de todos los ensayos repetidos y un resumen del resultado de la regresión.

3.3.3.6. Informe de validación de las herramientas. IN110

Este documento sólo es aplicable si existen en el proyecto herramientas de tipo T3.

El objetivo de este documento es asegurar que las herramientas que afectan al software como por ejemplo compiladores han sido ampliamente probadas o usadas en entornos similares. En caso de que no se pueda probar se han realizado tareas adicionales que detectan y controlan si las herramientas tienen o introducen errores en el software. Por último, en este documento se debe asegurar que las herramientas tienen salidas conforme a lo especificado.

Para analizar si la herramienta cumple con los puntos anteriores, es recomendable realizar un conjunto de pruebas de diferentes ámbitos y plasmar los resultados en el informe. Para ello se sugiere los siguientes elementos:

-Usos previos: Si la herramienta se ha usado en entornos similares y para funcionalidades relacionadas con el software actual.

-Detección de errores: Probar que la herramienta muestra los errores y controla los fallos que se den en el software. Para añadir seguridad en el proceso de detección de errores se puede añadir redundancia en la detección, esto añadiría un nivel adicional de seguridad en la herramienta.

-Análisis de riesgos: Un análisis de riesgos permitirá evaluar con más claridad.

Requisitos de validación de las herramientas

En este informe se debe dar el resultado de la validación de las herramientas, para ello la norma 50128 requiere que se definan una serie de puntos:

-Registro de las actividades de validación: todas las tareas que se realicen para validar deben dejarse constancia, este punto puede ser incluido dentro del informe de validación.

-Versión del manual y de la herramienta sobre la que se ha realizado la validación y se usa en el desarrollo del producto

-Funcionalidad: Uso que se le va a dar a la herramienta y que funcionan desempeña dentro del proyecto.

-Casos de ensayos: lista de las pruebas que se han realizado sobre la herramienta

-Conclusión a las que se ha llegado y si se ha encontrado alguna discrepancia.

3.3.3.7. Informe de validación del software. IN120

En el informe de validación se debe recoger los resultados del proceso de validación expuesto en el plan de validación. Para ellos se analizará la verificación, requisitos y pruebas del proyecto.

En el documento, aunque no es necesario se recomienda añadir un resumen del proceso de validación elegido en el plan de validación.

Se debe detallar el resultado de las pruebas de validación, se debe si estas son un subconjunto de otras pruebas de componentes, integración, vía… Se debe dar el número de pruebas realizadas y el número y cuáles de ellas se han superado, en caso de que haya pruebas de validación que no se han realizado se debe explicar el motivo y cómo afecta al resultado de la validación esta ausencia. Todos los requisitos deben estar probados, si una prueba no se realiza se debe asegurar que el requisito asociado ha quedado probado.

Deben incluir una lista de validación de los requisitos, esto significa que para todos los requisitos existe una lista de ensayos asociados que han sido realizados y con resultado positivo.

Por último, en la validación se debe incluir una evaluación de la verificación donde se compruebe que se ha realizado y esta ha sido completa.

3.3.3.8. Informe de verificación de diseño y arquitectura del software. IN130

La base para realizar este documento es: especificación de requisitos del software, la especificación de la arquitectura del software, la especificación del diseño del software, la especificación de ensayos de integración del software y la especificación de ensayos de integración del software/hardware.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento que deben quedar constancia son:

-Tiene que existir coherencia interna entre las especificaciones de arquitectura, interfaz y software.

-Las especificaciones de arquitectura, interfaz y software son implementan todos los requisitos funcionales que se describen en la especificación de requisitos del software y son coherentes con los mismos.

-La especificación de la arquitectura del software debe contener una descripción de toda la arquitectura y su viabilidad para el nivel de seguridad requerido. Debe dejar descrita las interacciones entre el hardware y el software. Una lista de los componentes breve con los requisitos relacionados en funcionalidad.

-Si existe software preexistente, el documento anterior debe contener una relación entre los requisitos y las funcionalidades para las que se usará el software preexistente, así como las interfaces que va a tener o usar. También se debe incluir el nivel de seguridad del software y si tiene alguna restricción.

-En la arquitectura se debe tratar la gestión de errores y el balance entre los tolerados y los evitados.

-En la especificación de interfaces debe existir una descripción de cada interfaz.

3.3.3.9. Informe de Verificación de Garantía de calidad del software. IN140

La base para realizar este documento es: los documentos de requisitos y la documentación necesaria para realizar la evaluación (pruebas, manuales, referencias de uso…).

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos que el plan de evaluación debe contener son:

-Lista de documentación a tener en cuenta al usar el software.

-Requisitos para realizar el informe de evaluación.

-Procedimientos para realizar la evaluación del software. Esto debe consistir en una lista detallada de procesos a seguir, criterio a evaluar y el criterio que se debe seguir para realizar el informe de evaluación del software.

3.3.3.10. Informe de verificación de la aplicación. IN150

A diferencia de los otros informes de verificación este informe depende exclusivamente de la aplicación genérica que se use y las condiciones que requiere, es recomendado para realizarlo exponer la condición o requisito y la evidencia de cumplimiento.

3.3.3.11. Informe de verificación de los datos/Algoritmo de aplicación. IN160

En este informe se debe dejar constancia o evidencia de que en él la especificación de ensayos de la aplicación cumple los siguientes objetivos:

-Existen ensayos para comprobar la correcta integración entre los datos/algoritmo en el hardware y software genérico y con la instalación completa.

3.3.3.12. Informe de Verificación de los requisitos del software. IN170

La base para realizar este documento es: la especificación de requisitos del software, de requisitos de seguridad del sistema, de ensayos del software en conjunto y el plan de garantía de calidad del software.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento que deben quedar constancia son:

-Los requisitos definen la funcionalidad, robustez, mantenibilidad, eficiencia, usabilidad y seguridad del desarrollo. Todos los requisitos están definidos en informativos, funcionales o no funcionales. En los funcionales queda comprendida toda la funcionalidad aplicada al desarrollo.

-Existe un requisito o se deja constancia del nivel de integridad que define al proyecto.

-Existe una trazabilidad al origen de los requisitos.

-La estructura de los requisitos está diseñada de forma que se puede verificar, someter a ensayo y mantener.

-El lenguaje usado es accesible a todo el personal implicado en el ciclo de vida del proyecto.

-Existen requisitos que traten con los interfaces internos y externos al programa, también los interfaces (y especialmente) los que incluyen los operadores.

-Existen requisitos que definen los modos de funcionamientos del software. En especial existen requisitos que definen el modo de fallo.

-Si existen los requisitos del software definen las incompatibilidades entre hardware y software

-Existen requisitos que definen la forma en la que el software y el hardware son capaces de detectar y emitir informes de errores.

-Existen requisitos que permiten probar todas las funciones de seguridad.

-Los requisitos deben ser claros sobre las funcionalidades a realizar.

-Deben diferenciar qué funcionalidades son de seguridad y cuales no afectan a la seguridad.

-La coherencia interna del documento.

3.3.3.13. Informe de verificación de preparación de la aplicación. IN180

A diferencia de los otros informes de verificación, el objetivo de informe es asegurar y documentar cada actividad descrita en el plan de preparación de la aplicación, realizada para garantizar que los datos/algoritmo del programa son correctos y coherentes con la arquitectura del programa. También se debe evaluar si los datos/algoritmo cumplen con los requisitos de la aplicación genérica.

3.3.3.14. Informe de verificación del código fuente del software. IN190

La base para realizar este documento es: toda la documentación.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento es atestiguar que el software cumple con lo que se ha especificado y son los siguientes:

-El código fuente está escrito bajo la responsabilidad del implantador y es acorde al diseño de los componentes del software.

-El tamaño y complejidad del código fuente está equilibrado. Si una funcionalidad está dividida en varios componentes, todos tienen una complejidad similar. También es importante señalar que a mayores complejidades es más fácil que la aplicación se vuelva

lenta. Para realizar este análisis se puede optar por calcular la complejidad matemática de cada componente o utilizar programas que evalúen estos parámetros.

-El código fuente debe ser comprensible. Aunque el concepto es subjetivo los pilares básicos suelen ser: nombres claros y descriptivos y comentarios descriptivos.

-En el informe se debe detallar si el código fuente se ha sometido al control de la configuración antes de realizar los ensayos y si han sido de forma satisfactoria.

3.3.3.15. Informe de verificación del diseño de los componentes del software. IN200

La base para realizar este documento es: la especificación del diseño del software, la especificación de diseño de los componentes software y la especificación de ensayos de los componentes software.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento que deben quedar constancia son:

-El documento de la Especificación de los Componentes del Software es coherente y satisface la especificación del diseño del software.

-Para componente se describe el autor, el historial de configuración (versión actual y todas las anteriores, fecha autor y modificaciones) y una descripción del componente. También se da una descripción de las interfaces del componente.

-Existe una trazabilidad entre elementos más pequeños como subrutinas con los componentes.

-Se analiza la viabilidad de las prestaciones definidas en el diseño de los componentes para cada componente. El diseño es mantenible, permite la posterior modificación de los componentes

-En el documento existe una relación de complejidad entre todos los componentes.

-La especificación de ensayos de componentes contiene pruebas para comprobar el correcto funcionamiento y es adecuado al diseño en el documento Diseño de los componentes del software. Todo componente debe poder someterse a ensayos. El documento define el nivel de cobertura requerido.

-Todos los ensayos cumplen con los objetivos: cumple la función prevista, las interacciones internas están controladas y son correctas y todas las partes del componente están sometidas a ensayos

3.3.3.16. Listado de elementos Hardware y Software. IN210

Pese a que la guía no especifica directamente un documento como este (y en concreto la CENELEC 50126) debido a otras necesidades (realizar los PPI) y tener trazado el hardware, es necesario un documento donde se lleve la trazabilidad del software y el hardware.

Como en otros documentos, el formato dependerá de las necesidades del proyecto y en esta guía solo se intenta dar un ejemplo.

Software

En este apartado el objetivo es tener trazado y controlado el software que se requiere para desarrollar el proyecto (no el que nosotros estamos produciendo). Un ejemplo de datos que deberíamos guardar sería el siguiente:

Nombre	Descripción
Nombre	
Referencia	
Ubicación	Donde se ha instalado el software.
Versión	-
Manual o doc. relacionada	Título de los documentos.
Cantidad	Si el software es licenciado y requiere un recuento de número de licencias.

Tabla 18 Plantilla de software

Hardware

Los elementos hardware incluidos serán todos los necesarios para el uso del software, PLCs, relés, elementos de comunicaciones…

Debido a las necesidades específicas del hardware y que el cliente o el certificador el que apruebe el hardware instalado se necesitará otros datos adicionales:

Nombre	Descripción
Nombre	
Referencia	
Ubicación	Donde se ha instalado el software.
Versión	-
Manual o doc. relacionada	Título de los documentos.
Cantidad	Para los elementos donde se usan un número elevado, pero no es importante su distinción (Ej.: conectores de fibra óptica).

Número de serie	Pueden existir múltiples equipos de mismo nombre.
Fecha y firma de recepción	
Fecha y firma de recepción	
Dirección IP	Donde aplique
Cuadro eléctrico	Para ayudar en la tarea de instalación y mantenimiento de los equipos este dato se añadirá como referencia.

Tabla 19 Características del hardware

Adicionalmente para los relés se añadirá el dato de la señal asociada.

Nombre	Descripción
Nombre	Nombre de la señal asociada

Tabla 20 Plantilla de Relés

3.3.3.17. Informe de Verificación de evaluación. IN220

La base para realizar este documento es: toda la documentación.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento es atestiguar que el software cumple con lo que se ha especificado y son los siguientes:

-El informe contiene el software y la configuración sobre la que se ha realizado la verificación.

-Una lista de elementos que contienen errores o soluciones ineficientes.

-Si el plan de verificación explica un método para realizar el informe y se decide usar otro (una desviación sobre el plan) el informe debe contener el motivo e informar de si es crítico la desviación.

-El informe debe contener un análisis de los componentes donde se analice si estos cumplen con las especificaciones. Esto se puede hacer de varias formas, mediante inspección de código, en cuyo caso toda la responsabilidad cae sobre el verificador o mediante pruebas de componentes donde la evidencia es el resultado de las pruebas.

Al final del documento se debe exponer un resumen del análisis donde exponga las deficiencias encontradas y si es posible una hipótesis de las mismas.

3.3.3.18. Informe de verificación del mantenimiento del software. IN230

La base para realizar este documento es: toda la documentación de diseño y desarrollo.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento que deben quedar constancia son:

-Se debe verificar que el plan de mantenimiento del software contiene una lista con los controles de registros de errores y formato de los registros. Una definición de quien autoriza los mantenimientos.

- El plan de mantenimiento debe contener el formato de registro de la configuración y el software, el mantenimiento y modificaciones.

3.3.3.19. Informe de verificación de la validación. IN240

La base para realizar este documento es: el plan de validación e informe de validación.

Además de lo expuesto aquí se debe incluir en este informe toda la información que se incluye en el Contenido general de la documentación apartado de informes de verificación.

Objetivos de la verificación específicos

Los objetivos específicos de este documento para el plan de validación que deben quedar para constancia son:

-La coherencia interna del documento.

-El plan de validación incluye un resumen de la estrategia de validación. Debe tener en cuenta las técnicas y ensayos usados.

-El plan identifica las etapas del proyecto para para demostrar la relación entre la especificación, los ensayos en conjunto y los requisitos de seguridad.

Los objetivos específicos de este documento para el informe de validación que deben quedar para constancia son:

-La coherencia interna del documento.

-El informe contiene los resultados que da el validador de la verificación

-El informe contiene la línea base que ha sido validada. La línea base son las especificaciones finales que contiene el proyecto

-Si existe alguna deficiencia en el software el informe debe reflejarla junto con el impacto que tiene.

-El informe contiene un análisis de objetivos cumplidos y sus desviaciones. El resultado de los ensayos y una evaluación de la cobertura de ensayos sobre los requisitos.

-En caso de que el validador realice sus propios ensayos debe dejarlos registrados en este documento.

-El informe cuenta con un análisis de la combinación de técnicas y medidas y que este es adecuado al nivel de integridad.

-El informe especifica

-El informe da una conclusión a si el software es válido para el uso previsto.

3.3.4. Guías

3.3.4.1. Guía de codificación de documentos. GU10

Pese a que la norma CENELEC 50128 no habla de realizar una guía de codificación de documentos, esta será recomendada para llevar a cabo el control de la documentación.

En esta guía se incluirá:

a. Guía de codificación.
b. Anexo de todos los códigos usados (como la expuesta en el ejemplo).

Guía de codificación

Se sugiere una codificación formada por:

-Código de proyecto.

-Si existen más de un ámbito o más de una aplicación se recomienda uno para cada elemento

-Tipo de documento.

-Número de documento.

-Nombre completo del documento

Ejemplo:

Proyecto	M1	
Ámbito	GE	General
	AP	ATP
	AO	ATO
TIPO	INF	1 Informe de verificación
		2 Informe de pruebas
		3 Informe de validación

	ESP	1	Especificación de verificación
		2	Especificación de pruebas
		3	Especificación de validación
	PLA	1	Plan de verificación
		2	Plan de pruebas
		3	Plan de validación

Tabla 21 Ejemplo de codificación

Ej.: M1-GE-INF1-Informe de verificación.doc

3.3.4.2. Guía de codificación del software. GU20

Este documento forma parte del diseño del software y dependerá de si se desarrolla un software o si se usa uno previamente certificado (Por ejemplo, PLCs y SIMATIC). Se especificará una guía de buenas prácticas siguiendo la Tabla A.12 de la norma 50128

En todo caso este documento contendrá:

a. Justificación del lenguaje utilizado
b. Si se usa software ya certificado una lista de todos los manuales que requiere el fabricante que se tienen que aplicar respecto al desarrollo del software o en caso contrario los manuales de desarrollo del lenguaje elegido.
c. Una guía de estricto cumplimiento de requisitos de codificación.
d. Una guía de estilo del software.
e. Una guía de límites, técnicas y medidas a aplicar en función del nivel SIL como explica la norma.
f. Un procedimiento para la documentación del código fuente

3.3.4.3. Guía de trazabilidad. GU30

Pese a que la norma CENELEC 50128 no habla de realizar una guía de trazabilidad, esta será recomendada para llevar a cabo el control de la documentación.

La guía de trazabilidad contará con el conjunto de puntos comunes al proyecto explicado en la guía "Contenido general de la documentación".

En la guía de trazabilidad se deberá de explicar cómo se realiza la trazabilidad de todos los documentos del proyecto y del software. Deberá incluir una:

a. Una lista de las herramientas encargadas de la trazabilidad (programas de gestión documental, repositorios, etc.) y si alguna es especifica un anexo o una referencia a un manual de uso. Se recomienda que prime la simpleza sobre otros aspectos, una herramienta compleja termina por no usarse.
b. Una descripción de los roles encargados de su control y uso, con una breve descripción. Se debería implementar mínimo dos un gestor de la trazabilidad y un usuario. Se recomienda que la trazabilidad entre

versiones la lleve el autor del documento en consenso con el gestor de la trazabilidad.

c. Una descripción de en qué consiste la trazabilidad de cada documento y software. Lo más simple y específica que se pueda, pensando siempre en las alteraciones que puede sufrir el proyecto a lo largo del tiempo y en la reticencia de los usuarios a seguirla.
d. Una descripción del procedimiento de cambio de versiones, condiciones que se deben dar para el cambio de versión.
e. Un procedimiento de evaluación de la trazabilidad del proyecto de forma periódica para comprobar que se está siguiendo.

Documentación

Para la documentación se recomienda el uso de un sistema de versiones y release.

Todos los documentos contarán con una sola cifra de versión formada por la unidad y decimales (Y. ZX). Se usarán los decimales para cuando se den cambios menores y la unidad cuando se den cambios mayores. Todas las versiones se considerarán internas del autor a excepción de las que aparezcan en las release.

De forma periódica y con una cadencia de entre una semana y un mes, pero manteniendo siempre la misma el jefe de proyecto lanzará un documento con toda la documentación asociada a todos los miembros del proyecto. Este documento contendrá el número de release y fecha, así como la versión de todos los documentos. De esta forma se sabrá en todo momento si un documento depende de otros cuáles son estos, por ejemplo, una matriz de requisitos y casos de pruebas. En este documento se añadirá la versión del software usado para esta release. Dado que no se darán siempre cambios en todos los documentos es recomendable señalar cuales han cambiado desde la última release.

Software

Se usará tres o cuatro dígitos, una letra y el cksum para definir el número de versión del software:

Nombre	Descripción
Mayor	Indicará mejoras y nuevas funcionalidades.
Menor	Corrección de errores y pequeños cambios.
Micro	Se aplica a pequeñas correcciones del software.
Módulos	De forma opcional si el software está formado por módulos independientes y podrá usar un conjunto de números que puedan definir de forma independiente todos los módulos del software (Ej.:

	A01B02 el módulo A tendría la versión 01, el módulo B la 02)
Fase	Con la letra “T” para testing y “R” para release.
Cksum	Como adjunto a la versión se añadirá el cksum del software.

Tabla 22 Ejemplo versionado de software

Ejemplo: 1.9.1T-26845

3.3.4.4. Gestión de cambios e incidencias. GU40

Como parte del plan de calidad el documento Gestión de cambios e incidencias se crea con el objetivo de definir los siguientes procesos:

- Cambios en la documentación.
- Cambios en la funcionalidad del software.
- Incidencia en el software.

Roles

El documento debe definir unos roles con el objetivo de que sea clara la lectura de la gestión de cambios e incidencias. Estos roles son solo para el proceso de gestión de cambios e incidencias y no están relacionados con la norma. Se recomienda los siguientes roles:

Nombre	Descripción
Responsable	Persona o grupo responsable del documento o software en el que se debe hacer una modificación o solventar una incidencia.
Publicador	Persona o grupo que ha detectado la incidencia o solicita el cambio.
Comité responsable	Grupo responsable, formado por el jefe de proyecto o autoridad competente y el responsable del documento o software.

Tabla 23 Roles de gestión de cambios

Objetivo

En muchos entornos de proyecto se dejan procedimientos largos y tediosos de seguir donde la persona que define el proceso no tiene que ejecutarlo o no conoce las situaciones de trabajo del proyecto.

Por estos motivos es importante recalcar que la gestión debe de adaptarse al personal. Realizar un proceso que requiere de mucho tiempo y muchas personas en un proyecto en el que el tiempo es ajustado y el equipo de trabajo es escaso el resultado es que se abandona la gestión y se generan problemas adicionales.

Por ello se describen los siguientes objetivos:

Trazabilidad: El objetivo principal de la gestión de cambios e incidencias es siempre saber por qué se ha realizado un cambio, quien lo ha aprobado y si la solución que se ha tomado es la correcta.

Simpleza: Cualquier elemento que no sirva para mantener la trazabilidad es superfluo y sólo sirve para que se abandone el uso de estos procedimientos.

Exactitud: Cualquier persona del entorno del proyecto debe poder leer el informe del cambio y entender la trazabilidad.

Procedimientos

Aunque el objetivo de esta guía es dar un boceto del documento, se debe detallar un texto que refleje los procedimientos, siempre es necesario adaptarlo a lo descrito en el apartado de objetivos.

Modificaciones

1 El publicador presenta ante el responsable la solicitud de modificación. Este verifica que la modificación requerida es correcta y factible.
2 Se presenta ante el comité responsable, que lo estudia y emite un veredicto.
3 En caso de ser aceptado se comunica a todos los afectados de forma directa o indirecta de los cambios que se realizarán (Por ejemplo, un cambio en un requisito puede implicar al equipo de desarrollo y al de pruebas).
4 Se realizan los cambios.
5 El responsable verifica que se ha realizado los cambios pertinentes e informa al comité responsable.

Incidencias

1 El publicador presenta ante el responsable la incidencia, en ella se especificará el origen, si es una prueba estática se adjuntará el registro de la prueba que consta de un registro de la fecha, la persona el error encontrado, su observación y si se realizó la corrección en el momento. En el caso de ser sobre una prueba de otro tipo se añadirá a lo anteriormente dicho el registro completo del caso de prueba, la disposición de los elementos de vía en el momento y la versión de software sobre la que se estaba realizando la prueba.
2 Se presenta ante el comité responsable, que lo estudia y emite un veredicto. En caso de que sea negativo deberá justificar por qué ante una prueba fallida se alega que no es necesario realizar los cambios pertinentes.
3 Se realizan los cambios pertinentes tanto en el software como en la documentación.
4 Se repite la prueba, se verifica y valida (si ya se habían realizado) y se emite un acta de resolución de incidencias.

3.3.5. Registros

3.3.5.1. Registro de la implantación. RE10

El registro debe justificar que se ha seguido el manual de implantación, así como la versión de software instalada.

3.3.5.2. Registro de modificación del software. RE20

Ante cualquier cambio del software se debe realizar un informe con el objetivo teórico de saber a qué afecta y el objetivo real de evitar repetir todas las pruebas del proyecto.

Para ello el informe incluirá la versión del software a la que hace referencia, todos los cambios realizados y a que componentes afecta, es vital delimitar correctamente el alcance de los cambios y fases del proyecto a las que afecta. Con esto y dado que por defecto implica repetir todos los ensayos realizados, se puede justificar que no es necesario repetir todas las pruebas y se pueden acotar a solo aquellas en las que ha habido cambios en su componente.

Este informe puede ser incremental, donde se añade toda la información de todas las versiones anteriores.

3.3.5.3. Registro de nueva configuración. RE30

La configuración la conforman todos los elementos alterables que no requieren modificar el software ni cambiar de versión como por ejemplo tiempos de espera. Este documento debe registrar una versión de configuración y sobre que versión de software aplica, así como los cambios con respectos a la anterior versión.

3.3.6. Otros

3.3.6.1. Acrónimos. OT10

Para evitar tener tablas de acrónimos y términos en todos los documentos del proyecto, se sugiere crear un documento con una tabla y todos los términos del proyecto, luego en cada documento en el apartado de Acrónimos y referencias, referenciar este documento.

3.3.6.2. Árbol de dependencias. OT20

Pese a que este documento no lo requiere la norma, simplifica la labor de modificar los documentos.

Un documento puede tener otros documentos de entrada. Siempre que un documento de entrada se modifique, el documento que toma esa entrada debe ser revisado por si es necesario realizar una modificación.

Ejemplo de árbol de dependencias simple:

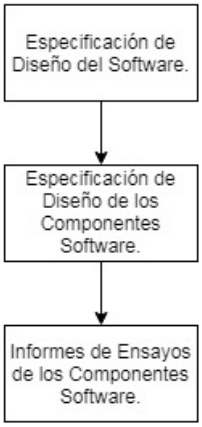

Ilustración 6 Ejemplo de dependencias

Una modificación en la Especificación de Diseño del Software implica una revisión de todos los demás documentos, esta revisión podría implicar o no cambios en el documento. Por otro lado, la modificación del Informe no implicaría una modificación de los demás documentos.

Fecha:		Nombre:	
Origen:			
Modificación:			
Aprobado:		Nombre:	
Cambios directos:			
Cambios indirectos:			

Tabla 24 Plantilla de modificaciones

3.3.6.3. Condiciones de aplicación del software genérico y de las herramientas de aplicación. OT30

Se debe incluir en el proyecto todos los manuales de usuario del software genérico de herramientas de la aplicación

Se debe incluir y dar evidencias de cumplimiento (en el pertinente informe) de todas las restricciones que tiene la aplicación, como reglas de codificación, redundancia necesaria para asegurar un sil o licencias.

4. Caso práctico

En los siguientes apartados se va a poner un ejemplo práctico de un proyecto de desarrollo desde el punto de vista de la redacción de la documentación. Partimos de la base de que el proyecto será SIL3 y el mantenimiento quedará a cargo del cliente.

En cada fase se detallan los documentos a realizar especificados en la tabla C.1 de la norma y los trabajos que se deben hacer.

4.1. Documentación de cada fase

Como se puede ver en la Figura 4 de la norma 50128 las fases forman una "V" donde cada fase tiene una serie de documentos asociados. Para cada fase se realizará un Informe de fin de fase.

4.2. Ciclo de vida del proyecto

4.2.1. Fase 0: Planificación

En esta fase el jefe de proyecto debe crear los grupos de trabajo respetando la independencia especificada en la norma como se explica en el apartado de roles. Habitualmente para este nivel es recomendable contar con cinco grupos: especificación, diseño e implementación, pruebas, validación y verificación.

En esta fase se crea la infraestructura necesaria para el proyecto como entornos de desarrollo, repositorios o laboratorios de pruebas. Debido al público tan diferente que puede tener estos programas, es aconsejado que sean de fácil uso.

En este punto el cliente ya ha expuesto un pliego de condiciones y aunque los requisitos no están aún especificados, existe un acuerdo sobre los trabajos a realizar y su alcance.

En esta fase se definen los planes que se ejecutaran durante la vida del proyecto en diferentes momentos, por eso es importante que estos sean claros y sean fáciles de ejecutar.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 0.

4.2.2. Fase 1: Requisitos del software

En esta fase se recogen los requisitos del software. Estos tendrán dos orígenes: requisitos "normales" y de seguridad en ambos casos el origen es externos a la empresa de desarrollo. En el caso de los requisitos normales es recomendable reunirse varias veces con el cliente para entender correctamente qué necesidades tiene y que entienda los requisitos correctamente, esto a pesar de parecer trivial puede reducir el tiempo empleado en otras fases. Los requisitos también se dividen en otras tres formas: relativos al hardware, al software e informativos (que no implican funcionalidad directa).

Los requisitos de seguridad deben ser entendidos perfectamente por todos los grupos implicados en el proyecto, aunque afecten en menor medida.

En resumen, los requisitos normales son los que definen la funcionalidad del proyecto y los requisitos de seguridad corresponden a las medidas que se deben tomar para reducir los riesgos asociados al proyecto.

En esta fase el equipo de implementación si ya está disponible puede ir formándose si es necesario para desarrollar la aplicación.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 1.

4.2.3. Fase 2: Arquitectura y diseño

En esta fase se dan dos procesos importantes, la definición completa del proyecto en forma de especificación funcional y el diseño de los componentes.

En esta fase es donde la realidad empieza a chocar con el mundo perfecto definido por la norma, aunque la especificación se realiza en base a los requisitos es muy posible que el cliente a partir de la especificación quiera modificarla y a su vez modificar los requisitos. Por eso es recomendable no hacer todo el trabajo y mostrarle el resultado final al cliente, ir especificando por subsistemas e ir colaborando con el cliente para que los cambios sean locales.

La especificación funcional es posiblemente el documento que más tiempo requiere por eso es buena idea ir elaborándolo de la forma más temprana posible y realizarlo de una forma modular para que los cambios sean aislados.

Si el proyecto es lo suficientemente grande y consta de partes bien diferenciadas, se pueden realizar dos especificaciones funcionales y tratas la interacción entre las dos partes como una interfaz más. Esto es aconsejado si existen varios grupos de desarrollo.

En la arquitectura se debe definir todas las interfaces del programa (como interactúa con otros programas o con otros elementos externos al proyecto), en este momento es recomendado reunirse con otras empresas que estén a cargo de las interfaces externas.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 2.

4.2.4. Fase 3: Diseño de componentes

A partir de la definición exhaustiva y la arquitectura se realiza el diseño de componentes. Esta es la unidad mínima que se usará en fases posteriores para ante un cambio definir qué pruebas asociadas se deben repetir. Por ello, es recomendable hacer los componentes lo más pequeño y con una funcionalidad lo más acotada que sea posible.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 3.

4.2.5. Fase 4: Implementación y ensayo de componentes

En esta fase se desarrolla el software. Para evitar un ciclo infinito de cambios-pruebas lo mejor es realizar todas las pruebas de componentes en fábrica por el equipo de desarrollo antes de sacar una nueva versión, de esta forma cuando el grupo de testing realice las pruebas se habrán depurado la mayoría de los errores, quedando solo aquellos que han pasado por alto.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 4.

4.2.6. Fase 5: Integración

En esta fase se usa el software en su hardware final y con todos los componentes corriendo juntos y las interfaces trabajando en común. Lo ideal es realizar el montaje en vía, pero habitualmente esto es imposible y se debe hacer mediante simulación del entorno en fábrica. Para ello, es necesario que todos los interfaces externos sean iguales a lo encontrado en campo o los resultados de los ensayos podrían variar.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 5.

4.2.7. Fase 6: Ensayos del software en conjunto/validación final

En esta fase se realizan todas las pruebas expuestas en el plan de ensayos, cualquier anomalía deberá desencadenar las acciones correspondientes y volver a la fase de diseño o implementación si es necesario.

Una vez realizados los ensayos el equipo de validación realizará las pruebas que crea oportunas entre las que se pueden incluir las ya realizadas por el equipo de testing. El objetivo es comprobar que cada requisito ha quedado plasmado en una funcionalidad y esta funciona correctamente

Esta fase es crítica y es donde habitualmente se muestran las debilidades del proyecto, una mala redacción de los requisitos, una mala implementación o funcionalidades que el cliente no son las que quiere llevarán a repetir parte del proyecto, incrementando en costes y tiempo.

Los documentos de esta fase son los marcados en la tabla de documentos y autores fase 6.

4.2.8. Fase 9: Implantación del software

En esta fase se permite el uso del software de forma comercial, el objetivo es que todo se disponga de la forma correcta, pero es más un trámite que otro proceso ya que el último punto importante es la validación final.

En este momento se entrega al cliente todos los elementos necesarios para que el producto entre en la fase de mantenimiento. Habitualmente se dejará un periodo de

vigilancia extensa corto y otro de supervisión de al menos un año si la seguridad del proyecto lo requiere.

5. Conclusiones

Si bien el objetivo principal ha sido crear una guía para la documentación, al final ha sido necesario crear un pequeño ecosistema de guías para entender cómo elaborar no solo los documentos necesarios, sino como se realiza un proyecto de esta envergadura.

Gracias a las especificaciones de roles y distribución de trabajos que definen en la norma y que se explican en este documento es fácil crear una base para cualquier proyecto que se quiera realizar. Como ejemplo está el apartado de ciclo de vida del proyecto donde se ve lo sencillo que puede ser esta labor.

Sobre el estudio de la norma, se podría resumir que deja bien definidos los procesos que deben realizar y quienes son los encargados de los mismos. Sin embargo, peca de ambigüedad en la descripción de algunas tareas, en especial en el apartado de técnicas y medidas.

Como posibles trabajos futuros, se podría añadir a este documento un estudio sobre otras metodologías de trabajo y si suponen una mejora para el ámbito ferroviario sobre la metodología en "v" propuesta por la norma. Ampliar las ayudas al jefe de proyecto, con un BPM donde se defina un flujo de trabajo estándar que permita adaptar al proyecto específico. Por último, un apartado que ahonde en la reutilización de documentación entre proyectos, generar una documentación genérica que reduzca la carga de trabajo para los proyectos.

6. Definiciones y acrónimos

PPI (*Planes de Puntos de Inspección*): Planos donde se indican los elementos importantes relativos a las instalaciones.

PLC (*Programmable Logic Controller*): Computadora especializada en automatización industrial.

Cenelec (*Comité Européen de Normalisation Electrotechnique*): Comité de normalización europea de electrónica.

SIL (*Safety integrity level*): Nivel de integridad que tiene un proyecto, refleje la reducción de riesgos que se ha tomado.

SCADA (*Supervisory control and data acquisition*): Programa que monitoriza y permite actuar sobre elementos físicos.

COTS (*Commercial off-the-shelf*): Software o conjunto de librerías comerciales.

BTS (*bug tracking system*): Software de seguimiento de errores.

CKSUM (*CheckSum*): función hash que sirve para representar la identidad única de un fichero.

Verificación: Es el conjunto de acciones que permite comprobar que un conjunto de procesos se ha realizado conforme a los procedimientos especificado.

Validación: Es el conjunto de acciones que permite comprobar que un conjunto de procesos se ha realizado conforme a lo especificado.

UNE-EN: Norma española conforme a los estándares europeos.

AENOR: Asociación española de normalización y certificación, organismo oficial español encargado del control y elaboración de las normas UNE.

IEC (*International Electrotechnical Commission*): Comité internacional encargado de la normalización de estándares relativos a la electrónica.

RAMS (*Reliability, Availability, Maintainability, and Safety*): atributos que debe tener el software para considerarse apropiado a para un entorno seguro.

PM (*project manager*): Jefe de proyecto.

RQM (*Requirements Manager*): Gestor de requisitos.

DES (*Designer*): Diseñador.

IMP: Implementador.

TST (*tester*): encargado de ensayos

VER: verificador

INT (*integrator*): integrador.

VAL (*validator*): validador.

ASR (*Assessor*): Evaluador.

CM (*configuration manager*): Gestor de la configuración.

BPM (*business process management*): Gestión de procesos de negocio.

7. Referencias

- [Normas, 2015] Guía europea sobre normas y reglamentación. (2015). Recuperado de https://www.une.org/normalizacion_documentos/Guia_30_2015.pdf
-European Standards (2019). Recuperado de https://www.cenelec.eu/ standardsdevelopment/ourproducts/europeanstandards.html
- [Coallier, 1994] F. Coallier, "How ISO 9001 fits into the software world," in IEEE Software, vol. 11, no. 1, pp. 98-100, Jan. 1994.
- [Steven, 2001] Steven R. Rakitin, “Software Verification and Validation for Practitioners and Managers”, 2001.
- [Hong Zhu, 1997] Hong Zhu, Patrick A. V. Hall, John H. R. May, Software unit test coverage and adequacy, ACM Computing Surveys (CSUR) Surveys Homepage archive Volume 29 Issue 4, Dec. 1997
- [RACI, 2016] ¿Qué es una matriz RACI y cómo usarla? (2016). https://www.laboratorioti.com/2016/02/22/ticcionario-una-matriz-raci-usarla/
- [Balaji, 2012] Balaji, S., & Murugaiyan, M. S. (2012). Waterfall vs. V-Model vs. Agile: A comparative study on SDLC. International Journal of Information Technology and Business Management, 2(1), 26-30.
- [Beresford, 2011] Beresford, D. (2011). Exploiting siemens simatic s7 plcs. *Black Hat USA*, *16*(2), 723-733.

8. Bibliografía

EN 50126, Railway applications - The specification and demonstration of Reliability, Availability, Maintainability and Safety (RAMS). Part 1: Basic requirements and generic process.

EN 50158, Railway applications - Communication, signaling and processing systems - Software for railway control and protection systems

EN 50159, Railway applications. Communication, signaling and processing systems. Safety-related communication in transmission systems.

ISO 9001:2015, Quality management systems – Requirements

9. Anexo

9.1. Anexo 1: Documentos y autores

Código	Documento	Autor	Fase
ES10	Especificación de diseño del software	DES	2
ES20	Especificación de ensayos de integración del software	TST	1
ES30	Especificación de ensayos de integración del software/Hardware	TST	1
ES40	Especificación de ensayos de la aplicación	TST	1
ES50	Especificación de ensayos del software en conjunto	TST	1
ES60	Especificación de ensayos de los componentes software	DES	3
ES70	Especificación de la arquitectura del software	DES	2
ES80	Especificación de la interfaz del software	DES	2
ES90	Especificación de requisitos de la aplicación	REQ	1
ES100	Especificación de requisitos del software para el software genérico	REQ	1
ES110	Especificación de requisitos del software	REQ	1
ES120	Especificación del diseño de los componentes software	DES	3
ES130	Arquitectura y diseño de la aplicación	DES	2
ES140	Herramientas, técnicas y medidas	*	2
GU10	Guía de codificación de documentos	*	1
GU20	Guía de codificación del software	*	1
GU30	Guía de trazabilidad	*	1
GU40	Gestión de cambios e incidencias	*	1
IN10	Informe de ensayos de integración software	INT	5
IN20	Informe de ensayos de integración software/Hardware	INT	5
IN30	Informe de ensayos de la aplicación	TST	1
IN40	Informe de ensayos de los componentes software	TST	3
IN50	Informe de ensayos del software en conjunto	TST	6
IN60	Informe de ensayos del software	TST	6
IN70	Informe de fin de fase	VER	Todas
IN80	Informe de la implantación	VER	4
IN90	Informe de la verificación de la integración software	VER	5
IN100	Informe de mantenimiento del software	*	7
IN110	Informe de validación de las herramientas	VAL	6
IN120	Informe de validación del software	VAL	6
IN130	Informe de verificación de diseño y arquitectura del software	VER	2
IN140	Informe de Verificación de Garantía de calidad del software	VER	0
IN150	Informe de verificación de la aplicación	VER	6
IN160	Informe de verificación de los datos/Algoritmo de aplicación	VER	4
IN170	Informe de Verificación de los requisitos del software	VER	1
IN180	Informe de verificación de preparación de la aplicación	VER	4
IN190	Informe de verificación del código fuente del software	VER	4
IN200	Informe de verificación del diseño de los componentes del software	VER	4
IN210	Listado de elementos Hardware y Software	*	6
IN220	Informe de Verificación de evaluación	VER	6

IN230	Informe de verificación del mantenimiento del software	VER	6
IN240	Informe de verificación de la validación	VER	6
PL10	Plan de evaluación del software	ASR	0
PL20	Plan de Garantía de calidad del software	*	0
PL30	Plan de implantación y versión de software publicado	IMP	0
PL40	Plan de mantenimiento del software	*	0
PL50	Plan de preparación de la aplicación	REQ	0
PL60	Plan de validación	VAL	0
PL70	Plan de verificación	VER	0
PL80	Plan documental	*	0
RE10	Registro de la implantación	*	0
RE20	Registro de modificación del software	*	0
RE30	Registro de nueva configuración	*	0
OT10	Acrónimos	*	0
OT20	Árbol de dependencias	*	0
OT30	Condiciones de aplicación del software genérico y de las herramientas de aplicación	EXT	2

9.2.Anexo 2: Indice detallado

#TODO cuando este terminado todo