\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### **1.1 Nodo Maestro (master\_node)**

#### **Descripción**

El nodo maestro es el coordinador global del sistema. Supervisa y controla los demás nodos, administra la conexión con el robot y centraliza la configuración global del sistema. Además, proporciona servicios para tareas de alto nivel, recopila estados de los demás nodos para generar una visión completa del estado del robot y permite reconfigurar parámetros dinámicos en tiempo real.

#### **Funciones**

1. Gestionar la conexión inicial con el robot a través del nodo de comunicación serial.
2. Supervisar el estado general del sistema solicitando información a otros nodos (movimiento, seguridad, energía, odometría y miscelánea).
3. Centralizar la configuración dinámica de los parámetros del sistema y proporcionar un único punto de control para el robot.
4. Publicar un estado global del robot accesible para otros componentes o sistemas externos.
5. Ofrecer servicios para reconfigurar dinámicamente parámetros de cualquier nodo del sistema.
6. Gestionar los comandos de movimiento, LEDs, botones y sonidos del robot.

#### **Interfaces**

**Parámetros:**

* connection\_method: Define el método de conexión al robot (usb o tcp).
  + Valores posibles:
    - **usb**: Conexión mediante un puerto serial.
    - **tcp**: Conexión mediante una dirección IP y puerto.
* connection\_time: Marca temporal del inicio de la conexión con el robot (formato ISO 8601).
* state\_update\_rate: Frecuencia de actualización del estado global del sistema (en Hz).
* dynamic\_reconfigure\_enabled: Activa o desactiva la posibilidad de reconfiguración dinámica de los parámetros.

Lista de parámetros remoto:

dynamic\_parameters = {

"communication\_node": {

"movement\_frequency": 10,

"security\_frequency": 5,

"power\_frequency": 5,

"misc\_frequency": 2,

"connection\_mode": "FULL"

},

"security\_node": {

"fit\_cliff\_left": 50,

"fit\_cliff\_front\_left": 50,

"fit\_cliff\_right": 50,

"fit\_cliff\_front\_right": 50,

"safe\_zone\_radius": 1.0,

"wall\_proximity\_threshold": 30,

"safety\_check\_rate": 10,

"publish\_emergency\_events": True

},

"motion\_control\_node": {

"max\_linear\_velocity": 0.5,

"max\_angular\_velocity": 1.0,

"odom\_frequency": 10,

"brush\_motor\_power": 128,

"vacuum\_power": 200,

"linear\_tolerance": 0.01,

"angular\_tolerance": 0.01

},

"charging\_node": {

"low\_battery\_threshold": 20,

"energy\_check\_rate": 1,

"overload\_protection\_enabled": True,

"return\_to\_dock\_enabled": True

},

"miscellaneous\_node": {

"led\_update\_rate": 5,

"sound\_volume": 128,

"default\_led\_color": 255,

"default\_led\_intensity": 200

},

"odometry\_node": {

"odom\_frame\_id": "odom",

"base\_frame\_id": "base\_link",

"wheel\_base": 0.35,

"wheel\_radius": 0.05,

"publish\_rate": 10,

"noise\_filter\_enabled": True

}

}

**Tópicos:**

* **Publica:**
  + /roomba/system/state/global: Publica el estado global del robot, recopilando información de todos los nodos. **Contenido del estado global:**
    - **Energía:** Carga de la batería, voltaje, temperatura.
    - **Movimiento:** Velocidad actual, posición y orientación.
    - **Seguridad:** Alertas de colisiones, caídas o emergencias.
    - **Miscelánea:** Botones presionados, estado de LEDs, reproducción de sonidos.
  + /roomba/system/command/movement: Publica comandos de movimiento para el robot.
  + /roomba/system/command/motor\_comman: Publica comandos de motores auxiliares para el robot.
* **Suscribe:**
  + /roomba/security/state: Información de seguridad enviada por el nodo de seguridad.
  + /roomba/motion/state: Información de movimiento enviada por el nodo de control de movimiento.
  + /roomba/power/state: Información de energía enviada por el nodo de energía.
  + /roomba/power/ir: Información de sensores ir enviada por el nodo de energía.
  + /roomba/power/overcurrent: Información de sobrecarga enviada por el nodo de energía.
  + /roomba/odom/state: Datos de estado enviados por el nodo de odometría.
  + /roomba/misc/state: Información miscelánea enviada por el nodo misceláneo.
  + /roomba/misc/buttons: Información sobre los botones enviada por el nodo misceláneo.
  + /roomba/odom/odom: Datos de odometría enviados por el nodo de odometría.
  + /roomba/odom/state: Datos de trnsformadas enviados por el nodo de odometría.

**Servicios:**

* /roomba/communica﻿tion/connect: Establece conexión inicial con el robot.
  + **Entrada:**
    - method: Método de conexión (usb o tcp).
    - details: Parámetros específicos según el método:
      * Para USB: Nombre del puerto serial.
      * Para TCP: Dirección IP y puerto.
  + **Salida:**
    - success: Booleano que indica si la conexión fue exitosa.
    - message: Mensaje adicional en caso de error.
* /roomba/system/status: Devuelve el estado global del sistema.
  + **Salida:**
    - Estado energético: Batería, voltaje, corriente y temperatura.
    - Estado de movimiento: Velocidades y posición del robot.
    - Estado de seguridad: Alertas de colisión, caídas y emergencias.
    - Estado misceláneo: Información de LEDs y botones.
* /roomba/system/reconfigure: Permite actualizar dinámicamente los parámetros de los nodos.
  + **Entrada:**
    - node: Nodo al que se aplica la reconfiguración.
    - parameter: Parámetro a modificar.
    - value: Nuevo valor del parámetro.
  + **Salida:**
    - success: Booleano que indica si el cambio fue exitoso.
    - message: Mensaje adicional en caso de error.
* /roomba/motion/command: Envía comandos de movimiento, LEDs, sonidos o display al robot.
  + **Entrada:**
    - type: Tipo de comando (movement, leds, sound, display).
    - data: Información específica del comando.
  + **Salida:**
    - success: Booleano que indica si el comando fue ejecutado con éxito.
    - message: Mensaje adicional en caso de error.

### 

### /roomba/motion/wheels\_motors

**Descripción:** Envía comandos de movimiento a los motores del Nodo de Movimiento.

**Entrada:**

**{**

**"linear\_velocity": 0.6,**

**"angular\_velocity": 0.2**

**}**

**Salida:**

* success: Booleano que indica si el comando se ejecutó correctamente.
* message: Mensaje adicional.

### 

### **/roomba/power/check\_status**

**Descripción:** Devuelve información detallada del estado energético del robot desde el Nodo de Carga.

**Entrada:**

* Sin parámetros de entrada.

**Salida:**

* success: Booleano que indica si el estado se obtuvo correctamente.
* data: Objeto con detalles del estado energético.

{

"battery\_charge": 85,

"voltage": 14.4,

"current": -1.5,

"temperature": 35,

"state": "discharging"

}

* message: Mensaje adicional.

### 

### **/roomba/odometry/reset**

**Descripción:** Resetea la posición y orientación del robot desde el Nodo de Odometría.

**Entrada:**

* Sin parámetros de entrada.

**Salida:**

* success: Booleano que indica si la operación se completó correctamente.
* message: Mensaje adicional.

### **/roomba/misc/set\_leds**

**Descripción:** Configura los LEDs del robot a través del Nodo de Miscelánea.

**Entrada:**

{

"power\_color": 128,

"power\_intensity": 255,

"status\_leds": {

"dock": true,

"spot": false

}

}

**Salida:**

* success: Booleano que indica si la configuración se aplicó correctamente.
* message: Mensaje adicional.

### 

### **/roomba/misc/play\_song**

**Descripción:** Reproduce una canción configurada en el Nodo de Miscelánea.

**Entrada:**

{

"song\_number": 1

}

**Salida:**

* success: Booleano que indica si la canción comenzó a reproducirse correctamente.
* message: Mensaje adicional.
* /roomba/communication/priority:

**Descripción:** Permite ajustar las prioridades de los mensajes.

* + **Entrada:**
    - message\_type: Tipo de mensaje a priorizar (por ejemplo, movimiento o seguridad).
    - priority: Nivel de prioridad (alto, medio, bajo).
  + **Salida:**
    - success: Booleano que indica si la reconfiguración fue exitosa.
    - message: Mensaje adicional en caso de error.

#### **Interacción**

* **Con otros nodos:**
  + **Nodo de Comunicación:**
    - Solicita establecer la conexión con el robot.
    - Envía comandos de configuración inicial.
  + **Nodo de Movimiento:**
    - Recibe información sobre velocidades, posición y orientación.
    - Envía comandos de movimiento.
  + **Nodo de Seguridad:**
    - Recibe alertas y estados críticos, como colisiones o caídas.
  + **Nodo de Energía:**
    - Recibe información sobre la batería, corriente y temperatura.
  + **Nodo de Odometría:**
    - Recibe datos procesados sobre la posición del robot.
  + **Nodo de Miscelánea:**
    - Recibe datos relacionados con botones, LEDs y sonidos.
* **Con sistemas externos:**
  + Publica el estado global del robot para aplicaciones externas o dashboards.

Ejemplo de Estado Global Publicado en /roomba/system/state/global:



#### **Consideraciones Técnicas:**

1. **Configuración Dinámica:** Utiliza el servicio /roomba/system/reconfigure para cambiar parámetros del robot en tiempo real sin necesidad de reiniciar los nodos.
2. **Frecuencia de Estado Global:** Configura state\_update\_rate según la capacidad de procesamiento y las necesidades del sistema. Valores típicos: 1-5 Hz.
3. **Manejo de Errores:** Implementa lógica robusta para gestionar fallos en la conexión con el robot o en la comunicación con otros nodos.

### **1.2 Nodo de Comunicación Serial (communication\_node)**

#### **Descripción**

Este nodo actúa como el intermediario exclusivo entre el sistema y el hardware del robot a través del puerto serial. Centraliza el acceso al puerto serie, asegurando que solo un nodo interactúe directamente con el robot. Publica datos provenientes de los sensores y recibe comandos de otros nodos para controlar el robot.

#### **Funciones**

1. Gestionar el acceso exclusivo al puerto serial para evitar conflictos entre nodos.
2. Publicar datos de los sensores y el estado del robot, como movimiento, energía y seguridad.
3. Recibir y procesar comandos de otros nodos, como movimiento, display, sonidos y LEDs, para enviarlos al robot.
4. Gestionar la prioridad de los mensajes para garantizar la ejecución eficiente de las acciones críticas.

#### **Interfaces**

**Parámetros:**

* connection\_serial: Nombre del puerto serial utilizado para conectarse al robot.
* connection\_mode: Modo de conexión con el robot:
  + **FULL:** Acceso completo al robot.
  + **SAFE:** Modo de seguridad, limita las acciones cuando los sensores de seguridad se activan.
* movement\_frequency: Frecuencia de publicación de los datos de movimiento (en Hz).
* security\_frequency: Frecuencia de publicación de los datos de seguridad (en Hz).
* power\_frequency: Frecuencia de publicación de los datos de energía (en Hz).
* misc\_frequency: Frecuencia de publicación de datos misceláneos (en Hz).
* state\_communication\_frequency: Frecuencia de publicación de datos estado (en Hz).
* movement\_filter: Lista de sensores de movimiento a monitorear.
* security\_filter: Lista de sensores de seguridad a monitorear.
* power\_filter: Lista de sensores de energía a monitorear.
* misc\_filter: Lista de otros sensores, como LEDs y botones.

**Tópicos:**

* **Publica:**
  + /roomba/communication**/**security: Datos de sensores relacionados con caídas, colisiones y proximidad.
    - Ejemplo: Señales de sensores de acantilado y colisión.
  + /roomba/communication**/**movement: Datos de los encoders, distancia y ángulo recorrido por el robot.
  + /roomba/communication**/**power: Información relacionada con el estado de la batería (carga, voltaje, corriente, temperatura).
  + /roomba/communication**/**misc: Datos misceláneos como estado de los botones, LEDs y sonidos.
  + /roomba/communication**/**state: Datos relacionados con el estado del nodo..
* **Suscribe:**
  + /roomba/motion/motors\_command: Recibe comandos de movimiento, como velocidades de las ruedas.
  + /roomba/misc/leds\_command: Recibe comandos relacionados con el display y los LEDs.
  + /roomba/misc/song\_command: Recibe comandos para reproducir sonidos y canciones.

**Servicios:**

* /roomba/communication/connect: Establece la conexión inicial con el robot.
  + **Entrada:**
    - port: Puerto serial donde se encuentra conectado el robot.
    - mode: Modo de conexión (**FULL** o **SAFE**).
  + **Salida:**
    - success: Booleano que indica si la conexión fue exitosa.
    - message: Mensaje adicional en caso de error.
* /roomba/communication/priority: Permite ajustar las prioridades de los mensajes.
  + **Entrada:**
    - message\_type: Tipo de mensaje a priorizar (por ejemplo, movimiento o seguridad).
    - priority: Nivel de prioridad (alto, medio, bajo).
  + **Salida:**
    - success: Booleano que indica si la reconfiguración fue exitosa.
    - message: Mensaje adicional en caso de error.

Permite actualizar dinámicamente los

* /roomba/system/reconfigure Permite actualizar dinámicamente los

#### **Entrada:**

* node: "communication\_node".
* parameter: Nombre del parámetro a modificar. Parámetros reconfigurables:
  + movement\_frequency: Frecuencia de publicación de los datos de movimiento (Hz).
  + security\_frequency: Frecuencia de publicación de los datos de seguridad (Hz).
  + power\_frequency: Frecuencia de publicación de los datos de energía (Hz).
  + misc\_frequency: Frecuencia de publicación de datos misceláneos (Hz).
  + connection\_mode: Define el modo de conexión (FULL o SAFE).

#### **Salida:**

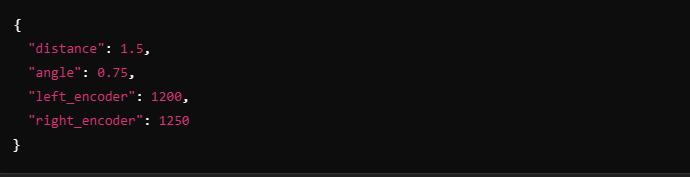
* success: Booleano que indica si el cambio fue exitoso.
* message: Mensaje adicional en caso de error.

#### **Interacción**

* **Con otros nodos:**
  + **Nodo Maestro:**
    - Establece la conexión inicial con el robot.
    - Recibe comandos para configurar la conexión serial y ajustar parámetros.
  + **Nodo de Movimiento:**
    - Recibe comandos relacionados con velocidades y motores.
    - Publica datos de movimiento como encoders, distancia y ángulo.
  + **Nodo de Seguridad:**
    - Publica datos de sensores de colisiones y caídas.
    - Recibe comandos para detener el robot en caso de emergencia.
  + **Nodo de Energía:**
    - Publica datos de energía como carga de batería, voltaje y temperatura.
  + **Nodo Misceláneo:**
    - Publica datos relacionados con botones, LEDs y sonidos.
* **Con el hardware del robot:**
  + Envia comandos seriales utilizando los protocolos del Open Interface.
  + Procesa y publica las respuestas recibidas desde el robot.

#### **Ejemplo de Datos Publicados**

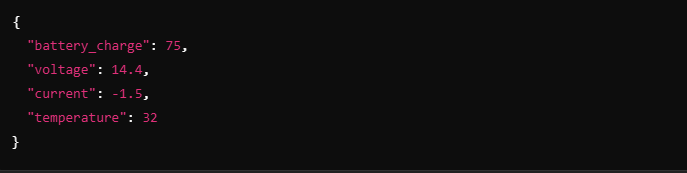
**Tópico /roomba/movement:**



Tópico /roomba/security:



Tópico /roomba/power:



#### **Consideraciones Técnicas:**

1. **Prioridades:** Utiliza el servicio /roomba/system/priority para priorizar mensajes críticos (por ejemplo, datos de sensores de seguridad sobre comandos de movimiento).
2. **Frecuencia de Publicación:** Configura las frecuencias de publicación (movement\_frequency, security\_frequency, etc.) según las necesidades del sistema.
3. **Modo de Conexión:** Define el modo de conexión más adecuado (FULL o SAFE) según el entorno y las posibles restricciones de seguridad.
4. **Errores de Conexión:** Implementa manejo de errores robusto en el servicio /roomba/system/connect para gestionar fallos en la conexión serial.

### **1.3 Nodo de Seguridad (security\_node)**

#### **Descripción**

El nodo de seguridad procesa y publica datos relacionados con los sensores de seguridad del robot, como detección de acantilados, colisiones y proximidad. Es responsable de definir un estado de seguridad para el robot, evitar situaciones de riesgo y reaccionar ante emergencias como colisiones o caídas.

#### **Funciones**

1. Leer datos de los sensores de seguridad publicados por el nodo de comunicación.
2. Procesar y evaluar la información de sensores para identificar riesgos como colisiones, caídas y proximidad a obstáculos.
3. Publicar estados de seguridad para que otros nodos los utilicen.
4. Detectar eventos de emergencia y notificar a otros nodos.
5. Permitir la configuración dinámica de parámetros relacionados con la seguridad, como zonas seguras y umbrales.

#### **Interfaces**

**Parámetros:**

* fit\_cliff\_left: Señal mínima aceptable para el sensor de acantilado izquierdo.
* fit\_cliff\_front\_left: Señal mínima aceptable para el sensor de acantilado frontal izquierdo.
* fit\_cliff\_right: Señal mínima aceptable para el sensor de acantilado derecho.
* fit\_cliff\_front\_right: Señal mínima aceptable para el sensor de acantilado frontal derecho.
* safe\_zone\_radius: Radio virtual alrededor del robot dentro del cual no se deben detectar obstáculos.
* wall\_proximity\_threshold: Distancia mínima (en señal) para considerar que una pared está demasiado cerca.
* virtual\_wall\_enabled: Habilita o deshabilita el uso de paredes virtuales.
* safety\_check\_rate: Frecuencia en Hz con la que se verifican los datos de sensores para tomar decisiones.
* publish\_emergency\_events: Activa o desactiva la publicación de eventos de emergencia.
* state\_security\_frequency: Frecuencia con la que se publican los estados de seguridad.

**Tópicos:**

* **Publica:**
  + /roomba/security/state: Publica el estado de seguridad del robot. **Ejemplo de estados:**
    - **SAFE:** Todo está bien.
    - **OBSTACLE:** Se detectó un obstáculo (choque, pared, etc.).
    - **CLIFF:** Riesgo de caída detectado.
    - **EMERGENCY:** Estado de emergencia, se ha solicitado una parada.
    - **COLLISION:** Se ha producido una colisión.
    - **HILLY:** Accidente detectado.
* **Suscribe:**
  + /roomba/security/security: Datos de los sensores relacionados con caídas, colisiones y proximidad, publicados por el nodo de comunicación.

**Servicios:**

* /roomba/motion/emergency\_stop: Solicita detener inmediatamente el robot en caso de emergencia.
  + **Entrada:** Ninguna.
  + **Salida:**
    - success: Booleano que indica si la parada fue exitosa.
    - message: Mensaje adicional en caso de error.
* /roomba/system/reconfigure; Permite actualizar dinámicamente los parámetros de los nodos.

#### **Entrada:**

* node: "security\_node".
* parameter: Nombre del parámetro a modificar. Parámetros reconfigurables:
  + fit\_cliff\_left, fit\_cliff\_front\_left, fit\_cliff\_right, fit\_cliff\_front\_right: Umbrales mínimos para los sensores de acantilado.
  + safe\_zone\_radius: Radio virtual alrededor del robot donde no debe haber obstáculos.
  + wall\_proximity\_threshold: Distancia mínima para considerar una pared como demasiado cercana.
  + safety\_check\_rate: Frecuencia de verificación de los datos de los sensores (Hz).
  + publish\_emergency\_events: Activa o desactiva la publicación de eventos de emergencia.

#### **Salida:**

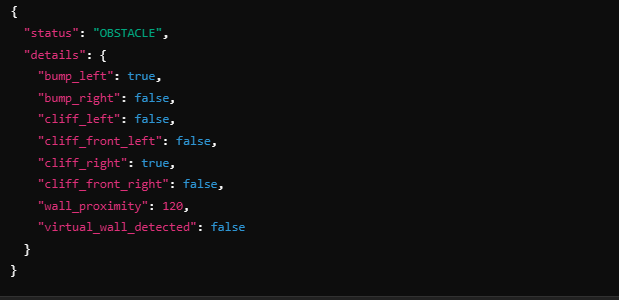
* success: Booleano que indica si el cambio fue exitoso.
* message: Mensaje adicional en caso de error.

#### **Interacción**

* **Con otros nodos:**
  + **Nodo de Comunicación:**
    - Recibe datos de los sensores del robot, como detección de acantilados, colisiones y proximidad.
  + **Nodo Maestro:**
    - Proporciona el estado de seguridad global para su integración en el estado general del robot.
    - Recibe configuraciones dinámicas de parámetros relacionados con la seguridad.
  + **Nodo de Movimiento:**
    - Proporciona alertas críticas para que el nodo de movimiento detenga al robot en caso de riesgo.
    - Solicita paradas de emergencia en situaciones críticas.

#### **Ejemplo de Datos Publicados**

**Tópico** /roomba/security/state**:**



#### **Consideraciones Técnicas:**

1. **Configuración de Umbrales:** Ajusta los parámetros como fit\_cliff\_\* y wall\_proximity\_threshold según el entorno operativo para minimizar falsos positivos.
2. **Frecuencia de Verificación:** Configura safety\_check\_rate para equilibrar la detección oportuna de riesgos y el uso eficiente de recursos.
3. **Eventos de Emergencia:** Utiliza el parámetro publish\_emergency\_events para activar/desactivar la publicación de eventos en función de los requisitos de la aplicación.
4. **Reconfiguración Dinámica:** El servicio /roomba/security/reconfigure permite modificar los parámetros sin reiniciar el sistema, facilitando la adaptación a nuevos entornos.

### **1.4 Nodo de Control de Movimiento (motion\_control\_node)**

#### **Descripción**

El nodo de control de movimiento se encarga de recibir y procesar los comandos de velocidad lineal y angular del robot, traduciendo estas órdenes en acciones concretas que los motores del robot puedan ejecutar. También gestiona los motores secundarios (cepillos y aspiradora).

#### **Funciones**

1. Calcular y publicar datos de odometría (posición y orientación del robot).
2. Controlar los motores secundarios del robot, como cepillos y aspiradora.
3. Implementar paradas de emergencia en caso de recibir alertas críticas desde el nodo de seguridad.
4. Generar las transformaciones necesarias para la navegación.

#### **Interfaces**

**Parámetros:**

* max\_linear\_velocity: Límite superior para la velocidad lineal (en m/s).
* max\_angular\_velocity: Límite superior para la velocidad angular (en rad/s).
* brush\_motor\_power: Potencia predeterminada de los motores de los cepillos (0-255).
* vacuum\_power: Potencia de succión del aspirador.
* emergency\_stop\_timeout: Tiempo de espera antes de deshabilitar los motores tras una emergencia (en segundos).
* linear\_tolerance: Tolerancia permitida en movimientos lineales antes de considerar el objetivo alcanzado.
* angular\_tolerance: Tolerancia permitida en movimientos angulares antes de considerar el objetivo alcanzado.
* motion\_command\_frequency: Frecuencia a la que se procesan los comandos de movimiento.
* estate\_motion\_frecuency: Frecuencia con la se envían los datos de estado.

**Tópicos:**

* **Publica:**
  + /roomba/motion/motors\_comman: Comandos de movimiento que el nodo traduce en acciones para los motores.
  + /roomba/motion/state: Publica el estado actual de los motores. **Ejemplo de estados:**
    - **RUNNING:** Los motores están activos y ejecutando movimientos.
    - **STOPPED:** Los motores están detenidos.
    - **EMERGENCY\_STOP:** Los motores se detuvieron por una emergencia.
* **Suscribe:**
  + /roomba/communication/movement: Datos de movimiento enviados por el nodo de comunicación.
  + /roomba/system/command/movement: Comandos de movimiento enviados por el nodo maestro.
  + /roomba/system/command/motor\_comman: Comandos de motores auxiliares enviados por el nodo maestro.

**Servicios:**

* /roomba/moti﻿on/emergency\_stop: Detiene inmediatamente los motores en caso de recibir un evento de emergencia.
  + **Entrada:** Ninguna.
  + **Salida:**
    - success: Booleano que indica si los motores se detuvieron exitosamente.
    - message: Mensaje adicional en caso de error.
* /roomba/motion/wheels\_motors: Permite enviar comandos directos a los motores de las ruedas.
  + **Entrada:**
    - linear\_velocity: Velocidad lineal deseada (en m/s).
    - angular\_velocity: Velocidad angular deseada (en rad/s).
  + **Salida:**
    - success: Booleano que indica si el comando fue ejecutado exitosamente.
    - message: Mensaje adicional en caso de error.
* /roomba/motion/command: Controla los motores secundarios (cepillos y aspiradora).
  + **Entrada:**
    - brush\_motor\_power: Potencia del motor del cepillo principal.
    - side\_brush\_power: Potencia del motor del cepillo lateral.
    - vacuum\_power: Potencia del motor del aspirador.
  + **Salida:**
    - success: Booleano que indica si los motores se configuraron correctamente.
    - message: Mensaje adicional en caso de error.
* /roomba/system/reconfigure: Permite actualizar dinámicamente los parámetros de los nodos.

#### **Entrada:**

* node: "motion\_control\_node".
* parameter: Nombre del parámetro a modificar. Parámetros reconfigurables:
  + max\_linear\_velocity: Límite superior para la velocidad lineal (m/s).
  + max\_angular\_velocity: Límite superior para la velocidad angular (rad/s).
  + odom\_frequency: Frecuencia de publicación de la odometría (Hz).
  + brush\_motor\_power: Potencia predeterminada del cepillo principal (0-255).
  + vacuum\_power: Potencia predeterminada de la aspiradora.
  + linear\_tolerance: Margen de error permitido para movimientos lineales.
  + angular\_tolerance: Margen de error permitido para movimientos angulares.

#### **Salida:**

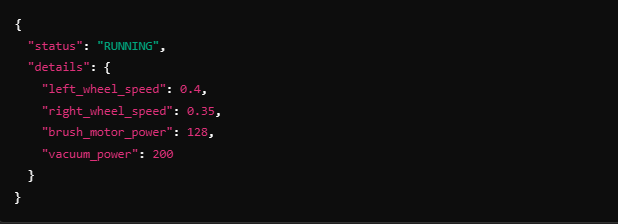
* success: Booleano que indica si el cambio fue exitoso.
* message: Mensaje adicional en caso de error.

#### **Interacción**

* **Con otros nodos:**
  + **Nodo Maestro:**
    - Recibe comandos de velocidad y configuraciones dinámicas.
    - Envia información de odometría y estado de motores.
  + **Nodo de Comunicación:**
    - Envía comandos directos a los motores.
    - Recibe información de los encoders y datos de movimiento.
  + **Nodo de Seguridad:**
    - Recibe alertas de emergencia y detiene los motores en caso de riesgo.
  + **Nodo de Navegación (si aplica):**
    - Recibe comandos de velocidad procesados para la planificación de rutas.

#### **Ejemplo de Datos Publicados**

Tópico /roomba/motion/state:



#### **Consideraciones Técnicas:**

1. **Tolerancias:** Ajusta linear\_tolerance y angular\_tolerance según los requisitos de precisión del sistema.
2. **Frecuencias:** Configura motion\_command\_frequency y odom\_frequency de acuerdo con las necesidades de navegación y los recursos disponibles.
3. **Paradas de Emergencia:** Asegúrate de que el servicio /roomba/emergency\_stop pueda ser invocado en cualquier momento, independientemente del estado actual del nodo.
4. **Reconfiguración Dinámica:** Utiliza el servicio /roomba/system/reconfigure (desde el nodo maestro) para modificar parámetros como límites de velocidad sin reiniciar el sistema.

### **1.6 Nodo de Gestión de Carga (charging\_node)**

#### **Descripción**

El nodo de gestión de carga supervisa todos los aspectos relacionados con el estado energético del robot. Monitorea continuamente el nivel de batería, voltaje, corriente y temperatura. Además, maneja interacciones relacionadas con la base de carga, sensores infrarrojos y situaciones de sobrecarga o sobrecalentamiento.

#### **Funciones**

1. Monitorear el estado de la batería, incluyendo carga, voltaje, corriente y temperatura.
2. Detectar y notificar sobre eventos de sobrecarga en los motores o sistemas de energía.
3. Coordinar con el nodo de navegación para planificar el regreso a la base de carga cuando el nivel de batería sea bajo.
4. Publicar información del estado energético para otros nodos.
5. Proporcionar servicios para activar modos de ahorro de energía o verificar el estado completo de la batería.

#### **Interfaces**

**Parámetros:**

* low\_battery\_threshold: Nivel de batería (en %) bajo el cual se recomienda regresar a la base de carga.
* energy\_check\_rate: Frecuencia en Hz a la que se verifica el estado energético.
* overload\_protection\_enabled: Habilita o deshabilita la protección contra sobrecargas.
* return\_to\_dock\_enabled: Activa o desactiva el regreso automático a la base de carga cuando el nivel de batería es bajo.
* power\_frecuency: Frecuencia con la que se publica los datos de energia

**Tópicos:**

* **Publica:**
  + /roomba/power/state/: Publica información detallada sobre el estado de la batería. **Ejemplo de contenido:**
    - battery\_charge: Carga actual de la batería (en %).
    - battery\_capacity: Capacidad máxima de la batería.
    - voltage: Voltaje actual.
    - current: Corriente consumida por el sistema.
    - temperature: Temperatura actual de la batería.
    - power\_state: Estado general (p. ej., "charging", "discharging").
  + /roomba/power/ir/: Publica datos de los sensores infrarrojos relacionados con la base de carga.
    - ir\_char\_omni: Señal infrarroja omnidireccional.
    - ir\_char\_left: Señal infrarroja detectada a la izquierda.
    - ir\_char\_right: Señal infrarroja detectada a la derecha.
  + /roomba/power/overcurrent/: Publica eventos de sobrecorriente en los motores.
    - left\_wheel: Sobrecorriente en la rueda izquierda.
    - right\_wheel: Sobrecorriente en la rueda derecha.
    - main\_brush: Sobrecorriente en el cepillo principal.
    - side\_brush: Sobrecorriente en el cepillo lateral.
* **Suscribe:**
  + /roomba/communication/power: Datos de energia publicados por el nodo de comunicación.

**Servicios:**

* /roomba/power/enable\_energy\_saving: Activa el modo de ahorro de energía.
  + **Entrada:** Ninguna.
  + **Salida:**
    - success: Booleano que indica si el modo fue activado con éxito.
    - message: Mensaje adicional en caso de error.
* /roomba/power/disable\_energy\_saving: Desactiva el modo de ahorro de energía.
  + **Entrada:** Ninguna.
  + **Salida:**
    - success: Booleano que indica si el modo fue desactivado con éxito.
    - message: Mensaje adicional en caso de error.
* /roomba/power/check\_status: Devuelve el estado completo de la batería.
  + **Salida:**
    - battery\_charge: Carga actual de la batería (en %).
    - voltage: Voltaje actual.
    - current: Corriente consumida.
    - temperature: Temperatura actual.
    - power\_state: Estado general de la batería (p. ej., "charging", "discharging").
* /roomba/system/reconfigure: Permite actualizar dinámicamente los parámetros de los nodos.

#### **Entrada:**

* node: "charging\_node".
* parameter: Nombre del parámetro a modificar. Parámetros reconfigurables:
  + low\_battery\_threshold: Nivel mínimo de batería para activar el regreso a la base de carga (%).
  + energy\_check\_rate: Frecuencia de verificación de los datos energéticos (Hz).
  + overload\_protection\_enabled: Activa o desactiva la protección contra sobrecargas.
  + return\_to\_dock\_enabled: Activa o desactiva el regreso automático a la base de carga.

#### **Salida:**

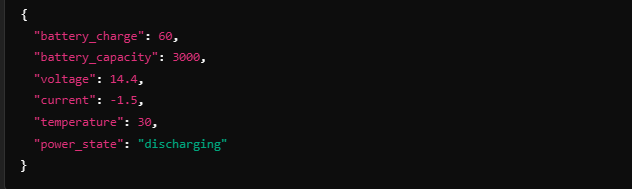
* success: Booleano que indica si el cambio fue exitoso.
* message: Mensaje adicional en caso de error.

#### **Interacción**

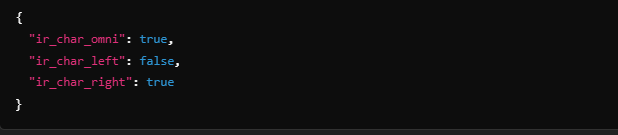
* **Con otros nodos:**
  + **Nodo Maestro:**
    - Proporciona información del estado energético.
    - Recibe comandos para activar o desactivar el modo de ahorro de energía.
  + **Nodo de Navegación:**
    - Coordina el regreso a la base de carga cuando el nivel de batería es bajo.
  + **Nodo de Movimiento:**
    - Recibe alertas de sobrecarga y detiene los motores en caso necesario.
  + **Nodo de Comunicación:**
    - Recibe información de los sensores de batería e infrarrojos desde el robot.
* **Con el hardware del robot:**
  + Supervisa datos energéticos directamente desde el robot, utilizando sensores de batería y señales infrarrojas.

#### **Ejemplo de Datos Publicados**

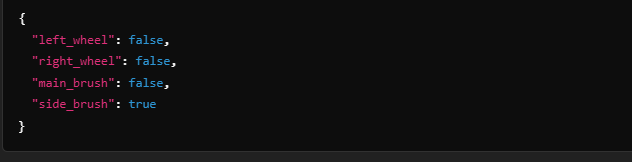
**Tópico /roomba/power/state/:**



Tópico /roomba/power/ir/:



Tópico /roomba/movement/overcurrent/:



#### **Consideraciones Técnicas:**

1. **Protección contra Sobrecarga:** Asegúrate de habilitar overload\_protection\_enabled para prevenir daños al sistema en caso de corrientes excesivas.
2. **Umbral de Batería Baja:** Configura low\_battery\_threshold para garantizar que el robot pueda regresar a la base antes de agotar completamente la batería.
3. **Sensores Infrarrojos:** Los datos de ir\_char\_\* deben utilizarse para alinear correctamente el robot con la base de carga.
4. **Frecuencia de Verificación:** Ajusta energy\_check\_rate para equilibrar el monitoreo constante y el uso eficiente de los recursos del sistema.

### **1.6 Nodo de Miscelánea (miscellaneous\_node)**

#### **Descripción**

El nodo de miscelánea gestiona los datos y comandos relacionados con funciones adicionales del robot, como los LEDs, botones y sonidos. Este nodo proporciona una interfaz centralizada para interactuar con estos elementos, permitiendo su control y monitoreo en tiempo real.

#### **Funciones**

1. Supervisar el estado de los botones físicos del robot y publicarlo para otros nodos.
2. Gestionar la reproducción de sonidos y canciones configurables.
3. Controlar los LEDs del robot, permitiendo ajustar colores, intensidades y patrones.
4. Proporcionar servicios para personalizar las configuraciones de sonidos y LEDs.

#### **Interfaces**

**Parámetros:**

* button\_check\_rate: Frecuencia en Hz con la que se verifican los estados de los botones.
* led\_update\_rate: Frecuencia en Hz con la que se actualizan los LEDs.
* sound\_volume: Nivel de volumen predeterminado para los sonidos (0-255).
* default\_led\_color: Color inicial predeterminado para los LEDs de estado.
* default\_led\_intensity: Intensidad inicial predeterminada para los LEDs de estado.
* state\_misc\_frecuency: Frecuencia con la que se envían los datos de estado.

**Tópicos:**

* **Publica:**
  + /roomba/misc/buttons: Publica el estado de los botones del robot. **Ejemplo de botones:**
    - clean
    - dock
    - spot
    - hour, minute, day, schedule.
  + /roomba/misc/status: Publica información sobre la canción y leds
    - song\_number: Número de la canción.
    - is\_playing: Indica si una canción está en reproducción.
    - power\_color
    - power\_intensity
    - status\_leds
  + /roomba/misc/leds\_comman: Publica comandos sobre leds al codo de comunicaciones.
  + /roomba/misc/song\_command: Publica comandos sobre sonidos al codo de comunicaciones.
* **Suscribe:**
  + /roomba/communication/misc: Datos de los botones publicados por el nodo de comunicación.

**Servicios:**

* /roomba/misc/set\_leds: Configura el estado de los LEDs del robot.
  + **Entrada:**
    - color: Color deseado del LED (ejemplo: RGB o escala de 0-255).
    - intensity: Intensidad deseada del LED (0-255).
  + **Salida:**
    - success: Booleano que indica si la configuración fue exitosa.
    - message: Mensaje adicional en caso de error.
* /roomba/misc/play\_song: Reproduce una canción preconfigurada.
  + **Entrada:**
    - song\_number: Número de la canción a reproducir.
  + **Salida:**
    - success: Booleano que indica si la canción comenzó a reproducirse.
    - message: Mensaje adicional en caso de error.
* /roomba/misc/stop\_song: Detiene cualquier canción que esté en reproducción.
  + **Entrada:** Ninguna.
  + **Salida:**
    - success: Booleano que indica si la reproducción se detuvo.
    - message: Mensaje adicional en caso de error.
* /roomba/misc/configure\_song: Configura una canción personalizada.
  + **Entrada:**
    - song\_number: Número de la canción a configurar.
    - notes: Secuencia de notas (ejemplo: [(nota, duración)]).
  + **Salida:**
    - success: Booleano que indica si la canción fue configurada correctamente.
    - message: Mensaje adicional en caso de error.
* /roomba/system/reconfigure: Permite actualizar dinámicamente los parámetros de los nodos.

#### **Entrada:**

* node: "miscellaneous\_node".
* parameter: Nombre del parámetro a modificar. Parámetros reconfigurables:
  + led\_update\_rate: Frecuencia de actualización de los LEDs (Hz).
  + sound\_volume: Nivel de volumen predeterminado para sonidos (0-255).
  + default\_led\_color: Color inicial predeterminado de los LEDs.
  + default\_led\_intensity: Intensidad inicial predeterminada de los LEDs.

#### **Salida:**

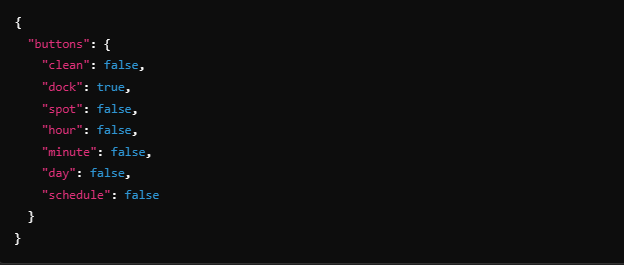
* success: Booleano que indica si el cambio fue exitoso.
* message: Mensaje adicional en caso de error.

#### **Interacción**

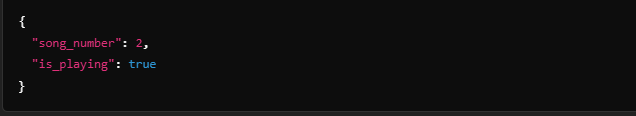
* **Con otros nodos:**
  + **Nodo Maestro:**
    - Recibe comandos para configurar LEDs y reproducir sonidos.
    - Envía el estado actual de los botones, LEDs y canciones.
  + **Nodo de Comunicación:**
    - Recibe comandos para interactuar con los elementos físicos del robot.
  + **Nodo de Seguridad:**
    - Puede activar LEDs o sonidos de advertencia en caso de alertas críticas.
* **Con el hardware del robot:**
  + Envía comandos para controlar los LEDs, botones y sonidos.
  + Recibe datos sobre el estado de los botones físicos y la reproducción de sonidos.

#### **Ejemplo de Datos Publicados**

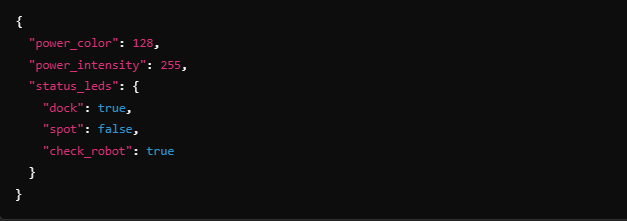
**Tópico /roomba/misc/buttons:**



Tópico /roomba/misc/songs:



Tópico /roomba/misc/leds:



#### **Consideraciones Técnicas:**

1. **LEDs Personalizables:** Utiliza el servicio /roomba/misc/set\_leds para configurar colores e intensidades que se ajusten a las necesidades de la aplicación (p. ej., advertencias o estados específicos).
2. **Canciones Personalizadas:** Configura canciones con /roomba/misc/configure\_song para añadir sonidos relevantes a diferentes eventos.
3. **Frecuencias de Actualización:** Ajusta button\_check\_rate y led\_update\_rate según las necesidades del sistema y la capacidad de procesamiento.
4. **Manejo de Errores:** Implementa un sistema robusto para manejar errores en comandos relacionados con LEDs y sonidos.

### **1.7 Nodo de Odometría (odometry\_node)**

#### **Descripción**

El nodo de odometría calcula y publica la posición y orientación del robot en el espacio utilizando datos de los encoders de las ruedas y otros sensores de movimiento. Este nodo genera transformaciones necesarias para la navegación y sirve como base para algoritmos de localización y mapeo.

#### **Funciones**

1. Procesar los datos de encoders para calcular la posición y orientación del robot (odometría).
2. Publicar las transformaciones necesarias para los sistemas de navegación (TF entre base\_frame\_id y odom\_frame\_id).
3. Filtrar y procesar datos de movimiento para mejorar la precisión de la odometría.
4. Proporcionar datos de odometría en tiempo real a otros nodos.

#### **Interfaces**

**Parámetros:**

* odom\_frame\_id: Identificador del marco de referencia de la odometría (por ejemplo, "odom").
* base\_frame\_id: Identificador del marco base del robot (por ejemplo, "base\_link").
* wheel\_base: Distancia entre las ruedas del robot (en metros).
* wheel\_radius: Radio de las ruedas del robot (en metros).
* ticks\_per\_revolution: Número de ticks del encoder por revolución de la rueda.
* odom\_frecuency: Frecuencia de publicación de los datos de odometría (en Hz).
* tf\_frecuency: Frecuencia de publicación de los datos de las transformadas (en Hz).
* noise\_filter\_enabled: Activa o desactiva un filtro para reducir el ruido en los datos de los encoders.

**Tópicos:**

* **Publica:**
  + /roomba/odom/odom: Publica datos de odometría, incluyendo posición (x, y, theta) y velocidades (lineal, angular).
  + /roomba/odom/tf: Publica transformaciones TF entre odom\_frame\_id y base\_frame\_id.
  + /roomba/odom/state: Publica el estado de la odometria.
* **Suscribe:**
  + /roomba/communication/movement: Datos de los encoders del robot, publicados por el nodo de comunicación.

**Servicios:**

* /roomba/odom/reset: Resetea la posición y orientación del robot a (0, 0, 0).
  + **Entrada:** Ninguna.
  + **Salida:**
    - success: Booleano que indica si el reseteo fue exitoso.
    - message: Mensaje adicional en caso de error.
* /roomba/system/reconfigure: Permite ajustar parámetros relacionados con la odometría dinámicamente.

#### **Entrada:**

* node: "odometry\_node".
* parameter: Nombre del parámetro a modificar. Parámetros reconfigurables:
  + odom\_frame\_id: Identificador del marco de referencia de odometría.
  + base\_frame\_id: Identificador del marco base del robot.
  + wheel\_base: Distancia entre las ruedas del robot (m).
  + wheel\_radius: Radio de las ruedas (m).
  + publish\_rate: Frecuencia de publicación de los datos de odometría (Hz).
  + noise\_filter\_enabled: Activa o desactiva el filtro de ruido en los encoders.

#### **Salida:**

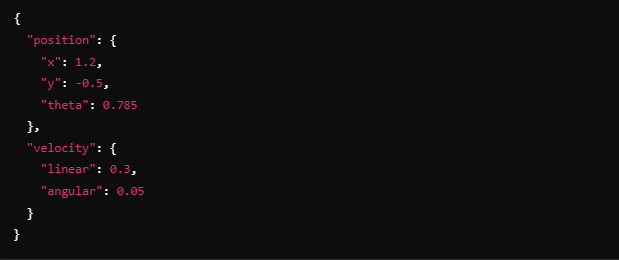
* success: Booleano que indica si el cambio fue exitoso.
* message: Mensaje adicional en caso de error.

#### **Interacción**

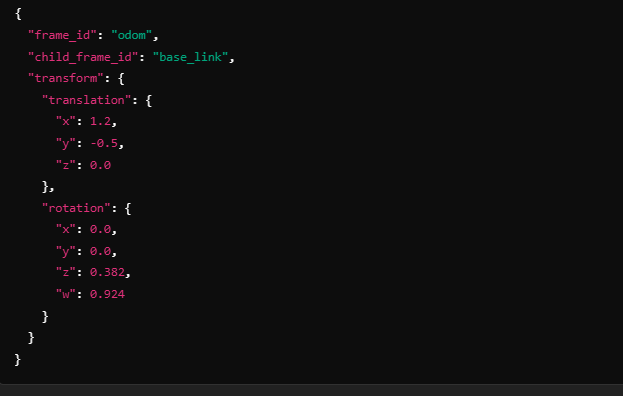
* **Con otros nodos:**
  + **Nodo Maestro:**
    - Proporciona datos de odometría para el estado global del robot.
    - Recibe configuraciones dinámicas de parámetros.
  + **Nodo de Movimiento:**
    - Recibe datos de movimiento para calcular odometría.
  + **Nodo de Navegación:**
    - Proporciona transformaciones y datos de posición y orientación para la planificación de rutas.
  + **Nodo de Seguridad:**
    - Recibe información sobre el estado del robot (por ejemplo, alertas de inactividad).
* **Con el hardware del robot:**
  + Procesa datos de encoders recibidos desde el nodo de comunicación.

#### **Ejemplo de Datos Publicados**

**Tópico /roomba/odom:**



Transformaciones Publicadas en /tf:



#### **Consideraciones Técnicas:**

1. **Calibración de Parámetros:** Asegúrate de configurar correctamente wheel\_base, wheel\_radius y ticks\_per\_revolution para obtener una odometría precisa.
2. **Filtro de Ruido:** Si noise\_filter\_enabled está activado, considera ajustar el filtro para reducir errores en datos provenientes de encoders.
3. **Frecuencia de Publicación:** Configura publish\_rate según las necesidades del sistema de navegación, equilibrando precisión y uso de recursos.
4. **Reseteo de Odometría:** Utiliza /roomba/odometry/reset para reiniciar la posición y orientación del robot en situaciones específicas, como cambios de contexto.

Siguientes nodos a desarrollar::

* **Nodo de Navegación**:
* **Nodo de Sensores Avanzados**: Para sensores adicionales, como cámaras o LiDAR,
* **Nodo de Interfaz Web/Usuario**: Para controlar y visualizar el estado del robot a través de una interfaz gráfica o web.
* **Nodo de Planificación de Tareas**:Nodo que actúa como planificador de tareas.