Evaluador de microcontroladores para misiones espaciales

Autor:
Gonzalo Nahuel Vaca

Director:
Ing. Roberto Cibils (INVAP)

Codirector:
Ing. Damián Rosetani (INVAP)

Esta planificación fue realizada en el curso de Gestión de proyectos entre el 24 de junio de 2021 y el 19 de agosto de 2021.

Índice

1. Descripción técnica-conceptual del proyecto a realizar **5**

2. Identificación y análisis de los interesados **6**

3. Propósito del proyecto **7**

4. Alcance del proyecto **7**

5. Supuestos del proyecto **7**

6. Requerimientos **7**
6.1 Interfaces externas 8
6.2 Funciones 9
6.3 Requisitos de rendimiento 10
6.4 Restricciones de diseño 10
6.5 Atributos del sistema 11

7. Historias de usuarios (*Product backlog*) **11**

8. Entregables principales del proyecto **11**

9. Desglose del trabajo en tareas **12**

10. Diagrama de Activity On Node **13**

11. Diagrama de Gantt **16**

12. Presupuesto detallado del proyecto **18**

13. Gestión de riesgos **18**

14. Gestión de la calidad **19**

15. Procesos de cierre **20**

Registros de cambios

Revisión	Detalles de los cambios realizados	Fecha
0	Creación del documento	24 de junio de 2021
1	Se completa hasta el punto 6 inclusive	04/07/2021
1.1	Se cambia el nombre del proyecto Se conforma con las políticas de confidencialidad de INVAP Correcciones de redacción Se realiza hasta el punto 8 inclusive	11/07/2021
2	Se cambia el logo de FIUBA Se actualiza el costo del proyecto Se completa hasta el punto 9 inclusive	13/07/2021

Acta de constitución del proyecto

Buenos Aires, 24 de junio de 2021

Por medio de la presente se acuerda con el Esp. Ing. Gonzalo Nahuel Vaca que su Trabajo Final de la Maestría en Internet de las Cosas se titulará "Evaluador de microcontroladores para misiones espaciales", consistirá esencialmente en la implementación de un firmware de autocomprobación para el microcontrolador seleccionado y un sistema de inyección de soft-errors, y tendrá un presupuesto preliminar estimado de 600 hs de trabajo y $100000, con fecha de inicio 24 de junio de 2021 y fecha de presentación pública 15 de mayo de 2022.

Se adjunta a esta acta la planificación inicial.

Ariel Lutenberg
Director posgrado FIUBA

Ing. Damián Rosetani
INVAP

Ing. Roberto Cibils
Director del Trabajo Final

1. Descripción técnica-conceptual del proyecto a realizar

Las misiones espaciales someten su electrónica a la radiación cósmica. Por esta razón, se necesitan componentes especiales que fueron sometidos a un largo y costoso proceso de diseño y calificación. Luego, la tecnología utilizada tiene un elevado costo y retraso tecnológico por sobre los productos del mercado masivo.

Actualmente existe una iniciativa comercial para el empleo en misiones espaciales de componentes sin calificación para uso espacial. Esta iniciativa es conocida como *New Space* y provee un contexto para el proyecto a realizar.

INVAP necesita evaluar si un microcontrolador en particular puede ser usado en sus misiones espaciales. El objetivo del proyecto es crear los instrumentos necesarios para realizar la evaluación. Las herramientas a desarrollar son:

- Inyector por consola de comando (CCI)
- Proceso de dispositivo bajo prueba (DUT)

La radiación cósmica está compuesta por partículas enérgicamente cargadas. Una de estas partículas puede impactar sobre un microcontrolador. Si esto ocurre, se produce una *no funcionalidad* debido a un pulso transitorio en la lógica del microcontrolador o sus circuitos de apoyo. Esta *no funcionalidad* se manifiesta como un *soft-error* no destructivo. Finalmente, los *soft-errors* son un tipo de error en donde una señal o dato es incorrecto.

Para realizar la inyección de *soft-errors* se propone construir un sistema como se muestra en la figura 1. Se observa que el usuario podrá describir el ensayo a realizar. Luego, se utilizará un servidor *OCD* para crear las instrucciones del *debugger*. Finalmente, se inyectarán los errores por protocolo *JTAG*.

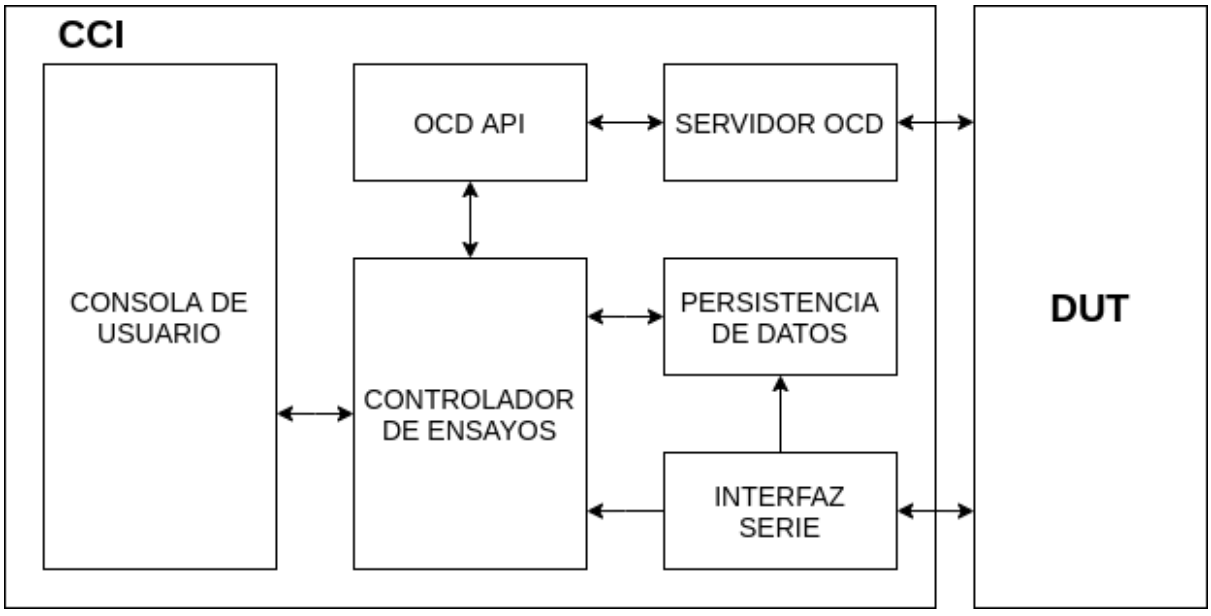

Figura 1. Diagrama en bloques del Inyector por consola de comando (CCI)

El segundo módulo del proyecto es el firmware de autocomprobación. En la figura 2 se puede observar el diagrama en bloques propuesto. Como se puede ver, este recurso deberá: verificar el estado de los periféricos, construir un informe y enviar los reportes para su análisis.

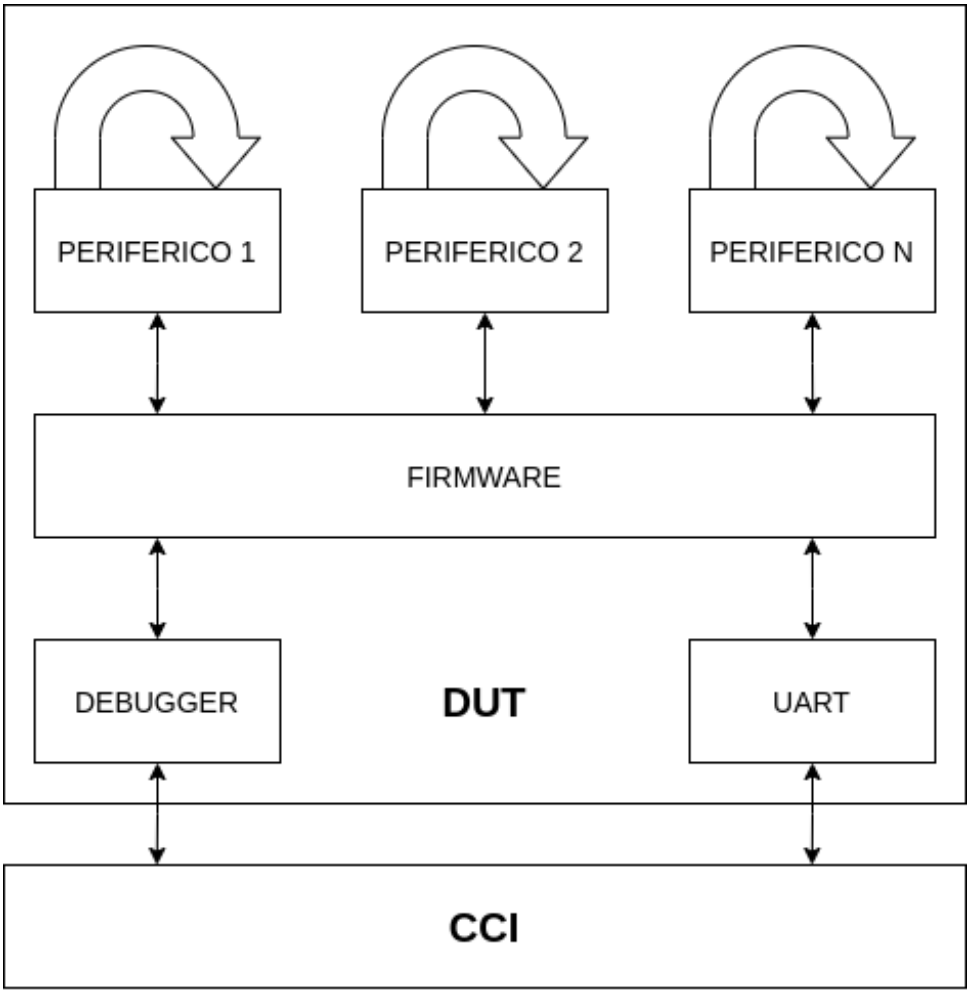

Figura 2. Diagrama en bloques del Proceso de dispositivo bajo prueba (DUT)

Se espera que el proyecto agregue valor al INVAP de las siguientes maneras:

- Simulando de forma acelerada la duración de una misión espacial
- Permitiendo presupuestar nuevo hardware para las misiones futuras

2. Identificación y análisis de los interesados

Rol	Nombre y Apellido	Organización	Puesto
Cliente	Ing. Damián Rosetani	INVAP	Ingeniero
Orientador	Ing. Roberto Cibils	INVAP	Ingeniero
Responsable	Gonzalo Nahuel Vaca	FIUBA	Alumno
Usuario final	Desarrollo de software	INVAP	-

- Cliente: cumple múltiples roles en el proyecto, valorar el tiempo que invierte.

3. Propósito del proyecto

El propósito de este proyecto es proporcionar herramientas para:

- Medir el nivel de susceptibilidad del software que se ejecuta en el DUT a los efectos de la radiación
- Evaluar el origen de dicha susceptibilidad
- Comparar la efectividad de distintas estrategias de mitigación incorporadas al software del DUT para mitigar los efectos de la radiación
- Obtener la figura de mérito (FOM) del DUT
- Simular los efectos del ambiente espacial en un microcontrolador.
- Evaluar si un dispositivo del mercado masivo puede ser utilizado en futuras misiones.

4. Alcance del proyecto

El proyecto incluye en su alcance la creación de:

- Un sistema de inyección de *soft-errors* controlado por consola de comandos
- Un firmware de autocomprobación de periféricos para el microcontrolador seleccionado por INVAP
- La ejecución de ensayos de *soft-errors* con la finalidad de validar el proyecto

El presente proyecto no incluye:

- El diseño de ensayos de *soft-errors*

5. Supuestos del proyecto

Para el desarrollo del presente proyecto se supone que:

- Se tendrá acceso irrestricto al microcontrolador propuesto por INVAP antes del día 01/01/2022

6. Requerimientos

Para comprender los requerimientos del proyecto, se enumeran las siguientes definiciones, acrónimos y abreviaturas:

1. Definiciones:

- Single event effect: efecto de una partícula enérgicamente cargada sobre un microcontrolador.
- Single event funtional interrupt: interrupción causada por el impacto de una sola partícula que conduce a una no funcionalidad temporal.
- Single event upset: pulso transitorio en la lógica o circuitos de apoyo. Son *soft-errors* no destructivos.
- Soft-error: tipo de error en donde una señal o dato es incorrecto.

2. Acrónimos:
   - API: interfaz de programación de aplicaciones.
   - DUT: dispositivo bajo prueba (microcontrolador).
   - FOM: figura de mérito.
   - IEEE: Instituto de Ingenieros Eléctricos y Electrónicos.
   - OCD: on-chip debugger.
   - SEE: single event effect.
   - SEFI: single event functional interrupt.
   - SEU: single event upset.
   - TBD: a ser determinado.
   - UART: universal asynchronous receiver-trasmitter.
3. Abreviaturas:
   - Std: estándar.

A continuación se enumeran los requerimientos del proyecto según lo especificado en el estándar IEEE Std. 830-1998:

6.1. Interfaces externas

1. CCI:
   1.1. Con usuario:
   - Deberá representar todos los caracteres de ISO Std. 10646
   - Conformará con las secuencias de escape de ISO Std. 6429
   - Usará el castellano como idioma conforme a la Real Academia Española
   - Se aceptarán barbarismos que conformen la interfaz con los sistemas UNIX
   - No deberá producir destellos ni cambios bruscos en su intensidad lumínica
   - No deberá producir sonidos
   - Títulos:
     - Los títulos deberán ser cortos
     - Los títulos deberán estar correctamente capitalizados
     - Los títulos deberán ser únicos
   - Comandos:
     - El sistema se iniciará con el comando “python sise.py”

- El sistema imprimirá en pantalla un manual de ayuda con el comando "python sise.py –help"
- Se podrá exportar la configuración del último ensayo realizado con el comando "python sise.py –export=Ruta"
- Se podrá importar la configuración de un ensayo a realizar con el comando "python sise.py –import=Ruta/Archivo"

▪ Menú:
- El sistema de menú tendrá una arquitectura de árbol
- La navegación entre los nodos del menú será consistente en todo el árbol
- Se indicará en todo momento el nodo actual y todos los nodos que lleven a la raíz del árbol

1.2. Con DUT:
- La comunicación con UART será en 9600 baudios, 8 bits de datos, 1 bit de parada y 0 bits de paridad
- La comunicación con el debugger se conformará con la configuración recomendada por el fabricante

2. Proceso de DUT:
- La comunicación con el debugger estará disponible durante todo el flujo de la secuencia
- Durante el flujo de la secuencia, la UART solo podrá transmitir información
- En el periodo entre secuencias, la UART podrá recibir y transmitir información

6.2. Funciones

1. CCI:
- Detendrá la secuencia de duración T del DUT en un momento t definido como $t \epsilon \mathbb{R}^+ \wedge \mathrm{t} < \mathrm{T}$
- Con la secuencia del DUT detenida, inyectará un SEFI-SEU que invertirá el valor de un bit de un registro interno
- La descripción del ensayo definirá el momento t de inyección de SEFI-SEU durante la secuencia de duración T y será un múltiplo de Δt definido como $\Delta t = T/N \, \forall \, N \epsilon \mathbb{N}$
- La descripción del ensayo definirá la cantidad M de registros involucrados en la prueba
- La cantidad de secuencias L a ejecutar quedará definida como $L = N \times M$
- Se ejecutará una secuencia de control sin inyección de SEFI-SEU antes de correr las L secuencias
- Por cada ejecución de una secuencia se obtendrá un valor de salida S del DUT
- Cada valor de salida S será persistido para su análisis
- Cada valor de salida S quedará asociado a su correspondiente secuencia con su inyección de SEFI-SEU y momento t
- Se generará un archivo de resultados llamado "resultados-AAAAMMDDHHmm.res", siendo AAAA el año del ensayo, MM el mes, DD el día, HH la hora y mm los minutos

- El archivo de resultados acumulará los SEFI y SEU de cada registro del DUT
- El archivo de resultados acumulará los SEU de cada periférico del DUT
- El archivo de resultados indicará el FOM del registro definido como: $FOM_{REG} = (1 - \frac{SEU}{SEFI})$
- El archivo de resultados indicará el FOM del DUT definido como: $FOM_{DUT} = \frac{1}{M} \times \sum_{i=1}^{i=M} FOM_i$ siendo i el número que representa un registro del DUT
- Se generará un archivo de histogramas llamado "histogramas-AAAAMMDDmm.his" siendo AAAA el año del ensayo, MM el mes, DD el día, HH la hora y mm los minutos
- El archivo de histogramas tendrá una tabla que indique la frecuencia de fallos como función de los SEFIs por registro del DUT
- El archivo de histogramas tendrá una tabla que indique la frecuencia de fallos como función de los SEFIs por periférico del DUT

2. Proceso de DUT:

- Deberá correr una secuencia de autoevaluación cuya ejecución durará un tiempo T
- Deberá producir una salida S que podrá ser un estado o una secuencia de estados
- Este proceso podrá tener una entrada E
- Deberá evaluar el estado de los periféricos del DUT
- Tendrá una función de evaluación para cada uno de los periféricos del DUT
- Se podrá definir por la entrada E si se desea excluir uno o más periféricos en la secuencia
- Manejará una interrupción del flujo normal de la secuencia y generará una salida S indicando la excepción, por ejemplo: interrupción por *watchdog*
- La salida S utilizará la UART del DUT para ser transmitida
- La entrada E utilizará la UART del DUT para ser recibida

6.3. Requisitos de rendimiento

- La inyección de SEFI-SEU podrá tener un desvío en su momento t de 10 ms
- El desvío tolerado de t deberá representar como máximo el 1 % de la duración T de la secuencia del DUT
- Aceptará un Δt que como mínimo represente el 5 % de la duración T de la secuencia del DUT

6.4. Restricciones de diseño

- Se utilizará como dispositivo principal el microcontrolador seleccionado por INVAP
- Se utilizará un sistema operativo de tiempo real para diseñar el Proceso de DUT

6.5. Atributos del sistema

1. Mantenibilidad:
   - El Proceso de DUT se desarrollará con un modelo de capas

Los requerimientos mencionados se detallan en el documento SISE-RS.

7. Historias de usuarios (*Product backlog*)

En esta sección se muestran dos historias de usuarios y su puntaje. El puntaje es una estimación del esfuerzo necesario para cumplir una historia. Para determinar la dificultad se consideran los siguientes factores:

- Complejidad ciclomática esperada antes del *refactoring*
- Cantidad de módulos necesarios
- Funciones que necesiten correr en tiempo real
- Dificultad para implementar *TDD* o *BDD*

El puntaje se expresa como un número primo entre 1 y 29. Entonces, existen 10 puntajes posibles. Sin embargo, una sucesión de números primos propone una escala no lineal. Finalmente, el número 1 corresponde a una historia trivial mientras que 29 se asigna a aquella que demande el máximo esfuerzo.

A continuación se enumeran las dos historias de usuario:

- Como *ingeniero de desarrollo* quiero *simular* los años previstos para una misión espacial en un periodo de 24 hs para *evaluar* las técnicas de mitigación de errores empleadas. *Story points: 7*, la historia de usuario requiere introducir *soft-errors* con una frecuencia proporcional al ensayo.
- Como *ingeniero de desarrollo* quiero *obtener* la figura de mérito del microcontrolador seleccionado para *decidir* si el dispositivo es adecuado para la misión. *Story points: 29*, la historia de usuario requiere realizar múltiples secuencias de autovalidación con inyecciones de *soft-errors* precisos.

8. Entregables principales del proyecto

Los entregables del proyecto son:

- Código fuente en el repositorio con control de versiones *Gitlab* de INVAP
- Documentación del código con *Doxygen*
- Documentación de las actividades semanales

9. Desglose del trabajo en tareas

1. Planificación
   1.1. Investigación de la problemática (10 hs)
   1.2. Relevamiento de las necesidades del cliente (15 hs)
   1.3. Exploración de las posibles tecnologías a emplear (20 hs)
   1.4. Investigación sobre el nivel de acceso posible al microcontrolador (5 hs)
   1.5. Creación del documento de especificaciones de requerimientos de software (15 hs)
   1.6. Elección final de las tecnologías e informe de justificación (15 hs)
   1.7. Confección de un documento de planificación del proyecto (20 hs)
   1.8. Reunión de control y aceptación de la documentación (1 hs)
   1.9. Creación del plan de pruebas y validación (20 hs)
   1.10. Reunión final de la planificación (1 hs)
2. Logística e infraestructura
   2.1. Establecimiento de un canal cifrado con el departamento de IT de INVAP (2 hs)
   2.2. Obtención de credenciales para acceder a la infraestructura de INVAP (2 hs)
   2.3. Obtención de credenciales para acceder al sistema de control de versiones de INVAP (2 hs)
   2.4. Configuración del entorno de trabajo conforme a los lineamientos de INVAP (3 hs)
   2.5. Obtención del hardware necesario para la realización del proyecto (10 hs)
3. Inyector por consola de comando
   3.1. Integración del servidor OCD (10 hs)
   3.2. Creacion de los archivos de configuración para el servidor OCD (30 hs)
   3.3. Pruebas de integración servidor OCD-Dispositivo bajo prueba (10 hs)
   3.4. Pruebas de rendimiento del servidor OCD (10 hs)
   3.5. Determinación de la media y desvío del error temporal de SEFI-SEU (5 hs)
   3.6. Creación de API para el servidor OCD (20 hs)
   3.7. Pruebas de aceptación de API para el servidor OCD (10 hs)
   3.8. Determinación y comparación de la media y desvío del error temporal de SEFI-SEU (5 hs)
   3.9. Servicio de interfaz serie para el dispositivo bajo prueba (20 hs)
   3.10. Pruebas de rendimiento de la interfaz serie (5 hs)
   3.11. Servicio de persistencia de datos (20 hs)
   3.12. Pruebas de aceptación del servicio de persistencia de datos (5 hs)
   3.13. Codificación del controlador de ensayos (40 hs)
   3.14. Pruebas del controlador de ensayos (10 hs)
   3.15. Validación manual de los informes generados (10 hs)
   3.16. Creación de la consola de usuario (15 hs)
   3.17. Creación de pruebas automáticas para la consola de usuario (10 hs)

3.18. Validación de la consola de usuario por parte del cliente (5 hs)

4. Proceso en DUT
   4.1. Diseño de la estrategia de autovalidación del estado general del DUT (25 hs)
   4.2. Diseño de la estrategia de la validación del estado de cada periférico (30 hs)
   4.3. Análisis de los posibles SEFI-SEU que alteren la secuencia de autovalidación general (25 hs)
   4.4. Diseño de estrategias de manejo de excepciones (25 hs)
   4.5. Reunión de control y aceptación del diseño (1 hs)
   4.6. Codificación del firmware (50 hs)
   4.7. Pruebas de manejo de excepciones (25 hs)
5. Procesos de cierre
   5.1. Pruebas automáticas de todo el sistema (6 hs)
   5.2. Obtención de la figura de mérito del microcontrolador evaluado (1 hs)
   5.3. Creación de la memoria del trabajo terminado (20 hs)
   5.4. Creación de la presentación para la defensa pública (5 hs)
   5.5. Creación del video para la defensa pública (5 hs)
   5.6. Defensa pública (0,5 hs)
   5.7. Agradecimiento a todos los involucrados (0,5 hs)

Cantidad total de horas: (600 hs)

10. Diagrama de Activity On Node

Armar el AoN a partir del WBS definido en la etapa anterior.

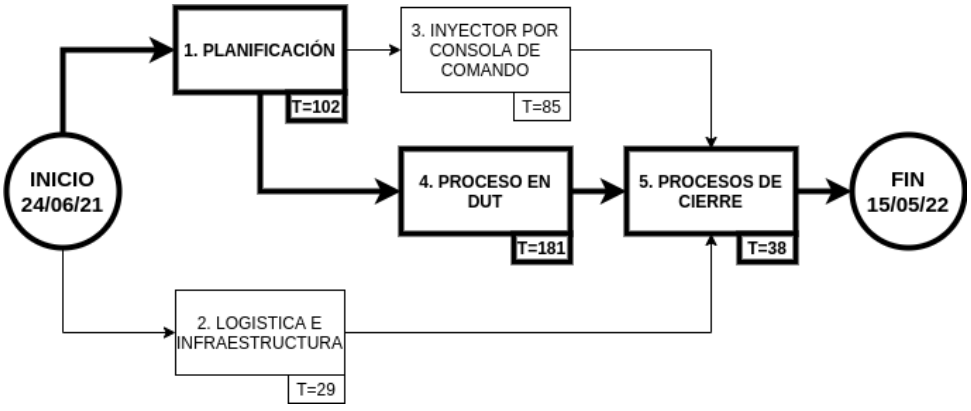

Figura 3. Diagrama en *Activity on Node*

Indicar claramente en qué unidades están expresados los tiempos. De ser necesario indicar los caminos semicríticos y analizar sus tiempos mediante un cuadro. Es recomendable usar colores y un cuadro indicativo describiendo qué representa cada color, como se muestra en el siguiente ejemplo:

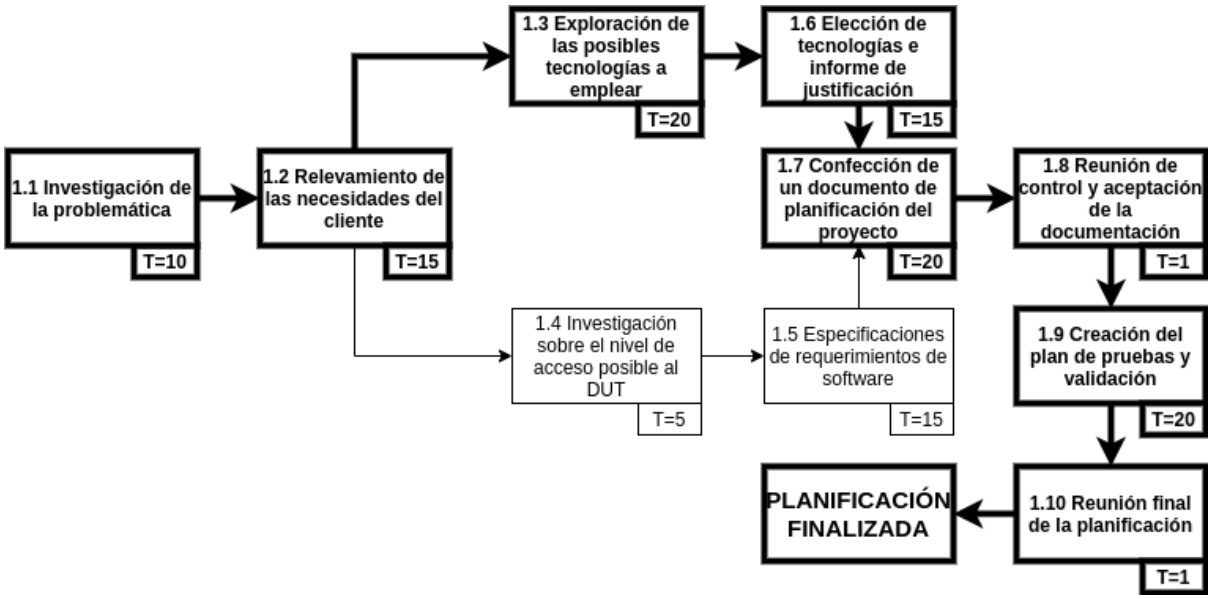

Figura 4. Diagrama en *Activity on Node*: planificación

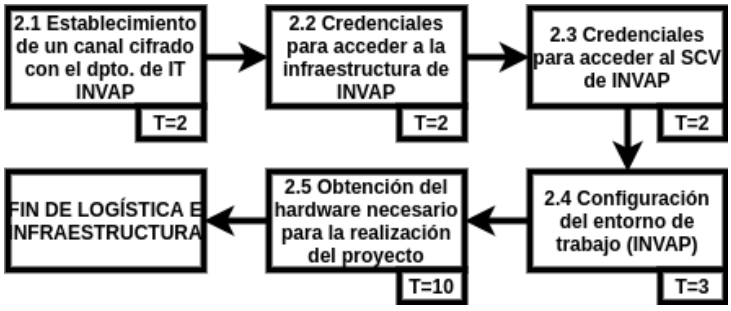

Figura 5. Diagrama en *Activity on Node*: logística e infraestructura

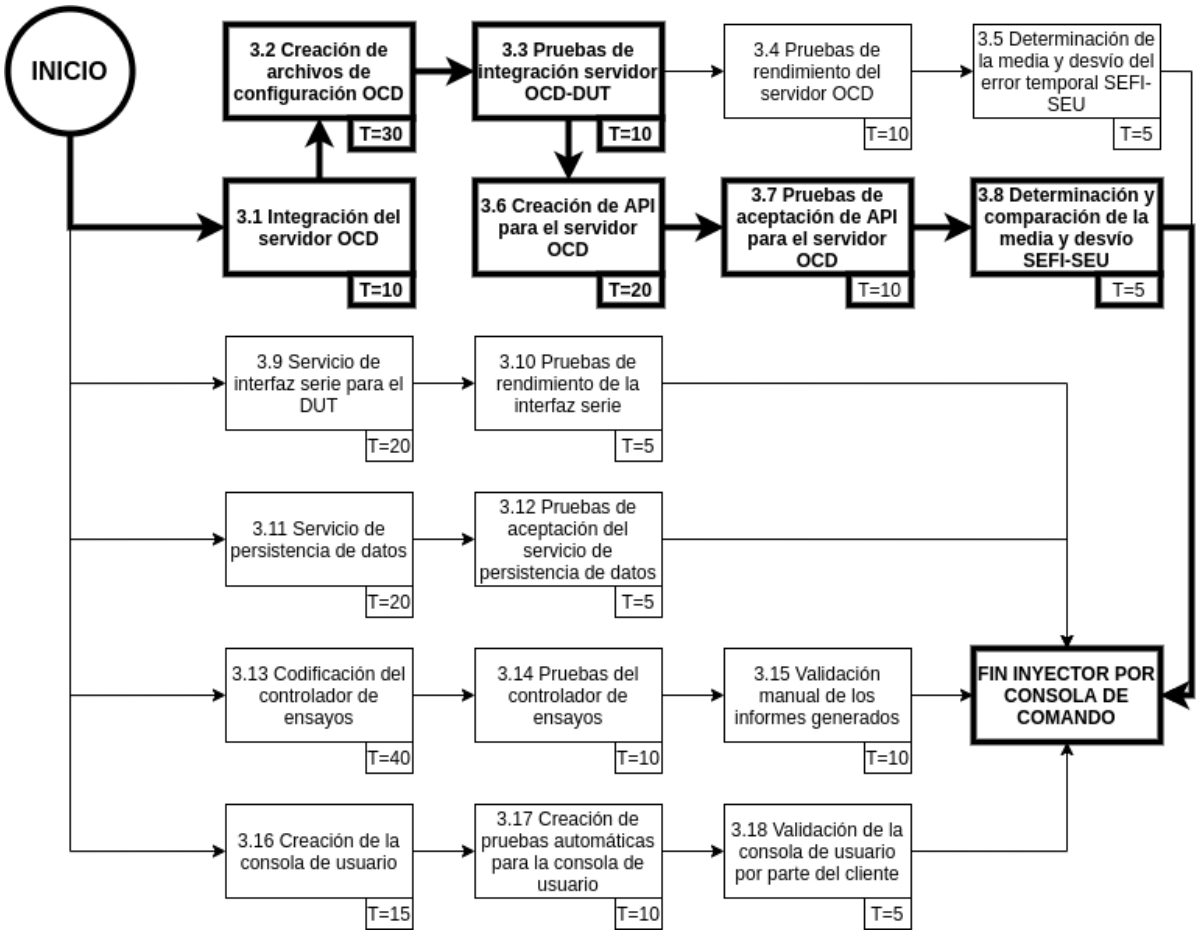

Figura 6. Diagrama en *Activity on Node*: inyector por consola de comando

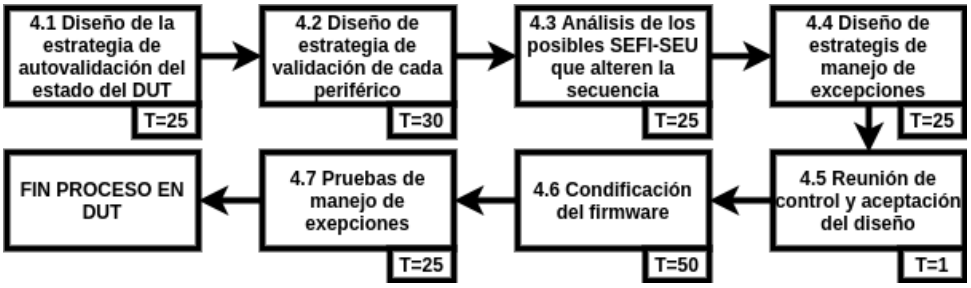

Figura 7. Diagrama en *Activity on Node*: proceso en DUT

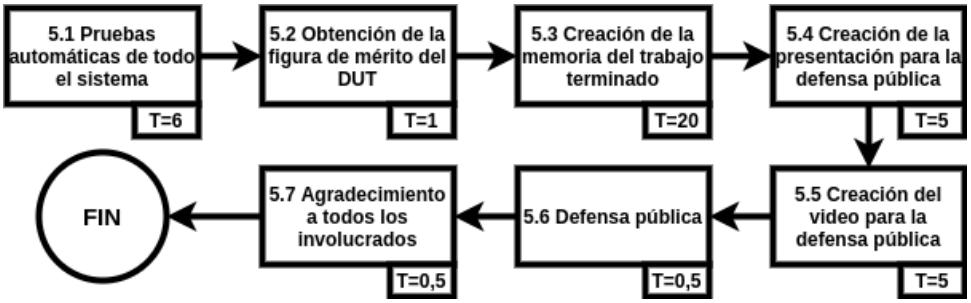

Figura 8. Diagrama en *Activity on Node*: procesos de cierre

11. Diagrama de Gantt

Existen muchos programas y recursos *online* para hacer diagramas de gantt, entre los cuales destacamos:

- Planner
- GanttProject
- Trello + *plugins*. En el siguiente link hay un tutorial oficial:
  `https://blog.trello.com/es/diagrama-de-gantt-de-un-proyecto`
- Creately, herramienta online colaborativa.
  `https://creately.com/diagram/example/ieb3p3ml/LaTeX`
- Se puede hacer en latex con el paquete *pgfgantt*
  `http://ctan.dcc.uchile.cl/graphics/pgf/contrib/pgfgantt/pgfgantt.pdf`

Pegar acá una captura de pantalla del diagrama de Gantt, cuidando que la letra sea suficientemente grande como para ser legible. Si el diagrama queda demasiado ancho, se puede pegar primero la "tabla" del Gantt y luego pegar la parte del diagrama de barras del diagrama de Gantt.

Configurar el software para que en la parte de la tabla muestre los códigos del EDT (WBS).
Configurar el software para que al lado de cada barra muestre el nombre de cada tarea.
Revisar que la fecha de finalización coincida con lo indicado en el Acta Constitutiva.

En la figura 8, se muestra un ejemplo de diagrama de gantt realizado con el paquete de *pgfgantt*. En la plantilla pueden ver el código que lo genera y usarlo de base para construir el propio.

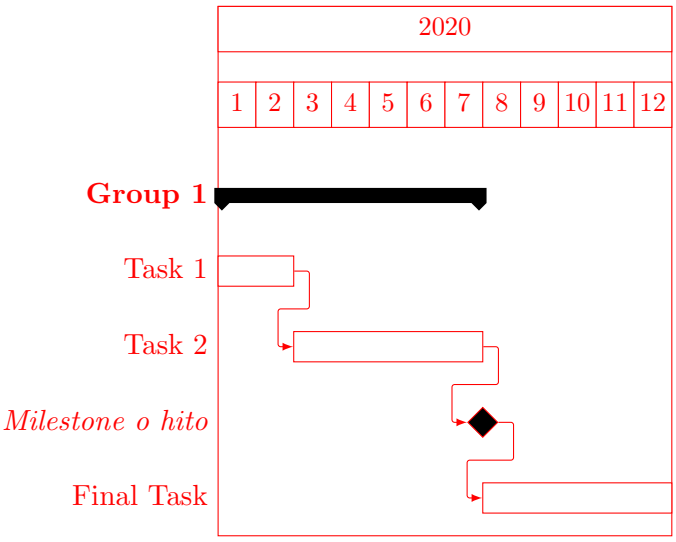

Figura 9. Diagrama de gantt de ejemplo

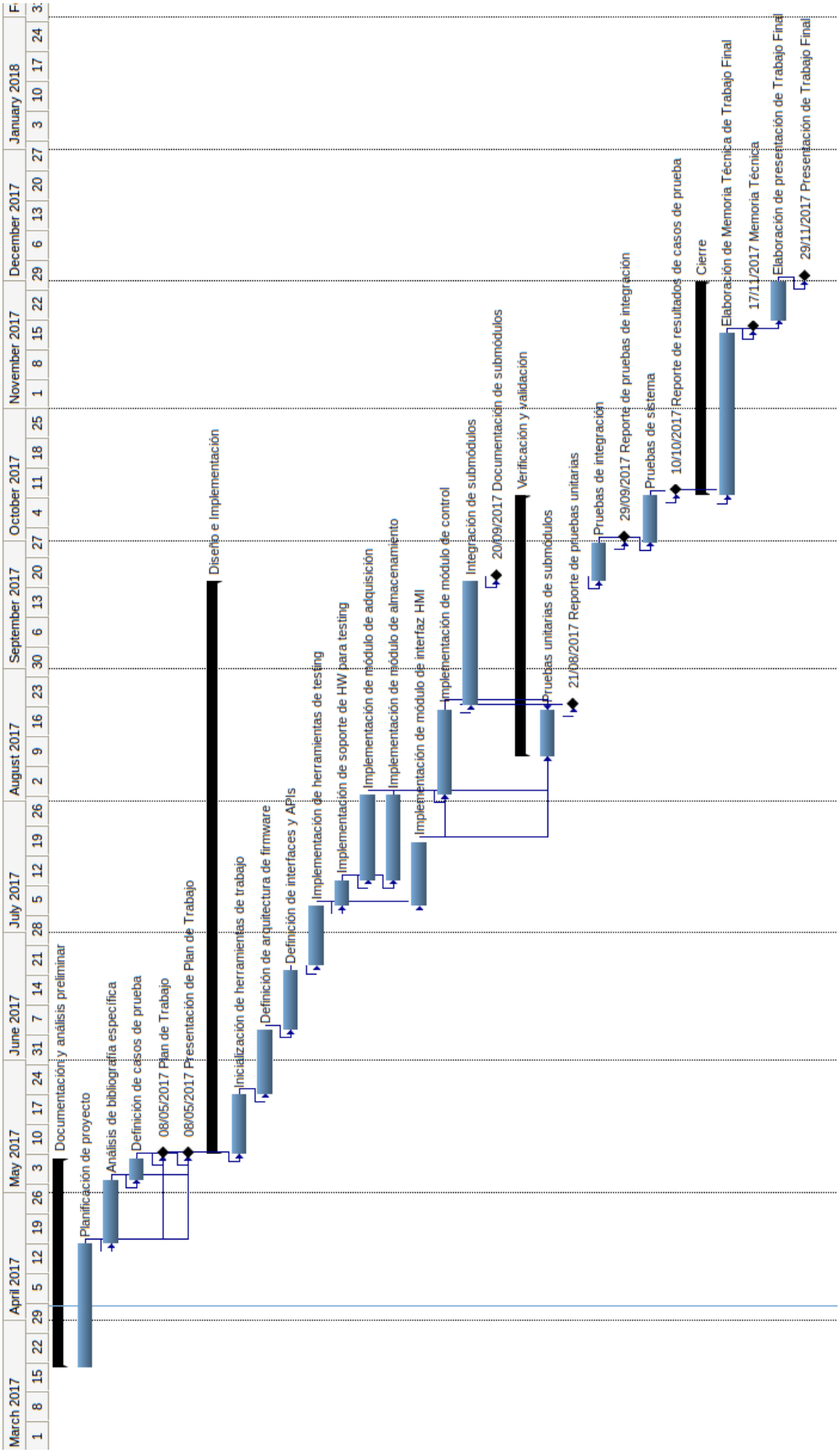

Figura 10. Ejemplo de diagrama de Gantt rotado

12. Presupuesto detallado del proyecto

Si el proyecto es complejo entonces separarlo en partes:

- Un total global, indicando el subtotal acumulado por cada una de las áreas.
- El desglose detallado del subtotal de cada una de las áreas.

IMPORTANTE: No olvidarse de considerar los COSTOS INDIRECTOS.

COSTOS DIRECTOS			
Descripción	Cantidad	Valor unitario	Valor total
SUBTOTAL			
COSTOS INDIRECTOS			
Descripción	Cantidad	Valor unitario	Valor total
SUBTOTAL			
TOTAL			

13. Gestión de riesgos

a) Identificación de los riesgos (al menos cinco) y estimación de sus consecuencias:

Riesgo 1: detallar el riesgo (riesgo es algo que si ocurre altera los planes previstos de forma negativa)

- Severidad (S): mientras más severo, más alto es el número (usar números del 1 al 10).
  Justificar el motivo por el cual se asigna determinado número de severidad (S).
- Probabilidad de ocurrencia (O): mientras más probable, más alto es el número (usar del 1 al 10).
  Justificar el motivo por el cual se asigna determinado número de (O).

Riesgo 2:

- Severidad (S):
- Ocurrencia (O):

Riesgo 3:

- Severidad (S):

- Ocurrencia (O):

b) Tabla de gestión de riesgos: (El RPN se calcula como RPN=SxO)

Riesgo	S	O	RPN	S*	O*	RPN*

Criterio adoptado: Se tomarán medidas de mitigación en los riesgos cuyos números de RPN sean mayores a...

Nota: los valores marcados con (*) en la tabla corresponden luego de haber aplicado la mitigación.

c) Plan de mitigación de los riesgos que originalmente excedían el RPN máximo establecido:

Riesgo 1: plan de mitigación (si por el RPN fuera necesario elaborar un plan de mitigación). Nueva asignación de S y O, con su respectiva justificación: - Severidad (S): mientras más severo, más alto es el número (usar números del 1 al 10). Justificar el motivo por el cual se asigna determinado número de severidad (S). - Probabilidad de ocurrencia (O): mientras más probable, más alto es el número (usar del 1 al 10). Justificar el motivo por el cual se asigna determinado número de (O).

Riesgo 2: plan de mitigación (si por el RPN fuera necesario elaborar un plan de mitigación).

Riesgo 3: plan de mitigación (si por el RPN fuera necesario elaborar un plan de mitigación).

14. Gestión de la calidad

Para cada uno de los requerimientos del proyecto indique:

- Req #1: copiar acá el requerimiento.
  - Verificación para confirmar si se cumplió con lo requerido antes de mostrar el sistema al cliente. Detallar
  - Validación con el cliente para confirmar que está de acuerdo en que se cumplió con lo requerido. Detallar

Tener en cuenta que en este contexto se pueden mencionar simulaciones, cálculos, revisión de hojas de datos, consulta con expertos, mediciones, etc. Las acciones de verificación suelen considerar al entregable como “caja blanca”, es decir se conoce en profundidad su funcionamiento interno. En cambio, las acciones de validación suelen considerar al entregable como “caja negra”, es decir, que no se conocen los detalles de su funcionamiento interno.

15. Procesos de cierre

Establecer las pautas de trabajo para realizar una reunión final de evaluación del proyecto, tal que contemple las siguientes actividades:

- Pautas de trabajo que se seguirán para analizar si se respetó el Plan de Proyecto original: - Indicar quién se ocupará de hacer esto y cuál será el procedimiento a aplicar.
- Identificación de las técnicas y procedimientos útiles e inútiles que se emplearon, y los problemas que surgieron y cómo se solucionaron: - Indicar quién se ocupará de hacer esto y cuál será el procedimiento para dejar registro.
- Indicar quién organizará el acto de agradecimiento a todos los interesados, y en especial al equipo de trabajo y colaboradores: - Indicar esto y quién financiará los gastos correspondientes.