Sistema de monitoreo y control de ambientes a distancia

Autor:
Ing. César Javier Fanelli

Director:
Ing. Fernando Lichtschein (FIUBA)

Esta planificación fue realizada en el curso de Gestión de proyectos entre el 21 de octubre de 2022 y el 8 de diciembre de 2022.

Índice

Registros de cambios

Revisión	Detalles de los cambios realizados	Fecha
1.0	Creación del documento	20/10/2022
1.1	Se completa hasta la sección 5 inclusive	3/11/2022
1.2	Se corrige hasta la sección 5 y se completa hasta la sección 9 inclusive	10/11/2022
1.3	Se corrige hasta la sección 9 y se completa hasta la sección 12 inclusive	17/11/2022
1.4	Se corrige hasta la sección 12 y se completa hasta la sección 15 inclusive (Se completa el plan)	24/11/2022

Acta de constitución del proyecto

Buenos Aires, 21 de octubre de 2022

Por medio de la presente se acuerda con el Ing. Ing. César Javier Fanelli que su Trabajo Final de la Carrera de Especialización en Internet de las Cosas se titulará "Sistema de monitoreo y control de ambientes a distancia", consistirá esencialmente en el diseño e implementación de un prototipo un sistema de control de ambientes por internet, y tendrá un presupuesto preliminar estimado de 642 horas de trabajo y $138344, con fecha de inicio 21 de octubre de 2022 y fecha de presentación pública a definir.

Se adjunta a esta acta la planificación inicial.

Dr. Ing. Ariel Lutenberg
Director posgrado FIUBA

Mg. Ing. Damián Corbalán
Movistar

Ing. Fernando Lichtschein
Director del Trabajo Final

1. Descripción técnica-conceptual del proyecto a realizar

Es sabido que la domótica, el monitoreo y el control de parámetros dentro de un ambiente está creciendo a grandes pasos. Tanto es así que empresas grandes como Google, Apple o Amazon desarrollaron sus propios sistemas de hogares inteligentes. Inclusive, están desarrollando en conjunto con otras empresas un sistema que sea compatible entre esas marcas, llamado Matter, de código abierto.

Así se puede ver el interés y hasta a veces la necesidad de conocer el estado y poder manejar parámetros a distancia de habitaciones, oficinas, salas de estar, y otros recintos. De esta manera se facilita la vida cotidiana y se pueden generar situaciones de confort con poco esfuerzo, dedicando el tiempo y la energía en otros aspectos de la vida cotidiana. Tan sólo con un click o pulsar una pantalla, se conocen o modifican los estados del sector que se desee.

Este proyecto, de índole personal o académica, surge de la idea principal de la tesis de grado de la carrera de la cual egresé en 2014 de ingeniero electrónico. El modelo anterior de sistema, era un prototipo básico de control inteligente de hogares, que si bien poseía comunicación inalámbrica y control de varios actuadores, carecía de varios aspectos, como ser, poder hacer el control mediante una PC o smartphone y la conexión a internet, que le aporta funcionalidad y versatilidad.

En la figura 1 se muestra un diagrama en bloques del sistema a desarrollar en el proyecto. Se observa que consta de 2 partes físicas diferenciables: el nodo, o actuador remoto, y el servidor backend, implementado en una Raspberry Pi. El nodo a su vez estará compuesto por el módulo de control (comandado por una placa de desarrollo con ESP32), un display para indicar los valores medidos y seteados, un potenciómetro digital para variar la temperatura, una salida de relé para el calefactor y una salida PWM para controlar una luminaria LED de 12 VCC. El usuario podrá acceder mediante una página web, con usuario y contraseña, a visualizar y cambiar los parámetros de la habitación donde esté instalado el sistema. En este caso, el sistema se implementará y probará con un solo nodo.

La diferencia con los sistemas ya existentes, es que sea fácilmente adaptable a entornos fuera de un hogar y una oficina, tales como una sala de servidores, un área limpia de un laboratorio, u otros lugares donde se requiera sensar distintos parámetros y poder además tener un historial de ellos.

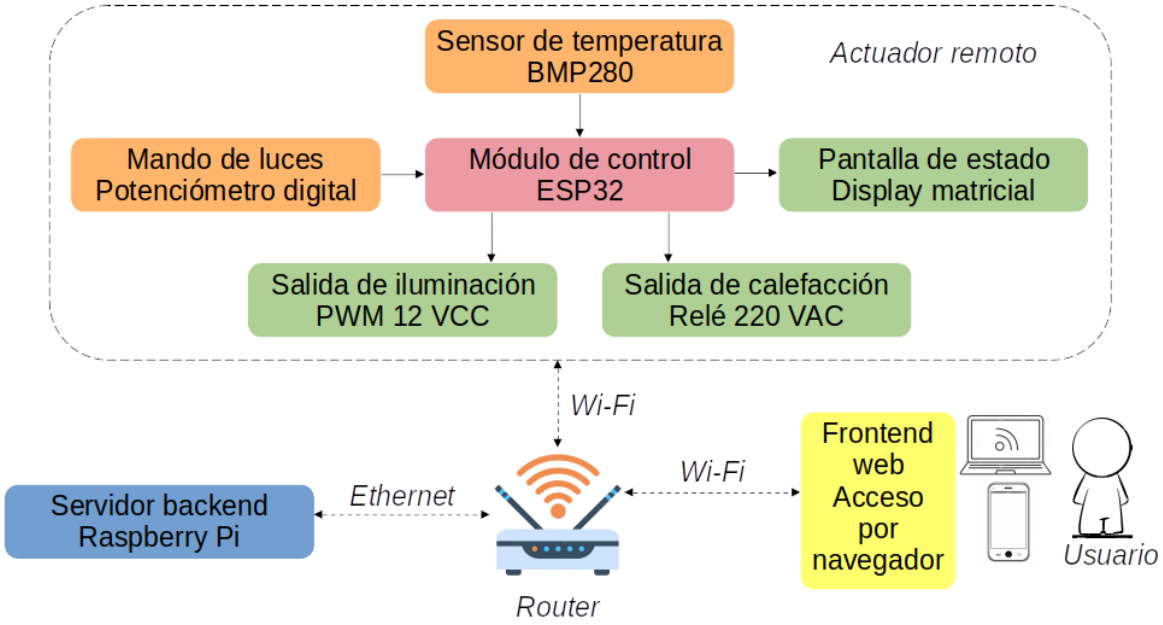

Figura 1. Diagrama en bloques del sistema.

2. Identificación y análisis de los interesados

Rol	Nombre y Apellido	Organización	Puesto
Cliente	Mg. Ing. Damián Corbalán	Movistar	Especialista de redes IP
Responsable	Ing. César Javier Fanelli	FIUBA	Alumno
Orientador	Ing. Fernando Lichtschein	FIUBA	Director Trabajo final
Usuario final	Usuarios que habiten o utilicen los recintos	-	-

- Cliente: Mg. Ing. Damián Rubén Corbalán. Es una persona exigente con muchos conocimientos en el tema.
- Responsable: Ing. César Javier Fanelli, único responsable a cargo del desarrollo del proyecto. Se ocupará de la planificación y ejecución de las tareas.
- Orientador: Ing. Fernando Lichtschein. Debido a su experiencia y conocimiento va a poder colaborar mucho con el desarrollo del proyecto y la resolución de los problemas que surjan en el camino.

3. Propósito del proyecto

El propósito de este proyecto es crear un prototipo de sistema de ambientes inteligentes, que sirva como puntapié inicial de un sistema más grande y más completo. Al ser personal la intención es diseñar un sistema que sea potencialmente escalable, al que se le puedan ir agregando

más módulos con distintas funcionalidades que sean de utilidad en la vida cotidiana. Dichas funciones futuras podrían ser controlar el acceso (que se lea desde un celular), integrar cámaras de seguridad o sensar y modificar de otros parámetros. El aprendizaje y el desafío son las motivaciones para hacer un proyecto de esta índole. Y si resultara siendo viable técnica y económicamente, en un futuro se analizará la posibilidad de un desarrollo de negocio.

4. Alcance del proyecto

El proyecto en cuestión incluye:

- Desarrollo y fabricación de un prototipo de placa que actúe una salida ON-OFF para controlar el calefactor y otra salida para la iluminación.
- Desarrollo e implementación de las conexiones entre la placa del punto anterior y la placa de desarrollo con el ESP32 (formando en conjunto el nodo).
- Desarrollo del software del nodo.
- Desarrollo del software del servidor.

El proyecto no incluye:

- Otros nodos cuyas funciones no sean las del ejemplo descripto en el diagrama en bloques.
- Software en Android o iOS para acceder a la plataforma.

5. Supuestos del proyecto

Para el desarrollo del presente proyecto se supone que:

- Se podrán conseguir todos los materiales necesarios para el armado del prototipo.
- Se contará con los recursos económicos para desarrollarlo con normalidad.
- No surgirán imprevistos personales que retrasen significativamente o pospongan el desarrollo del proyecto o la cursada del posgrado.
- Se adquirirán los conocimientos necesarios para la implementación del software y el sistema en su integridad.
- Se tendrá apoyo del director en temas netamente técnicos para el desarrollo.
- Se tendrá feedback constante del cliente para definir alcance, requerimientos y aprobar la funcionalidad del prototipo.
- El cliente cuenta con conexión a Internet y un hardware mínimo para conectar el sistema a su red local.

6. Requerimientos

Los requerimientos revisados en conjunto con el cliente y agrupados por afinidad son:

1. Requerimientos funcionales del sistema.
   1.1. El sistema deberá poder funcionar de forma automática o manual.
   1.2. El modo automático consistirá en poder programar horarios y niveles de las salidas, seteadas a través de la interfaz web.
   1.3. El modo manual consistirá en accionar las salidas desde la interfaz web o desde los comandos en el nodo.
   1.4. Al modificarse de forma manual cualquier parámetro, el nodo informará al servidor el nuevo valor del parámetro seteado.
   1.5. Al activar el modo manual, se interrumpirá el modo automático.
   1.6. Se guardarán los valores sensados y seteados en una base de datos dentro del servidor.
   1.7. Los usuarios podrán revisar los datos de los valores actuales e históricos.
2. Requerimientos asociados al nodo.
   2.1. El nodo estará implementado en una placa con un ESP32.
   2.2. Contará con un potenciómetro digital para elegir la temperatura e iluminación.
   2.3. Tendrá con un display que mostrará la temperatura sensada y el estado de las salidas.
   2.4. La frecuencia de sensado de la temperatura será de 1 minuto.
   2.5. Los niveles de tensión de las entradas y salidas lógicas serán adaptados a los valores lógicos correspondientes de los actuadores.
   2.6. Se alimentará con una fuente de 5 VCC y un cable microUSB.
3. Requerimientos asociados al servidor.
   3.1. El servidor estará montado en una Raspberry Pi con sistema operativo Linux.
   3.2. Alojará el código de la página web desde la que se ingresará al sistema.
   3.3. Alojará el código del backend del servidor y la base de datos.
   3.4. Se alimentará con una fuente de 5 VCC.
4. Requerimientos de la interfaz.
   4.1. Será intuitiva y sencilla de operar.
   4.2. Contará con logueo de usuario y contraseña para ver y cambiar parámetros.
5. Requerimientos opcionales (se llevarán a cabo si el tiempo es suficiente).
   5.1. El nodo contará con una salida de dimerización de 220 VCA para ventilación.
   5.2. El servidor (Raspberry Pi) funcionará como *access point* al que se conectará el nodo.

7. Historias de usuarios (*Product backlog*)

A continuación se enumeran las historias de usuario y se las pondera según la serie de Fibonacci. Los valores elegidos para la dificultad son 3, 5 y 8; para la complejidad 3, 5 y 8 y para la incertidumbre 2, 3 y 5.

- Como usuario deseo poder monitorear la temperatura en tiempo real de la habitación para saber si hace falta encender la calefacción.
  Dificultad: 5 - Complejidad: 3 - Incertidumbre: 3 - Total: 11 - Valor más cercano 13
- Como usuario deseo que el sistema funcione igualmente sin internet para garantizar el confort en la sala.
  Dificultad: 3 - Complejidad: 3 - Incertidumbre: 2 - Total: 8 - Valor más cercano 8
- Como usuario quiero que el sistema dé aviso por mail si la temperatura alcanza valores extremos para salvaguardar la integridad de los bienes.
  Dificultad: 5 - Complejidad: 5 - Incertidumbre: 3 - Total: 13 - Valor más cercano 13
- Como usuario quiero programar la regulación de las luces en horarios programados para simular la presencia de personas en una casa que no está siempre habitada.
  Dificultad: 5 - Complejidad: 8 - Incertidumbre: 3 - Total: 16 - Valor más cercano 21
- Como usuario quiero almacenar los valores sensados y los estados de las salidas para poder generar un posible ahorro de energía eléctrica.
  Dificultad: 5 - Complejidad: 5 - Incertidumbre: 3 - Total: 13 - Valor más cercano 13
- Como usuario quiero poder controlar las salidas desde un mando dentro de la habitación para no depender de una PC o teléfono. Dificultad: 8 - Complejidad: 8 - Incertidumbre: 3 - Total: 19 - Valor más cercano 21

8. Entregables principales del proyecto

Los entregables del proyecto son:

- Diagrama de conexión del nodo
- Diagrama de circuito esquemático de accesorios del nodo
- Código fuente del nodo (ESP32)
- Código fuente del frontend y del backend
- Informe final

9. Desglose del trabajo en tareas

Las tareas se han agrupado por tema y ordenado por secuencialidad.

1. Investigación y búsqueda de material (45 h)

1.1. Búsqueda y análisis de soluciones similares en el mercado (5 h)
1.2. Búsqueda de códigos fuente de ejemplo similares (10 h)
1.3. Búsqueda y análisis de los materiales a usar (5 h)
1.4. Listado de materiales necesarios y aspectos de red (5 h)
1.5. Estudio de funcionamiento básico de cada componente del sistema (10 h)
1.6. Estudio y diseño de diagrama en bloques del sistema (5 h)
1.7. Organizar la documentación y archivar lo importante (5 h)

2. Adquisición de materiales (15 h)

2.1. Búsqueda de materiales (3 h)
2.2. Análisis de alternativas (2 h)
2.3. Armado y análisis de costos (5 h)
2.4. Negociación con proveedores y compra de materiales (5 h)

3. Diseño y puesta en marcha de hardware (85 h)

3.1. Instalación de sistema operativo Linux en servidor (5 h)
3.2. Pruebas de funcionalidad básicas (5 h)
3.3. Diseño de circuitos esquemático e impreso del hardware adicional al nodo (10 h)
3.4. Armado y pruebas del hardware adicional (20 h)
3.5. Conexión y pruebas del hardware adicional con ESP32 (40 h)
3.6. Documentar brevemente lo sucedido en este punto (5 h)

4. Instalación y pruebas de la red local (10 h)

4.1. Instalación y configuración de la red local (3 h)
4.2. Pruebas iniciales de conexión del servidor a la red (2 h)
4.3. Pruebas de conexión del nodo a la red (3 h)
4.4. Documentar brevemente lo sucedido en este punto (2 h)

5. Desarrollo del software del servidor (frontend y backend) (292 h)

5.1. Instalación y configuración del software de la base de datos (5 h)
5.2. Instalación y configuración del servidor web (5 h)
5.3. Pruebas avanzadas de conexión a la red local e internet (2 h)
5.4. Creación del software para la verificación constante de conexión a internet (5 h)
5.5. Creación del software para conocer la hora y temperatura local por internet (5 h)
5.6. Creación del script para correr los programas automáticamente al encender el servidor (5 h)
5.7. Diseño e implementación de la base de datos (10 h)
5.8. Diseño y desarrollo de las funcionalidades del backend (40 h)
5.9. Pruebas de integración con la base de datos (10 h)
5.10. Desarrollo y configuración del código para enviar mails (10 h)
5.11. Diseño y maquetación de la página web (20 h)
5.12. Desarrollo de la página web (40 h)

5.13. Pruebas de funcionalidad de la página web (20 h)
5.14. Integración de la página web con el software backend (30 h)
5.15. Resolución de problemas de integración (20 h)
5.16. Resolución de problemas de funcionamiento (40 h)
5.17. Diseño de reportes a presentar por la aplicación (20 h)
5.18. Documentar brevemente lo sucedido en este punto (5 h)

6. Desarrollo del software de nodo (100 h)
   6.1. Diseño de base de software de inicialización y configuración del nodo (5 h)
   6.2. Diseño de software de lectura de entradas y actuación de salidas (20 h)
   6.3. Diseño de software de comunicación con el servidor (30 h)
   6.4. Pruebas de comunicación con el servidor (20 h)
   6.5. Resolución de problemas (20 h)
   6.6. Documentar brevemente lo sucedido en este punto (5 h)
7. Instalación y pruebas funcionales (25 h)
   7.1. Instalación del nodo y adicionales en la habitación (4 h)
   7.2. Pruebas funcionales de los comandos locales dentro de la habitación (3 h)
   7.3. Análisis y resolución de posibles fallas (3 h)
   7.4. Pruebas reales de funcionalidad en un ambiente (5 h)
   7.5. Pruebas de alarmas y funcionalidad de aplicación web (5 h)
   7.6. Documentar brevemente lo sucedido en este punto (5 h)
8. Elaboración de documentación (70 h)
   8.1. Redacción del manual de uso (10 h)
   8.2. Redacción del informe de avance (10 h)
   8.3. Elaboración de la memoria técnica (50 h)

Cantidad total de horas: (642 h).
Si en los cálculos anteriores llegara a haber sobrante de tiempo, se puede analizar diseñar e implementar el control de la salida de 220 VCA para controlar la ventilación. Dicha duración de tareas extra (opcionales) se estima de la siguiente manera:

- Diseño e implementación del control de ventilación mediante salida dimerizable (50 h)
  1. Diseño de placa de control de la salida de 220 VCA (10 h)
  2. Armado de placa para dimerizar salida de 220 VCA (10 h)
  3. Conexión al nodo y pruebas (5 h)
  4. Modificación del software del servidor (5 h)
  5. Modificación del software del nodo (20 h)

10. Diagrama de Activity On Node

A continuación se muestra el diagrama *Activity on Node*. Las unidades de tiempo están expresadas en horas (h). El camino crítico está marcado en color rojo. La duración total del camino crítico es de 612 horas.
Por la forma de diagramar las tareas en el WBS y al ser el único recurso para hacer todas las actividades, resulta difícil desdoblar la gran mayoría de ellas, por lo que el resultado es el expresado en la imagen.

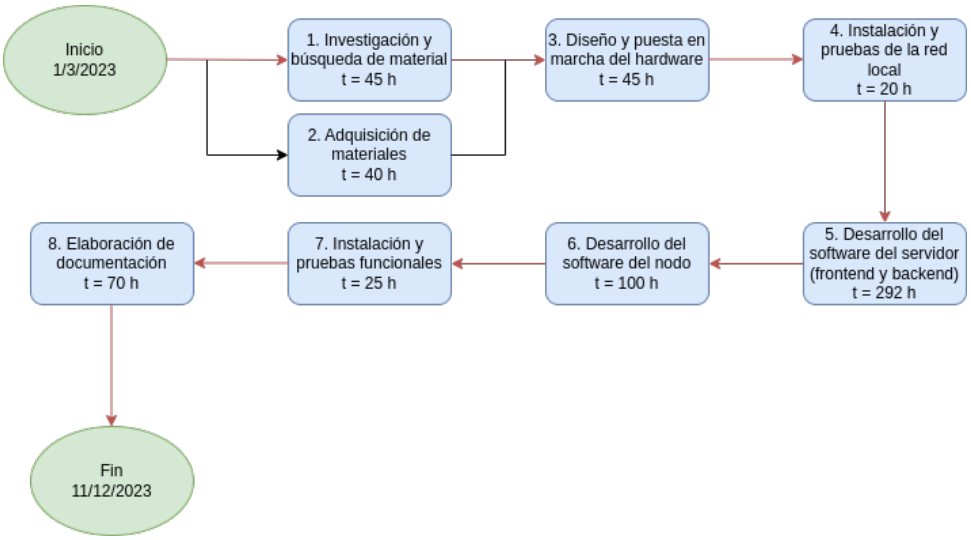

Figura 2. Diagrama en *Activity on Node*.

11. Diagrama de Gantt

A continuación se observa el diagrama de Gantt subdividido en varias partes para que sea legible. La carga horaria semanal promedio se hizo teniendo en cuenta días laborales y días no laborales. En promedio, en días laborales se trabajará 10 horas semanales y en fin de semana se trabajará en promedio 6 horas, dando un total en promedio una carga horaria semanal de 16 hs. Esta carga variará además en base la carga que tengan las materias del posgrado.
En la primera imagen se ven los 8 grupos de tareas con la duración total de la realización del proyecto. Luego se ve el desglose en varias imágenes, agrupando las tareas en cuanto a similitud o temas comunes.

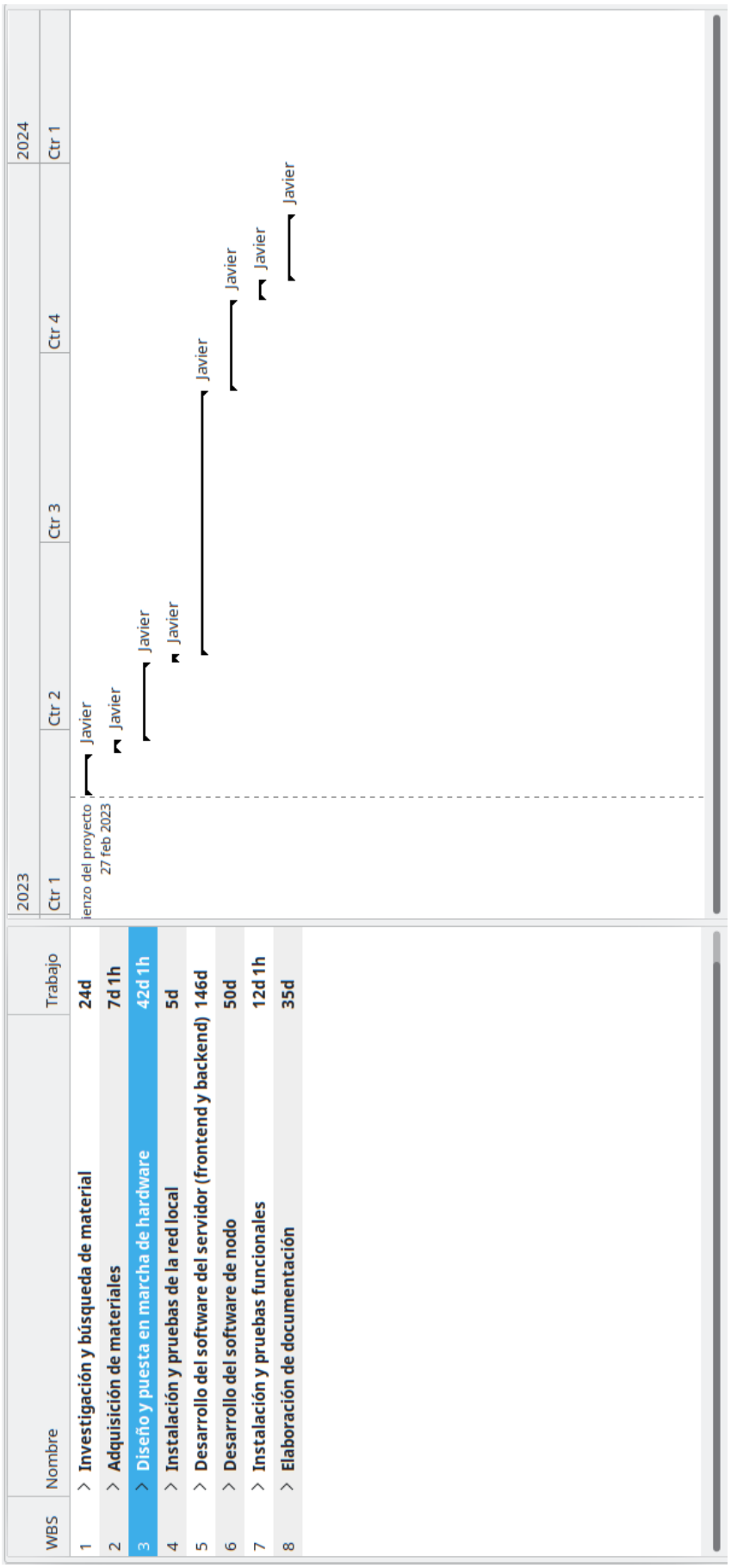

Figura 3. Resumen de gurpos de tareas.

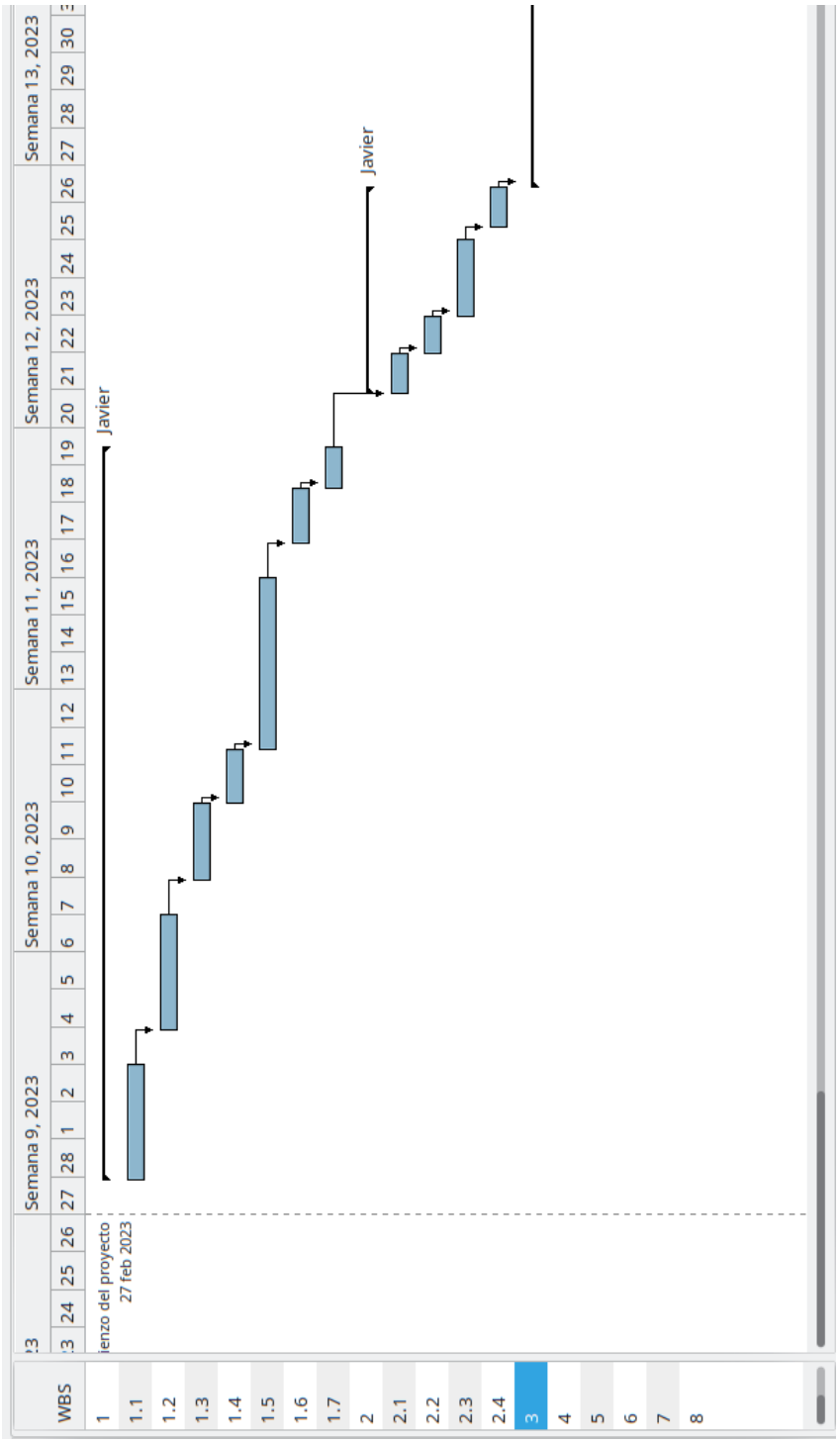

Figura 4. Inicio y pasos preliminares.

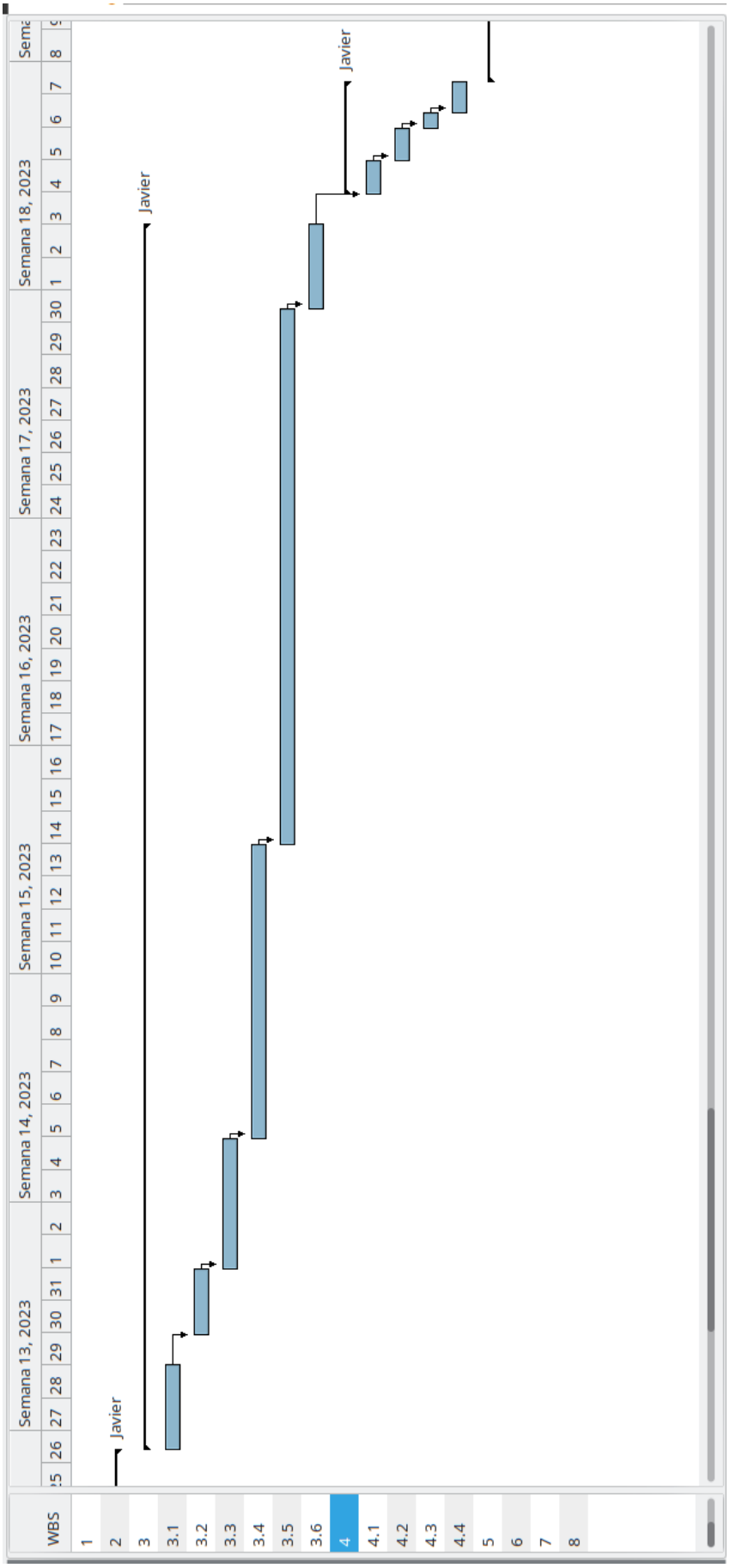

Figura 5. Adecuación del hardware.

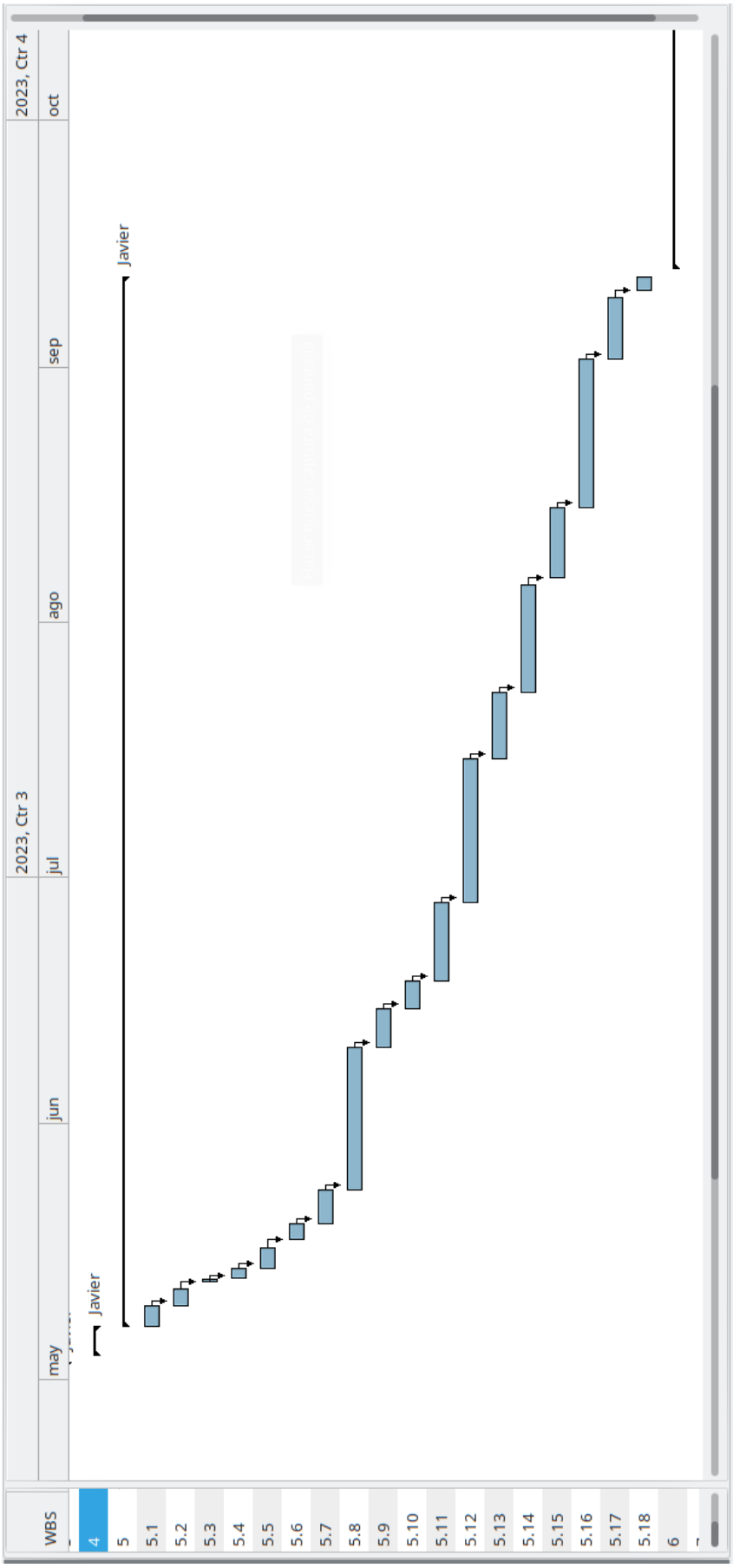

Figura 6. Desarrollo del software del servidor.

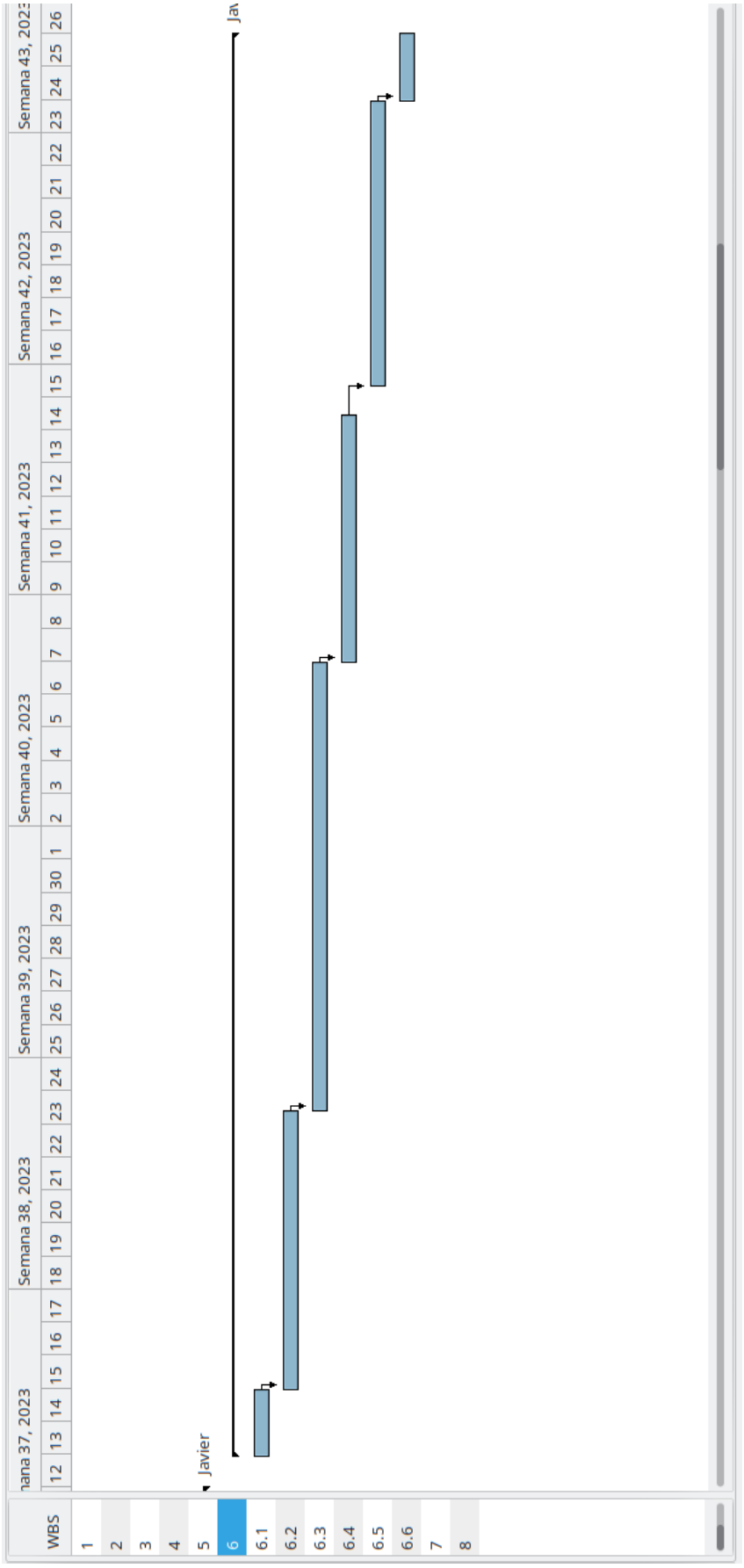

Figura 7. Desarrollo del software del nodo.

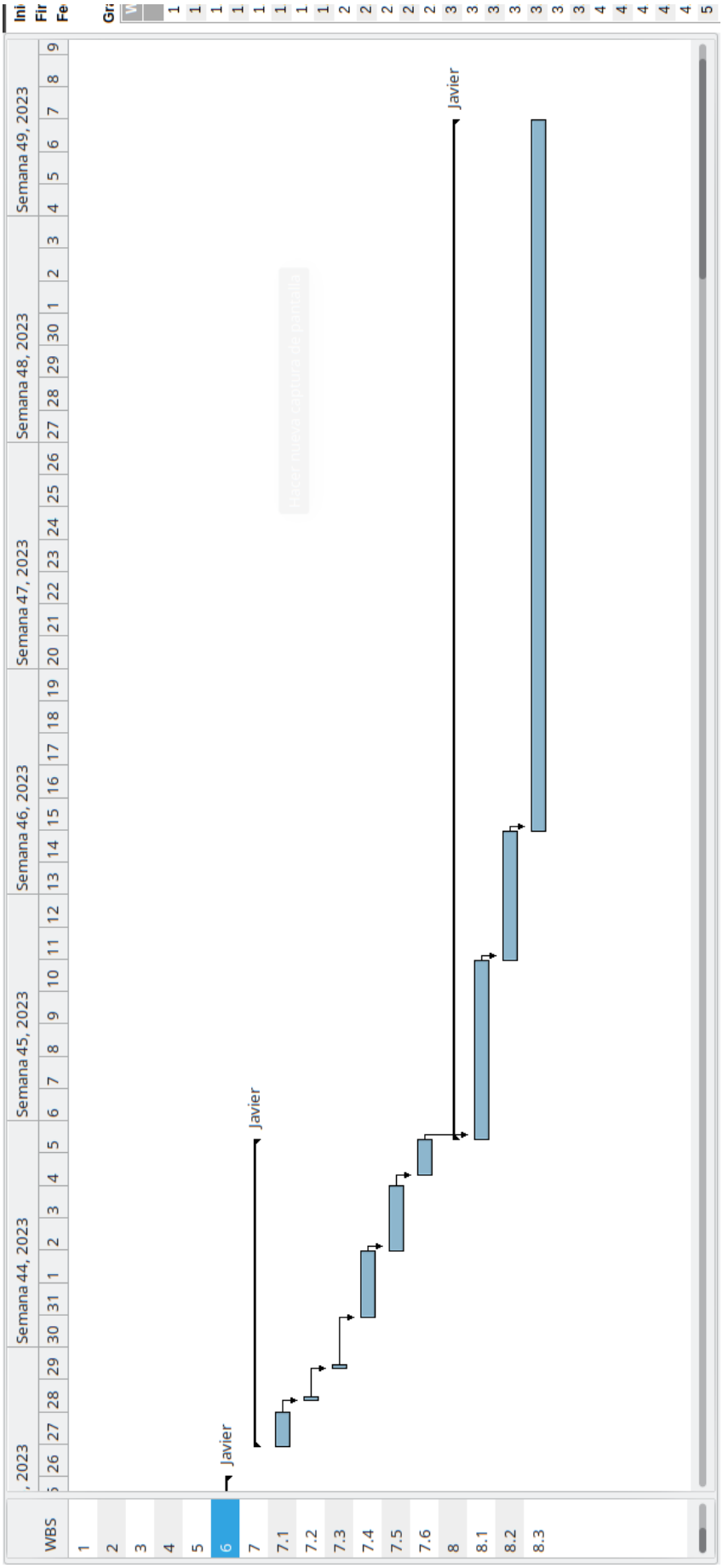

Figura 8. Pruebas finales y desarrollo de documentación.

12. Presupuesto detallado del proyecto

COSTOS DIRECTOS			
Descripción	Cantidad	Valor unitario	Valor total
Placa desarrollo ESP32C3	1	$7239	$7239
Display Oled 1.3 pulgadas Azul 128x64 I2C	1	$2685	$2685
Encoder rotativo con pulsador	1	$360	$360
Placa experimental perforada simple faz 7x9 cm	2	$779	$1558
Sensor de presión y temperatura I2C BMP280	1	$404	$404
Cables, conectores y componentes genéricos	1	$2500	$2500
Fuente 5 VCC 500 mA miniUSB	1	$500	$500
Raspberry Pi 4 Model B 4 GB	1	$80000	$80000
Fuente 5 VCC 3000 mA USB-C	1	$500	$500
Memoria Micro SD 32 GB 100 MB/s	1	$3500	$3500
SUBTOTAL			$99246
COSTOS INDIRECTOS			
Descripción	Cantidad	Valor unitario	Valor total
40 % de los costos directos	1	$39098	$39098
SUBTOTAL			$39098
TOTAL			$138344

El presupuesto anterior da un total estimado de $138344. Ante el escenario inflacionario actual los precios varían mes a mes, por lo que se estima el presupuesto en dólares. Tomando un valor de $169 por dólar al 17/11/2022, el mismo da un estimado de USD818,60.

13. Gestión de riesgos

a) Identificación de los riesgos y estimación de sus ocurrencias:

Los riesgos listados a continuación son ponderados del 1 al 10 en severidad (S) y probabilidad de ocurrencia (O).

Riesgo 1: desabastecimiento de componentes claves del sistema.

- Severidad (8): de no conseguir los componentes más importantes del sistema habrá que buscar opciones y re adaptar parte del código, lo cual introduciría demoras.
- Ocurrencia (7): hay escasez a nivel mundial las placas Raspberry Pi y Argentina está en un período de dificultades con las importaciones.

Riesgo 2: incompatibilidad del hardware adicional con la placa del ESP32.

- Severidad (4): el hardware no funciona de la manera esperada por incompatibilidad entre las tensiones, corrientes o frecuencias.
- Ocurrencia (4): se tiene experiencia previa en desarrollo de hardware similar y son problemas frecuentes.

Riesgo 3: demoras en el desarrollo del software del servidor.

- Severidad (7): es el software con más carga de trabajo a realizar y más complejo del sistema y sin este no hay funcionalidad en una plataforma de IoT.
- Ocurrencia (7): no se tiene la experiencia necesaria al momento del desarrollo del presente documento para hacer la tarea en su totalidad.

Riesgo 4: problemas de comunicación entre el nodo y el servidor.

- Severidad (6): si bien el nodo podrá programarse de forma local, perdería la capacidad de tomar parámetros pasados de forma remota y el servidor perdería los parámetros que está sensando el nodo.
- Ocurrencia (4): al ser programados en plataformas y lenguajes distintos, es probable que haya problemas de envío o interpretación de datos.

Riesgo 5: inestabilidad en la conexión con la red.

- Severidad (5): la conexión del nodo a la red es indispensable para poder leer las entradas y modificar las salidas.
- Ocurrencia (3): al estar conectado de forma inalámbrica puede ser que la señal de *Wi-Fi* sea débil en la ubicación del nodo.

Riesgo 6: daño de la tarjeta de memoria del servidor.

- Severidad (9): en la tarjeta de memoria de la Raspberry Pi se guardan todos los datos tomados, corre el sistema operativo y todo el software desarrollado en el proyecto.
- Ocurrencia (4): al tener varias lecturas y escrituras se corre el riesgo de que se fatigue la memoria micro SD y se pierda la información.

Riesgo 7: imposibilidad de hacer una instalación básica para pruebas.

- Severidad (8): sin una instalación básica de pruebas no se puede hacer la depuración completa del prototipo.
- Ocurrencia (2): las instalaciones se pueden hacer en lugares de forma básica y hasta en una maqueta.

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

Criterio adoptado: Se tomarán medidas de mitigación en los riesgos cuyos números de RPN sean mayores a 25 puntos. 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: desabastecimiento de componentes claves del sistema.

- Plan de mitigación: adelantar las compras para cuando se tenga tiempo y los recursos materiales.

Riesgo	S	O	RPN	S*	O*	RPN*
1. Desabastecimiento de componentes claves del sistema	8	7	56	8	2	16
2. Incompatibilidad del hardware adicional con la placa del ESP32	4	4	16			
3. Demoras en el desarrollo del software del servidor	7	7	49	7	3	21
4. Problemas de comunicación entre el nodo y el servidor	6	4	24			
5. Inestabilidad en la conexión con la red	5	3	15			
6. Daño de la tarjeta de memoria del servidor	9	4	36	3	4	12
7. Imposibilidad de hacer una instalación básica para pruebas	8	2	16			

- Severidad (8): La severidad sigue siendo la misma, ya que si no se dispone del hardware no se puede desarrollar el prototipo para el proyecto.
- Ocurrencia (2): hacia fines del 2022 todavía hay stock de la placa Raspberry Pi y ESP32, por lo que se planea hacerlo en breve.

Riesgo 3: demoras en el desarrollo del software del servidor.

- Plan de mitigación: se buscará información y se capacitará, se comenzarán con pruebas previas al desarrollo del proyecto durante los meses diciembre, enero y febrero previos al inicio del desarrollo del proyecto.
- Severidad (7): la severidad sigue siendo la misma ya que por más que se adelanten tareas, sin este desarrollo el sistema no es de IoT.
- Ocurrencia (3): al adelantar la adquisición de algunos conocimientos, los preparativos del entorno de la Raspberry Pi y probablemente alguna parte del código bajan las chances de sufrir demoras.

Riesgo 6: daño de la tarjeta de memoria del servidor.

- Plan de mitigación: se harán backups cada semana o cambio importante de la información creada y código desarrollado, y se creará una imagen de la memoria.
- Severidad (3): se disminuye la severidad ya que si se dañara o se perdiera información el único perjuicio es incurrir en una demora para la recuperación de la información.
- Ocurrencia (4): la probabilidad de ocurrencia es la misma ya que puede pasar de igual manera, y se trabaja en mitigar la severidad del riesgo creando copias.

14. Gestión de la calidad

1. Requerimientos funcionales del sistema.

   1.1. El sistema deberá poder funcionar de forma automática o manual.
   - Verificación: se creará el software con 2 modos, incluyendo un script para conocer la hora local y adicionando al nodo hardware para la modificación de parámetros.
   - Validación: se validará la existencia de los modos desde la aplicación web.

1.2. El modo automático consistirá en poder programar horarios y niveles de las salidas, seteadas a través de la interfaz web.
   - Verificación: se incorporará un script para saber la hora local al software y se programarán funciones con horarios.
   - Validación: se programará funciones con horarios y se validará el accionamiento de las salidas.

1.3. El modo manual consistirá en accionar las salidas desde la interfaz web o desde los comandos en el nodo.
   - Verificación: se analizarán las hojas de datos del ESP32 y del hardware utilizado realizando pruebas funcionales.
   - Validación: se accionarán los controles del nodo modificando los parámetros, validando el cambio de las salidas.

1.4. Al modificarse de forma manual cualquier parámetro, el nodo informará al servidor el nuevo valor del parámetro seteado.
   - Verificación: se incorporará una función de envío de parámetros dentro del software del nodo y se verificará el envío de datos.
   - Validación: se accionarán los controles viendo al mismo tiempo que se modifiquen en la aplicación web.

1.5. Al activar el modo manual, se interrumpirá el modo automático.
   - Verificación: se incorporará una función en el backend para que dé aviso al nodo el cambio del modo a automático y se verificará el envío de datos.
   - Validación: se programará el modo automático y se verificará de cambio del modo en el nodo.

1.6. Se guardarán los valores sensados y seteados en una base de datos dentro del servidor.
   - Verificación: se verificará el envío periódico de datos del nodo al servidor informando el estado de las salidas y los sensores conectados y muestra de estos valores.
   - Validación: muestra de los valores en tiempo real en la aplicación web.

1.7. Los usuarios podrán revisar los datos de los valores actuales e históricos.
   - Verificación: se incorporará de una base de datos en el servidor que almacene los parámetros y se verificará la cantidad y calidad de estos datos.
   - Validación: se presentarán en pantalla a través de la aplicación web los parámetros almacenados en la base de datos.

2. Requerimientos asociados al nodo.

2.1. El nodo estará implementado en una placa con un ESP32.
   - Verificación: se estudiarán los aspectos necesarios en las hojas de datos de la placa ESP32.
   - Validación: se adjuntarán imágenes del nodo con la placa conectada dentro de él.

2.2. Contará con un potenciómetro digital para elegir la temperatura e iluminación.
   - Verificación: se estudiarán las hojas de datos del potenciómetro, se medirá con osciloscopio o similar y se verificará el funcionamiento en placa de prueba.
   - Validación: se accionará el potenciómetro para modificar parámetros.

2.3. Tendrá con un display que mostrará la temperatura sensada y el estado de las salidas.

- Verificación: se estudiarán las hojas de datos del display y se verificará que se muestren los datos.
- Validación: se mostrará la lectura de pantalla con información y corroboración visual del estado de las salidas.

2.4. La frecuencia de sensado de la temperatura será de 1 minuto.
- Verificación: se programará el software del nodo para tomar lectura de los sensores canda un minuto usando timers internos y se verificará el envío de datos en ese intervalo de tiempo.
- Validación: se mostrarán los datos sensados en la pantalla del nodo y en la aplicación web, actualizándose con cada lectura y envío de información.

2.5. Los niveles de tensión de las entradas y salidas lógicas serán adaptados a los valores lógicos correspondientes de los actuadores.
- Verificación: se utilizarán transistores bipolares o FET para conmutar las salidas de la placa del ESP32. Se verificarán las hojas de datos de todos los componentes.
- Validación: se detectará y mostrará el accionamiento correcto de las salidas.

2.6. Se alimentará con una fuente de 5 VCC y un cable microUSB.
- Verificación: se estudiarán las hojas de datos de la placa del ESP32.
- Validación: se validará el encendido y correcto funcionamiento.

3. Requerimientos asociados al servidor.

3.1. El servidor estará montado en una Raspberry Pi con sistema operativo Linux.
- Verificación: se comprobará el funcionamiento del sistema operativo con conexión a una pantalla y se harán algunas pruebas de software.
- Validación: se adjuntarán imágenes del hardware del servidor.

3.2. Alojará el código de la página web desde la que se ingresará al sistema.
- Verificación: se desarrollará el software frontend dentro de la memoria que alojará la Raspberry Pi.
- Validación: se mostrará el encendido del servidor y se accederá a la aplicación web.

3.3. Alojará el código del backend del servidor y la base de datos.
- Verificación: se desarrollará el software backend dentro de la memoria que alojará la Raspberry Pi.
- Validación: se comprobarán los datos recopilados y guardados en la base de datos.

3.4. Se alimentará con una fuente de 5 VCC.
- Verificación: se estudiarán las hojas de datos de la Raspberry Pi.
- Validación: se mostrará un esquema de conexión del hardware del servidor.

4. Requerimientos de la interfaz.

4.1. Será intuitiva y sencilla de operar.
- Verificación: se desarrollará la aplicación web en base a experiencias de usuario y se harán pruebas funcionales con clientes potenciales.
- Validación: se demostrará la aplicación web dando un breve recorrido y explicación y se escucharán sugerencias y reclamos.

4.2. Contará con logueo de usuario y contraseña para ver y cambiar parámetros.
- Verificación: se desarrollará y probará el frontend y del backend con un usuario y clave.
- Validación: se creará de un usuario para que el cliente pueda ver los estados y datos almacenados.

15. Procesos de cierre

- Análisis de las pautas de trabajo del desarrollo del Plan de Proyecto.
  - Responsable: Ing. César Javier Fanelli.
  - Participación: Mg. Ing. Damián Rubén Corbalán (cliente).

  En este proceso se analizarán: el cumplimiento los requerimientos y los entregables en conjunto con el cliente; que se haya respetado la fecha de entrega y si hubo desvíos en la duración de las tareas con respecto al diagrama de Gantt planteado;