Maestría en Tecnología Informática y de Comunicaciones

Cohorte 80 (2018-2020)

“USO INTELIGENTE DE SALAS DE REUNIÓN CON DETECCIÓN DE MOVIMIENTO”

Autor: Santiago Javier Alvarez

Director del Trabajo Final: *Mg Diego Otero*

Institución a la que pertenece: UADE UBS

Fecha de entrega: *28/11/2019*

ABSTRACT

The management of conference room and meeting points is a recurrent issue in large office spaces or commercial buildings. Availability is generally low, especially when organizing spontaneous meetings with colleagues or customers. There are numerous solutions aimed at the optimization of the static scheduling system, and this investigation seeks to refine some of the existing ideas that use passive infrared devices for detection movement while at the same time simplifying the solution enough to avoid expensive implementation costs.

This proposal integrates simple PIR devices that have IP presence and are powered over Ethernet with an Outlook booking system and a custom web application for user interface. It's presented as an unobtrusive alternative solution to the existing booking system, allowing it to thrive even without mass adoption and providing statistical reports to measure its effectiveness.

Keywords— conference room scheduling; office space management; impromptu meetings; space optimization; smart rooms;

ÍNDICE

1. INTRODUCCIÓN

Un problema recurrente en grandes oficinas o edificios de uso comercial es la administración de espacios comunes y salas de reunión. El problema tiene múltiples factores, como la reserva de forma repetitiva de espacios comunes, las cancelaciones informales que no liberan el recurso de la sala, la sobredimensión del tiempo de ocupación o simplemente el olvido. Estos problemas suelen abordarse por medio de políticas de etiqueta en el uso de espacios comunes; sin embargo, la implementación de procesos rígidos afecta negativamente la flexibilidad y las nuevas formas de trabajo (MARGINALIA, 2018).

Con la aparición y masificación de tecnologías relacionadas con Internet of Things (IoT) se realizaron variados estudios y sistemas para el mejor aprovechamiento de los espacios comunes y calidad de vida en general, pero los resultados pueden ser muy costosos y complejos de implementar: utilizar tecnologías experimentales y potencialmente vulnerables en términos de seguridad; o que no permiten medir explícitamente el éxito de su implementación. Además, la resolución de este problema debe ser suficientemente amigable como para que la adopción sea masiva y prolongada en el tiempo: caso contrario, habrá sido una pérdida de tiempo y dinero. (MADRID, Ignacio. 2018) (BUILDINGS, 2017)

La detección de movimiento provee un valor binario, confiable y de manejo sencillo. Combinado con un segundo valor binario de estado de reserva de la sala se obtiene un estado de cuatro valores posibles (ocupado y reservado, reservado sin ocupar, disponible y ocupado o disponible sin ocupar) que puestos a disposición en tiempo real para el usuario brindan información valiosa a la hora de organizar una reunión espontánea. La persistencia de estos estados en el tiempo permite también la generación de estadísticas de uso y aprovechamiento de las salas monitoreadas, que sumados a los reportes de uso de la interfaz web permitirán determinar el impacto de la adopción del sistema propuesto.

La posibilidad de incorporar distintos sensores IoT a los mismos dispositivos propone opciones de escalabilidad para el desarrollo de salas inteligentes utilizando la misma infraestructura.

1.1. Descripción del problema

En ciertos ambientes laborales puede ser frecuente o por lo menos recurrente la necesidad de realizar una reunión espontánea en un lugar que permita la comodidad de dos o más personas para discutir sin molestar o interferir con el trabajo de otros o ser interrumpidos. Especialmente

cuando es un grupo de más de dos personas suele ser necesario también proyectar algún tipo de material digital en una pantalla o realizar una teleconferencia con audio o video. Incluso en los diseños de oficina más modernos de espacios abiertos con puestos de trabajo móviles y sectores de encuentro espontáneo aún se requiere extensivamente el uso de salas de reunión tradicionales, con soporte para proyección y tele/video conferencias. (SCHWAB, 2019)

El uso y reserva de salas de conferencia suele gestionarse a través de calendarios con las salas individualizadas, prevención de conflicto y orden de llegada. Mientras más solicitados sean los recursos será necesario mayor antelación en la reserva, y esto provoca inevitablemente cancelaciones, modificaciones y alteraciones de los tiempos reservados.

Incluso desarrollando y logrando practicar adecuadamente un reglamento, conducta o etiqueta en el sistema de reservas, el mismo uso adecuado generará espacios de reservas durante los cuales la sala no será ocupada, ya sea porque la reunión se canceló y no se canceló la reserva (sea justificada dicha acción o no) o porque simplemente se la reservó por más tiempo del que se la necesitó. Aquí sí podemos justificar que es preferible que el tiempo de reserva sobre y no que deba ser interrumpida una reunión importante simplemente porque fue demasiado ajustada la previsión del tiempo requerido.

En la práctica, lo descripto ocasiona que ante la necesidad de realizar una reunión espontánea con todas las salas reservadas para ese momento el usuario logre encontrar algún lugar que se encuentre vacío y que no vaya a ser utilizado hasta el comienzo de la próxima reserva. Dependiendo del diseño de la oficina, esta búsqueda puede ser sencilla si uno se encuentra en una única planta de espacio abierto con salas vidriadas donde su real ocupación resulta evidente a simple vista, o prácticamente imposible si el diseño no es abierto, las puertas y paredes son opacas o el diseño de piso es compartimentado o en varios niveles. En cualquiera de los dos extremos o sus situaciones intermedias, resulta trabajoso y difícil de coordinar con el resto de los participantes.

1.2. Objetivos

Desarrollar un sistema que permita extender y optimizar el proceso de reserva de salas de reunión en oficinas o espacios de trabajo y maximizar la productividad de los activos de la empresa o establecimiento que lo adopte.

1.2.1. Objetivos Particulares

Determinar la eficiencia del uso de las salas monitoreadas mediante el sistema implementado.

Identificar la oportunidad de mejora en la eficiencia con el análisis de los datos reunidos.

1.2.2. Alcance

Monitorear la actividad de reservas y movimiento de al menos dos salas distintas en las oficinas de Munro, Buenos Aires. Obtener información en esta locación durante un mínimo de tres meses sin interrupción.

No forma parte de este trabajo la adopción masiva del sistema por parte de los usuarios, por lo que la posibilidad de mejora de eficiencia es especulativa en base a la información disponible.

1.3. Glosario

IoT: Internet of Things (Internet de las cosas). Son sistemas de dispositivos, máquinas digitales o mecánicas con identificadores unívocos que pueden transmitir datos a través de una red autónomamente.

Open Office (Layout) Oficina Abierta: Diseño de planta de oficinas en donde los trabajadores y el equipamiento se encuentras mayormente agrupadas en un gran espacio sin paredes o divisiones. Son generalmente acompañadas por una política de escritorios limpios (clean desk) y sin lugares fijos asignados.

PoE: Power over Ethernet (Alimentación a través de Ethernet). Estándar por el cual la energía eléctrica necesaria para alimentar un dispositivo o sistema ad-hoc es provista a través del par trenzado de cobre o Ethernet

Sensores PIR: Passive Infrared Sensor (Sensor Infrarrojo Pasivo): Sensor eléctrico que mide la luz infrarroja irradiada por objetos en su campo de visión, generalmente utilizado como detectores de movimiento en sistemas de alarmas o en automatización de luminarias.

1.4. Marco metodológico

La definición del problema parte de la experiencia personal en espacios de oficina. A través de entrevistas informales y el resultado de un grupo de enfoque se consolidaron los objetivos y el alcance de una solución. Mediante investigación de documentación académica se estudiaron implementaciones de distintos sistemas orientados a la resolución de la misma problemática. El enfoque es por lo tanto cualitativo. La ausencia de datos previos sobre la utilización de las salas es justamente uno de los puntos a resolver.

El diseño del trabajo es descriptivo, estableciendo la tecnología, infraestructura y software a utilizar para la implementación del sistema. Aplicando un producto mínimo viable, se recolectaron datos de uso real de dos salas durante tres meses.

2. MARCO CONCEPTUAL Y ESTADO DEL ARTE

La disponibilidad y administración de espacios de reunión es un tema de interés en grandes oficinas, especialmente debido a que cada vez más empresas adoptan el concepto de open office: según un estudio del Internacional Facility Management Association, en el 2010 el 68% de las personas entrevistadas había trabajado en alguna oficina sin paredes o cubículos. (SCHWAB, 2019)

Las salas de reuniones ya no se utilizan para situaciones excepcionales y planificadas con antelación, sino que:

- se vuelven necesarias para la privacidad a la hora de tener conversaciones, como negociaciones de términos contractuales de un servicio, el precio de un producto o detalles de contrataciones laborales;

- son indispensables para la colaboración, trabajo en equipo y transferencia de conocimiento, especialmente para compartir una proyección o discutir abiertamente sin interferir con el trabajo de los demás en un espacio común;

- representan un recurso costoso, ya que cada metro cuadrado destinado a salas de reunión es un lugar no disponible para un empleado de tiempo completo; y

- el uso cotidiano de estos espacios dista de ser óptimo, resultando incluso en uso ineficiente de la energía

2.1.Salas inteligentes

El problema de la baja optimización de espacios de reunión, incluso cuando no representa aún dificultad directa en su disponibilidad, es al menos planteado a la hora de diseñar espacios de trabajo de forma eficiente. Por lo tanto, existen diversas aplicaciones de la tecnología enfocadas a resolver o mejorar el uso de espacios comunes, algunas de las cuales se exponen en este capítulo.

2.1.1. Administradores de reservas

Variadas propuestas comerciales apuntan a esta problemática desarrollando soluciones orientadas principalmente a la simplificación de los sistemas existentes por medio de visualizaciones enfocadas en la experiencia de usuario. Conocidas como Sistema de Reservas de Salas de Reunión (Meeting Room Booking Systems) pueden reemplazar o extender la funcionalidad del sistema de

oficina existente por medio de dispositivos tipo tablet distribuidos en los accesos a los pisos y a las salas permitiendo una visualización geográfica de la distribución y la ocupación en tiempo real de todas las salas cercanas. Generalmente aportan una opción amigable para reservas espontáneas, cancelaciones rápidas para liberar un espacio, solicitud de confirmación al iniciar cada reunión (check-in), análisis de uso y/o filtros de búsqueda rápida.

En su mayoría estos son productos cerrados que requieren un nivel de personalización bajo a nivel software como la carga y diseño de planos de distribución, pero una complejidad mayor en la distribución e instalación de dispositivos. La mayoría incluye una pantalla por cada sala y algunas en puntos estratégicos como las entradas de cada piso y los lugares de mayor tránsito (TRUSTRADIUS. 2019)

Un subgrupo considerable entre los administradores de reservas son los algoritmos de priorización y asignación de salas de operación en el ámbito quirúrgico. El caso es más de nicho, pues la disponibilidad de los quirófanos es mucho más limitada debido al equipamiento necesario para provisionarla. Asimismo, los tiempos de ocupación son sumamente variables, sujetos a complicaciones inesperadas con probabilidades de riesgo de vida y merecen un estudio completamente aparte (DURÁN, Guillermo. 2017).

Debido a la naturaleza estocástica del problema (los tiempos de ocupación tienen una distribución de probabilidad aleatoria que puede ser analizada estadísticamente pero difícilmente predecible) éste recibe considerable atención en la literatura científica. Los trabajos están principalmente orientados al desarrollo de algoritmos y modelos adaptativos para la reserva, modificación y gestión de los tiempos de uso teniendo en cuenta prioridades de los pacientes, complejidad de la intervención y el riesgo de vida asociado (XIAO, Guanlian. 2018).

2.1.2. Minutas automáticas

Otro enfoque de aplicación de tecnología para resolver el problema de los tiempos de ocupación de las salas es trabajar sobre la optimización de los tiempos de reunión a través del desarrollo de salas inteligentes que permitan registrar y analizar el contenido de las discusiones y decisiones para generar minutas automáticas.

La premisa sobre la que se basan estas propuestas es que la efectividad de las reuniones de trabajo es disminuida por la calidad de las notas tomadas manualmente durante la discusión, por ser generalmente incompletas, imprecisas o subjetivas (BELL, Gordon. 2007).

En una investigación del estado del arte de salas inteligentes se relevaron numerosos trabajos orientados a desarrollos de sistemas de reuniones basadas en distintas tecnologías, que van desde la captura física y análisis estructural hasta el procesamiento semántico del diálogo. Dichas técnicas incluyen los campos de procesamiento de voz e imagen, interacción hombre-máquina, visión computacional y computación ubicua (integración de la informática en el entorno de la persona). Este tipo de herramientas aún no han sido expandidas masivamente, no sólo por su costo sino

porque aún quedan problemas sin resolver como el bajo rendimiento del reconocimiento de voz, falta de infraestructura y características limitadas. Es un campo de investigación abierto y existen distintas líneas de investigación orientadas a solucionar estas trabas (YU, Zhiwen. 2010).

Un trabajo más reciente profundiza sobre una de estas propuestas en la especificación de un producto de Sala Virtual o V-ROOM con toma de minutas automáticas, fundamentando con una encuesta que la captura de minutas es la mayor demanda en las características de salas inteligentes o virtuales en el ambiente corporativo. Esta investigación avanza sobre la captura de voz y análisis semántico y contextual de palabras clave, buscando extraer sentido y resumen conceptual de la conversación capturada. (JAMES, Anne. 2016)

2.1.3. Ocupación real por sensores de movimiento

Este tipo de acercamiento al problema de la ocupación subóptima de salas debido a la variabilidad e imprevisibilidad de las duraciones de las reuniones en el ambiente laboral también fue abordado por una investigación reciente de la Universidad de Vietnam. En esta publicación se presenta un modelo de sistema basado en detectores de movimiento del tipo pasivo infrarrojo, haciendo una comparación con otras propuestas en cuanto a costo y complejidad. Los dispositivos se interconectan por la red Ethernet, existentes en virtualmente cualquier espacio de oficina, y los datos se recolectan en un servidor web local.

La propuesta incluye el desarrollo de un sistema de reservas a medida que incorpore los datos recolectados por los dispositivos, así como un sistema de decisión en las reservas y políticas de uso de las salas. Esta propuesta requiere la adopción masiva del sistema expuesto y no contempla la integración con políticas preestablecidas o software de reserva actual. (TRAN, Linh. 2016)

2.2.Sensores de movimiento

En el marco de esta propuesta, la tecnología de detección de movimiento ofrece distintos medios. A continuación, se describen y comparan brevemente las tecnologías aplicadas más utilizadas.

2.2.1.1.Ultrasónicos

Estos sensores entran en la categoría de “activos”. Un transductor emite una onda sonora a frecuencias mayores a las audibles por el oído humano (ultrasónicas) y recibe el reflejo en los objetos cercanos. Las ondas son recibidas en fase constante cuando no hay movimiento en el área; cuando algo se mueve, la señal transmitida cambia de fase antes de ser recibida y el comparador de fase incluido en el dispositivo detecta el diferencial e inicia la condición de alarma.

Las ondas ultrasónicas pueden atravesar ciertas paredes, pudiendo resultar en falsos positivos. Partículas de polvo pueden acumularse en el transductor y causar vibraciones que produzcan

sonidos agudos audibles, o que sin llegar a serlo provoquen presión en los tímpanos llegando a ser molesta en exposiciones prolongadas (Filippini, Steve. 2015)

En la Figura 2.2.1 se observa el esquema de funcionamiento de los sensores ultrasónicos, indicando la interacción de ondas entre el Emisor/Receptor y el objeto ubicado a una distancia *r* menor o igual al rango de alcance del dispositivo (PAPOUTSIDAKIS, Michail. 2016).

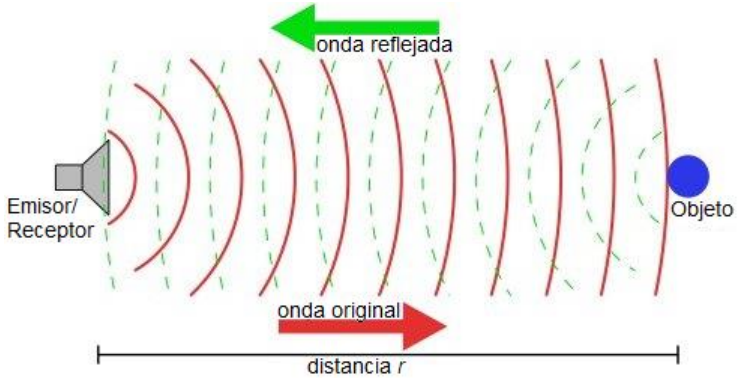

Figura 2.2.1 - Sensores Ultrasónicos.

2.2.2. Microondas

Estos sensores detectan movimiento a través del principio de radar Doppler, como el que se encuentra en los radares de velocidad de automóviles en la ruta. Una ola de microondas continua es emitida y la fase cambia al ser reflejadas en objetos en movimiento. Muy similar en principio al descripto previamente, pero utilizando otro tipo de onda y frecuencia. También entra en la categoría de sensores activos porque tiene un emisor de ondas.

En la figura 2.2.2 se aprecia el comportamiento del sensor y los dispositivos de amplificación y procesamiento digital (ITEAD. 2019).

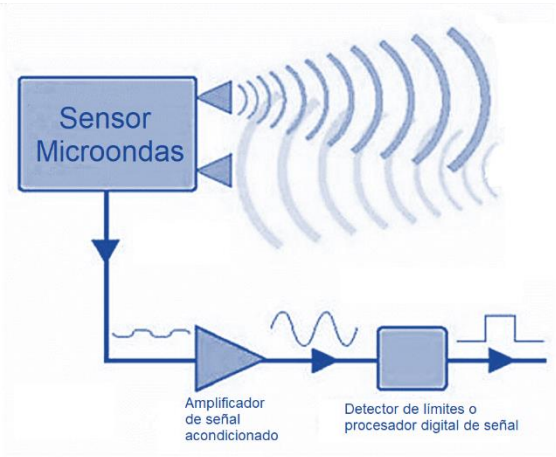

Figura 2.2.2 – Sensor microondas.

2.2.3. Procesamiento de imágenes

Esta solución aparece con la proliferación de cámaras digitales y la disminución en sus costos, y consiste en la captura constante de imágenes y el procesamiento de estas a través de software especializado.

Este tipo de solución resulta conveniente si se desea comenzar a filmar al detectar movimiento o si la captura de video forma parte del objetivo final, pero no tanto cuando el único propósito del dispositivo es detectar movimiento. También puede representar la mejor opción cuando la prioridad es evitar falsos positivos, ya que el software de detección puede optimizarse y adaptarse al tipo de movimiento o variación que no desea detectarse.

Requiere almacenamiento y procesamiento digital externo al dispositivo de captura, como computadoras o servidores conectados. (YONG, Ching. 2011)

Figura 2.2.3 – Procesamiento de imágenes.

2.2.4. Infrarrojo Pasivo

La tecnología de infrarrojo pasivo (PIR) es la más conocida y utilizada en detección de movimiento, aunque también la que mayores falsos positivos provoca.

Todos los objetos, incluidas personas y animales, emiten energía en forma de calor; esta energía no está dentro del espectro visible para el ojo humano, pero puede ser detectada por sensores especializados; por eso es infrarroja, que significa que tiene una frecuencia menor a la del color rojo que se encuentra en el límite del espectro. El nombre de 'pasivos' se da porque estos detectores no emiten ningún tipo de señal o energía, sino que monitorea un rango energético específico dentro del rango de 'visión' del detector. Dicho campo es generalmente ampliado por un lente Fresnel que sectoriza el campo de detección y permite activar el evento de movimiento al detectar que el diferencial de energía pasa de un sector a otro.

En el esquema de la figura 2.2.4 se grafica un corte horizontal del rango de cobertura de un sensor PIR con lente Fresnel, especificando el alcance de cada sector según su orientación. (MUKHOPADHYAY, Bodhibrata. 2018)

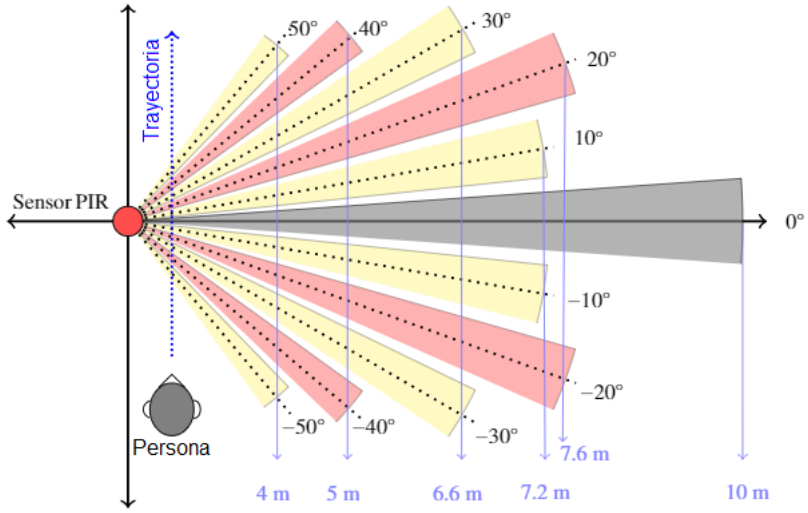

Figura 2.2.4 – Campo de visión del sensor PIR con lente Fresnel.

2.3. Dispositivos IoT

Internet de las cosas (Internet of Things – IoT) como tecnología comenzó a desarrollarse en 1999 por el sistema de identificación de frecuencia de radio (Radio-Frequency Identification – RFID) propuesto por el Auto-ID Labs del Massachussetts Institute of Technology (MIT). Con la aplicación y desarrollo de nuevas tecnologías informáticas, la connotación y extensión del término Internet de las Cosas pasó por grandes cambios. Actualmente IoT puede ser definido como una enorme red basada en internet que conecta "cosas" tanto físicas como virtuales con protocolos de comunicación de interoperabilidad estándar. Mas específicamente, todo objeto como ser un sensor o dispositivo que posea una identificación y atributos únicos, comunicándose e intercambiando mensajes para generar información como posicionamiento inteligente, rastreo, identificación o monitoreo. IoT se ha introducido en una variedad de actividades humanas, desde el hogar inteligente hasta la asistencia médica o el control industrial. (SHI, Xiaojie. 2019)

Sobre este último aspecto, Internet de las Cosas fue uno de los habilitadores de una importante transformación en la fabricación de productos a través de la digitalización de la industria. Esta transición se la está llamando "Industry 4.0" por considerarse la cuarta revolución industrial. Desde la primera (la mecanización a través de maquinarias a vapor) hasta la producción masiva y líneas de ensamblaje eléctricas, esta nueva revolución continúa con lo que comenzó la adopción de computadoras y automatización de procesos para mejorarlos a través de la constante recolección de datos y el aprendizaje automático (machine learning). (MARR, Bernard. 2018)

3. DESARROLLO

La propuesta de solución al problema de ocupación subóptima de las salas propone la utilización de tecnología probada y accesible para brindar información sobre el uso real de las salas y facilitar el aprovechamiento de espacios comunes para reuniones espontáneas.

Como información adicional al estado de ocupación de las salas en el sistema de reservas, se presenta otro dato de valor binario en base a la presencia real de personas ocupando el espacio. En este caso fue seleccionada la tecnología de sensores infrarrojos pasivos (PIR) conectados a dispositivos tipo Internet de las Cosas (IoT) cuya presencia en la red permitan consultar en tiempo real la existencia de movimiento y su registro a lo largo del tiempo. A través de una interfaz de usuario se presenta en pantalla la información que de otro modo requeriría la inspección física, sumado a los datos útiles de cuándo se registró movimiento por última vez y cuánto tiempo queda hasta la próxima reserva. Con este conocimiento se puede proceder a ocupar una sala haciendo uso de la reserva existente, sin perjuicio de nadie y permitiendo un uso más eficiente del espacio

3.1. Hardware

3.1.1. Detectores de movimiento

Como se detalla en 2.2, la detección de movimiento (o más específicamente de presencia) es posible por medio de distintas tecnologías con distintos rangos de costo, confiabilidad, accesibilidad de la tecnología y complejidad de instalación. Entre ellas, la menos costosa y más ampliamente utilizada es la de infrarrojo pasivo. Si bien no es la más confiable en cuanto a la probabilidad de falsos positivos cuando el movimiento que se pretende capturar es la presencia humana, es suficientemente precisa para el presente objetivo. Adicionalmente, las detecciones de movimiento aisladas o irrelevantes pueden ajustarse luego por lógica de la aplicación.

Además del tipo de tecnología, continúa el proceso de selección del tipo de dispositivo PIR a adquirir. Se definen entonces los criterios utilizados y se evalúan posteriormente algunas de las opciones disponibles en el mercado.

3.1.2. Criterios de selección

- Accesibilidad de los datos. Se refiere al método o procedimiento de extracción de información registrada por el dispositivo. La existencia de interfaz programable de aplicación y la variedad de lenguajes para la que esté programada
- Sustentabilidad. La infraestructura necesaria para el mantenimiento y operabilidad del hardware. Se refiere a la conectividad requerida por cada dispositivo, la infraestructura

necesaria para su funcionamiento, la alimentación eléctrica y la conectividad con la red de datos.

- Complejidad de implementación. Es la medida de costo y tiempo en el que se puede tener los dispositivos funcionales e instalados desde el momento que salen de la caja (out of the box). Contempla el ensamble, la disposición física,
- Costo. Los criterios anteriores cubren las consideraciones de tiempos y costos de implementación. En este caso el costo comprende los precios de compra del hardware necesario
- Seguridad. Los aspectos de seguridad y adhesión a los estándares de seguridad de los dispositivos con presencia de red. Su posición en la jerarquía no implica que la seguridad sea un aspecto menor en el sistema, sino que la complejidad de los dispositivos IoT suele ser baja y se entiende que el esfuerzo de analizar las vulnerabilidades que puede presentar este tipo de dispositivos para lograr una eventual excepción de seguridad es menor que la de conseguir dispositivos que adhieran a todos los estándares preexistentes en cada ambiente.

3.1.3. Evaluación

En la figura 3.1.3 se realiza la evaluación de algunas de las opciones contempladas y su puntuación de acuerdo con los criterios señalados anteriormente, donde - - es la menor calificación y + + la mayor.

	Accesibilidad	*Sustentabilidad*	*Implementación*	*Costo*	*Seguridad*
Arduino	+ +	+	- -	-	+ +
Alarma	- -	- -	+	+	+
Brick WiFi	+ +	+ +	+ +	-	-
Brick PoE	+ +	+ +	+ +	-	+

Figura 3.1.3 – Tabla comparativa

Arduino es una compañía de desarrollo que se enfoca en acercar y facilitar el uso de la electrónica y programación de placas con sistemas integrados. En este caso, la implementación presenta su principal punto en contra por la complejidad y el costo en tiempo de diseño, prototipado y prueba del dispositivo. Parte del punto más básico y permite el mayor grado de adaptabilidad: esto le brinda la alta evaluación de accesibilidad, pero a costa de la complejidad en su implementación. Esta misma característica es la que permite adaptarlo a cualquier norma de seguridad,

especialmente debido a que en el caso de esta investigación la complejidad es baja: el influjo de datos es binario de un solo canal (movimiento detectado o no). Pero un factor que aparece en este caso es además el tamaño y la practicidad, porque si bien la lógica de los datos no es compleja de programar, otras cuestiones básicas toman el costo en tamaño e infraestructura necesaria como la alimentación eléctrica y la conectividad en la red.

Los sistemas de alarmas hogareñas presentan su ventaja en la masividad de su comercialización y su disponibilidad en cualquier mercado local. Esto lo sitúa en el único lugar donde el costo es un aspecto positivo. Por ser sistemas cerrados que proveen su propia infraestructura no presentan problemas significativos en seguridad o implementación. Pero esta misma particularidad les da la mayor desventaja en los criterios más importantes: no cuentan con interfaces programables y requieren su propia infraestructura.

En el punto opuesto se encuentran los Bricks, que por ser específicamente diseñados como dispositivos IoT con escalación incremental nos da la mayor ventaja en accesibilidad. Cada nivel de Brick tiene sus propias interfaces programables (APIs) en los lenguajes más utilizados, incluyendo Java, C#, .NET y Python. La alimentación por USB o Power Over Ethernet (PoE) habilita la conexión con el cableado de oficina y la opción WiFi brinda independencia en su ubicación y facilidad de instalación. Sin embargo, la opción WiFi aún depende de una alimentación, por lo que a menos que se contemple la adición de baterías y su correspondiente mantenimiento no nos ofrece completa independencia en su ubicación. Además, en ciertos ambientes la red WiFi interna puede estar limitada por hardware ya que presenta una vulnerabilidad mayor que el acceso por Ethernet.

En primera instancia la opción del Brick WiFi fue instalada las salas de la oficina, pero las normas de seguridad impedían que dispositivos no certificados por la compañía tengan acceso a la red inalámbrica de acceso a servidores internos. Este factor determina finalmente la elección de la opción cableada con PoE.

3.1.4. Especificaciones

3.1.4.1. Lente Fresnel

Como se explica anteriormente, este tipo de lente sectoriza el campo de detección y permite activar el evento de movimiento al detectar que el diferencial de energía pasa de un sector a otro. Las dimensiones del lente Fresnel que utiliza el dispositivo seleccionado se detallan en la figura 3.1.4.a

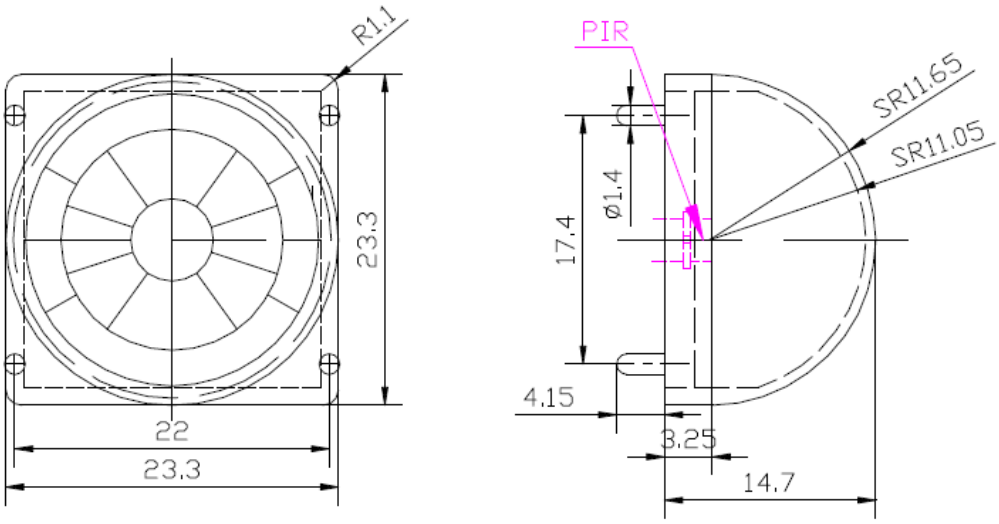

Figura 3.1.4.a – Dimensiones del lente.

En la figura 3.1.4.b se ilustra el rango de detección obtenido por estos lentes en metros (horizontal y vertical respectivamente):

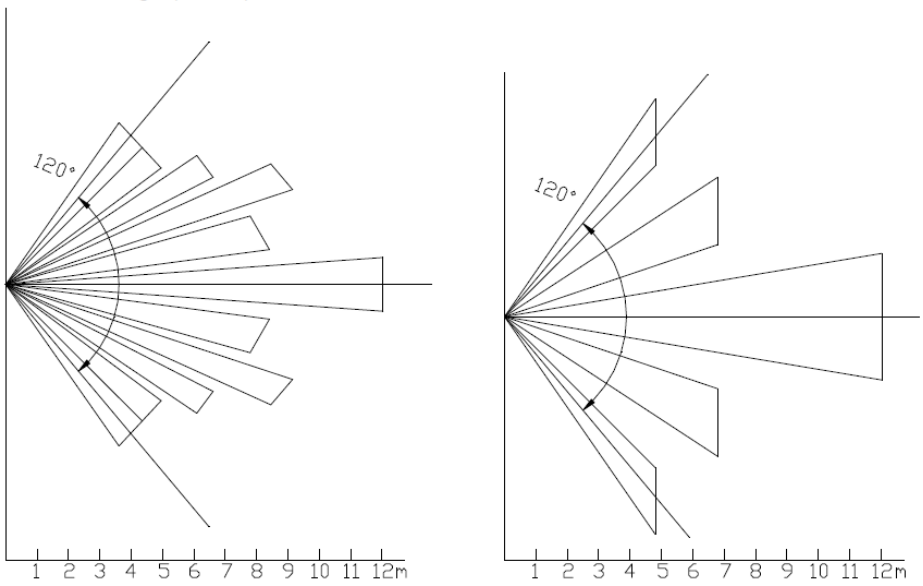

Figura 3.1.4.b – Rangos de detección.

3.1.4.2.Sensor Radial Infrarrojo

En el centro del lente se ubica el detector piroeléctrico AS612.

3.1.4.3.Motion Detector Bricklet

El sensor y el lente se montan en una placa madre con un software de base que permite el acceso a los datos capturados por el sensor. El Bricklet también cuenta con dos potenciómentros que regulan la sensibilidad del detector y el retardo. Por retardo se entiende el lapso entre el instante en que se detectó el último movimiento y que se interrumpe la señal enviada. Un LED ubicado en el dispositivo permite apreciar el ajuste de retardo prendiéndose al detectar un movimiento y apagándose transcurrido el tiempo estipulado.

3.1.4.4.Master Brick

El bricklet se conecta al master brick por medio de un cable y puerto propietarios, llamados bricklet ports. Cada master brick es la pieza principal del dispositivo al que se pueden conectar hasta cuatro bricklets por dichos puertos. Esto provee a cada unidad la escalabilidad para agregar nuevas funcionalidades con otros bricklets de detección de movimiento en caso de salas de gran tamaño, así como con otros bricklets de distintas funciones.

La funcionalidad principal del Master brick es proveer comunicación para el stack (conjunto de bricks, bricklets y extensiones), enrutando todos los mensajes entre las placas del dispositivo. Un mismo stack puede tener más de un master brick, pero el que se encuentre en el extremo inferior de la pila actuará como maestro; el resto sólo proveerá a los bricklets que tenga conectados directamente.

El puerto principal de comunicación y alimentación eléctrida de un master brick es un único USB. Para extender la conectividad y fuentes de alimentación se requieren extensiones

3.1.4.5.Ethernet Master Extension (with PoE)

Las extensiones tienen conexión el master brick a través de un puerto propietario y se conectan entre sí en forma apilable. El master y sus extensiones conforman una pila o ‘stack’.

El Ethernet extension with PoE provee de conectividad a red ethernet de 10/100 Mbit/s y alimentación eléctrica a todo el stack.

3.1.4.6.Stack

Los componentents mencionados conforman el stack que se observa en la figura 2.4.4.3. En la mitad superior se encuentran el Master Brick con su extensión Ethernet, conectados a la derecha por el cable UTP azul. Por la izquierda, un cable serial conectado al bricklet port se comunica con el Motion Detector Bricklet y el lente Fresnel en la mitad inferior.

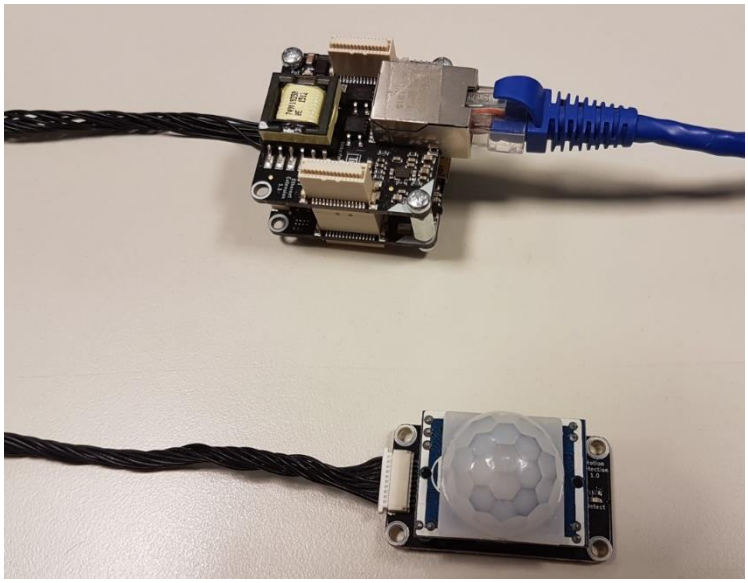

Figura 3.1.4.c – Máster, extensión PoE y bricklet.

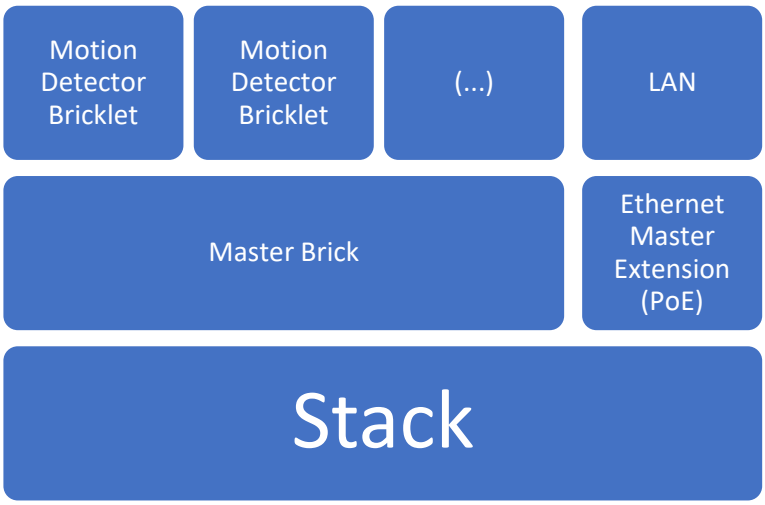

Figura 3.1.4.d – Esquema del stack.

3.2. Infraestructura y red

3.2.1. Dispositivos

Las especificaciones del sensor describen un alcance de hasta 12 metros en un barrido de 120°, por lo que en salas de 7m por 10m (con una diagonal de 12m) un único sensor es suficiente para cubrir toda su superficie.

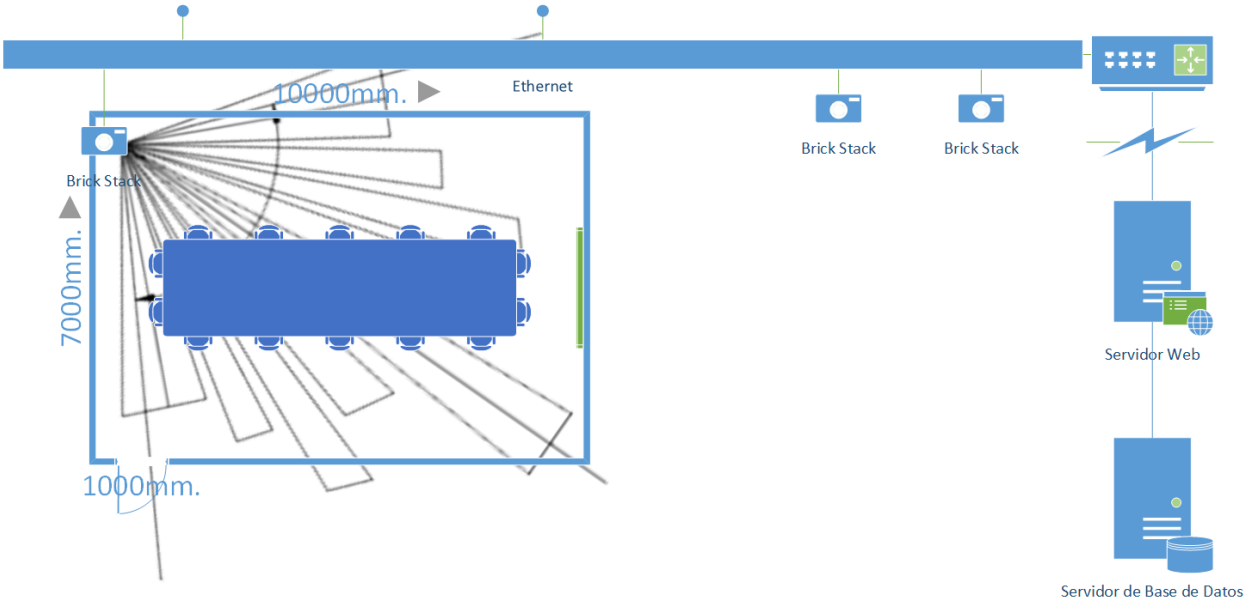

Figura 3.2.1.a – Plano de sala y diagrama de red.

El cableado estructurado del edificio tiene una troncal Ethernet sobre el techo de la sala, permitiendo la conexión ethernet y alimentación a través de PoE del Brick. Por medio de los paneles de mantenimiento se conecta el Brick a la red, y por una perforación en el techo se pasa el cable serial del Bricklet. Éste se coloca en la esquina de las salas orientándolo a lo largo de la diagonal para optimizar el alcance. En la figura 2.4.4.5 se observa el Bricklet conectado y en posición.

Figura 3.2.1.b – Bricklet con soporte.

El cableado Ethernet provee acceso directo a la red local (LAN) de la compañía. Por medio de la conexión USB del Master Brick y utilizando el software de interfaz del mismo (brickd) se le asigna una dirección IP estática dentro del rango de subred del sitio. Cada una de estas direcciones IP debe ser previamente reservada en el servicio de DHCP (Dynamic Host Configuration Protocol) de Active Directory para evitar conflictos con las asociaciones dinámicas y otras configuraciones

estáticas. Al mismo tiempo, permite la identificación unívoca de cada dispositivo sin necesidad de registrar nombres de host.

3.2.2. Web Server

La recolección de datos de los dispositivos, su administración e interfaz con el usuario y la herramienta de gestión de salas es a través de una aplicación web. La aplicación requiere de .NET Framework 4.5 en un servidor de Internet Information Service (IIS)

La infraestructura de web hosting predominante en la red donde se desarrolla el trabajo es de Microsoft IIS con .NET Framework en servidores centrales con instancias compartidas con distintas aplicaciones y servicios.

La granja de servidores compartidos de IIS se encuentra en los datacenter centrales conectado a través de la WAN en una única red con el mismo enrutamiento para todas las direcciones internas, lo que provee visibilidad entre el servidor web y los Stacks instalados en las salas.

El servicio se brinda sólo para uso interno y por defecto la visibilidad de la aplicación es sólo interna. Para la publicación en internet se utiliza un servicio de reverse proxy sitio por sitio pero que no se contempla en el alcance de este trabajo. Por esta razón y por el tipo de información no sensible que se almacena y transmite no es necesario la implementación del protocolo seguro y se mantiene en http plano.

3.2.3. Base de datos

La persistencia de los datos se realiza en un conjunto de tablas para acceso y almacenamiento en base de datos tipo SQL. La estructura de datos no presenta complejidad, por lo que es compatible con cualquier tipo de base.

La estructura predominante de bases de datos en este caso también es de Microsoft SQL Server, con disponibilidad de instancias compartidas en los datacenter centrales sin representar costo adicional.

La base de datos no tiene un alto nivel transaccional y consta de no más de cinco tablas y stored procedures.

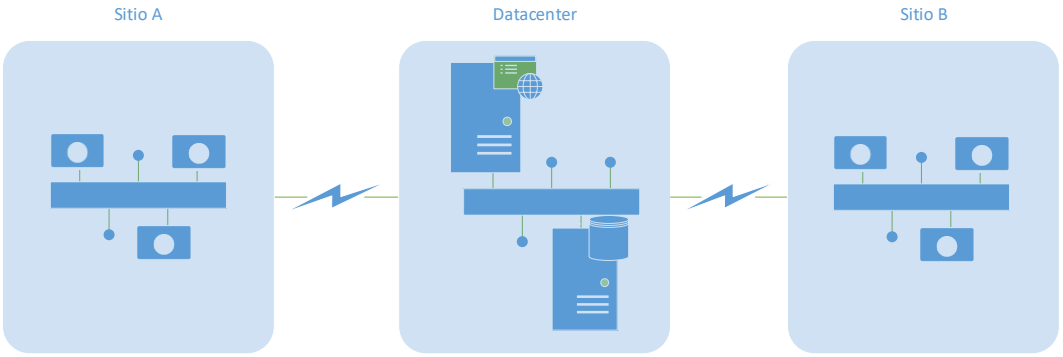

Figura 3.2.3 – Diagrama de red

3.3.Software

3.3.1. Interfaz de Usuario - Vista

Al acceder a la página principal, la aplicación muestra de forma esquemática todas las salas de reunión de la oficina del usuario, separada por pisos e indicando claramente el nombre y la capacidad de cada una. Para facilitar la implementación en distintas plantas las salas no están ordenadas o distribuidas de acuerdo con un plano de piso.

Cada sala tiene un código de coloración que indicará su estado actual y muestra la información relevante para cada situación.

- Libre – No reservada, sin movimiento

  La sala en color verde significa que no está reservada y se verifica que efectivamente no se registra movimiento.

  Se muestra el tiempo que falta hasta que comience la próxima reserva, obtenida de las reservas agendadas en Outlook.

  En este caso, el usuario simplemente reserva la sala Libre más conveniente y procede a ocuparla

- Ocupada – Reservada, con movimiento

  El color rojo indica que la sala está reservada en Outlook y que se registró movimiento en los últimos 5 minutos. Se consideran estos 5 minutos como prudenciales para mantener el estado de ocupado en los casos que los ocupantes no estén activando el sensor constantemente. Es una forma de aumentar por software el retardo ajustable manualmente del bricklet, que es de un máximo de 2 minutos.

  En este estado se muestra el tiempo restante hasta que la sala ya no esté reservada. Se tienen en cuenta todas las reservas consecutivas y se calcula el tiempo hasta la finalización de la última de la cadena.

- Disponible – Reservada, sin movimiento

  Es la situación que brinda nueva información y que representa el campo de acción de esta solución para resolver el problema planteado. En color azul, indica que, si bien la sala está reservada no se registra movimiento en los últimos 5 minutos. Este único dato no es suficiente para realizar una acción en consecuencia, por lo que se brindan los siguientes dos lapsos:

  - Inactiva por: la cantidad de tiempo en minutos en la que no se detectó movimiento *desde el inicio de la reserva actual*. Esto

- quiere decir que se calcula primero el tiempo transcurrido desde la última vez que se detectó movimiento y luego el tiempo desde que comenzó la reserva actual; se muestra *el menor* de estos dos valores.

- Próxima reserva en: la cantidad de tiempo desde el momento actual hasta que comience la próxima reserva.

Con estos dos datos adicionales, el usuario puede elegir entre las salas que se encuentren en este estado aquella que tenga el tiempo suficiente hasta la próxima reserva para sus necesidades y priorizar la que tenga mayor tiempo inactivo para asegurarse que realmente esté disponible. Una vez seleccionada la sala, sólo con ocuparla y comenzar a registrar movimiento el estado pasará automáticamente a Ocupada (color rojo) ya que seguirá reservada y registrará movimiento. Se aconseja en estos casos contactar a quien hizo la reserva.

- Tomada – No reservada, con movimiento

Cuando la sala no tiene una reserva en Outlook, pero se detecta movimiento, el color es amarillo y aporta información al momento de priorizar la selección de una sala. Si bien no tiene una reserva, es probable que exista una reunión espontánea y que no sea ideal para su uso.

Se brinda adicionalmente el tiempo transcurrido desde el último momento en el que el registro de movimiento fue falso. Indica por cuánto tiempo la sala estuvo ocupada o tomada, para distinguir entre movimientos aislados y ocupación real y sostenida.

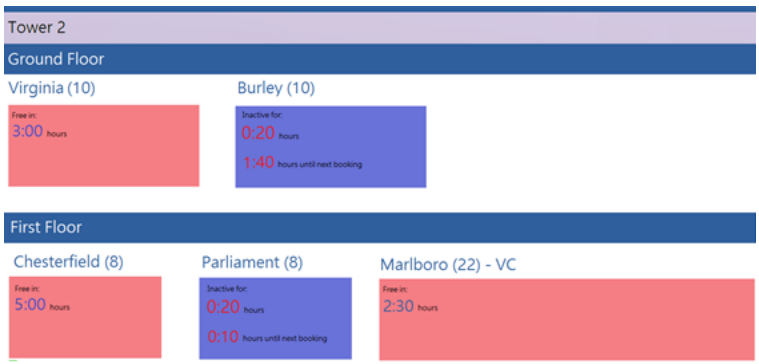

Figura 3.3.1– Esquema de interfaz de usuario.

3.3.2. Servicios

Los servicios de la aplicación se agrupan en las siguientes según su interfaz y se describen brevemente a continuación.

- ActiveDirectoryService. Realiza consultas de Windows Active Directory Services para obtener propiedades del usuario a través de System.DirectoryServices. La aplicación se

publica en una red interna, por lo que todo usuario fue previamente autenticado por Active Directory y el servicio puede consultar su nombre y ubicación. La ubicación se utiliza para determinar el edificio y las salas correspondientes que la aplicación muestra por defecto.

- ExchangeServerService. Consulta la disponibilidad de cada sala. Todas las reservas se realizan a través de Exchange y la información está disponible utilizando la librería Microsoft.Exchange.WebServices.Data. Todas las operaciones son de sólo lectura.
- MotionService. Establece la conexión con los Bricks y consulta la detección por movimiento por dispositivo.
- RoomService. Administra las salas, consultando y persistiendo sus propiedades

Se muestra en el diagrama de clases de la figura *3.3.2* el detalle de RoomService y el modelo asociado. Se describen algunas sus funciones.

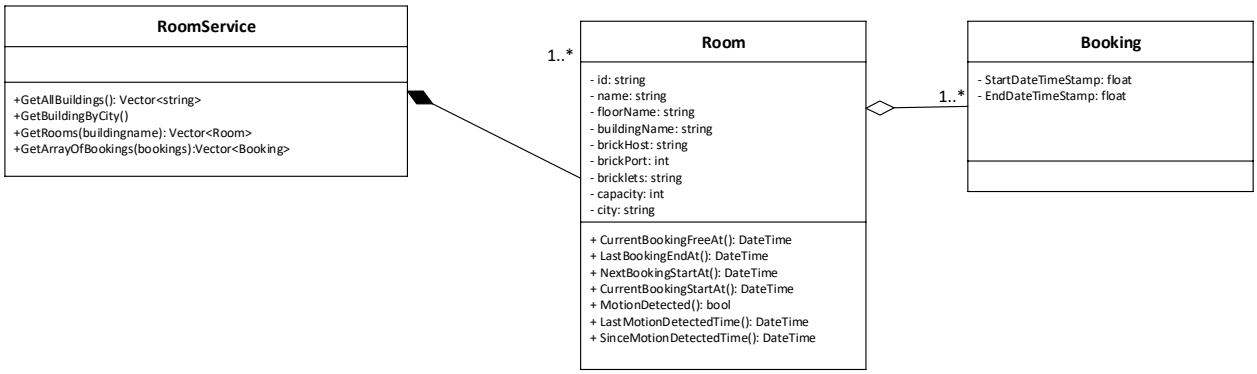

Figura 3.3.2 – Diagrama de clases.

- GetAllBuildings(): Devuelve todos los edificios registrados en la tabla de configuración de la base de datos
- GetBuildingByCity(city): A partir del nombre de la ciudad, que es una propiedad obtenida de los usuarios de active directory, devuelve el edifcio asociado a dicha ciudad según la relación establecida en la tabla de configuración.
- GetRooms(buildingname): A partir del nombre del edificio correspondiente a la selección de usuario la función devuelve la lista de salas correspondientes al edificio

correspondiente. Utiliza para ello un stored procedure de la base de datos para consulta de salas, asigna los atributos y la lista de reservas asociada a cada una.

- CurrentBookingFreeAt(): Devuelve fecha y hora de la finalización de la reserva en curso al momento de la consulta.
- LastBookingEndAt(): De las reservas consecutivas que tenga la sala, devuelve fecha y hora de finalización de la última de la serie.
- MotionDetected(): Devuelve verdadero si existe registro de movimiento en los últimos 5 minutos
- SinceMotionDetectedTime(): Devuelve el tiempo desde el que se comenzó a registrar movimiento continuo.

3.3.3. Persistencia

La persistencia se realiza en base de datos relacional en las tablas que se muestran en el diagrama relacional de la figura 3.3.3

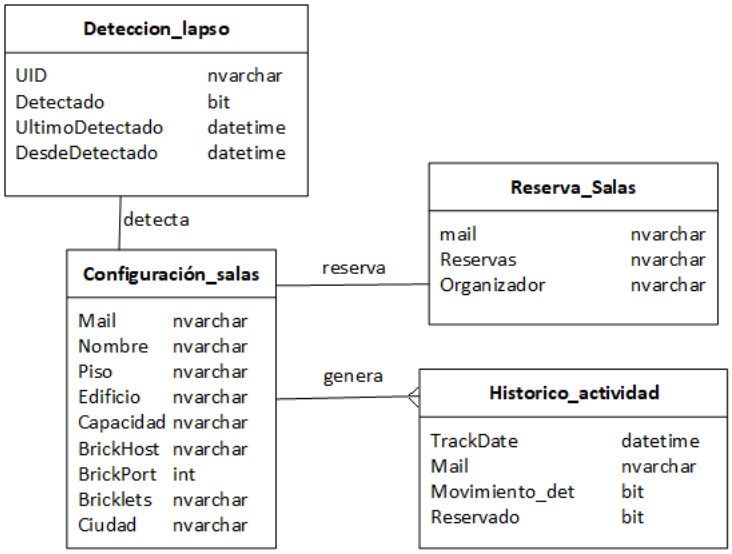

Figura 3.3.3 – Diagrama Relacional

- En la tabla de configuración se registran todas las salas que se utilizan en la herramienta. Se utiliza la dirección de correo de la sala como identificador primario de la tabla y para localizar el recurso en Exchange Web Services.

  Los campos de Nombre y Capacidad son los nombres amigables para mostrar en la interfaz de usuario en cada sala, por lo que son de texto libre. Lo mismo ocurre con Piso y Edificio, pero para valores iguales agrupa las salas en el piso o edificio correspondiente.

  BrickHost y BrickPort almacenan la dirección IP y el puerto del dispositivo instalado en la sala. Bricklets es el identificador o lista de identificadores en caso de ser una sala grande y

tenga más de un detector de movimiento conectado al bricklet (que permite hasta cuatro conexiones).

Ciudad también es de texto libre y permite valores nulos, pero no se muestra en la interfaz de usuario. Se utiliza para comparar el valor ingresado para la sala y compararlo con el atributo "city" de usuario de Active Directory y mostrarle por defecto el edificio asociado a esa ciudad.

Estos valores deben ingresarse manualmente con cada instalación de un nuevo brick.

En la Tabla 3.3.3 se muestra un ejemplo de un registro de configuración_sala.

Tabla 3.3.3 – Ejemplo de configuración

Mail	Nombre	Piso	Edificio	Capacidad	BrickHost	BrickPort	Bricklets	Ciudad
r-CRChesterfield@pmi.com	Chesterfield	1	PMLAS	8	10.202.34.61	4223	wsz	Munro

- Reserva_salas persiste la última consulta de próximas reservas, utilizando el mail como llave principal y utilizando pares de timestamps como inicio y fin de las próximas reuniones (inicio1,fin1;inicio2,fin2; ...). Organizador contiene la lista de organizadores de las reservas correspondientes.

- Deteccion_lapso permite consultar el último segmento de actividad de cada bricklet. El valor binario "detectado" indica si registró movimiento en la última consulta, y los timestamps de UltimoDetectado y DesdeDetectado indican la duración de la última detección. Con estos valores es posible calcular los lapsos mostrados en la interfaz de usuario para los distintos estados de sala: la cantidad de tiempo desde que registró el último movimiento (en el estado azul) o el lapso durante el cual hubo movimiento registrado (en el estado amarillo). También se usa esta tabla para extender por lógica el tiempo de retardo del dispositivo a 5 minutos para el estado rojo, como se describe en 3.3.1.

- Historico_Actividad es el registro minuto a minuto del estado de cada sala, indispensable para la generación de reportes. Trackdate es la marca de tiempo en la que se realiza cada registro, mail es el identificador de sala, y los bits de movimiento y reserva son los valores que determinan los cuatro estados posibles de la sala.

3.3.4. Interfaz de Bricklets

Cada componente del brick stack cuenta con sus propias interfaces de aplicación (APIs) en más de 15 lenguajes entre los que se encuentran C, C#, Go, Java, JS, Perl, Python, .NET y PHP. Se utiliza en este caso la interfaz para C# con las siguientes funciones básicas.

- class **`BrickletMotionDetector`**(String uid, IPConnection ipcon)

El constructor de la clase, que permite crear objetos para cada bricklet

- byte `BrickletMotionDetector.`**`GetMotionDetected`**()
  Devuelve el valor de byte con 1 si el sensor detecta movimiento en ese instante o si aún se encuentra en la ventada de retardo (ver 3.1.4). La función incluye además las siguientes constantes:
    - BrickletMotionDetector.MOTION_NOT_DETECTED = 0
    - BrickletMotionDetector.MOTION_DETECTED = 1

La interfaz posee otras funciones más específicas que no se utilizan en este trabajo pero que se mencionan a continuación.

void `BrickletMotionDetector.`**`SetStatusLEDConfig`**(byte config)
byte `BrickletMotionDetector.`**`GetStatusLEDConfig`**()
byte[] `BrickletMotionDetector.`**`GetAPIVersion`**()
bool `BrickletMotionDetector.`**`GetResponseExpected`**(byte functionId)
void `BrickletMotionDetector.`**`SetResponseExpected`**(byte functionId, bool responseExpected)
void `BrickletMotionDetector.`**`SetResponseExpectedAll`**(bool responseExpected)
void `BrickletMotionDetector.`**`GetIdentity`**(out string uid, out string connectedUid, out char position, out byte[] hardwareVersion, out byte[] firmwareVersion, out int deviceIdentifier)

3.3.5. Interfaz Exchange

Para la consulta de reservas existentes de salas en el servidor de Exchange, se utiliza la clase Microsoft.Exchange.WebServices.Data

- GetUserAvailabilityResults GetUserAvailability(IEnumerable<AttendeeInfo>, TimeWindow, AvailabilityData)
  Devuelve información detallada sobre la disponibilidad de un grupo de usuarios, salas y recursos durante el período determinado
- FreeBusyStatus (Propiedad)
  Devuelve el estado de disponibilidad del evento asociado
- FindItemsResults<Appointment> FindAppointments(FolderId, CalendarView)
  Obtiene la lista de citas buscando los contenidos de determinada carpeta y vista de calendario
- ServiceResponseCollection<ServiceResponse> LoadPropertiesForItems(IEnumerable<Item>, PropertySet)
  Carga las propiedades de un grupo de ítems en una sola llamada a Exchange Web Services

- ServiceResponseCollection<ServiceResponse> LoadPropertiesForItems(IEnumerable<Item>, PropertySet)

3.3.6. Interfaz Base de datos

Para la persistencia de los datos capturados y la consulta de actividad histórica se utiliza la clase SqlConnection de .NET Framework en System.Data. Se describen las operaciones básicas del manejo de la conexión y la creación de comandos.

- class **SqlConnection**(String connectionString)

  El constructor de la clase, que permite crear un objeto para la base de datos definida en el String de entrada

- void **Open**()

  Abre la conexión definida en el connectionString

- void **Dispose**()

  Cierra y dispone de la conexión luego de ejecutar un comando o atrapar una excepción

- SqlCommand **CreateCommand**()

Devuelve una instancia de la clase SqlCommand

4. DATOS Y ANÁLISIS

Se instalaron los dispositivos detectores de movimiento en la red local de una oficina situada en Munro, en un primer piso con dos salas de reunión y alrededor de 50 puestos de trabajo ocupados. El software y la base de datos se configuraron y alojaron en el datacenter de la compañía, y se procedió a capturar información de los dispositivos y ambas salas a modo de piloto, durante el transcurso del año 2017.

Los datos registrados en la tabla histórica de la base de datos se extrajeron para su análisis con la herramienta R Studio, con el procedimiento descripto en este capítulo.

4.1. Carga de Datos

De la base de datos se extraen todos los campos de la tabla Room_History_Status (Histórico_actividad) en formato de archivo CSV, y se utiliza directamente ese formato para importar los datos al dataframe dfroomhistory

```
dfroomhistory <- read.csv("C:/.../Room_History_Status.csv")
```

A fin de verificar los datos importados se muestra la estructura del dataframe

```
str(dfroomhistory)

## 'data.frame':    1850164 obs. of  4 variables:
##  $ ï..TrackDate  : Factor w/ 1413304 levels "2017-01-01 00:00:11.8000000",..: 1 2 3 4 5 6 7 8 9 10 ...
##  $ RoomMailBox   : Factor w/ 9 levels "CaptainBlack.CRPMICAToronto@pmi.com",..: 5 4 2 9 6 7 3 1 5 4
##  $ MotionDetected: int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Booked        : int  0 0 0 0 0 0 0 0 0 0 ...
```

4.2. Preparación de datos

Durante el período seleccionado también se capturó información de otras salas en distintas oficinas: dos en New York, cuatro en Lausanne y las dos en Buenos Aires mencionadas previamente. Estas últimas mantuvieron un funcionamiento prolongado con monitoreo frecuente, por lo que seleccionaremos sólo los valores r-CRChesterfield@pmi.com y r-CRParliament@pmi.com para este análisis. Se renombran además los encabezados del dataframe.

```
colnames (dfroomhistory) <- c("DiaHora","Sala","Movimiento","Reservado")

dfroomhistory<-dfroomhistory[dfroomhistory$Sala == 'r-CRChesterfield@pmi.com' | dfroomhistory$Sala
== 'r-CRParliament@pmi.com',]
dfroomhistory$Sala<-factor(dfroomhistory$Sala)
```

Por defecto, la conversión de CSV a dataframe asignó a las columnas *DiaHora* y *Sala* el tipo factor multinivel y a las columnas *Movimiento* y *Reservado* el tipo entero. Para optimización de los datos y manejo más sencillo se convierten los enteros a tipo lógico donde los únicos valores de 0 y 1 se convierten a Falso (FALSE) y Verdadero (TRUE) respectivamente.

En el caso de la fecha, se utiliza la librería lubridate para transformar el tipo de datos factor al tipo fecha POSIXct en formato *año-mes-día hora:minuto:segundo* y asignando las fechas originales a la zona horaria donde fueron capturados (CET - Central European Time). En un paso siguiente, la función with_tz permite adaptar las fecha y horas originales a la zona horaria de Buenos Aires (ART)

Para las salas, se reemplazan los nombres de formato correo a los valores más sencillos de SalaC y SalaP. A continuación, se muestra la nueva estructura con los cambios realizados.

```
library(lubridate)

dfroomhistory$Movimiento <- as.logical(dfroomhistory$Movimiento)
dfroomhistory$Reservado <- as.logical(dfroomhistory$Reservado)
dfroomhistory$DiaHora <- as.character(dfroomhistory$DiaHora)
dfroomhistory$DiaHora <- ymd_hms(dfroomhistory$DiaHora,tz="CET")
dfroomhistory$DiaHora <- with_tz(dfroomhistory$DiaHora,tzone="America/Argentina/Buenos_Aires")

levels(dfroomhistory$Sala) <- c("Sala C", "Sala P")

str(dfroomhistory)
```

```
'data.frame':    462540 obs. of  4 variables:
 $ DiaHora   : POSIXct, format: "2016-12-31 20:00:11" "2016-12-31 20:00:11" ...
 $ Sala      : Factor w/ 2 levels "Sala C","Sala P": 1 2 1 2 1 2 1 2 1 2 ...
 $ Movimiento: logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
 $ Reservado : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

4.3.Corrección y ajuste

El servicio web de la aplicación recogió minuto a minuto la información de las salas de Outlook y los valores de detección de movimiento de los dispositivos. Si el dispositivo no da respuesta o no se puede encontrar, el valor por defecto es siempre FALSE. Podemos entonces detectar el primer momento en que se registran valores TRUE cuando comienzan a funcionar los dispositivos infrarrojos.

Se verifica el funcionamiento de ambos detectores por separado. Tiempo de primera detección:

```
min(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala C"])
```

```
2017-01-02 06:58:13 -03
```

```
min(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala P"])
```

2017-01-02 06:01:13 -03

Ambos se registran el mismo día por la mañana. El tiempo de la última detección, sin embargo, revela que la Sala C registró movimiento durante doce meses mientras que la sala P sólo durante tres.

```
max(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala C"])
```

2018-01-13 00:17:14 -03

```
max(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala P"])
```

2017-03-18 06:56:03 -03

Estos valores nos permiten entonces separar el dataframe según la sala, y además acotar los datos a las fechas obtenidas para realizar el análisis sólo en los períodos en los que cada detector estuvo activo. Se utiliza en este caso la librería dplyr para filtrar y reagrupar los dataframes. Se remueve la columna de Sala luego de filtrar ya que se vuelve irrelevante. Se verifica luego la cantidad de registros en cada dataframe.

```
library(dplyr)

dfSalaC <- dfroomhistory %>%
        filter(Sala == "Sala C") %>%
          filter(DiaHora<"2018-01-13") %>%
            filter(DiaHora>"2017-01-02") %>%
              select (-Sala)
dfSalaP <- dfroomhistory %>%
        filter(Sala == "Sala P") %>%
          filter(DiaHora<"2017-03-18") %>%
            filter(DiaHora>"2017-01-02") %>%
              select (-Sala)
nrow(dfSalaC))
```

Sala C: 228372

```
nrow(dfSalaP))
```

Sala P: 95420

4.4.Análisis

Se analiza en primera instancia el porcentaje de ocupación real de cada sala. Nos interesa ver en un gráfico sencillo para cada minuto en que la sala estuvo reservada si se detectó o no algún

movimiento Utilizando nuevamente la librería dplyr generamos un nuevo dataframe para cada sala donde sólo se registran aquellos momentos en que la sala estuvo reservada.

```
soloReservadoC <- dfSalaC %>%
        select(-DiaHora) %>% #Remover DiaHora
        group_by(Movimiento,Reservado) %>% #Agrupar valores lógicos
        filter(Reservado) #Filtrar sólo reservado


soloReservadoP <- dfSalaP %>%
        select(-DiaHora) %>% #Remover DiaHora
        group_by(Movimiento,Reservado) %>% #Agrupar valores lógicos
        filter(Reservado)  #Filtrar sólo reservado
```

Con las librerías ggplot2 y gridExtra generamos dos gráficos de barra lado a lado para visualizar los datos obtenidos. Los dataframes difieren en su tamaño, razón por la cual se realizan gráficos separados puesto lado a lado. De esta forma se mantiene la comparación de las proporciones a pesar de la diferencia en la muestra.

```
library(ggplot2)
library(gridExtra)

plotC <- ggplot(soloReservadoC)+ geom_bar(aes(x=Reservado,fill=Movimiento))
plotP <- ggplot(soloReservadoP)+ geom_bar(aes(x=Reservado,fill=Movimiento))
grid.arrange(plotC,plotP,ncol=2)
```

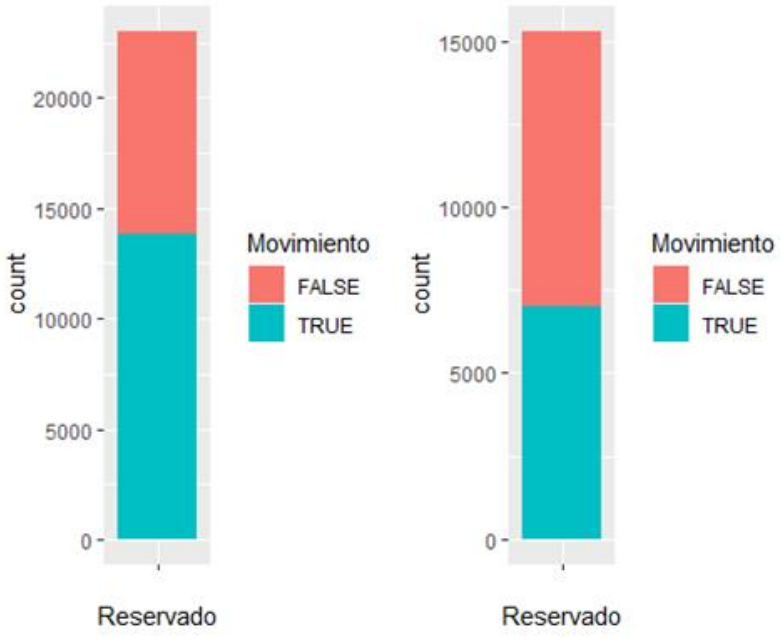

Figura 4.4.a – Gráfico de ocupación en las reservas.

Tomamos nuevamente la muestra completa, pero filtrando los horarios de oficina únicamente. Si bien hay actividad a partir de las 7 y hasta las 19, el horario central de 9 a 17 concentra la mayor cantidad de empleados en el sitio.

```
nine2fiveC <- dfSalaC %>%
              mutate(dsem = weekdays(DiaHora)) %>% #traigo el día de la semana
             filter(hour(DiaHora)>=9,hour(DiaHora)<=16,
                  dsem!='domingo',dsem!='sabado')

nine2fiveP <- dfSalaP %>%
              mutate(dsem = weekdays(DiaHora)) %>% #traigo el día de la semana
             filter(hour(DiaHora)>=9,hour(DiaHora)<=16,
                  dsem!='domingo',dsem!='sabado')
```

Volvemos a utilizar los gráficos lado a lado para comparar proporciones. Se observa en este caso que los períodos acumulados de sala reservada (TRUE) no parecen ser la ampliamente mayores (incluso en la sala C es considerablemente menor), pero se debe tener en cuenta que los mediodías y días feriados o de menor actividad se conservan en la muestra.

```
plotC2 <-ggplot(nine2fiveC)+ geom_bar(aes(x=Reservado,fill=Movimiento)) + labs(title="Sala C")
plotP2 <-ggplot(nine2fiveP)+ geom_bar(aes(x=Reservado,fill=Movimiento)) + labs(title= "Sala P")
grid.arrange(plotC2,plotP2,ncol=2)
```

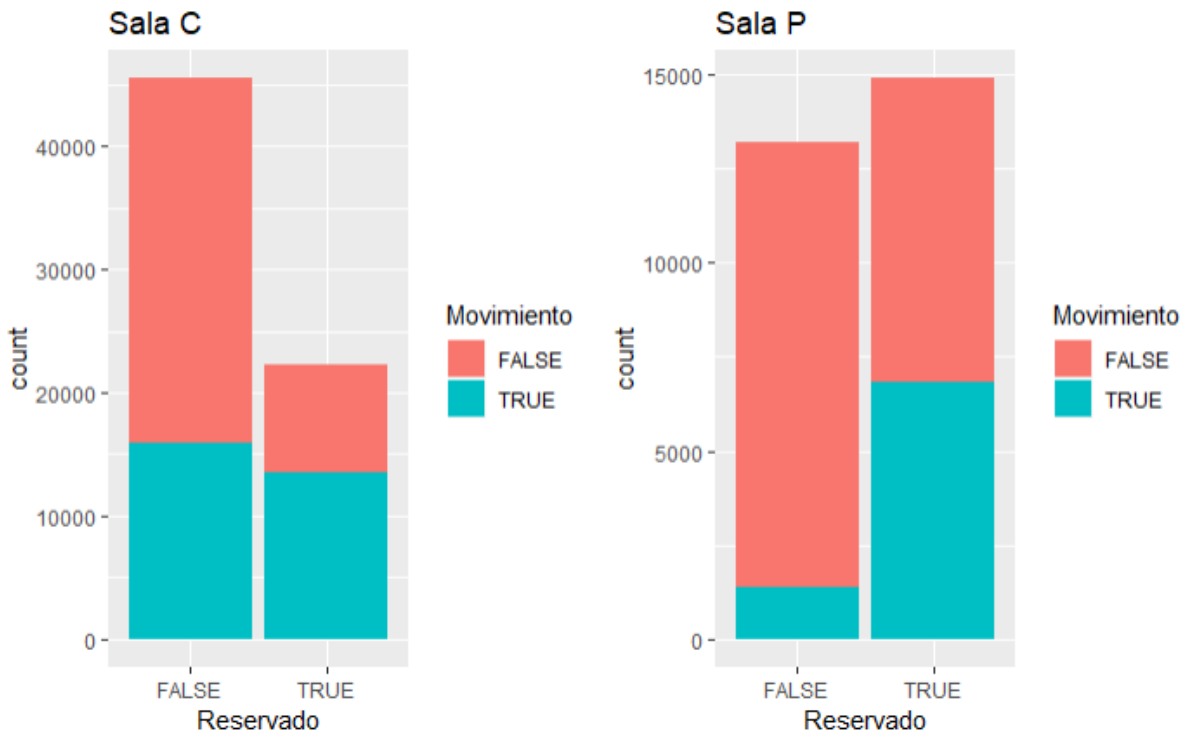

Figura 4.4.b – Gráfico general de salas.

El caso concreto de análisis son los segmentos de oportunidad en los cuales una sala presenta períodos aprovechables, en los que aunque esté reservada se encuentra vacía por tiempo suficiente como para ocuparse con una reunión espontánea. Dentro de los datos registrados, estos segmentos son aquellos registros en los cuales el estado para una misma sala es de Moviemiento=FALSE y Reservado=TRUE en forma continua. Para encontrarlos se utiliza la función rle (run length encoding - codificación de longitud de ejecución), que devuelve las longitudes y valores de valores equivalentes en un vector. Se convierten entonces los dataframes en vectores, filtrándolos por la combinación de valores mencionada.

Cada número en los resultados obtenidos representa entonces la cantidad de minutos continuos en los que la sala estuvo reservada sin registrar movimiento. Cabe aclarar que ante un sólo movimiento registrado por el dispositivo su valor de Movimiento=TRUE se mantiene por dos minutos, para evitar falsos negativos cuando los ocupantes se encuentran relativamente quietos. Sin embargo, se aprecia que se registra una gran cantidad de segmentos de 1 a 5 minutos precisamente por estos casos de relativa quietud. Otros segmentos de 5 a 15 minutos reflejarán las demoras en iniciar una reunión o aquellas que terminan poco antes de lo previsto.

Sala C. Total de segmentos, total de menores a 5, totales entre 10 y 15

```
segmentosC <- rle(with(nine2fiveC, !Movimiento & Reservado))
vectorC<-segmentosC$lengths[segmentosC$values]
length(vectorC[vectorC<5])
```

Menos de 5 minutos: 693

```
vectorC[vectorC>=5&vectorC<15])
```

De 5 a 15 minutos: 224

```
length(vectorC)
```

Más de 15 minutos: 1066

```
segmentosP <- rle(with(nine2fiveP, !Movimiento & Reservado))
vectorP<-segmentosP$lengths[segmentosP$values]
length(vectorP[vectorP<5])
```

Menos de 5 minutos: 319

```
length(vectorP[vectorP>=5&vectorP<15])
```

De 5 a 15 minutos: 175

```
length(vectorP)
```

Más de 15 minutos: 637

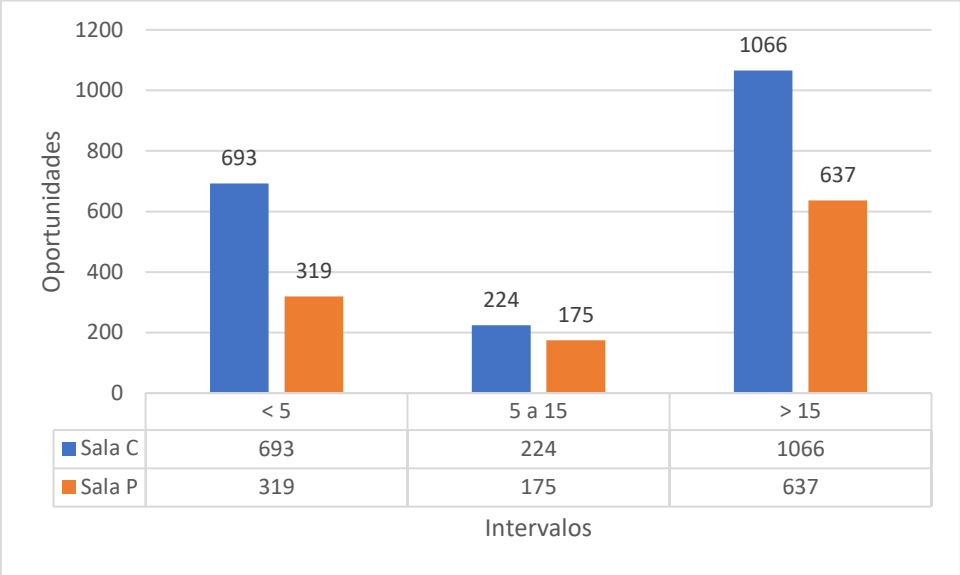

Figura 4.4.c – Intervalos de oportunidad.

Asignando una ventana arbitraria se puede definir un margen de oportunidad para las reuniones espontáneas a partir de la consulta de la herramienta. Se puede suponer reuniones de al menos 20 minutos, sumando un tiempo prudencial de al menos 10 minutos que una sala se detecta sin movimiento y otros 10 minutos como margen de detección de la oportunidad, se fija una ventana de 40 minutos para contabilizar la cantidad de oportunidades por sala.

```
ventana <- 40
oportunidadesC <- segmentosC$lengths[segmentosC$values]
#oportunidadesC[oportunidadesC>ventana

length(oportunidadesC[oportunidadesC>ventana]
```

Cantidad de 'Oportunidades' de la Sala C: 47

```
sum(oportunidadesC[oportunidadesC>ventana])
```

Total de minutos aprovechables de Sala C: 3246

```
oportunidadesP <- segmentosP$lengths[segmentosP$values]
length(oportunidadesP[oportunidadesP>ventana])
```

Cantidad de 'Oportunidades' de la Sala P: 50

```
sum(oportunidadesP[oportunidadesP>ventana])
```

Total de minutos aprovechables de Sala P: 3647

Se representa gráficamente en la figura 4.4.d el porcentaje de mejora en la utilización de las salas con el valor arbitrario de 40 minutos continuados. Los valores de la tabla son en minutos.

Salas/Minutos	Aprovechado	Oportunidades >40	No aprovechables	Total Reservado
SalaC	13569	3246	5453	22268
SalaP	6829	3647	4427	14903

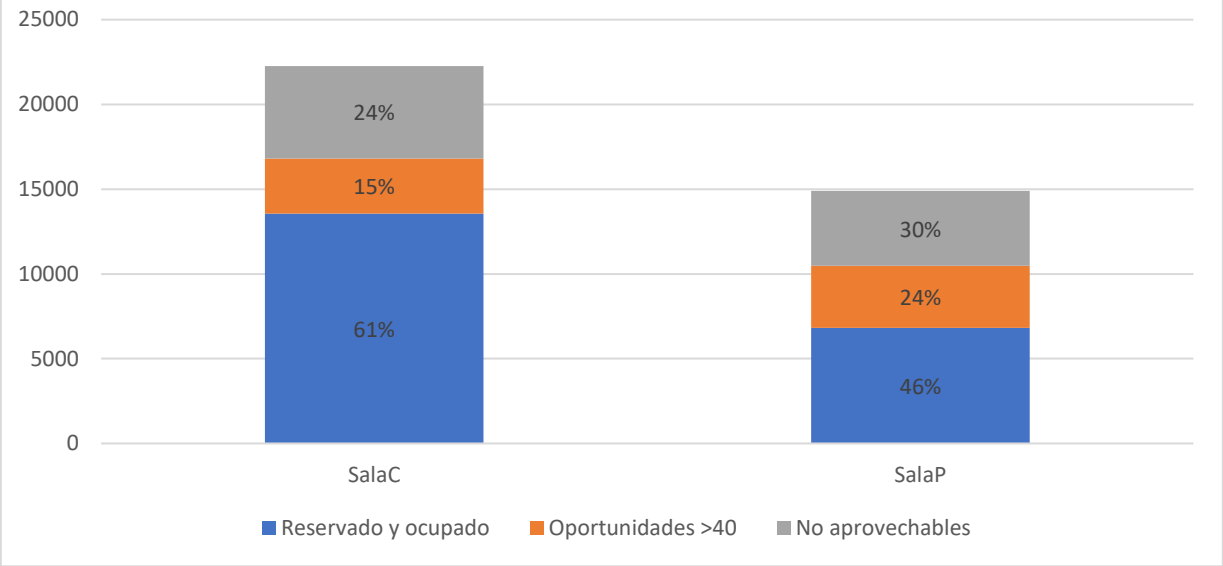

Figura 4.4.d – Porcentaje de mejora.

Se destaca que pese a ser casi cuatro veces más la cantidad de tiempo capturado para la sala C, presenta casi la misma cantidad de oportunidades que la sala C. También cabe aclarar que estas son oportunidades no tomadas o no detectadas. Muchos de estos valores de ocupación mientras la sala está reservada pueden ser "legítimos" o "aprovechados". En el caso de Sala C, que contiene datos de un año completo equivalente a unas 53 semanas, se puede aprovechar un promedio de una reunión de una hora por semana. En el caso de la Sala P, que tiene datos de sólo dos meses y medio o unos 50 días de semana, se puede aprovechar a razón de una reunión diaria de más de una hora. Se debe tener en cuenta que la muestra de la sala P no sólo es más reducida, sino que la información fue capturada durante los meses de verano cuando la mayoría de los trabajadores suele tomar vacaciones.

5. RESULTADOS

5.1. Desarrollo e implementación del software

La codificación de la lógica para el registro se realizó en el lenguaje ASP.NET con la lógica y requerimientos mencionados anteriormente. Tanto el desarrollo como la implementación se realizaron en forma de prueba piloto como un MVP (mínimo producto viable). Se priorizó el funcionamiento de las reglas de negocio y la recolección de datos, dejándose para futuras iteraciones el desarrollo de una interfaz de usuario gráfica y una herramienta de reportes que permita explotar los datos en un panel de control.

El paquete .NET se desplegó en el ambiente de desarrollo de un servidor Microsoft Internet Information Services compartido con distintas aplicaciones corporativas, mientras que la base de datos se creó en una instancia de Microsoft SQL también compartida con otras bases de datos en ambiente de desarrollo. No es necesaria ninguna infraestructura dedicada y sólo se requirió la creación de un hostname para el sitio y la base de datos en el servicio de directorio de nombres (DNS), así como una cuenta de servicio en Active Directory para la administración los mismos.

5.2. Adquisición e instalación de equipamiento

Los dispositivos de detección de movimiento, el stack con extensión de Power Over Ethernet y el bricklet de sensor infrarrojo, se adquirieron en las oficinas centrales en Europa y se distribuyeron a los sitios piloto en las distintas locaciones.

Una vez recibidos se conectaron al cableado Ethernet existente en las salas. En primera instancia se utilizaron las fichas disponibles para dispositivos de usuario en la sala, pero para evitar desconexiones y no ocupar los puertos disponibles se los conectó directamente a la columna de cableado estructurado sobre el cielorraso y a través de una perforación se mantuvo descubierto únicamente el sensor PIR.

El cableado de la oficina de Buenos Aires, así como la de todos los sitios de oficinas, tiene conectividad directa con la red del datacenter, por lo cual los dispositivos adquirieron una dirección reservada del servicio de asignación dinámica (DHCP) y pudieron ser contactados desde el servicio web de la aplicación.

El costo de implementación de la infraestructura consistió principalmente en el valor de adquisición de los dispositivos, de aproximadamente 100 USD cada uno. Se utilizó infraestructura preexistente para el cableado, servidor web y base de datos sin generar gastos adicionales, por lo que la proyección de costo de implementación no tuvo variaciones con los reales.

5.3. Ejecución del piloto

El piloto en la oficina de Buenos Aires se mantuvo por unos meses recolectando datos de las salas. Si bien la interfaz de usuario era accesible, no fue difundida por no tener en la presente instancia la experiencia de usuario necesaria para ser publicada como herramienta

Se analizó el comienzo de una nueva iteración para el desarrollo del front end, pero por haberse elegido una solución diferente para la administración y mejora del uso de las salas, el piloto se vio interrumpido a los tres meses de su lanzamiento. Sin embargo, uno de los disipositivos continuó capturando los datos que se utilizaron para este estudio por el resto del año.

6. CONCLUSIONES

Si bien el trabajo remoto y los diseños de oficina de espacio abierto pueden sugerir que las salas de reunión tendrán menos uso, lo cierto es que la necesidad de espacios de interacción personal y privada seguirá siendo una prioridad en el día a día de las oficinas en la actualidad.

La prueba de concepto confirmó el funcionamiento de la lógica propuesta para esta aplicación, tanto en la presentación de la información en tiempo real para los usuarios como la recolección de datos minuto a minuto para su análisis. Los costos reales de su implementación fueron exactamente los previstos al comienzo.

El costo y esfuerzo de escalar el mínimo producto viable a una implementación de gran escala consiste exclusivamente en la adquisición de nuevos dispositivos, su instalación en la red local y la actualización de una tabla de configuración con un nuevo registro por cada detector.

Los datos recolectados y analizados dieron pruebas del uso subóptimo de las salas de reunión y mostraron oportunidades de mejora; sin embargo, su efectivo aprovechamiento a través de la herramienta de usuario no pudo ser probada por no masificarse su utilización. En una situación ideal en la que todo intervalo de más de 40 minutos sea aprovechado la optimización total en cada una de las salas monitoreadas sería de 15% y 24%.

Por lo visto, se recomienda en cada caso comenzar con un tiempo de captura de datos antes de alentarse el uso de la herramienta a fin de poder comparar los porcentajes de uso efectivo de los espacios de reunión antes y después de su uso.

7. BIBLIOGRAFÍA

BELL, Gordon y GEMMELL, Jim. 2007. A digital life. En: *Scientific American.* Marzo 2007, vol. 296, no. 3. ISSN 0036-8733. Disponible en: Business Source Complete. EBSCO Host.

DE PAOLI, Donatella, ARGE, Kirsten y BLAKSTAD, Siri Hunnes. Creating business value with open space flexible offices. En: *Journal of Corporate Real Estate*. Septiembre 2013, vol. 15, no. 3/4, pp. 181-193. ISSN 1463-001X. Disponible en: Business Source Complete. EBSCO Host.

DURÁN, Guillermo, REY, Pablo y WOLFF, Patricio. Solving the operating room scheduling problem with prioritized lists of patients. En: *Annals of Operations Research*. Noviembre 2017, vol. 258, no. 2, pp. 239-414. ISSN 0254-5330. Disponible en: Academic Search Premier. EBSCO Host

FILIPPINI, Steve. Magic Detectors. En: *Sound & Video Contractor*. Octubre 2015, vol. 23, no. 10, pp. 30-34. ISSN 0741-1715. Disponible en: Academic Search Premier. EBSCO Host.

Finding Secure IoT Devices. En: *Buildings*. Julio 2017, vol. 111, no. 7, p19. ISSN 0007-3725. Disponible en: Business Source Complete. EBSCO Host

ITEAD STUDIO [en línea]. 2019 [consulta: 21 septiembre 2019]. Disponible en: https://www.itead.cc/hb100-miniature-microwave-motion-sensor.html

JAMES, Anne, NANOS, Antonios y THOMPSON, Phillip. V-ROOM: a virtual meeting system with intelligent structured summarisation. En: *Enterprise Information Systems*. Noviembre 2016, vol. 10, no. 8, pp. 863-892). Disponible en: EBSCO Host.

MADRID, Ignacio. Internet of Things en la vida cotidiana. En: *Documentos de Trabajo*. Noviembre 2018, no. 665, pp. 1-77. ISSN 1668-4575. Disponible en: Academic Search Premier. EBSCO Host.

MARGINALIA. The new rules of meeting room etiquette. En: *Marginalia* [en línea]. 19 enero 2018 [consulta: 06 septiembre 2019]. Disponible en: https://web.archive.org/web/20180724134712/http://www.marginalia.online/the-new-rules-of-meeting-room-etiquette/

MARR, Bernard. What is Industry 4.0? Here's A Super Easy Explanation For Anyone. En: Forbes [en línea]. 02 septiembre 2018 [consulta: 21 septiembre 2019]. Disponible en: https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone

MUKHOPADHYAY, Bodhibrata, SRIRANGARAJAN, Seshan y KAR, Subrat. Modeling the Analog Reposnse of Passive Infrared Sensor. En:*Sensors and Actuators A Physical.* Agosto 2018, no. 279, pp. 65-74. DOI 10.1016/j.sna.2018.05.002.

PAPOUTSIDAKIS, Michail *et al.* Motion Sensors and Transducers to Navigate an Intelligent Mechatronic Platform for Outdoor Applications. En: Sensors and Transducers. Marzo 2016, vol. 3, no. 198, pp. 16-24.

SCHWAB, Katherine. The slow death of open offices. En: *Fast Company*. Febrero 2019, no. 230, pp. 10-12. ISSN 1085-9241. Disponible en: Business Source Complete. EBSCO Host.

TRAN, Linh *et al*. A smart meeting room scheduling and management system with utilization control and ad-hoc support based on real-time occupancy detection. En: *IEEE Sixth International Conference on Communications and Electronics*. Junio 2016. DOI: 10.1109/CCE.2016.7562634

TRUSTRADIUS | Meeting Room Booking System [en línea] © 2019. [consulta: 28 septiembre 2019]. Disponible en: https://www.trustradius.com/meeting-room-booking-system

SHI, Xiaojie *et al*. State-of-the-Art Internet of Things in Protected Agriculture. En: *Sensors*. Abril 2019, vol. 19, no. 8, p. 1833. ISSN 1424-8220. Disponible en: Academic Search Premier. EBSCO Host.

XIAO, Guanlian *et al*. Models, algorithms and performance analysis for adaptive operating room scheduling. En: *International Journal of Production Research*. Febrero 2018, vol. 56, no. 4, pp. 1389-1413. Disponible en: Academic Search Premier. EBSCO Host.

YONG, Ching, SUDIRMAN, Rubita, CHEW, Kim Mey. Motion Detection and Analysis with Four Different Detectors. En: *Computational Intelligence, Modelling & Simulation*. Agosto 2011. DOI: 10.1109/CIMSim.2011.18

YU, Zhiwen y NAKAMURA, Yuichi. Smart Meeting Systems: A Survey of State-of-the-Art and Open Issues. En: *ACM Computing Surveys*. Febrero 2010, vol. 42, no. 2, pp 8-20. ISSN 0360-0300. Disponible en: EBSCO Host.

8. ANEXOS

8.1.Anexo 1: Archivo fuente RStudio (.rnd)

````
---
title: "MAKERoom"
output: word_document
---
```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)
```
# Carga de datos
De la base de datos se extraen todos los campos de la tabla Room_History_Status (Historico_actividad) en formato CSV y se utiliza directamente ese formato para importar los datos al dataframe dfroomhistory
```{r}
dfroomhistory <- read.csv("C:/Users/salvarez/OneDrive - Philip Morris International/TIC/TrabajoFinal/Material/Room_History_Status.csv")
```
A fin de verificar los datos importados se muestra la estructura del dataframe y un sumario
```{r}
str(dfroomhistory)
summary (dfroomhistory)
```
#Preparación de datos
Durante el período seleccionado se capturó información de distintas salas en distintas oficinas: dos en New York, cuatro en Lausanne y dos en Buenos Aires. Estas últimas estaban eran las dos salas principales del primer piso del edificio y mantuvieron un funcionamiento prolongado con monitoreo frecuente, por lo que seleccionaremos sólo los valores r-CRChesterfield@pmi.com y r-CRParliament@pmi.com para este análisis. Se renombran además los encabezados
```{r}
colnames (dfroomhistory) <- c("DiaHora","Sala","Movimiento","Reservado")
dfroomhistory<-dfroomhistory[dfroomhistory$Sala == 'r-CRChesterfield@pmi.com' | dfroomhistory$Sala == 'r-CRParliament@pmi.com',]
dfroomhistory$Sala<-factor(dfroomhistory$Sala)
```
Por defecto, la conversión de CSV a dataframe asigó a las columnas DiaHora y Sala el tipo factor multinivel y a las columnas Movimiento y Reservador el tipo entero.Para optimización de los datos y manejo más sencillo se convierten los enteros a tipo lógico donde los únicos valores de 0 y 1 se convierten a Falso (FALSE) y Verdadero (TRUE) respectivamente.
En el caso de la fecha, se utiliza la librería lubridate para transformar el tipo de datos factor al tipo fecha POSIXct en formato año-mes-día hora:minuto:segundo y asignando las fechas originales a la zona horaria donde fueron capturados (CET - Central European Time). En un paso siguiente, la función with_tz permite adaptar las fecha y horas originales a la zona horaria de Buenos Aires (ART)
Para las salas, se reemplazan los nombres de formato correo a los valores más sencillos de SalaC y SalaP
````

A continuación se muestra la nueva estructura con los cambios realizados
```{r}
library(lubridate)
dfroomhistory$Movimiento <- as.logical(dfroomhistory$Movimiento)
dfroomhistory$Reservado <- as.logical(dfroomhistory$Reservado)
dfroomhistory$DiaHora <- as.character(dfroomhistory$DiaHora)
dfroomhistory$DiaHora <- ymd_hms(dfroomhistory$DiaHora,tz="CET")
dfroomhistory$DiaHora <-
with_tz(dfroomhistory$DiaHora,tzone="America/Argentina/Buenos_Aires")
levels(dfroomhistory$Sala) <- c("Sala C", "Sala P")
str(dfroomhistory)
```
#Corrección y ajuste

El servicio web de la aplicación recogió minuto a minuto la información de las salas de Outlook y los valores de detección de movimiento de los dispositivos. Si el dispositivo no da respuesta o no se puede encontrar, el valor por defecto es siempre FALSE. Podemos entonces detectar el primer momento en que se registran valores TRUE cuando comienzan a funcionar los dispositivos infrarrojos.

Se verifica el funcionamiento de ambos detectores por separado. Tiempo de primera detección:
```{r}
min(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala C"])
min(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala P"])
```
Ambos se registran el mismo día por la mañana. El tiempo de la última detección, sin embargo, revela que la Sala C registró movimiento durante doce meses mientras que la sala P sólo durante tres.
```{r}
max(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala C"])
max(dfroomhistory$DiaHora[dfroomhistory$Movimiento & dfroomhistory$Sala=="Sala P"])
```
Estos valores nos permiten entonces separar el dataframe según la sala, y además acotar los datos a las fechas obtenidas para realizar el análisis sólo en los períodos en los que cada detector estuvo activo. Se utiliza en este caso la librería dplyr para filtrar y reagrupar los dataframes. Se remueve la columna de Sala luego de filtrar ya que se vuelve irrelevante.
Se verifica luego la cantidad de registros en cada dataframe.
```{r}
library(dplyr)
dfSalaC <- dfroomhistory %>%
            filter(Sala == "Sala C") %>%
              filter(DiaHora<"2018-01-13") %>%
                filter(DiaHora>"2017-01-02") %>%
                  select (-Sala)
dfSalaP <- dfroomhistory %>%
            filter(Sala == "Sala P") %>%
```

```
              filter(DiaHora<"2017-03-18") %>%
                filter(DiaHora>"2017-01-02") %>%
                  select (-Sala)
c(noquote('Sala C: '),nrow(dfSalaC))
c(noquote('Sala P: '),nrow(dfSalaP))
```
#Análisis
Se analiza en primera instancia el porcentaje de ocupación real de cada sala. Nos interesa ver en un gráfico sencillo para cada minuto en que la sala estuvo reservada si se detectó o no algún movimiento
Utilizando nuevamente la librería dplyr generamos un nuevo dataframe para cada sala donde sólo se registran aquellos momentos en que la sala estuvo reservada
Vemos primero un pie chart considerando sólo los valores de ocupación mientras la sala está reservada
```{r}
soloReservadoC <- dfSalaC %>%
            select(-DiaHora) %>% #Remover DiaHora
            group_by(Movimiento,Reservado) %>% #Agrupar valores lógicos
            filter(Reservado) #Filtrar sólo reservado
soloReservadoP <- dfSalaP %>%
            select(-DiaHora) %>% #Remover DiaHora
            group_by(Movimiento,Reservado) %>% #Agrupar valores lógicos
            filter(Reservado)  #Filtrar sólo reservado

```
Con las librerías ggplot2 y gridExtra generamos dos gráficos de barra lado a lado para visualizar los datos obtenidos. Los dataframes difieren en su tamaño, razón por la cual se realizan gráficos separados puesto lado a lado. De esta forma se mantiene la comparación de las proporciones a pesar de la diferencia en la muestra.
```{r}
library(ggplot2)
library(gridExtra)
plotC <- ggplot(soloReservadoC)+ geom_bar(aes(x=Reservado,fill=Movimiento))
plotP <- ggplot(soloReservadoP)+ geom_bar(aes(x=Reservado,fill=Movimiento))
grid.arrange(plotC,plotP,ncol=2)
```
Tomamos nuevamente la muestra completa pero filtrando los horarios de oficina únicamente. Si bien hay actividad a partir de las 7 y hasta las 19, el horario central de 9 a 17 concentra la mayor cantidad de empleados en el sitio.
```{r}
nine2fiveC <- dfSalaC %>%
                mutate(dsem = weekdays(DiaHora)) %>% #traigo el día de la semana
                filter(hour(DiaHora)>=9,hour(DiaHora)<=16,
                       dsem!='domingo',dsem!='sabado')
nine2fiveP <- dfSalaP %>%
                mutate(dsem = weekdays(DiaHora)) %>% #traigo el día de la semana
                filter(hour(DiaHora)>=9,hour(DiaHora)<=16,
                       dsem!='domingo',dsem!='sabado')
```

Volvemos a utilizar los gráficos lado a lado para comparar proporciones. Se observa en este caso que los períodos acumulados de sala reservada (TRUE) no parecen ser la ampliamente mayores (incluso en la sala C es considerablemente menor), pero se debe tener en cuenta que los mediodías y días feriados o de menor actividad se conservan en la muestra.

```{r}
plotC2 <-ggplot(nine2fiveC)+ geom_bar(aes(x=Reservado,fill=Movimiento)) + labs(title="Sala C")
plotP2 <-ggplot(nine2fiveP)+ geom_bar(aes(x=Reservado,fill=Movimiento)) + labs(title= "Sala P")
grid.arrange(plotC2,plotP2,ncol=2)
```

El caso concreto de análisis son los segmentos de oportunidad en los cuales una sala presenta períodos aprovechables, en los que aunque esté reservada se encuentra vacía por tiempo suficiente como para ocuparse con una reunión espontánea. Dentro de los datos registrados, estos segmentos son aquellos registros en los cuales el estado para una misma sala es de Moviemiento=FALSE y Reservado=TRUE en forma continua. Para encontrarlos se utiliza la función rle (run length encoding - codificación de longitud de ejecución), que devuelve las longitudes y valores de valores equivalentes en un vector. Se convierten entonces los dataframes en vectores, filtrándolos por la combinación de valores mencionada.

Cada número en los resultados obtenidos representa entonces la cantidad de minutos continuos en los que la sala estuvo reservada sin registrar movimiento. Cabe aclarar que ante un sólo movimiento registrado por el dispositivo su valor de Movimiento=TRUE se mantiene por dos minutos, para evitar falsos negativos cuando los ocupantes se encuentran relativamente quietos. Sin embargo, se aprecia que se registra una gran cantidad de segmentos de 1 a 5 minutos precisamente por estos casos de relativa quietud. Otros segmentos de 5 a 15 minutos reflejarán las demoras en iniciar una reunión o aquellas que terminan poco antes de lo previsto.

Sala C. Total de segmentos, total de menores a 5, totales entre 10 y 15

```{r}
segmentosC <- rle(with(nine2fiveC, !Movimiento & Reservado))
vectorC<-segmentosC$lengths[segmentosC$values]
c('Menos de 5 minutos',length(vectorC[vectorC<5]))
c('De 5 a 15 minutos',length(vectorC[vectorC>=5&vectorC<15]))
c('Más de 15 minutos',length(vectorC))
```

```{r}
segmentosP <- rle(with(nine2fiveP, !Movimiento & Reservado))
vectorP<-segmentosP$lengths[segmentosP$values]

c('Menos de 5 minutos',length(vectorP[vectorP<5]))
c('De 5 a 15 minutos',length(vectorP[vectorP>=5&vectorP<15]))
c('Más de 15 minutos',length(vectorP))
```

Asignando una ventana arbitraria se puede definir un margen de oportunidad para las reuniones espontáneas a partir de la consulta de la herramienta. Se puede suponer reuniones de al menos 20 minutos, sumando un tiempo prudencial de al menos 10 minutos que una sala se detecta sin movimiento y otros 10 minutos como

margen de detección de la oportunidad, se fija una ventana de 40 minutos para contabilizar la cantidad de oportunidades por sala.

```{r}
ventana <- 40
oportunidadesC <- segmentosC$lengths[segmentosC$values]
#oportunidadesC[oportunidadesC>ventana]
c(noquote('Cantidad de \'Oportunidades\' de la Sala C:
'),length(oportunidadesC[oportunidadesC>ventana]))
c(noquote('Total de minutos aprovechables de Sala C:
'),sum(oportunidadesC[oportunidadesC>ventana]))
noquote('')
oportunidadesP <- segmentosP$lengths[segmentosP$values]
c(noquote('Cantidad de \'Oportunidades\' de la Sala P:
'),length(oportunidadesP[oportunidadesP>ventana]))
c(noquote('Total de minutos aprovechables de Sala P:
'),sum(oportunidadesP[oportunidadesP>ventana]))
```