Universidad de las Ciencias Informáticas
Vertex, Facultad de Tecnologías Interactivas

Aplicación para el manejo remoto de un brazo robótico para la siembra automática de posturas

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor: Keily Marrero Martínez

Tutores: Dr.C Omar Correa Madrigal

Ing.A Julio Alberto Leyva Durán

La Habana, 2024

Dedicatoria

Para mi familia por todo su apoyo incondicional

Agradecimientos

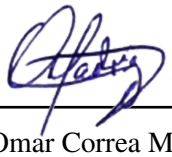
A mi amigo Yordenis, a mi amiga Tania, a Harold y a Roger.Gracias por ser mi familia en el campus. A Yosiel por todo su apoyo. A Dani. Gracias por todo.

Declaración de autoría

Declaro ser autor de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales sobre esta, con carácter exclusivo.

Para que así conste firmamos la presente a los 5 días del mes de diciembre del año 2024.

Keily Marrero Martínez
Autor



Dr.C Omar Correa Madrigal
Tutor

Ing.A Julio Alberto Leyva Durán
Tutor

Resumen

Esta investigación se centra en el desarrollo de un sistema de manejo remoto para un brazo robótico, con el objetivo de facilitar tareas automatizadas en entornos agrícolas y de jardinería. El objeto de estudio abarca la integración de hardware y software, buscando mejorar el proceso de siembra y solventar la escacez de mano de obra. Para llevar a cabo esta investigación, se empleó una metodología ágil basada en eXtreme Programming (XP), que permitió una iteración continua en el diseño y desarrollo del sistema, así como la implementación de pruebas de aceptación para validar las funcionalidades. Los principales hallazgos incluyen la efectividad del sistema en el control remoto del brazo robótico sin embargo, también se presentan dificultades técnicas. Los resultados obtenidos demuestran que a pesar de que el sistema cumple con la mayoría de requisitos funcionales, aún no está listo para su uso en entornos agrícolas. En conclusión, la investigación resalta la importancia de un enfoque estructurado y colaborativo en el desarrollo de tecnologías robóticas, evidenciando su potencial para aportar a la productividad y eficiencia en aplicaciones prácticas dando como resultado final un prototipo funcional que puede ser mejorado en proyectos posteriores.

Palabras clave: Brazo robótico,control remoto,automatización agrícola.

Índice general

Índice de figuras

Índice de tablas

Introducción

La agricultura enfrenta desafíos significativos que amenazan su sostenibilidad y capacidad para alimentar a una población mundial en crecimiento. Entre los problemas más críticos se encuentran el cambio climático, la escacez de mano de obra y la creciente demanda de alimentos. Estos factores no solo afectan la producción agrícola, sino que también impactan negativamente en la vida de millones de pequeños agricultores que dependen de la tierra para su subsistencia. Con una población agrícola envejecida y una migración significativa hacia las ciudades, hay una creciente escasez de mano de obra en el campo. Los brazos robóticos pueden realizar tareas repetitivas y físicamente exigentes, permitiendo que los agricultores se concentren en aspectos más estratégicos de su trabajo.Un brazo robótico para la siembra no solo representa una innovación tecnológica, sino que también se presenta como una solución viable ante los múltiples desafíos que enfrenta la agricultura moderna. Su implementación podría transformar radicalmente las prácticas agrícolas, haciendo frente a problemas críticos como la degradación del suelo, la escasez de recursos hídricos y sobre todo la escases de recursos humanos.

Se tiene como **problema científico:** ¿cómo automatizar la siembra de posturas?

Se define como **objeto de estudio:** la robótica. Este incluye sistemas de control, y la integración de tecnologías como sensores y actuadores que permiten su operación remota. Se analizarán aspectos como la cinemática, el control de movimientos, y la interacción con el entorno, se define como **campo de acción:** : las aplicaciones para el manejo remoto de brazos robóticos para la siembra.

El **objetivo general:** desarrollar un sistema de manejo remoto para un brazo robótico que permita realizar tareas de siembra de forma automática.

Para abordar el problema de investigación sobre la siembra automatizada de posturas, se proponen las siguientes **Tareas de investigación:**

- Elaboración del marco teórico de la investigación a través del estudio del estado del arte para una mejor comprensión de la investigación.
- Investigación de tecnologías existentes en el control remoto de brazos robóticos, como sistemas de comunicación como 5G y Bluetooth.
- Análisis de estudios previos sobre aplicaciones agrícolas de brazos robóticos y su efectividad en la siembra.
- Implementación un prototipo funcional del brazo robótico, considerando aspectos mecánicos y electrónicos.

- Desarrollo de un sistema de control que permita manejar el brazo robótico a distancia.
- Realización de pruebas para la validación del sistema.

Como **Métodos de investigación científica** s a utilizar con el fin de solucionar las tareas planteadas anteriormente tenemos:

Métodos teóricos:

- Histórico-lógico: Este método permitirá realizar un análisis del desarrollo histórico de la robótica agrícola, enfocándose en la evolución de los brazos robóticos y su aplicación en la siembra. Se estudiarán las tendencias actuales en tecnología robótica y control remoto, así como los avances en sistemas de comunicación y automatización que son relevantes para la implementación del brazo robótico. Este enfoque ayudará a identificar las características necesarias y deseables que debe tener el sistema propuesto, asegurando que se alinee con las mejores prácticas y tecnologías disponibles.
- Revisión Documental: La revisión documental será importante para fundamentar el propósito de la investigación. Se recopilarán estudios previos, artículos académicos, y documentos técnicos que aborden el uso de brazos robóticos en la agricultura, así como investigaciones sobre control remoto y teleoperación. Esta revisión permitirá desarrollar un marco teórico sólido que respalde la propuesta de investigación, además de identificar vacíos en el conocimiento actual que el proyecto podría abordar.
- Modelación: El método de modelación se utilizará para diseñar y simular el funcionamiento del brazo robótico. A través de modelos teóricos y prácticos, se podrán confeccionar representaciones del sistema que integren los componentes mecánicos, electrónicos y de software necesarios para su operación. Este proceso ayudará a prever el comportamiento del brazo robótico en diferentes escenarios de siembra, permitiendo ajustar su diseño antes de la construcción del prototipo.

Métodos empíricos:

- Observación: La observación se empleará para analizar diferentes brazos robóticos ya existentes y sus aplicaciones en la agricultura. Al observar cómo funcionan estos sistemas en entornos reales, se podrán identificar características clave y elementos fundamentales que deben ser considerados en el diseño del nuevo brazo robótico. Esta información será valiosa para establecer comparaciones entre las soluciones existentes y la propuesta, garantizando que se aborden las limitaciones actuales.

Estructura de la investigación:

- **Capítulo 1: Fundamentación Teórica.** Este capítulo se encargará de definir los elementos teóricos necesarios para el desarrollo de la investigación sobre el manejo remoto de un brazo robótico para la siembra automática de posturas. Se incluirán definiciones y explicaciones sobre brazos robóticos, control remoto, y siembra automatizada, análisis de soluciones similares en el ámbito de la robótica agrícola, incluyendo estudios previos y tecnologías actuales, selección y justificación de la metodología de desarrollo de software que se utilizará en el proyecto, así como las herramientas y tecnologías específicas (como lenguajes de programación, plataformas de control, etc.).

- **Capítulo 2: Características y Diseño del Sistema.** Este capítulo detallará las características del sistema propuesto y su diseño. Incluirá descripción de las funcionalidades que debe cumplir el brazo robótico, como la siembra automatizada, la capacidad de detectar posturas, y el control remoto. Consideraciones sobre rendimiento, usabilidad, seguridad y escalabilidad del sistema. Presentación de diagramas y artefactos que representen el diseño del sistema, incluyendo diagramas de flujo, diagramas de clases y esquemas del sistema de control.
- **Capítulo 3: Implementación y pruebas.** Este capítulo se centrará en la implementación del brazo robótico y las pruebas necesarias para garantizar su correcto funcionamiento. Se abordarán los siguientes aspectos: Descripción de los estándares y buenas prácticas que se seguirán durante la implementación del software que controla el brazo robótico. Detalles sobre el proceso de implementación del prototipo físico y la integración del software con el hardware. Planificación y ejecución de diferentes tipos de pruebas (unitarias, integradas, funcionales) para verificar que el sistema cumpla con los requisitos establecidos. Se incluirán resultados obtenidos durante las pruebas para evaluar la efectividad del sistema en situaciones reales.

CAPÍTULO 1

Fundamentación Teórica

Este capítulo se organiza en varias secciones que abordan los fundamentos teóricos y metodológicos relacionados con el manejo remoto de brazos robóticos. Se explorarán las tecnologías utilizadas, el estado actual del objeto de estudio y las variables que afectan su implementación. También se presentarán casos concretos que ilustran cómo estas herramientas están siendo utilizadas en diferentes contextos agrícolas. Al finalizar este capítulo, se espera proporcionar una visión integral sobre el impacto que el manejo remoto de brazos robóticos puede tener en la agricultura moderna. La investigación no solo busca validar la efectividad técnica de estas herramientas, sino también explorar cómo su integración puede contribuir a un futuro más sostenible para las prácticas agrícolas.

1.1. Fundamentos teórico-metodológicos de la robótica aplicada a la agricultura

El manejo remoto de brazos robóticos para la siembra de posturas se fundamenta en una serie de teorías y metodologías que han evolucionado a lo largo del tiempo. Estos fundamentos abarcan desde los principios básicos de la robótica hasta las técnicas avanzadas de control y automatización. La robótica es una disciplina que integra mecánica, electrónica e informática para diseñar máquinas capaces de realizar tareas específicas. Los brazos robóticos están compuestos por varios componentes, incluidos actuadores, sensores y sistemas de control.

Agricultura 4.0

La agricultura 4.0 representa una evolución significativa en las prácticas agrícolas, integrando tecnologías avanzadas para mejorar la eficiencia, sostenibilidad y productividad en el sector. Algunas de sus ventajas son: (Rubio, 2021)

- Eficiencia Productiva: La implementación de tecnologías avanzadas puede aumentar la productividad en un 20 por ciento mientras se reduce el desperdicio de recursos hasta en un 30 por ciento.

- Sostenibilidad Ambiental: Al optimizar el uso de insumos como agua y fertilizantes, se minimiza el impacto ambiental, contribuyendo a prácticas más sostenibles.
- Toma de Decisiones Informadas: El análisis de datos permite a los agricultores anticipar problemas como plagas o enfermedades, mejorando la respuesta ante estos desafíos.
- Reducción de Costos: Aunque la inversión inicial en tecnología puede ser alta, los ahorros derivados de una gestión más eficiente pueden compensar rápidamente esos costos.

Conceptos Clave de la Agricultura 4.0

- Agricultura de Precisión: Este enfoque se basa en la observación y respuesta a las variabilidades dentro de los campos, utilizando datos para optimizar el uso de recursos como agua, fertilizantes y pesticidas. Esto se logra mediante tecnologías como GPS y sensores.
- Internet de las Cosas (IoT): La interconexión de dispositivos que recopilan y comparten datos en tiempo real, permitiendo un monitoreo constante de las condiciones del cultivo y del entorno.
- Big Data y Análisis de Datos: La recopilación y análisis de grandes volúmenes de datos generados por diversas tecnologías para tomar decisiones informadas sobre prácticas agrícolas.
- Inteligencia Artificial (IA): El uso de algoritmos que permiten a las máquinas aprender de los datos y hacer predicciones o tomar decisiones automatizadas.

Internet de las Cosas

El Internet de las Cosas (Iot, por sus siglas en inglés) es un concepto que se refiere a la interconexión de objetos físicos a través de Internet, permitiendo que estos dispositivos se comuniquen entre sí y con sistemas centrales para recopilar, enviar y recibir datos. Esta red de dispositivos incluye desde electrodomésticos y vehículos hasta sensores industriales y dispositivos de salud(Gates, 2023).

Beneficios del IoT en la Agricultura

- Eficiencia Mejorada: El uso del IoT permite a los agricultores reducir costos operativos al optimizar el uso de recursos como agua y fertilizantes, lo que resulta en una producción más sostenible.
- Toma de Decisiones Basada en Datos: La recopilación continua de datos permite a los agricultores tomar decisiones informadas que pueden mejorar los rendimientos y reducir riesgos relacionados con plagas o condiciones climáticas adversas.
- Aumento en la Productividad: Al utilizar tecnologías avanzadas, los agricultores pueden aumentar su producción al mismo tiempo que minimizan el impacto ambiental.
- Sostenibilidad: La implementación del IoT promueve prácticas agrícolas más sostenibles al reducir el uso excesivo de recursos y minimizar el desperdicio.

Brazos Robóticos

Los brazos robóticos son dispositivos mecánicos que imitan la función de un brazo humano, diseñados para realizar tareas específicas en diversas industrias. Su versatilidad y precisión los han convertido en herramientas esenciales en la automatización de procesos(Rubio, 2021).

Tipos de Brazos Robóticos

- Articulados: Estos brazos cuentan con múltiples articulaciones que les permiten moverse en diferentes direcciones. Son ideales para tareas complejas como soldadura y pintura.
- SCARA (Selective Compliance Assembly Robot Arm): Con cuatro ejes, estos brazos son rápidos y precisos, utilizados comúnmente en aplicaciones de ensamblaje y manipulación de materiales.
- Cartesianos: Se mueven en un sistema de coordenadas tridimensional (X, Y, Z) y son utilizados para tareas que requieren movimientos lineales, como el mecanizado y la impresión 3D.
- Cilíndricos: Combinan movimientos verticales y rotacionales, siendo adecuados para tareas de ensamblaje y soldadura en espacios reducidos.
- Paralelos/Delta: Con un diseño único que permite alta velocidad y precisión, estos brazos son ideales para tareas ligeras y rápidas, como el empaquetado.
- Colaborativos (Cobots): Diseñados para trabajar junto a humanos, estos brazos incluyen características de seguridad que reducen el riesgo de accidentes en entornos compartidos.

Beneficios

- Eficiencia Aumentada: Los brazos robóticos pueden operar a velocidades superiores a las humanas, lo que reduce el tiempo de producción.
- Precisión Mejorada: Son capaces de realizar tareas repetitivas con una alta precisión, minimizando errores y mejorando la calidad del producto final.
- Seguridad Laboral: Al asumir tareas peligrosas o físicamente exigentes, los brazos robóticos reducen el riesgo de lesiones entre los trabajadores.
- Reducción de Costos: Aunque la inversión inicial puede ser alta, los ahorros a largo plazo por la reducción de mano de obra y mejora en la calidad pueden ser significativos (Medeiros, 2023).

Manejo Remoto

Los brazos robóticos pueden ser controlados de diversas maneras, lo que permite su uso en una variedad de aplicaciones. Los diferentes tipos de manejo remoto para brazos robóticos ofrecen versatilidad y adaptabilidad a diversas aplicaciones industriales, médicas y agrícolas. La elección del método depende del tipo de tarea a realizar, los requisitos de precisión y la naturaleza del entorno operativo. Con el avance continuo de las tecnologías, se espera que estos métodos se vuelvan aún más sofisticados y accesibles en el futuro(Trujillo, 2023).

Tipos de Manejo Remoto

- Control Remoto Directo: Este método implica que un operador controle el brazo robótico en tiempo real a través de un dispositivo, como un joystick o una interfaz gráfica en una computadora. Este tipo de control es común en aplicaciones donde se requiere una alta precisión y respuesta inmediata, como en la cirugía robótica o en tareas de ensamblaje.
- Control por Gestos: El control por gestos utiliza tecnologías de reconocimiento de movimiento para permitir que el usuario controle el brazo robótico mediante gestos físicos. Este enfoque proporciona

una interacción más intuitiva y natural, permitiendo que los operadores realicen acciones sin necesidad de dispositivos físicos. Se ha demostrado su efectividad en entornos donde la rapidez y la precisión son cruciales.

- Control a través de Software: Los brazos robóticos pueden ser programados mediante software especializado que permite al usuario definir secuencias de movimientos y tareas. Este método es útil para operaciones repetitivas, donde la programación puede optimizar la eficiencia y reducir errores. Los usuarios pueden crear scripts o utilizar interfaces gráficas para programar las acciones del brazo.
- Control Inalámbrico: El uso de tecnologías como Wi-Fi, Bluetooth o módulos XBee permite el control remoto del brazo robótico sin necesidad de cables. Esto no solo facilita la movilidad del operador, sino que también reduce el riesgo de tropiezos o accidentes causados por cables enredados. La comunicación inalámbrica es especialmente útil en entornos industriales donde los brazos robóticos deben operar en espacios amplios.
- Control a través de Redes 5G: La implementación de redes 5G ofrece la posibilidad de controlar brazos robóticos con baja latencia y alta capacidad de transmisión de datos. Esto es especialmente relevante para aplicaciones críticas donde el tiempo de respuesta es fundamental, como en la telecirugía o en operaciones remotas complejas. La tecnología 5G permite una comunicación más fluida entre el operador y el brazo robótico, mejorando la precisión del control.
- Control Híbrido: Este enfoque combina diferentes métodos de control, como la programación predefinida junto con la intervención manual. Por ejemplo, un brazo robótico puede seguir un programa específico pero permitir ajustes manuales en tiempo real si es necesario. Este tipo de control es útil en situaciones donde se requiere flexibilidad y adaptabilidad.

En la presente investigación el control inalámbrico adquiere gran relevancia como potencial solución.

1.2. Sistematización de fundamentos teóricos-metodológicos y referentes internacionales

El manejo remoto de brazos robóticos se basa en una serie de teorías que abarcan diversas disciplinas, incluyendo la ingeniería mecánica, electrónica, informática y agronomía. Estos fundamentos permiten entender cómo los brazos robóticos pueden ser utilizados para mejorar la eficiencia y precisión en la siembra de posturas. La teoría de control es fundamental para el funcionamiento de los brazos robóticos. Esta teoría se ocupa del comportamiento dinámico de sistemas y permite diseñar algoritmos que regulan el movimiento del robot. Por ejemplo, los controladores) son comúnmente utilizados para ajustar la posición y velocidad del brazo robótico, asegurando que las tareas se realicen con precisión. La robótica móvil es otra área relevante, ya que muchos brazos robóticos están montados sobre plataformas móviles que les permiten desplazarse por el campo. La integración de sensores como GPS y LIDAR permite a estos robots navegar de forma autónoma, identificando obstáculos y ajustando su trayectoria en tiempo real.

1.2.1. Referentes internacionales

Los referentes internacionales son cruciales para entender cómo se ha desarrollado el manejo remoto de brazos robóticos en diferentes contextos. A nivel internacional, países como Japón, Suecia y España han sido pioneros en la investigación y aplicación de tecnologías robóticas en la agricultura.
Nissan, el fabricante de coches japonés, ha desarrollado un prototipo que puede significar el vínculo definitivo entre la tecnología y los métodos de cultivo más tradicionales: el pato robot. El dispositivo surca el agua a través de los cultivos para dificultar la fotosíntesis de las malas hierbas e impedir su crecimiento.El pato robot emplea Wi-Fi, baterías, energía solar y un sistema de navegación GPS para navegar entre los campos de arroz. El dispositivo tiene un peso de 1,5kg y una superficie de aproximadamente 60 centímetros, un diseño que se asemeja al de los aspiradores de hogar Roomba. Su tecnología de conexión Wi-Fi puede ser siginificativa para el prototipo de brazo (Claver, 2021).

Figura 1.1. Pato Robot

Robot YuMi:Un robot automatizado que va a ayudar a repoblar hasta 22.000 hectáreas de la selva amazónica. Se trata de un proyecto piloto realizado por la empresa ABB Robotics y la ONG estadounidense Junglekeepers. El robot, YuMi, trabajará con tecnología en la nube para tratar de realizar la reforestación de la Amazonía más eficiente y rápidamente. El robot YuMi automatizará las tareas de plantación en un laboratorio de la jungla, acelerando el proceso y sembrando él mismo las semillas. Además, estará controlado desde Västerås, Suecia, a unos 12.000 kilómetros desde donde operará en la Amazonía peruana. Será, por tanto, el robot más remoto del mundo (HuamÃ¡n, 2023).
YuMi ayudará a los voluntarios a sembrar semillas. El proceso comenzará por cavar un hoyo, dejar caer la

semilla, compactar el suelo y marcar el lugar con una etiqueta codificada por colores. Con la ayuda de este robot se podrá replantar cada día el tamaño equivalente a dos campos de fútbol(Scaliter, 2023).

Figura 1.2. Robot Yumi

El impacto social y económico del manejo remoto de brazos robóticos es significativo, especialmente en comunidades rurales donde la agricultura es una fuente principal de ingresos.La implementación de brazos robóticos ha demostrado aumentar significativamente la productividad agrícola al permitir una siembra más rápida y precisa. Esto no solo reduce costos operativos sino que también mejora la calidad del cultivo al asegurar una distribución uniforme.

AINIA ha desarrollado un sistema autónomo de detección y recolección de frutas del suelo basado en tecnologías fotónicas, visión 3D, inteligencia artificial y robótica móvil. La combinación de estas tecnologías ha hecho posible el desarrollo de un robot móvil que se desplaza de manera autónoma por el campo detectando, recolectando y almacenando frutas para su posterior procesamiento. En esta segunda fase del proyecto que se inició en 2021, se han integrado los tres módulos que componen la plataforma móvil robótica: un robot móvil autónomo equipado con GPS y sensores de visión; un sistema de visión con cámaras 2D y un brazo colaborativo con una garra que facilita la recolección de la fruta. El módulo de navegación autónoma permite al robot desplazarse por el campo en busca de los frutos caídos al suelo. Para ello, se ha desarrollado un sistema de algoritmos que traza la ruta óptima para detectar obstáculos y evitarlos. Además, de crear trayectorias en la parcela para una recolección eficiente. El módulo de visión inteligente, con cámaras 2D y 3D, se encarga de detectar los frutos en el suelo y localizar las coordenadas utilizando inteligencia artificial(LÃ^3*pez*, 2023). En este caso se estima la integración de sensores para una mayor precisión.

Figura 1.3. Robot de AINIA

Robotnik RB-ROBOUT+: Este manipulador móvil combina un brazo robótico con capacidades de movilidad y es utilizado en aplicaciones industriales y logísticas. Aunque su enfoque principal es la manipulación de objetos, también se destaca por su interfaz intuitiva que permite a los operadores interactuar fácilmente con el robot a través de paneles táctiles o aplicaciones específicas, facilitando la programación y el control en tiempo real. El RB-ROBOUT+ puede ser controlado a través de un panel táctil que proporciona una experiencia de usuario amigable. Este panel permite a los operadores acceder a diferentes funciones del robot con simples toques, lo que reduce la curva de aprendizaje y mejora la eficiencia operativa(LÃ^3*pez*, 2023).

Figura 1.4. RB-ROBOUT+

Metodología XP (eXtreme Programming)

La metodología eXtreme Programming (XP, por sus siglas en inglés) es adecuada para trabajar en proyectos de automatización y robótica, incluyendo aquellos que utilizan plataformas como Arduino(Kettunen, 2022). A continuación, se detallan las razones por las cuales XP es la metodología seleccionada para este proyecto.

.

Adaptabilidad a Cambios

XP se basa en la premisa de que los requisitos pueden cambiar durante el desarrollo. Esto es especialmente relevante en proyectos de robótica, donde la naturaleza del hardware y los objetivos del proyecto pueden evolucionar a medida que se avanza en el desarrollo. La flexibilidad de XP permite ajustar rápidamente el software para adaptarse a nuevas necesidades o descubrimientos.

Desarrollo Iterativo e Incremental

La metodología XP promueve ciclos cortos de desarrollo, lo que permite realizar entregas frecuentes. Esto es beneficioso en proyectos de robótica, ya que los desarrolladores pueden probar rápidamente nuevas funcionalidades y recibir retroalimentación, lo que ayuda a identificar y corregir problemas antes de que se conviertan en obstáculos mayores.

Enfoque en la Calidad del Código

XP enfatiza prácticas como Test-Driven Development (TDD) y la programación en pareja, lo que resulta en un código más limpio y menos propenso a errores. En el contexto de Arduino y robótica, donde la precisión es crucial, estas prácticas ayudan a asegurar que el software funcione correctamente con el hardware.

A continuación se muestra una tabla comparativa de XP con otra metodología:

Tabla 1.1. Comparación entre XP y Scrum

Aspecto	**XP**	**Scrum**
Enfoque	Enfoque en la calidad y la entrega rápida de software	Enfoque en la entrega de valor a los clientes
Roles	Programador, Tester, Coach, Customer	Product Owner, Scrum Master, Development Team
Iteraciones	Iteraciones cortas (1-4 semanas)	Iteraciones cortas (1-4 semanas)
Planificación	Planificación informal, no hay un plan detallado	Planificación formal, plan de product backlog
Desarrollo	Desarrollo en equipo, refactorización continua	Desarrollo en equipo, refactorización continua
Pruebas	Pruebas automatizadas y manuales, testing continuo	Pruebas automatizadas y manuales, testing continuo
Revisión	Revisión continua, retroalimentación constante	Revisión continua, retroalimentación constante
Adaptabilidad	Flexibilidad para cambiar el enfoque según sea necesario	Flexibilidad para cambiar el enfoque según sea necesario

Aspecto	XP	Scrum
Diseño	Diseño flexible, no hay un diseño previo	Diseño flexible, no hay un diseño previo
Documentación	Documentación mínima, se enfoca en la entrega de software	Documentación mínima, se enfoca en la entrega de software

Casos Reales

Un estudio mencionado indica que se ha utilizado una metodología basada en XP para desarrollar software orientado a robots educativos utilizando plataformas como ROS (Robot Operating System) junto con Arduino. Este enfoque ha demostrado ser efectivo al combinar las prácticas ágiles con un diseño evolutivo e incremental, adaptándose bien a las necesidades cambiantes de los proyectos educativos y de investigación(G. Parnell, 2021).

La metodología XP es altamente adecuada para proyectos de automatización y robótica que utilizan Arduino debido a su flexibilidad, enfoque en la calidad del código y capacidad para adaptarse a cambios rápidos en los requisitos. Su énfasis en la colaboración continua y la retroalimentación también asegura que los productos finales sean más alineados con las expectativas del usuario, lo cual es fundamental en un campo tan dinámico como la robótica.

1.3. Estado actual de las soluciones robóticas en la siembra automatizada

El presente epígrafe se dedica a la descripción y análisis del estado actual del manejo remoto de brazos robóticos, con un enfoque particular en su aplicación para la siembra de posturas. Este análisis incluye la identificación de las variables que se estudian en este objeto de investigación y el diagnóstico realizado antes de iniciar la investigación, lo que demuestra la pertinencia y veracidad de la situación problemática planteada.

1.3.1. Descripción del Estado Actual

El uso de brazos robóticos en la agricultura ha crecido considerablemente en los últimos años, impulsado por la necesidad de aumentar la eficiencia y reducir costos en un sector que enfrenta desafíos significativos, como el cambio climático y la escasez de mano de obra. Según un informe de Faster Capital (2023), se estima que el uso de tecnologías robóticas podría aumentar la producción agrícola en un 30 por ciento para 2030. Esto se debe a que los brazos robóticos pueden realizar tareas repetitivas con mayor precisión y rapidez que los humanos.

Los brazos robóticos modernos están equipados con sistemas avanzados de control y sensores que les permiten operar en entornos complejos. Por ejemplo, el brazo robótico ÜR5"de Universal Robots ha sido ampliamente utilizado por su versatilidad y facilidad de programación, lo que lo convierte en una opción popular para diversas aplicaciones agrícolas, desde la siembra hasta el riego(Medeiros, 2023).

1.3.2. Variables Estudiadas

- Eficiencia operativa: Medida a través del tiempo requerido para completar tareas específicas como la siembra.
- Precisión: Evaluación de cuán exactas son las acciones del brazo robótico en comparación con las realizadas manualmente.
- Costo: Análisis del costo operativo versus el costo de mano de obra humana.
- Sostenibilidad: Impacto ambiental asociado al uso de tecnologías robóticas en comparación con métodos tradicionales.

  Cada una de estas variables es crucial para determinar la viabilidad y efectividad del uso de brazos robóticos en la agricultura. Por ejemplo, estudios han demostrado que los sistemas automatizados pueden reducir el uso de agua y fertilizantes al aplicar estos recursos solo donde son necesarios.

1.4. Tecnologías informáticas utilizadas en el desarrollo de brazos robóticos para la siembra

El uso de la tecnología Arduino en el manejo remoto de brazos robóticos para la siembra de posturas representa una innovación significativa en el ámbito agrícola. Esta sección se dedica a sistematizar los fundamentos teórico-metodológicos que respaldan la elección de Arduino como herramienta clave en esta investigación, así como a describir las tecnologías y herramientas utilizadas para lograr los resultados deseados.

1.4.1. Arduino

Arduino es una plataforma de hardware y software de código abierto que permite a los usuarios crear prototipos electrónicos de manera sencilla y económica. Consiste en una placa con un microcontrolador que puede ser programada para interactuar con diversos sensores y actuadores, lo que la convierte en una herramienta ideal para aplicaciones en robótica agrícola(Hamdi., 2023).

Las características distintivas de Arduino incluyen su bajo costo, facilidad de uso y flexibilidad. Esto permite que tanto aficionados como profesionales desarrollen proyectos complejos sin necesidad de un profundo conocimiento previo en electrónica o programación. Además, su naturaleza abierta fomenta la colaboración y el intercambio de ideas entre desarrolladores, lo que lleva a soluciones innovadoras en el campo agrícola(ibíd.).

Ventajas del Uso de Arduino en Agricultura

- Versatilidad: Una de las principales ventajas del uso de Arduino en tecnología agrícola es su versatilidad. Los microcontroladores pueden ser programados para realizar diversas tareas, como monitorear condiciones ambientales (humedad, temperatura) y controlar sistemas automatizados (riego, fertilización). Esto permite a los agricultores optimizar sus operaciones basándose en datos en tiempo real, mejorando así la eficiencia y la calidad del cultivo.

- Automatización: La automatización es otro aspecto crítico donde Arduino brilla. Con la integración de sensores y actuadores, es posible automatizar procesos como el riego y la aplicación de fertilizantes, lo que no solo ahorra tiempo y costos laborales, sino que también asegura que los insumos se apliquen de manera precisa. Por ejemplo, un sistema automatizado puede regar solo cuando los niveles de humedad del suelo caen por debajo de un umbral específico, evitando así el desperdicio de agua.
- Monitoreo Remoto: Arduino también permite el monitoreo remoto de las operaciones agrícolas. Los agricultores pueden acceder a datos sobre sus cultivos desde cualquier lugar mediante aplicaciones móviles o plataformas web. Esto reduce la necesidad de desplazamientos frecuentes al campo, ahorrando tiempo y recursos(Mathews, 2023).

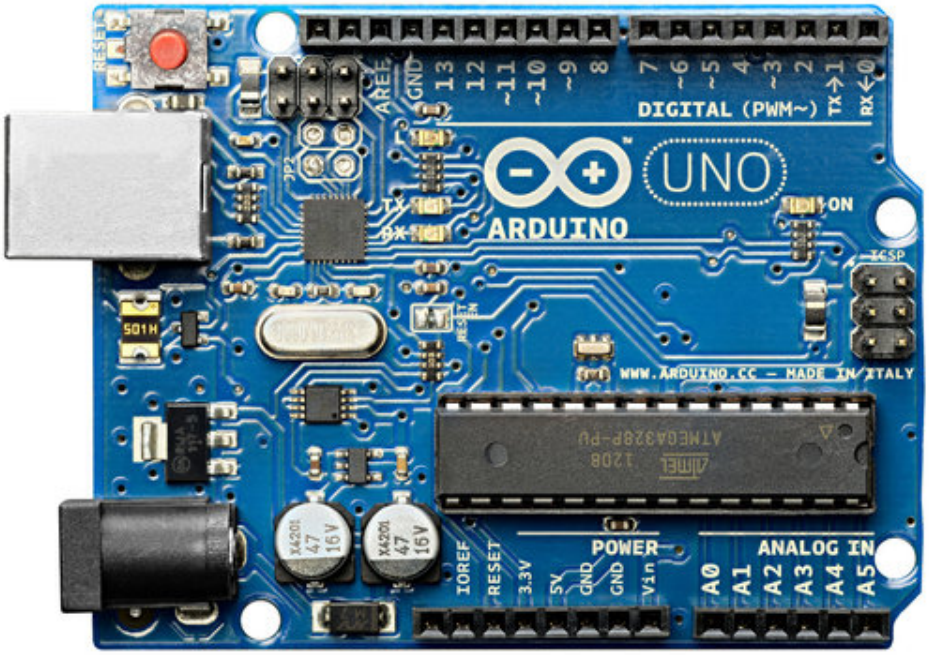


Figura 1.5. Placa de Arduino UNO

1.4.2. Tinkerkit Braccio

El TinkerKit Braccio es una herramienta poderosa y flexible para proyectos robóticos que requieren control preciso y adaptabilidad. Su integración con Arduino y la posibilidad de manejarlo remotamente lo convierten en una opción ideal para desarrollar soluciones innovadoras en diversas áreas.

Características:

- Estructura Modular: El Braccio puede ser ensamblado de diversas maneras, lo que permite adaptarlo a diferentes tareas. Esto incluye la posibilidad de agregar accesorios como cámaras o herramientas específicas para manipular objetos.

- Control a través de Arduino: Utiliza una placa Arduino para el control de sus servomotores, lo que facilita la programación y personalización del brazo. Los usuarios pueden aprovechar la amplia comunidad y recursos disponibles para Arduino.
- Braccio Shield: Incluye un escudo (shield) que permite conectar los servos directamente a la placa Arduino. Este escudo tiene características como un interruptor de encendido y fusibles de protección para los motores, asegurando un funcionamiento seguro.
- Movimientos Controlados: El brazo tiene múltiples grados de libertad, permitiendo movimientos precisos en varias direcciones. Los servos pueden ser controlados mediante comandos simples en el software Arduino, facilitando la programación de movimientos complejos(Elvira, 2021).

Componentes

- Servomotores: El Braccio utiliza servomotores que permiten un control preciso sobre los movimientos del brazo. Cada motor está conectado a un pin específico en el escudo, lo que permite su control individual.
- Conectores TinkerKit: El escudo incluye conectores estándar que simplifican la conexión de componentes adicionales, como sensores o módulos de comunicación.
- Biblioteca Braccio: Para facilitar la programación, se proporciona una biblioteca específica que incluye funciones para inicializar el brazo y controlar los movimientos de los servos. Esto permite a los desarrolladores implementar rápidamente funcionalidades sin necesidad de escribir todo el código desde cero(ibíd.).

Figura 1.6. Brazo Robótico Tinkerkit

1.4.3. Python

Python es un lenguaje de programación de alto nivel, versátil y ampliamente utilizado, conocido por su legibilidad y sencillez. Fue creado por Guido van Rossum y lanzado en 1991. Su diseño enfatiza la claridad del código, lo que permite a los desarrolladores escribir programas con menos líneas en comparación con otros lenguajes(Bradley, 2021).

Características Principales de Python

- Multi-paradigma: Python soporta varios paradigmas de programación, incluyendo la programación estructurada, orientada a objetos y funcional. Esto permite a los desarrolladores elegir el estilo que mejor se adapte a sus necesidades.
- Tipado Dinámico: Python es un lenguaje dinámicamente tipado, lo que significa que no es necesario declarar el tipo de una variable al momento de su creación; el tipo se determina en tiempo de ejecución.
- Interprete: Python se ejecuta en un sistema interpretado, lo que permite ejecutar el código inmediatamente después de escribirlo, facilitando la prototipación rápida.
- Amplia Biblioteca Estándar: Python incluye una extensa biblioteca estándar que proporciona herramientas para tareas comunes como manipulación de archivos, acceso a bases de datos y desarrollo web, entre otros.
- Compatibilidad Multiplataforma: Python es compatible con diversos sistemas operativos como Windows, macOS y Linux, lo que lo hace accesible para una amplia gama de desarrolladores.

Usos Comunes

- Desarrollo Web:Se puede usar para crear aplicaciones web del lado del servidor.
- Ciencia de Datos y Análisis: Es popular en la comunidad de análisis de datos y aprendizaje automático.
- Automatización y Scripting: Se utiliza para automatizar tareas repetitivas y escribir scripts.
- Desarrollo de Software: Facilita la creación de software tanto para prototipos como para aplicaciones listas para producción.

Interfaces gráficas

- Tkinter: Tkinter es la biblioteca estándar de Python para crear interfaces gráficas. Viene preinstalada con la mayoría de las distribuciones de Python, lo que facilita su uso. Proporciona una amplia gama de widgets como botones, etiquetas, cuadros de texto y menús. Permite personalizar la apariencia y el comportamiento de los elementos gráficos.Utiliza un bucle principal (‘mainloop‘) para manejar eventos y actualizaciones en la interfaz.
- PyQt: PyQt es un conjunto de enlaces de Python para las bibliotecas Qt, que son muy potentes para el desarrollo de aplicaciones gráficas. Ofrece un diseño visual a través de Qt Designer, permitiendo crear interfaces complejas fácilmente. Soporta características avanzadas como gráficos en 2D y 3D, así como animaciones. Ideal para aplicaciones comerciales y de escritorio que requieren una interfaz rica y profesional.

- wxPython: wxPython es una biblioteca que permite crear aplicaciones nativas en múltiples plataformas utilizando Python. Proporciona acceso a las API nativas del sistema operativo, lo que resulta en aplicaciones que se ven y se comportan como aplicaciones nativas. Incluye una variedad de widgets y herramientas para construir interfaces complejas.

1.5. Conclusiones Parciales

Al finalizar este capítulo se llegaron a las siguientes conclusiones:

- La investigación realizada ha demostrado que el uso de brazos robóticos para la siembra de posturas representa una solución innovadora y efectiva para enfrentar los desafíos actuales en el sector agrícola. La automatización de tareas agrícolas no solo mejora la eficiencia, sino que también contribuye a la sostenibilidad al optimizar el uso de recursos.
- Los fundamentos teóricos y metodológicos asociados al manejo remoto de brazos robóticos han proporcionado un marco robusto para entender cómo estas tecnologías pueden ser implementadas efectivamente en la agricultura. Estos principios son esenciales para el diseño y desarrollo de sistemas que respondan a las necesidades específicas del entorno agrícola.
- La implementación de brazos robóticos puede aumentar significativamente la productividad agrícola. Al reducir el tiempo y los costos asociados con la siembra, los agricultores pueden obtener mejores rendimientos, lo que es vital en un contexto donde la demanda alimentaria sigue creciendo.
- Arduino, XP y Python son las herramientas más adecuada para el desarrollo de brazos robóticos por su versatilidad, bajo costo y facilidad de uso.
  Estas conclusiones establecen una base sólida para continuar explorando el potencial del manejo remoto de brazos robóticos en la agricultura, reafirmando su papel como un componente esencial en la búsqueda de soluciones innovadoras para los desafíos actuales del sector

CAPÍTULO 2

Características y Diseño del Sistema

En el ámbito del desarrollo de software, la creación de sistemas robustos y eficientes requiere un enfoque metódico que integre las necesidades del usuario, la planificación adecuada y prácticas de diseño efectivas. Este capítulo se centra en el diseño y desarrollo de un sistema de control para un brazo robótico, utilizando metodologías ágiles, específicamente eXtreme Programming (XP), que promueven la colaboración continua y la adaptabilidad a los cambios.

El capítulo comienza con una breve descripción de la solución propuesta para el sistema. Luego muestra la identificación de las historias de usuario, que son fundamentales para comprender las expectativas y necesidades del cliente. A continuación se presenta la estimación de esfuerzos y un plan de iteraciones que organiza el trabajo en ciclos cortos y manejables. Además, se exploran los patrones de diseño aplicados en el desarrollo del sistema, tanto desde la perspectiva de GRASP (General Responsibility Assignment Software Patterns) como de GoF (Gang of Four). Finalmente, se utilizan tarjetas CRC (Clase-Responsabilidad-Colaboración) para visualizar la estructura del sistema y definir claramente las responsabilidades y colaboraciones entre las clases involucradas.

Este capítulo proporciona un marco integral para el diseño y desarrollo del sistema del brazo robótico, destacando la importancia de una planificación cuidadosa, una comunicación efectiva y prácticas de diseño sólidas en la creación de software exitoso.

2.1. Modelado del diseño y estimación de esfuerzo

2.1.1. Propuesta de solución

Para dar respuesta a la problemática se propone una solución integral que aborde tanto los aspectos técnicos como los operativos. A continuación, se presenta una propuesta de solución que incluye componentes de hardware, software y consideraciones operativas:

Se determina un brazo robótico con 5 grados de libertad equipados con servomotores para permitir movimientos precisos y flexibles. Además integra un sensor ultrasónico para detectar la presencia de posturas a

una distancia entre 5 y 30 cm y un modulo Bluetooth HC-05 para la conectividad y garantizar el manejo a distancia. Se emplea una placa Arduino como controlador central para gestionar los movimientos del brazo y la comunicación con el software.
Se desarrolla una aplicación que permita a los usuarios controlar el brazo robótico de manera remota mendiante la conectividad Bluetoth. Esta interfaz contiene una consola de salida con mensajes que actualizan al usuario del estado del brazo y las acciones que lleva a cabo en el momento además de mostrar el tiempo que le tomó realizar la acción y la cantidad de posturas sembradas. El usuario tiene las opciones de iniciar el proceso de siembra, pausarlo y retomarlo en cualquier momento que el decida y dar por finalizado el proceso. También tiene la opción de guardar las acciones y estados del brazo mediante ficheros que contendrán la fecha para una mejor identificación.

2.1.2. Requisitos funcionales, no funcionales e historias de usuario

Los Requisitos No Funcionales (RNF) de software forman una parte significativa de la especificación de requisitos y en algunos casos estos son críticos para el éxito del producto. Con frecuencia estos requisitos son ignorados o subestimados debido a que para muchos proyectos estos implican una cantidad considerable de trabajo y esfuerzo; resultan ser más complejos y requieren un mayor nivel de conocimiento (Salamea, 2021). Los Requisitos Funcionales (RF) son esenciales para garantizar que el sistema del brazo robótico opere de manera efectiva, segura y eficiente, cumpliendo con las expectativas tanto técnicas como operativas.
Requisitos funcionales

- RF 01. Debe integrarse un sensor ultrasónico que detecte la presencia de objetos a una distancia de entre 5 y 30 cm.
- RF 02. El sistema debe incluir un módulo Bluetooth HC-05 para permitir el control remoto del brazo robótico.
- RF 03. La aplicación debe permitir a los usuarios controlar el brazo robótico de manera remota.
- RF 04. Debe incluir una consola de salida que muestre mensajes sobre el estado del brazo y las acciones que está realizando.
- RF 05. La interfaz debe mostrar el tiempo que tomó realizar cada acción y la cantidad de posturas sembradas.
- RF 06. Los usuarios deben tener la opción de iniciar, pausar, y retomar el proceso de siembra en cualquier momento.
- RF 07. La aplicación debe permitir a los usuarios guardar las acciones y estados del brazo en ficheros que contengan la fecha para una mejor identificación.
- RF 08. La aplicación debe actualizar al usuario en tiempo real sobre el estado del brazo robótico y las acciones realizadas.

Requisitos no funcionales de usabilidad

- RNF 01. El sistema debe tener una interfaz intuitiva que permita a los usuarios nuevos aprender a utilizar el software sin necesidad de formación extensa.
- RNF 02. Los usuarios deben poder completar tareas comunes en un tiempo razonable.
- RNF 03. El sistema debe proporcionar retroalimentación clara después de cada acción realizada.
- RNF 04. Proporcionar indicadores visuales claros sobre el estado del brazo robótico para mantener informado al usuario sobre las operaciones en curso.
- RNF 05. Al realizar acciones críticas el software debe mostrar mensajes de confirmación que aseguren al usuario que su comando ha sido recibido y ejecutado correctamente.

Requisitos no funcionales de apariencia o interfaz externa

- RNF 06. Todos los elementos de la interfaz deben seguir un estilo visual coherente para facilitar la comprensión del sistema.
- RNF 07. Los botones y controles deben ser fácilmente identificables, utilizando etiquetas descriptivas que indiquen claramente su función.
- RNF 08. La interfaz debe adaptarse a diferentes tamaños de pantalla y dispositivos.
- RNF 09. La interfaz debe tener una gama de colores relacionados con la agricultura y la siembra.
- RNF 10. Asegurar que exista un contraste suficiente entre el texto y el fondo para facilitar la lectura.

Requisitos no funcionales de software

- RNF 11. El sistema debe responder a las acciones del usuario en menos de 7 segundos
- RNF 12. El sistema debe ser funcional en ordenadores portátiles o de escritorio.

Historias de Usuario

Las historias de usuario representan la técnica utilizada por XP para especificar los requisitos de software. Las mismas deben ser programadas en un período de tiempo comprendido entre una y tres semanas aproximadamente. Si, por el contrario, la estimación es superior a las tres semanas, se debe dividir la historia en dos o más y si es menos de una semana, se debe combinar con otra respectivamente. A la hora de estimar el esfuerzo asociado a la implementación de las historias, los programadores deben utilizar como medida de referencia el punto. Un punto es equivalente a una semana ideal de programación (5 días hábiles). Las historias valen, generalmente; de 1 a 3 puntos en esta escala (Beck, 2022). A continuación, se describen las historias de usuario definidas para llevar a cabo el desarrollo de los módulos propuestos en la solución.

Tabla 2.1. Historia de usuario # 1

Historia de usuario	
Número: 1	**Nombre:** Controlar el brazo robótico de forma remota
Usuario: Usuario del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** bajo

Continúa en la próxima página

Tabla 2.1. Continuación de la página anterior

Puntos estimados: 1	**Iteración asignada:** 1
Programador responsable: Keily Marrero Martínez	
Descripción: El brazo robótico debe ser controlado a distancia	
Observaciones: -	

Tabla 2.2. Historia de usuario # 2

Historia de usuario	
Número: 2	**Nombre:** Detección precisa de posturas
Usuario: Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** medio
Puntos estimados: 2	**Iteración asignada:** 1
Programador responsable: Keily Marrero Martínez	
Descripción: El brazo robótico debe ser capaz de detectar posturas a través de sensor ultrasónico	
Observaciones: El sensor tiene un tiempo de espera de 2 segundos antes de hacer una detección para una mejor precisión.	

Tabla 2.3. Historia de usuario # 3

Historia de usuario	
Número: 3	**Nombre:** Mostrar consola de estado en la aplicación
Usuario: Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** bajo
Puntos estimados: 1.5	**Iteración asignada:** 2
Programador responsable: Keily Marrero Martínez	
Descripción: La interfaz debe mostrar una consola con los estados del brazo, tiempo de trabajo y cantidad de posturas sembradas.	
Observaciones: -	
Interfaz: 	

Tabla 2.4. Historia de usuario # 4

Historia de usuario	
Número: 4	**Nombre:** Guardar ficheros
Usuario: Usuario del sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** bajo
Puntos estimados: 1	**Iteración asignada:** 2
Programador responsable: Keily Marrero Martínez	
Descripción: Al finalizar el proceso el usuario guarda los estados mostrados en la consola en un fichero	
Observaciones: El fichero tiene como título la fecha en que se realizó el proceso.	
Interfaz: 	

El resto de las historias de usuario se muestran en el apéndice *A* de los anexos.

Estimación de esfuerzo por Historias de usuario

La estimación de esfuerzo en el desarrollo de software es un proceso crucial que implica predecir el tiempo, recursos y costos necesarios para completar un proyecto. Este proceso se basa en la identificación y análisis de los requisitos del software, así como en la experiencia previa y el conocimiento del equipo de desarrollo. El principal objetivo de la estimación es proporcionar una base realista para la planificación del proyecto, permitiendo asignar recursos adecuadamente y establecer plazos realistas(Sommerville, 2022).

Tabla 2.5. Tabla de Iteraciones y Duración de Historias de Usuario

Iteración	**Historias de usuario**	**Duración (semanas)**
1	Controlar el brazo robótico de forma remota	1.0
	Detección precisa de posturas	0.5
	Conectividad Bluetooth fiable	1.0
2	Mostrar consola de estado en la aplicación	0.5
	Guardar ficheros	1.0
3	Mostrar tiempo de ejecución	0.5
	Pausar y retomar el proceso	1.0
	Visualización de posturas sembradas	0.5
Total		6.0

2.2. Plan de iteraciones y patrones de diseño y arquitectónicos

2.2.1. Desarrollo del plan de iteraciones

Un plan de iteración es un componente esencial en metodologías ágiles como eXtreme Programming (XP) y Scrum, que organiza el trabajo en ciclos cortos y manejables. Este enfoque permite a los equipos de desarrollo entregar incrementos funcionales del producto de manera regular, facilitando la adaptación a cambios y la incorporación de retroalimentación continua.

Tabla 2.6. Plan de iteraciones

Iteración	**Historias de usuario**	**Duración (semanas)**
1	Controlar el brazo robótico de forma remota Detección precisa de posturas Conectividad Bluetooth fiable	2.5
2	Consola de estado en la aplicación Guardar ficheros	1.5
3	Mostrar tiempo de ejecución Pausar y retomar el proceso Visualización de posturas sembradas	2.0
Total		6.0

Tabla 2.7. Plan de Entregas

Iteraciones	**Fecha de inicio**	**Fecha de fin**
Iteración 1	2 de septiembre de 2024	19 de septiembre de 2024
Iteración 2	19 de septiembre de 2024	30 de septiembre de 2024
Iteración 3	30 de septiembre de 2024	13 de octubre de 2024

2.2.2. Patrones de arquitectura de software

La arquitectura de software se refiere a la estructura y diseño de un sistema de software, definiendo cómo los componentes del software interactúan entre sí y cómo cumplen con los requisitos funcionales y no funcionales. Proporcionan un marco claro para organizar el código y los componentes del sistema. Facilitan la reutilización de soluciones probadas para problemas comunes. Mejoran la mantenibilidad del software al separar preocupaciones y responsabilidades. Ayudan a diseñar sistemas que pueden escalar fácilmente para manejar un mayor volumen de usuarios o datos. La elección del patrón arquitectónico adecuado es crucial

para el éxito del desarrollo del software, ya que influye en su rendimiento, escalabilidad y capacidad de adaptación a cambios futuros(Sommerville, 2022).

Arquitectura en Capas

La arquitectura en capas organiza el sistema en niveles (capas) que representan diferentes responsabilidades.Cada capa puede desarrollarse y mantenerse de forma independiente,lo que facilita la implementación y el mantenimiento.Permite a los desarrolladores enfocarse en una capa específica sin preocuparse por las demás.Cada capa se puede probar individualmente,lo que mejora la calidad del software (Len Bass, 2021). La arquitectura para este proyecto se dividirá en cuatro capas principales:Capa de Presentación,Capa de Control, Capa de Acceso a Datos y Capa de Comunicación.

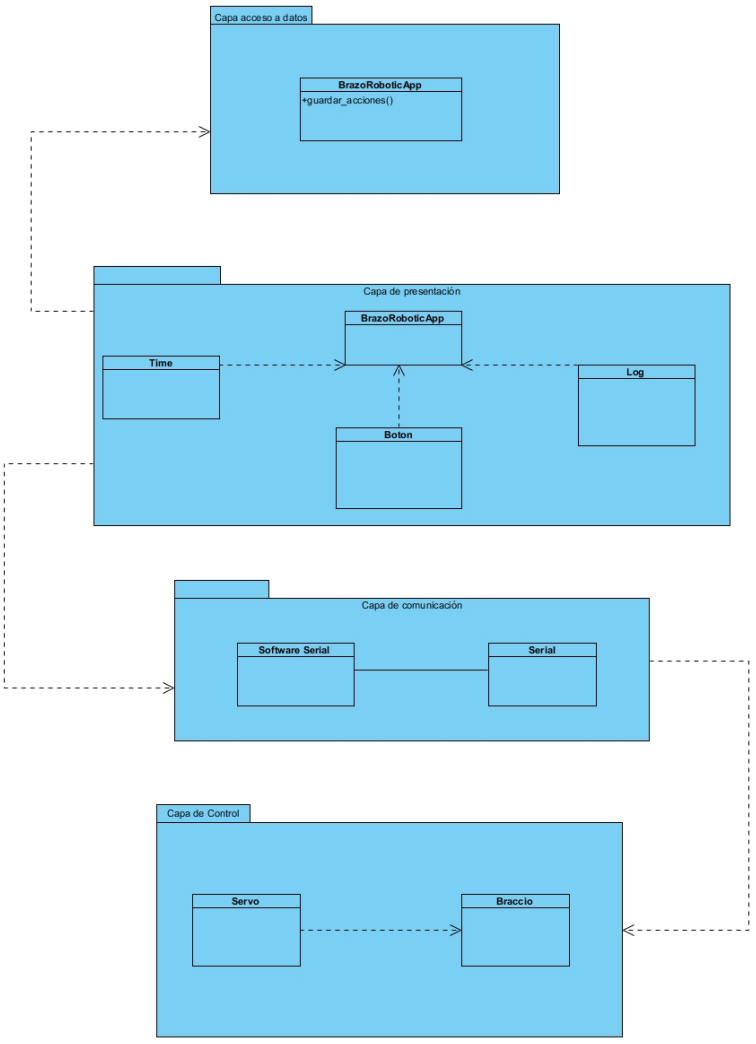


Figura 2.1. Diseño Arquitectónico

Capas de la arquitectura

- Capa de Presentación: Esta capa es responsable de la interfaz de usuario (UI) y la interacción con

el usuario. Se implementará en Python utilizando bibliotecas como Tkinter. Permite al usuario enviar comandos al brazo robótico. Muestra información actualizada sobre el estado del brazo. Recibe retroalimentación del usuario sobre las acciones realizadas.

- Capa de Control: Esta capa contiene las clases principales encargadas de manejar y controlar los movimientos del brazo.
- Capa de Acceso a Datos: Esta capa se encarga de gestionar el almacenamiento y recuperación de datos. En este caso, se utilizarán ficheros para guardar los datos mostrados en la consola. Guarda registros sobre las acciones realizadas por el brazo robótico, incluyendo fechas y tiempos.
- Capa de comunicación: esta capa es un enlace entre la capa de presentación y la capa de control, ya que contiene las clases necesarias para establecer la comunicación con el módulo Bluetooth y así conectar el brazo con la aplicación.

2.3. Patrones de Diseño

Los patrones de diseño son soluciones reutilizables a problemas comunes que surgen en el desarrollo de software. Estos patrones proporcionan un enfoque estructurado y eficiente para resolver problemas recurrentes, facilitando la comunicación entre desarrolladores y mejorando la calidad del software. Para diseñar la solución se usarán algunos patrones de diseño GRASP Y GoF (Bradley, 2021).

2.3.1. Patrones GRASP

Creator (Creador) Descripción: Este patrón sugiere que una clase debe ser responsable de crear instancias de otras clases si tiene una relación directa con ellas. Aplicación en el Código: La clase Braccio es responsable de manejar el movimiento del brazo robótico. Esta clase tiene la responsabilidad de crear y gestionar los servomotores y otros componentes del brazo.

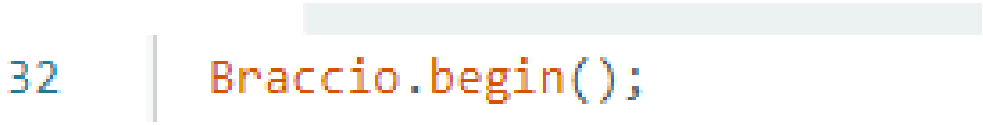


Figura 2.2. Patrón Creator

Controller (Controlador) Descripción: Este patrón se refiere a la asignación de la responsabilidad de manejar eventos y coordinar la interacción entre el modelo y la vista.
La clase BrazoRoboticoApp actúa como un controlador al gestionar los eventos generados por los botones (iniciar, pausar, reanudar, finalizar, guardar). Cada botón está vinculado a un método correspondiente que maneja la lógica de negocio para cada acción.

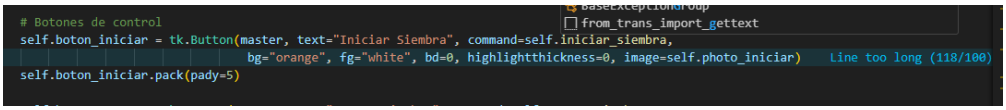


Figura 2.3. Patrón Controller

Information Expert (Experto en Información) Descripción: Este patrón sugiere que la responsabilidad debe ser asignada a la clase que tiene la información necesaria para cumplirla. Aplicación en el Código: La lógica para calcular la distancia utilizando el sensor ultrasónico se encuentra dentro del loop(), donde se utiliza información sobre el tiempo de duración del pulso para determinar la distancia. La clase que maneja el sensor es responsable de estos cálculos.

```
duracion = pulseIn(ecco, HIGH);
distancia = duracion / 58.2;
```

Figura 2.4. Patrón Information Expert

2.3.2. Patrones GoF

Command Descripción: Este patrón encapsula una solicitud como un objeto, permitiendo parametrizar clientes con diferentes solicitudes y soportar operaciones que pueden deshacerse. Aplicación en el Código: El uso de comandos enviados a través del módulo Bluetooth (miBT) para iniciar (’1’) y finalizar (’2’) las operaciones del brazo robótico puede considerarse una implementación del patrón Command, ya que cada comando desencadena una acción específica en el sistema.

```
if (dato == '1') {
  isRun = true;
  Serial.println("iniciado");
} else if (dato == '2') {
  isRun = false;
  Serial.println("finalizado");
  Braccio.ServoMovement(20,     0, 90,  90,   90,  90,   10);
}
```

Figura 2.5. Patrón Command

Observer Descripción: Este patrón define una relación uno-a-muchos entre objetos, de modo que cuando un objeto cambia su estado, todos sus dependientes son notificados y actualizados automáticamente. Aunque no hay un patrón Observer explícito implementado, la forma en que se actualiza la consola con mensajes podría adaptarse a este patrón. La consola actúa como un .ºbservador"del estado del brazo robótico, ya que se actualiza cada vez que se realiza una acción (iniciar, pausar, etc.).

```
tiempo_actual = datetime.now().strftime("%Y-%m-%d %H:%M:%S")
self.consola.insert(tk.END, f"{tiempo_actual} - {mensaje}\n")
```

Figura 2.6. Patrón Observer

2.4. Tarjetas de Contenido, Responsabilidad y Colaboración

Las tarjetas CRC son una herramienta efectiva para la planificación y el diseño de software orientado a objetos, y se pueden aplicar en este contexto para identificar y organizar las clases que componen el sistema, así como sus responsabilidades y colaboraciones(Kettunen, 2022).

Tabla 2.8. Tarjeta CRC # 1

Tarjeta CRC	
Clase: Braccio	
Responsabilidad	**Colaboración**
• Inicializar y controlar los servomotores. • Realizar movimientos del brazo robótico. • Gestionar la lógica de siembra.	ControladorBluetooth ServoControlador SensorUltrasonico

Tabla 2.9. Tarjeta CRC # 2

Tarjeta CRC	
Clase: ServoControlador	
Responsabilidad	**Colaboración**
• Manejar la posición de los servomotores individuales (base, hombro, codo, muñeca). • Proveer métodos para mover los servos a posiciones específicas	Braccio

Tabla 2.10. Tarjeta CRC # 3

Tarjeta CRC	
Clase: BrazoRoboticoApp	
Responsabilidad	**Colaboración**

Continúa en la próxima página

Tabla 2.10. Continuación de la página anterior

• Inicializar la interfaz gráfica. • Gestionar el estado del brazo robótico. • Proporcionar métodos para iniciar, pausar, reanudar y finalizar la siembra. • Guardar acciones en un archivo	datetime para gestionar tiempos y fechas tk (Tkinter) para la interfaz gráfica.

Tabla 2.11. Tarjeta CRC # 4

Tarjeta CRC	
Clase: SensorUltrasonico	
Responsabilidad	**Colaboración**
• Medir la distancia utilizando un sensor ultrasónico. • Proveer datos sobre la presencia de objetos o posturas.	Braccio

El resto de las tarjetas crc se muestran en el apéndice *B* de los anexos.

2.5. Conclusiones parciales

Con la realización de este capítulo se llegaron a las siguientes conclusiones:
El uso de la metodología ágil XP fue clave para el diseño del sistema. Se identificaron 8 historias de usuarios que facilitaron la priorización de tareas y ayudaron a entender mejor los requisitos funcionales. La organización del trabajo en tres iteraciones mejoró la gestión del tiempo y la calidad del producto final y la estimación de esfuerzo fue crucial para la planificación del proyecto, permitiendo asignar recursos y establecer un plazo realista.La aplicación de patrones de diseño, como GRASP y GoF, proporcionaron soluciones estructuradas a problemas comunes en el desarrollo. Estos patrones no solo mejoraron la calidad del software tanto del brazo como de la aplicación, sino que también facilitaron su escalabilidad. Se identificaron además requsitos no funcionales que fueron claves para la usabilidas, experiencia del usuario, la retroalimentación clara y el rendimiento del sistema. La arquitectura en capas propuesta permite una clara separación de responsabilidades entre los diferentes componentes del sistema, lo que facilitará su desarrollo, pruebas y mantenimiento.
Un enfoque metódico, combinado con prácticas ágiles y un sólido uso de patrones de diseño, puede llevar al desarrollo exitoso de un sistema complejo como el control de un brazo robótico. La atención a los detalles en la planificación, diseño e interacción con el usuario son elementos clave para lograr un producto final eficiente y satisfactorio.

CAPÍTULO 3

Validación del sistema

En el desarrollo de software, la fase de implementación y pruebas es fundamental para garantizar que el sistema cumpla con los requisitos establecidos y funcione de manera efectiva en un entorno real. Este capítulo se centra en la implementación del sistema de manejo remoto del brazo robótico, donde se llevarán a cabo las tareas ingenieriles necesarias para traducir el diseño conceptual en un producto funcional.
La implementación implica la integración de diversos componentes, incluyendo el hardware del brazo robótico y el software de control desarrollado. Durante esta fase, se abordarán aspectos técnicos como la configuración del entorno de desarrollo, la programación de las funcionalidades definidas en las historias de usuario y la conexión entre el sistema de control y el hardware. Se presentarán las tareas específicas realizadas por el equipo de desarrollo, destacando los desafíos enfrentados y las soluciones implementadas. Además, se desarrollarán pruebas de aceptación para cada historia de usuario, que son esenciales para validar que el sistema cumple con las expectativas del usuario final. Estas pruebas se diseñarán para evaluar cada funcionalidad implementada, asegurando que el sistema no solo funcione correctamente, sino que también satisfaga las necesidades y requisitos específicos del cliente. Cada historia de usuario se convertirá en un criterio de éxito, permitiendo al equipo verificar que todas las características deseadas están presentes y operativas.

3.1. Tareas de Ingeniería

Las tareas de ingeniería en software abarcan diversas actividades esenciales para el desarrollo de aplicaciones y sistemas informáticos. Estas incluyen el análisis de requerimientos para entender las necesidades del cliente, el diseño de sistemas para crear la arquitectura del software, y el desarrollo del código necesario para implementar funcionalidades. Además, se realizan pruebas y validación para detectar errores y asegurar un correcto funcionamiento, así como mantenimiento y actualizaciones continuas. La gestión de proyectos es crucial para planificar y supervisar el progreso, mientras que la documentación proporciona guías técnicas útiles (Bruce R. Maxim, 2021).

La fase de implementación y pruebas del sistema de manejo remoto del brazo robótico implica una serie de tareas ingenieriles críticas que aseguran la funcionalidad, eficiencia y calidad del producto final. A continuación, se describen las principales tareas que se llevarán a cabo durante esta fase.

Tabla 3.1. Tarea de ingeniería # 1

Tarea	
Número de tarea: 1	**Número de Historia de usuario:** 1
Nombre de la tarea: Implementar la comunicación remota con el brazo	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 2 de septiembre de 2024	**Fecha de fin:** 9 de septiembre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se implementa la lógica para enviar comandos desde la aplicación al brazo robótico.	

Tabla 3.2. Tarea de ingeniería # 2

Tarea	
Número de tarea: 2	**Número de Historia de usuario:** 2
Nombre de la tarea: Integrar sensores para detección de posturas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 9 de septiembre de 2024	**Fecha de fin:** 14 de septiembre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se integra un sensor ultrasónico que permite detectar las posturas.	

Tabla 3.3. Tarea de ingeniería # 3

Tarea	
Número de tarea: 3	**Número de Historia de usuario:** 3
Nombre de la tarea: Configurar el módulo Bluetooth	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 14 de septiembre de 2024	**Fecha de fin:** 19 de septiembre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se incorpora el módulo Bluetoth HC-05	

Tabla 3.4. Tarea de ingeniería # 4

Tarea	
Número de tarea: 4	**Número de Historia de usuario:** 4
Nombre de la tarea: Implementar consola para mostrar estado	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5

Continúa en la próxima página

Tabla 3.4. Continuación de la página anterior

Fecha de inicio: 19 de septiembre de 2024	**Fecha de fin:** 23 de septiembre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se desarrolla una consola que muestra el estado actual del brazo robótico en tiempo real.	

El resto de las tareas ingenieriles se muestran en el apéndice *C* de los anexos.

3.2. Pruebas de aceptación

Las pruebas de aceptación son un componente crítico en el desarrollo de software, diseñadas para validar que el sistema cumple con los requisitos y expectativas del usuario final(Klein, 2021). Se realizaron pruebas por cada historia de usuario. A continuación se muestran dichas pruebas por cada interación.

Tabla 3.5. Prueba de aceptación # 1

Caso de prueba de aceptación	
Código: HU1_P1	**Historia de usuario:** 1
Nombre: Iniciar proceso automático	
Descripción: Prueba para la funcionalidad: Controlar el brazo robótico de forma remota	
Condiciones de ejecución: El brazo robótico debe estar conectado a la aplicación	
Pasos de ejecución: • Se accede a la aplicación • Se selecciona el botón de iniciar • El brazo comienza a realizar el proceso de siembra automática	
Resultados esperados: Satisfactorio	

Tabla 3.6. Prueba de aceptación # 2

Caso de prueba de aceptación	
Código: HU1_P2	**Historia de usuario:** 1
Nombre: Verificar siembra de posturas	
Descripción: Prueba la funcionalidad de registrar una persona.	
Condiciones de ejecución: El brazo robótico debe estar conectado a la aplicación	

Continúa en la próxima página

Tabla 3.6. Continuación de la página anterior

Pasos de ejecución: • Se accede a la aplicación • Se selecciona el botón de iniciar • El brazo comienza a realizar el proceso de siembra automática
Resultados esperados: Insatisfactorio.

Tabla 3.7. Prueba de aceptación # 3

Caso de prueba de aceptación	
Código: HU2_P1	**Historia de usuario:** 2
Nombre: Detectar la presencia de posturas después de 2 segundos	
Descripción: Prueba para la funcionalidad: Detección precisa de posturas	
Condiciones de ejecución: El sensor debe estar correctamente instalado y funcionando.	
Pasos de ejecución: • Se accede a la aplicación • Se selecciona el botón de iniciar • Se coloca una bandeja con posturas • El brazo comienza a moverse dos segundos después de la detección	
Resultados esperados: Satisfactorio	

Tabla 3.8. Prueba de aceptación # 4

Caso de prueba de aceptación	
Código: HU2_P2	**Historia de usuario:** 2
Nombre: Detectar vacío	
Descripción: Prueba para la funcionalidad: Detección precisa de postura	
Condiciones de ejecución: El sensor debe estar correctamente instalado y funcionando.	
Pasos de ejecución: • Se accede a la aplicación • Se selecciona el botón de iniciar • Se coloca una bandeja con posturas • El brazo comienza a moverse dos segundos después de la detección • Se retira bandeja de posturas • El brazo termina la ejecución	
Resultados esperados: Satisfactorio	

Tabla 3.9. Prueba de aceptación # 5

Caso de prueba de aceptación	
Código: HU3_P1	**Historia de usuario:** 3
Nombre: Conectar el brazo a la aplicación	
Descripción: Prueba para la funcionalidad: Conectividad Bluetooth fiable	
Condiciones de ejecución: El módulo Bluetooth debe estar configurado y emparejado con el brazo	
Pasos de ejecución: • Se activa el módulo Bluetooth. • Se intenta conectar desde la aplicación. • Se verifica que la conexión se establece sin errores y se mantiene estable durante el uso.	
Resultados esperados: Satisfactorio	

El resto de las pruebas de aceptación se muestran en el apéndice *D* de los anexos.

3.3. Análisis de las pruebas de aceptación

Durante las pruebas de aceptación se realizaron un total de 12 casos de pruebas divididos en 3 iteraciones. En el siguiente gráfico se muestra la estadística de los resultados:

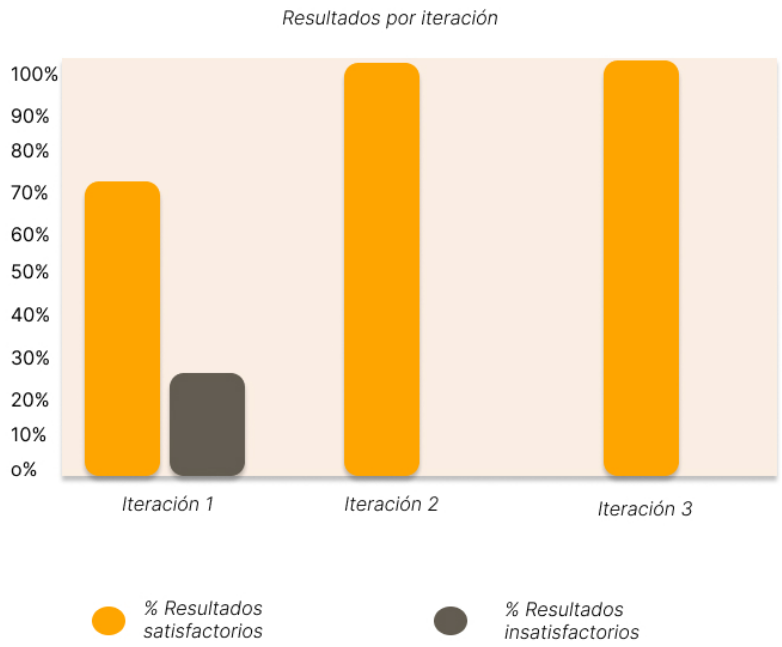


Figura 3.1. Resultado Pruebas

De la iteración 1 se obtuvo el 71 por ciento (5 pruebas) como satisfactorias y el 29 por ciento (2 pruebas) como insatisfactorias. Se detectó que en el proceso de siembra el brazo no cuenta con un agarre adecuado para asegurar la captura de las posturas y tampoco cuenta con una buena precisión para moverse en ángulos pequeños. Además, el sensor ultrasónico tiene dificultades para detectar objetos que no son densos, lo que puede traer consigo problemas si las posturas están muy separadas una de otra. La conectividad Bluetooth es buena, sin embargo se presentaron problemas con respecto al tiempo de respuesta con la interfaz y el brazo. En el resto de las iteraciones se obtuvieron resultados satisfactorios del 100 por ciento por lo que mayoritariamente se cumplieron los requisitos funcionales.

3.4. Conclusiones parciales

En este capítulo se realizó la implementación del sistema a través del desglose de las historias de usuario en tareas de ingeniería. Se realizaron pruebas de aceptación para cada historia de usuario, validando que el sistema cumple con los requisitos establecidos. A lo largo de las iteraciones, se observaron resultados mixtos en las pruebas. Aunque la mayoría cumplieron con los criterios establecidos, se identificaron áreas de mejora, como la precisión del brazo robótico y la efectividad del sensor ultrasónico. A pesar de los desafíos enfrentados durante la implementación, como problemas con la conectividad Bluetooth y la detección precisa por parte del sensor se presentaron soluciones implementadas para superar estos obstáculos.
Al finalizar esta fase, se cuenta con un sistema robusto y confiable. La atención a los detalles durante la implementación y pruebas garantizó que el producto final cumpla con los estándares requeridos.

Conclusiones

La realización de este trabajo finalmente llevó a la conclusión de que el manejo remoto de brazo robótico para la siembra automática como prototipo funcional es una base sólida para proyectos a gran escala. Desde la experiencia práctica se identificó que un brazo con cinco grados de libertad (como el utilizado en el prototipo) o más es lo más adecuado para este tipo de aplicación que necesita de una herramienta flexible. La combinación técnica de sensores de percepción con un módulo bluetooth constituye una solución interesante y efectiva. El uso de una aplicación para su control a distancia en un mundo cada vez más informatizado y amigable con la tecnología hace que la interacción del usuario con el robot sea más fácil. A pesar de los desafíos técnicos en su desarrollo se ha logrado implementar este prototipo funcional que demuestra el potencial de esta solución para su uso en aplicaciones prácticas.

Recomendaciones

Para futuros proyectos se recomienda tener en cuenta la investigación sobre el uso de plataformas móviles para su implementación en el prototipo y su aplicación en exteriores. También se recomienda la investigación e implementación sobre sensores de precisión para una captura más exacta de las posturas y algoritmos de control para detectar la pérdida en la precisión del robot por fallas del hardware.

Referencias bibliográficas

BECK, Kent, 2022. *Extreme programming explained: Embrace change*. Addison-Wesley Professional (vid. pág. 20).

BRADLEY, S., 2021. *Design Patterns in Python: Common GOF (Gang of Four) Design Patterns implemented in Python*. Independently published (vid. págs. 16, 25).

BRUCE R. MAXIM, Roger S. Pressman y, 2021. *Software engineering: A practitioner's approach (9th ed.)* McGraw-Hill Education (vid. pág. 29).

CLAVER, Hugo, 2021. Nissan robot duck keeps paddy fields clear of weeds. *Future Farming* (vid. pág. 8).

ELVIRA, Augusto HernÃ¡ndez, 2021. *Tinkerkit Braccio and the use of the ROS framework for learning and development of manipulator robots*. Tesis de maestría. Universidad Obrera de CataluÃ±*a* (vid. pág. 15).

G. PARNELL, J. Bishop y, 2021. Using agile methods to develop educational robotics applications. *Proceedings of the International Conference on Robotics and Automation Engineering* (vid. pág. 12).

GATES, James, 2023. Soluciones de visiÃ³*nporordenadorrevolucionandoindustrias*.. *Faster Capital* (vid. pág. 5).

HAMDI., Sami, 2023. Arduino Uno: explorando el poder de la tecnologÃa agrÃcola inteligente. *Full Skills* (vid. pág. 13).

HUAMÃ¡N, Jorge, 2023. YuMi, el robot que estÃ¡ reforestando la selva amazÃ³*nicacontecnologaenlanube*.. *El Comercio* (vid. pág. 8).

KETTUNEN P & Laanti, M., 2022. *Agile Software Development: An Empirical Study of the Benefits and Challenges of XP*. Ed. por PONSARD, Christophe. Journal of Software: Evolution y Process (vid. págs. 11, 27).

KLEIN, Corbett S., 2021. User Acceptance Testing. *Quality Assurance* (vid. pág. 31).

LÃ³*PEZ*, *JessA*., 2023. Robots agrÃcolas, una vÃa para consolidar un campo sustentable. *CIMMYT* (vid. págs. 9, 10).

LEN BASS, Paul Clements y Rick Kazman, 2021. *Software architecture in practice*. Addison-Wesley (vid. pág. 24).

MATHEWS, Frederic, 2023. Ventajas de Arduino. *The Green Monkey* (vid. pág. 14).

MEDEIROS, Javier, 2023. Aprendiendo sobre robÃ³*ticaagrcolaconArduino*. *Cursa* (vid. págs. 6, 12).

RUBIO, Manchado, 2021. *Desarrollo de un robot para la caracterizaciÃ³nyeltratamientodelasplantasenagriculturadepreci* Tesis doctoral. Escuela TÃ©cnica Superior de Ingenieros Industriales. (vid. págs. 4, 5).

SALAMEA, MarÃa JosÃ©, 2021. *EstimaciÃ³nypriorizaci*´*nderequisitosno*−*funcionalesparadesarrollodesoftware*. Tesis doctoral. Universitat PolitÃ¨cnica de Catalunya (vid. pág. 19).

SCALITER, Juan, 2023. Este robot planta mÃ¡s de 100 Ã¡rboles cada dÃa. *La RazÃ³n* (vid. pág. 9).

SOMMERVILLE, I., 2022. *Software Engineering, 10th Edition. Pearson* (vid. págs. 22, 24).

TRUJILLO, Manuel, 2023. CÃ³*momovertubrazorob*´*ticoKimoconcontrolremoto*.. *RobÃ³ticadeColombia* (vid. pág. 6).

Generado con LaTeX: 2 de diciembre de 2024: 10:54pm

Apéndices

APÉNDICE A

Historias de Usuario

Tabla A.1. Historia de usuario # 5

Historia de usuario	
Número: 5	**Nombre:** Mostrar tiempo de ejecución
Usuario: Sistema	
Prioridad en negocio: Media	**Riesgo en desarrollo:** bajo
Puntos estimados: 0.5	**Iteración asignada:** 3
Programador responsable: Keily Marrero Martínez	
Descripción: Al finalizar un proceso la consola debe mostrar el tiempo de ejecución.	
Observaciones: -	
Interfaz: 	

Tabla A.2. Historia de usuario # 6

Historia de usuario	
Número: 6	**Nombre:** Pausar y retomar el proceso
Usuario: Usuario del sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** bajo
Puntos estimados: 1	**Iteración asignada:** 3

Continúa en la próxima página

Tabla A.2. Continuación de la página anterior

Programador responsable: Keily Marrero Martínez
Descripción: El usuario debe pausar y retomar el proceso de siembra en cualquier momento luego de iniciado.
Observaciones: -
Interfaz: 

Tabla A.3. Historia de usuario # 7

Historia de usuario	
Número: 7	**Nombre:** Conectividad Bluetoth fiable
Usuario: Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** medio
Puntos estimados: 1.5	**Iteración asignada:** 1
Programador responsable: Keily Marrero Martínez	
Descripción: La conexión Bluetooth debe funcionar sin interrupciones para garantizar un control fluido y confiable del brazo robótico a distancia	
Observaciones: -	

Tabla A.4. Historia de usuario # 8

Historia de usuario	
Número: 8	**Nombre:** Mostrar posturas sembradas
Usuario: Sistema	
Prioridad en negocio: Alta	**Riesgo en desarrollo:** medio
Puntos estimados: 1.5	**Iteración asignada:** 3
Programador responsable: Keily Marrero Maertínez	
Descripción: La consola debe mostrar la cantidad total de posturas sembradas hasta el momento, para tener un registro claro del progreso realizado durante cada sesión.	
Observaciones: La cantidad de posturas se guardará en el fichero.	

Continúa en la próxima página

Tabla A.4. Continuación de la página anterior

Interfaz:

APÉNDICE B

Tarjetas CRC

Tabla B.1. Tarjeta CRC # 5

Tarjeta CRC	
Clase: SensorUltrasonico	
Responsabilidad	**Colaboración**
• Medir la distancia utilizando un sensor ultrasónico. • Proveer datos sobre la presencia de objetos o posturas.	Braccio

Tabla B.2. Tarjeta CRC # 6

Tarjeta CRC	
Clase: ControladorBluetooth	
Responsabilidad	**Colaboración**
• Gestionar la comunicación con el módulo Bluetooth. • Leer datos de comandos enviados desde la aplicación móvil. • Enviar información sobre el estado del brazo robótico.	Braccio

Tabla B.3. Tarjeta CRC # 7

Tarjeta CRC
Clase: Timer

Continúa en la próxima página

Tabla B.3. Continuación de la página anterior

Responsabilidad	**Colaboración**
• Medir el tiempo de ejecución de las acciones del brazo robótico. • Proveer información sobre el tiempo total de trabajos.	Braccio

Tabla B.4. Tarjeta CRC # 8

Tarjeta CRC	
Clase: Log	
Responsabilidad	**Colaboración**
• Actualizar la consola con mensajes de estado. • Formatear mensajes con fecha y hora	BrazoRoboticoApp (se utiliza dentro de esta clase).

Tabla B.5. Tarjeta CRC # 9

Tarjeta CRC	
Clase: Botones	
Responsabilidad	**Colaboración**
• Proveer botones para controlar el brazo robótico (iniciar, pausar, reanudar, finalizar y guardar). • Asociar cada botón a su respectivo método en BrazoRoboticoApp	BrazoRoboticoApp (los botones son parte de esta clase).

Tabla B.6. Tarjeta CRC # 10

Tarjeta CRC	
Clase: Estado	
Responsabilidad	**Colaboración**
• Mantener el estado actual del brazo robótico (desconectado, siembra en curso, siembra finalizada). • Proveer métodos para actualizar y recuperar el estado.	BrazoRoboticoApp (el estado se gestiona dentro de esta clase)

Tabla B.7. Tarjeta CRC # 11

Tarjeta CRC	
Clase: Acciones	
Responsabilidad	**Colaboración**
• Contar el número de posturas sembradas. • Guardar información sobre las acciones realizadas en un archivo de texto	BrazoRoboticoApp (las acciones se gestionan dentro de esta clase).

APÉNDICE C

Tareas de Ingeniaría

Tabla C.1. Tarea de ingeniería # 5

Tarea	
Número de tarea: 5	**Número de Historia de usuario:** 4
Nombre de la tarea: Implementar consola para mostrar estado	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 19 de septiembre de 2024	**Fecha de fin:** 23 de septiembre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se desarrolla una consola que muestra el estado actual del brazo robótico en tiempo real.	

Tabla C.2. Tarea de ingeniería # 6

Tarea	
Número de tarea: 6	**Número de Historia de usuario:** 5
Nombre de la tarea: Implementar botón de guardar fichero.	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 23 de septiembre de 2024	**Fecha de fin:** 30 de septiembre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se implementa un botón en la interfaz que permite guardar un fichero con los datos mostrados en la consola.	

Tabla C.3. Tarea de ingeniería # 7

Tarea	
Número de tarea: 7	**Número de Historia de usuario:** 6
Nombre de la tarea: Mostrar el tiempo total transcurrido	

Continúa en la próxima página

Tabla C.3. Continuación de la página anterior

Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 30 de septiembre de 2024	**Fecha de fin:** 4 de octubre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se implementa un temporizador que muestra el tiempo total transcurrido durante las operaciones del brazo robótico.	

Tabla C.4. Tarea de ingeniería # 8

Tarea	
Número de tarea: 8	**Número de Historia de usuario:** 7
Nombre de la tarea: Implementar botón de pausa y reanudación del proceso	
Tipo de tarea: Desarrollo	**Puntos estimados:** 1
Fecha de inicio: 4 de octubre de 2024	**Fecha de fin:** 11 de octubre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se desarrollan funcionalidades que permiten pausar y reanudar las operaciones del brazo robótico mediante botones en la interfaz.	

Tabla C.5. Tarea de ingeniería # 9

Tarea	
Número de tarea: 9	**Número de Historia de usuario:** 8
Nombre de la tarea: Mostrar cantidad de posturas sembradas	
Tipo de tarea: Desarrollo	**Puntos estimados:** 0.5
Fecha de inicio: 11 de octubre de 2024	**Fecha de fin:** 13 de octubre de 2024
Programador responsable: Keily Marrero Martínez	
Descripción: Se desarrolla una funcionalidad que muestre en la consola las posturas sembradas hasta el momento.|	

APÉNDICE D

Pruebas de Aceptación

Tabla D.1. Prueba de aceptación # 6

Caso de prueba de aceptación	
Código: HU3_P2	**Historia de usuario:** 3
Nombre: Verificar comunicación fluida	
Descripción: Prueba para la funcionalidad: Conectividad Bluetooth fiable	
Condiciones de ejecución: El módulo Bluetooth debe estar configurado y emparejado con el brazo	
Pasos de ejecución: • Se activa el módulo Bluetooth. • Se intenta conectar desde la aplicación. • Se verifica que la conexión se establece sin errores y se mantiene estable durante el uso. procede a usar la aplicación. .	
Resultados esperados: Insatisfactorio	

Tabla D.2. Prueba de aceptación # 7

Caso de prueba de aceptación	
Código: HU4_P1	**Historia de usuario:** 4
Nombre: Verificar consola de estado	
Descripción: Prueba para la funcionalidad: Consola de estado en la aplicación	
Condiciones de ejecución: -	
Pasos de ejecución: • Se accede a la aplicación. • Se observa el estado actual del brazo robótico. • Se verifica que el estado mostrado sea correcto según las acciones realizadas	

Continúa en la próxima página

Tabla D.2. Continuación de la página anterior

Resultados esperados: Satisfactorio.

Tabla D.3. Prueba de aceptación # 8

Caso de prueba de aceptación	
Código: HU5_P5	**Historia de usuario:** 5
Nombre: Guardar fichero con datos de la consola	
Descripción: Prueba para la funcionalidad: Guardar ficheros	
Condiciones de ejecución: La aplicación debe tener acceso al sistema de archivos del dispositivo.	
Pasos de ejecución: • Se realiza una acción con el brazo robótico. • Se guarda el estado actual utilizando la función correspondiente. • Se verifica que el archivo se crea correctamente y contiene los datos esperados.	
Resultados esperados: Satisfactorio	

Tabla D.4. Prueba de aceptación # 9

Caso de prueba de aceptación	
Código: HU6_P1	**Historia de usuario:** 6
Nombre: Verificar tiempo de ejecución	
Descripción: Prueba para la funcionalidad: Mostrar tiempo de ejecución.	
Condiciones de ejecución: El sistema debe estar en funcionamiento durante un periodo determinado.	
Pasos de ejecución: • Se inicia una operación con el brazo robótico. • Se observa el temporizador mientras se realizan acciones. • Al finalizar, se verifica que el tiempo mostrado es correcto según las operaciones realizadas y su duración total	
Resultados esperados: Satisfactorio.	

Tabla D.5. Prueba de aceptación # 10

Caso de prueba de aceptación	
Código: HU7_P1	**Historia de usuario:** 7
Nombre: Verificar pausar y retomar el proceso	
Descripción: Prueba para la funcionalidad: Pausar y retomar el proceso	

Continúa en la próxima página

Tabla D.5. Continuación de la página anterior

Condiciones de ejecución: El sistema debe estar en funcionamiento y ejecutando una operación continua..
Pasos de ejecución: • Se inicia una operación con el brazo robótico. • Se presiona el botón "Pausar". • Se verifica que la operación se detiene correctamente. • Se presiona Reanudar". • Se verifica que la operación continúa desde donde se detuvo sin errores ni pérdida del estado anterior.
Resultados esperados: Satisfactorio.

Tabla D.6. Prueba de aceptación # 11

Caso de prueba de aceptación	
Código: HU8_P1	**Historia de usuario:** 8
Nombre: Actualizar posturas sembradas.	
Descripción: Prueba para la funcionalidad: Visualización de posturas sembradas	
Condiciones de ejecución: El sistema debe estar en funcionamiento.	
Pasos de ejecución: • Se inicia el proceso • El brazo comienza el proceso de siembra • La consola va actualizando la cantidad de posturas sembradas	
Resultados esperados: Satisfactorio.	