

1. De Requisitos a Casos de Uso:

Propósito: Asegura que cada caso de uso esté justificado por un requisito real.

Método: Vincular cada caso de uso directamente al requisito que aborda. Esto se puede hacer utilizando una matriz de trazabilidad o herramientas especializadas de gestión de requisitos.

2. De Casos de Uso a Diagramas de Diseño:

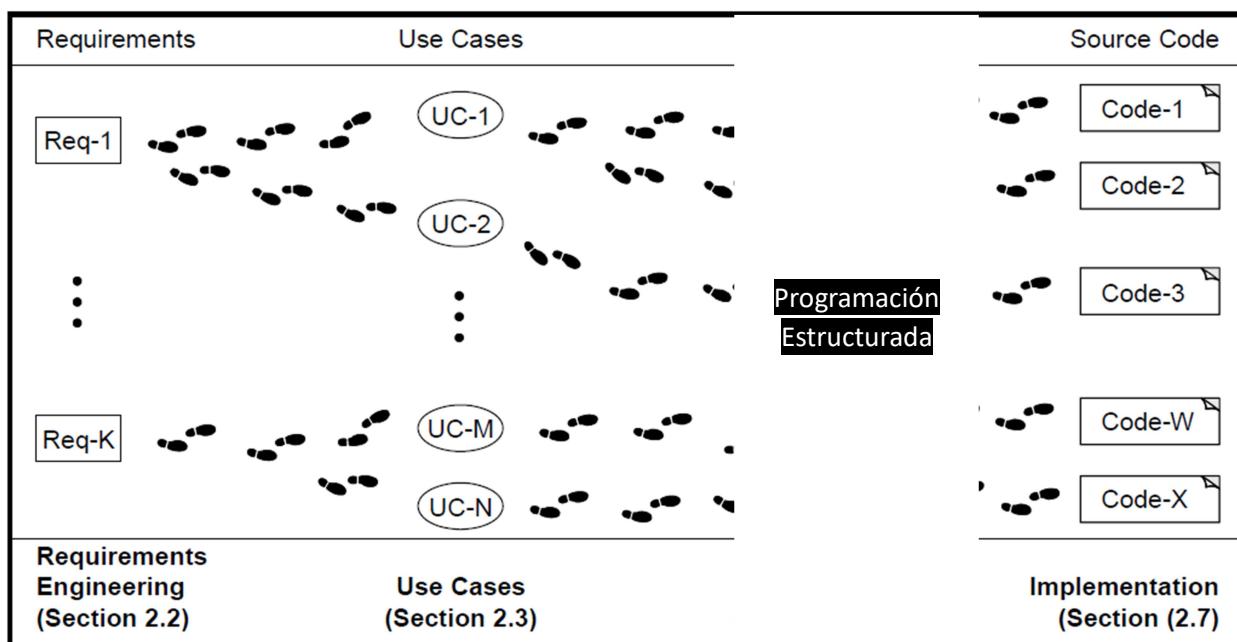
Propósito: Asegura que el diseño proporciona una solución para cada caso de uso.

Método: Los diagramas de diseño deben estar anotados o vinculados a los casos de uso que implementan. Esto se puede visualizar en herramientas que soportan UML y otras notaciones de diseño.

3. De Diagramas de Diseño a Código Fuente:

Propósito: Asegura que el código implementa el diseño de manera precisa.

Método: Los repositorios de código fuente pueden incluir referencias a documentos de diseño, y los IDE modernos pueden vincular el código a modelos.



Problemática: Sistema de Control de Acceso Doméstico

Descripción Breve:

Desarrollo de un sistema integral de control de acceso residencial que administra cerraduras y se sincroniza con dispositivos del hogar (iluminación, climatización, alarmas), ofreciendo seguridad y comodidad avanzadas.

Objetivos:

- Control eficiente de múltiples puntos de acceso.
- Autenticación y validación de usuarios para entrada segura.
- Monitoreo y reporte de seguridad en tiempo real.
- Interacción inteligente con dispositivos domésticos.
- Funciones de simulación de presencia y operación remota para usuarios.

Enfoque de Solución:

Adopción de una arquitectura descentralizada que minimiza la dependencia de una red centralizada, facilitando la gestión y mantenimiento del sistema y mejorando la respuesta ante intentos de intrusión.

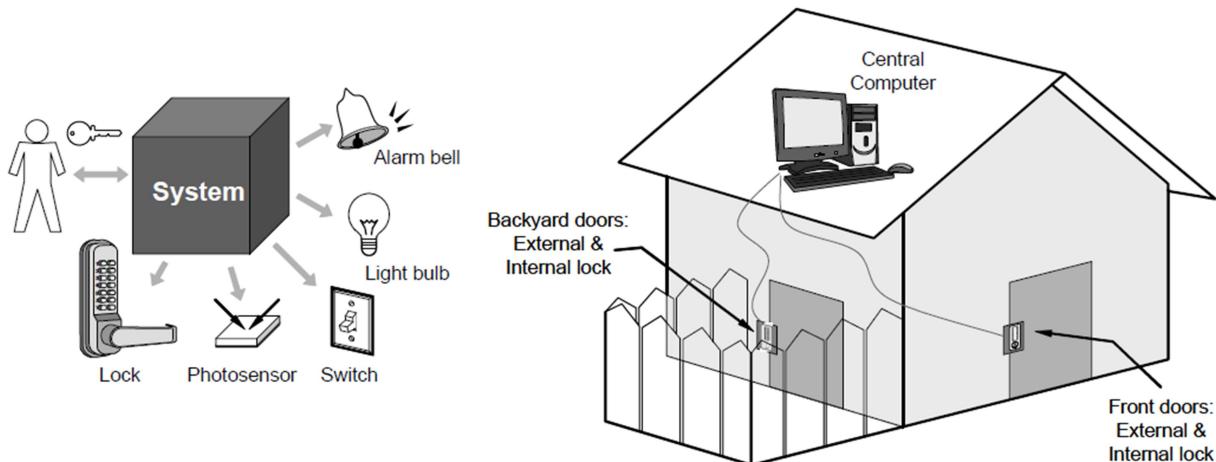


Tabla 1: Requisitos para el primer caso de estudio, sistema de acceso seguro al hogar.

| Identificador | Prioridad | Requisito |
|---------------|-----------|---|
| REQ1 | 5 | El sistema deberá mantener la puerta cerrada con llave en todo momento, a menos que un usuario autorizado ordene lo contrario. Cuando la cerradura esté desactivada, se iniciará una cuenta regresiva al final de la cual la cerradura se cerrará automáticamente (si aún está abierta). |
| REQ1a | | El sistema debe mantener las puertas cerradas en todo momento, a menos que un usuario autorizado ordene lo contrario. |
| REQ1b | | Cuando la cerradura esté desarmada, se iniciará una cuenta regresiva al final de la cual la cerradura se armará automáticamente (si aún está desarmada). |
| REQ2 | 2 | El sistema deberá cerrar la puerta con llave cuando se ordene presionando un botón dedicado. |
| REQ3 | 5 | El sistema deberá proporcionar un clave de código válido para desbloquear la puerta y activar otros dispositivos. |
| REQ4 | 4 | El sistema debería permitir errores al ingresar el clave de código válido. Sin embargo, para resistir los "ataques de diccionario", el número de intentos fallidos permitidos deberá ser pequeño, digamos tres, después de los cuales el sistema se bloqueará y sonará |

| Identificador | Prioridad | Requisito |
|---------------|-----------|---|
| | | la campana de alarma. |
| REQ5 | 2 | El sistema deberá mantener un registro histórico de todos los accesos intentados para su posterior revisión. |
| REQ6 | 2 | El sistema debería permitir agregar nuevas personas autorizadas en tiempo real o eliminar las existentes. |
| REQ7 | 2 | El sistema deberá permitir configurar las preferencias para la activación de dispositivos cuando el usuario proporcione un clave de código válido, así como cuando se detecte un intento de robo. |
| REQ8 | 1 | El sistema debería permitir buscar en el registro histórico especificando uno o más de estos parámetros: el marco de tiempo, el rol del actor, la ubicación de la puerta o el tipo de evento (desbloqueo, bloqueo, fallo de energía, etc.). Esta función deberá estar disponible en la Web apuntando un navegador a una URL especificada. |
| REQ9 | 1 | El sistema debería permitir presentar consultas sobre accesos "sospechosos". Esta función deberá estar disponible en la Web. |

Tabla 2: Historias de Usuario

| Identificador | Historia de Usuario | Requisitos | Story Points (Estimación) |
|---------------|---|------------|---------------------------|
| ST-1 | Como persona autorizada (inquilino o propietario), puedo mantener las puertas cerradas en todo momento. | REQ1 | 5 puntos |
| ST-2 | Como persona autorizada (inquilino o propietario), puedo cerrar las puertas a demanda. | REQ1 | 3 puntos |
| ST-3 | La cerradura debería bloquearse automáticamente después de un período de tiempo definido. | REQ2 | 8 puntos |
| ST-4 | Como persona autorizada (inquilino o propietario), puedo desbloquear las puertas. (Prueba: Permitir un pequeño número de errores, digamos tres.) | REQ3, REQ4 | 13 puntos |
| ST-5 | Como propietario , puedo gestionar en tiempo real a las personas autorizadas. | REQ6 | 13 puntos |
| ST-6 | Como persona autorizada (inquilino o propietario), puedo ver accesos pasados. | REQ8 | 8 puntos |
| ST-7 | Como inquilino , puedo configurar las preferencias para la activación de varios dispositivos. | REQ7 | 8 puntos |
| ST-8 | Como inquilino , puedo presentar quejas sobre accesos "sospechosos". | REQ9 | 8 puntos |

La Tabla muestra las historias de usuario para el control de acceso al hogar. Si comparamos estas historias con los requisitos derivados anteriormente (Tabla 1), encontraremos que las historias ST-1 y ST-2 corresponden aproximadamente al requisito REQ1, la historia ST-3 corresponde a REQ2, la historia ST-4 corresponde a REQ3 y REQ4, y la historia ST-6 corresponde a REQ8.

Estimación de esfuerzo

Es el proceso de predecir la cantidad de esfuerzo y tiempo que se necesitará para completar una tarea o un proyecto. El esfuerzo se puede estimar en **puntos de historia (story points)**, son una unidad de medida común en las metodologías ágiles como **Scrum**.

Arquitectura de software

La arquitectura planteada será una **cliente-servidor** es un modelo de diseño de software donde las tareas o cargas de trabajo se distribuyen entre los proveedores de recursos o servicios, llamados servidores, y los solicitantes de servicios, llamados clientes.

Estructura

Servidor:

Servidor de Aplicaciones: Aloja la lógica de negocio central, incluyendo la gestión de usuarios, el control de accesos, la integración con dispositivos inteligentes y la lógica de seguridad.

Base de Datos: Almacena información sobre usuarios, registros de acceso, configuraciones de dispositivos y otros datos relevantes.

Servidor de Autenticación: Maneja la autenticación y autorización de usuarios.

API de Servicios: Proporciona endpoints para que los clientes interactúen con el sistema, como abrir/cerrar puertas, configurar preferencias y revisar registros de acceso.

Servidor de Eventos: Maneja eventos en tiempo real, como alertas de seguridad o cambios en el estado de los dispositivos.

Cliente:

Aplicación Web/Móvil: Interfaz de usuario para que los inquilinos y propietarios interactúen con el sistema. Permite realizar acciones como desbloquear puertas, configurar preferencias y revisar registros.

Dispositivos IoT: Actúan como clientes que se comunican con el servidor para recibir comandos (por ejemplo, cerraduras inteligentes, sistemas de alarma, termostatos).

Comunicación:

HTTP/HTTPS: Protocolo estándar de comunicación entre clientes y servidores.

WebSockets: Para comunicación en tiempo real, permitiendo que el servidor envíe actualizaciones inmediatas a los clientes.

MQTT o CoAP: Protocolos ligeros de mensajería para la comunicación con dispositivos IoT.

Seguridad:

Encriptación: Todos los datos transmitidos entre clientes y servidores deben estar encriptados utilizando TLS/SSL.

Autenticación y Autorización: Implementar mecanismos robustos para verificar la identidad de los usuarios y dispositivos, y asegurar que solo tengan acceso a las funciones permitidas.

Ejemplo de Flujo de Operación:

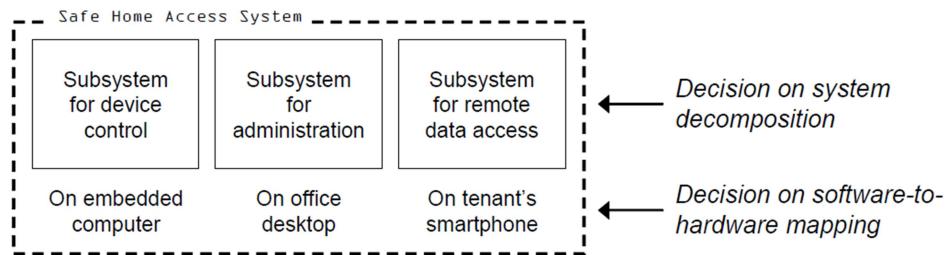
1. **Usuario Interactúa con la Aplicación:** Un inquilino usa la aplicación móvil para desbloquear la puerta de su apartamento.
2. **Autenticación:** La aplicación envía la solicitud al servidor de autenticación para verificar la identidad del usuario.
3. **Lógica de Negocio:** Una vez autenticado, el servidor de aplicaciones procesa la solicitud, verifica los permisos y envía un comando al dispositivo IoT correspondiente.
4. **Acción del Dispositivo:** La cerradura inteligente recibe el comando y desbloquea la puerta.
5. **Registro y Respuesta:** El servidor registra el evento y envía una confirmación a la aplicación móvil.

Beneficios de esta Arquitectura:

- Centralización de la Gestión: Facilita la administración y el mantenimiento del sistema.
- Escalabilidad: Permite escalar el sistema añadiendo más recursos al servidor o balanceando la carga entre múltiples servidores.
- Flexibilidad: Los clientes pueden ser diversos (aplicaciones web, móviles, dispositivos IoT) y comunicarse con el mismo servidor de aplicaciones.
- Seguridad Mejorada: Permite una gestión centralizada de la seguridad, lo que es crucial para un sistema de control de acceso.

Diagrama de bloques "Sistema de Acceso Seguro a la Vivienda".

El diagrama está dividido en tres subsistemas principales:

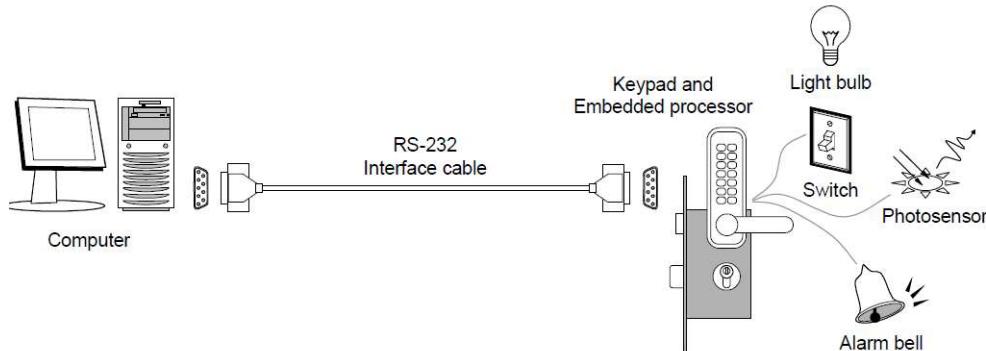


- Subsistema para control de dispositivos: Este subsistema está alojado en un ordenador embebido.
- Subsistema para administración: Este subsistema se ejecuta en un escritorio de oficina.
- Subsistema para acceso remoto a datos: Este subsistema se encuentra en el smartphone del inquilino.

El diagrama también destaca dos decisiones importantes que se han tomado:

- Decisión sobre la descomposición del sistema.
- Decisión sobre el mapeo de software a hardware.

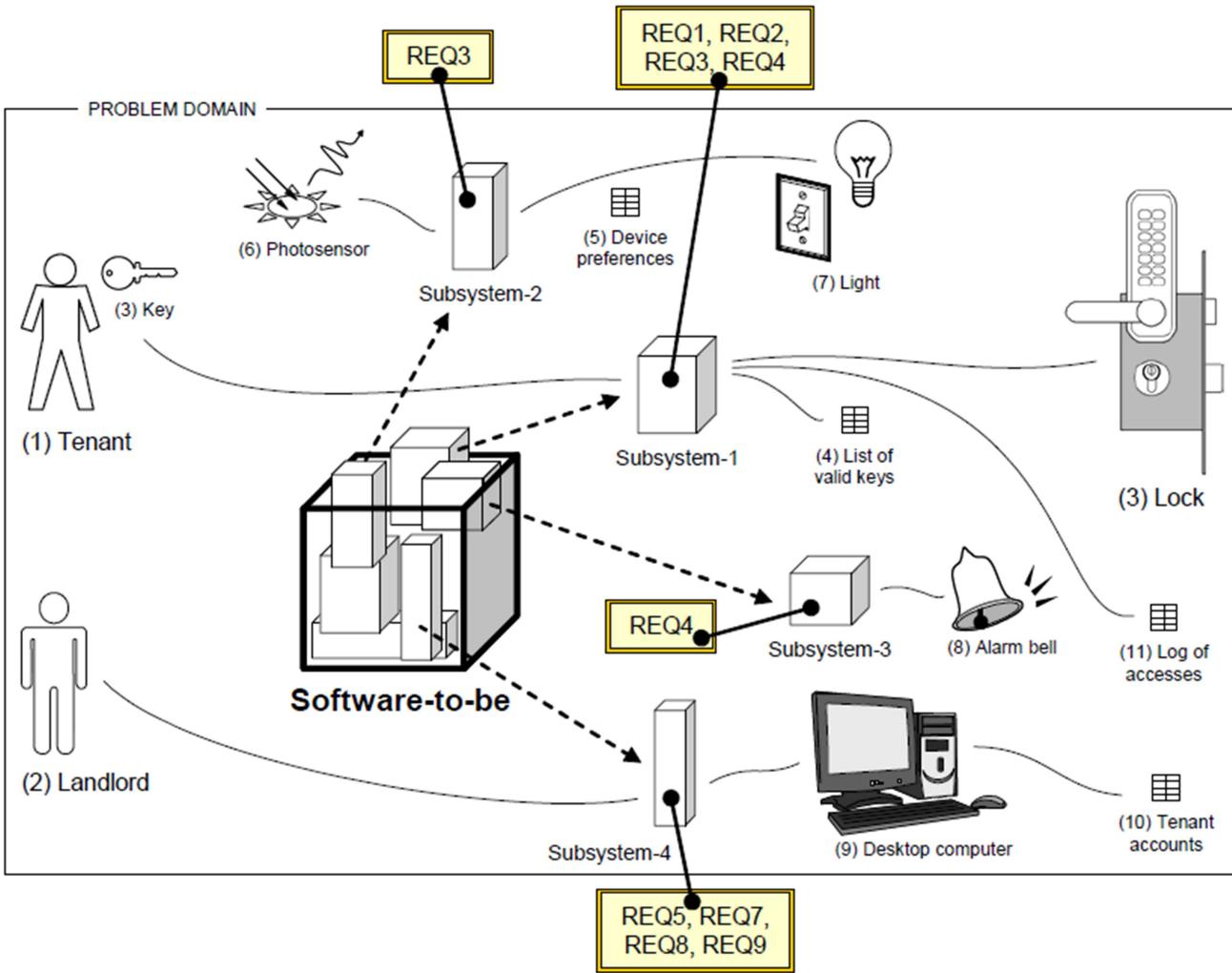
Componentes de hardware para la implementación del sistema



Componentes de hardware para la implementación de un sistema.

- "Computadora" conectado a través de un "Cable de interfaz RS-232" a un "Teclado y Procesador Embebido" que está incorporado en una cerradura de puerta.
- A la derecha del teclado y procesador embebido, hay un "Interruptor" conectado a una "Bombilla" y a un "Fotosensor" que está alineado con la bombilla. Además, hay una "Campana de Alarma" que parece estar también conectada al sistema.

Elementos del dominio del problema



Once subdominios en el dominio del problema

Los subdominios clave son:

- el inquilino (1)
- el propietario (2)
- la cerradura (3)

Algunos subdominios son personas u objetos físicos y algunos subdominios son artefactos digitales, como:

- la lista de claves válidas (4)
- las cuentas de los inquilinos (10)
- el registro de accesos (11)

Otros:

- Dispositivos (5)
- Fotosensor (6)
- Luz del foco (7)
- Campana (8)
- Computadora (9)

Subsistemas

Subsistema 1

| Identificador | Prioridad | Requisito |
|---------------|-----------|--|
| REQ1 | 5 | El sistema deberá mantener la puerta cerrada con llave en todo momento, a menos que un usuario autorizado ordene lo contrario. Cuando la cerradura esté desactivada, se iniciará una cuenta regresiva al final de la cual la cerradura se cerrara automáticamente (si aún está abierta). |
| REQ2 | 2 | El sistema deberá cerrar la puerta con llave cuando se ordene presionando un botón dedicado. |
| REQ3 | 5 | El sistema deberá proporcionar un código de CLAVE válido para desbloquear la puerta y activar otros dispositivos. |
| REQ4 | 4 | El sistema debería permitir errores al ingresar el código de llave. Sin embargo, para resistir los "ataques de diccionario", el número de intentos fallidos permitidos deberá ser pequeño, digamos tres, después de los cuales el sistema se bloqueará y sonará la campana de alarma. |

Subsistema 2

| Identificador | Prioridad | Requisito |
|---------------|-----------|---|
| REQ3 | 5 | El sistema deberá proporcionar un código de CLAVE válido para desbloquear la puerta y activar otros dispositivos. |

Subsistema 3

| Identificador | Prioridad | Requisito |
|---------------|-----------|---|
| REQ4 | 4 | El sistema debería permitir errores al ingresar el código de llave. Sin embargo, para resistir los "ataques de diccionario", el número de intentos fallidos permitidos deberá ser pequeño, digamos tres, después de los cuales el sistema se bloqueará y sonará la campana de alarma. |

Subsistema 4

| Identificador | Prioridad | Requisito |
|---------------|-----------|---|
| REQ5 | 2 | El sistema deberá mantener un registro histórico de todos los accesos intentados para su posterior revisión. |
| REQ6 | 2 | El sistema debería permitir agregar nuevas personas autorizadas en tiempo real o eliminar las existentes. |
| REQ7 | 2 | El sistema deberá permitir configurar las preferencias para la activación de dispositivos cuando el usuario proporcione un código de llave válido, así como cuando se detecte un intento de robo. |
| REQ8 | 1 | El sistema debería permitir buscar en el registro histórico especificando uno o más de estos parámetros: el marco de tiempo, el rol del actor, la ubicación de la puerta o el tipo de evento (desbloqueo, bloqueo, fallo de energía, etc.). Esta función deberá estar disponible en la Web apuntando un navegador a una URL especificada. |
| REQ9 | 1 | El sistema debería permitir presentar consultas sobre accesos "sospechosos". Esta función deberá estar disponible en la Web. |

Casos de uso

El modelado de casos de uso es una técnica fundamental en la ingeniería de requisitos y el diseño de sistemas. Un caso de uso es una descripción de cómo un usuario utilizará el sistema planificado para lograr objetivos de negocio.

Actores y Sus Metas

Un actor es cualquier entidad (humana, objeto físico u otro sistema) externa al sistema que interactúa con él. Los actores tienen sus responsabilidades y buscan la asistencia del sistema para gestionar esas responsabilidades.

Para alcanzar sus metas, un actor realiza algunas acciones. Una acción es el desencadenamiento de una interacción con el sistema.

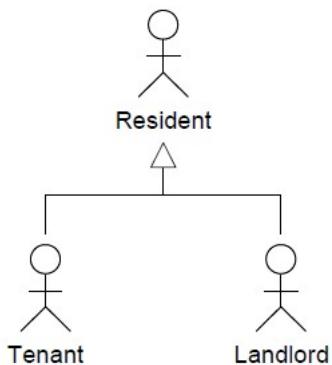
Actores:

- **Inquilino:** ocupante de la vivienda.
- **Propietario:** dueño de la propiedad o administrador.
- **Dispositivo:** dispositivo físico que será controlado por el sistema, como mecanismos de cerradura y interruptores de luz.
- **Otros actores potenciales:** Mantenimiento, Policía, etc.

Un actor puede ser una persona u otro sistema que interactúa con nuestro sistema por desarrollar. Hay dos categorías principales de actores, definidas en relación con un caso de uso particular:

- **Actor iniciador** (también llamado actor primario o simplemente usuario): inicia el caso de uso para realizar un objetivo, que depende de las responsabilidades del actor y el contexto actual.
- **Actor participante** (también llamado actor secundario): participa en el caso de uso, pero no lo inicia; hay dos subcategorías:
 - **Actor de apoyo:** ayuda al sistema por desarrollar a completar el caso de uso, es decir, nuestro sistema por desarrollar inicia al actor de apoyo.
 - **Actor fuera de escena:** participa pasivamente en el caso de uso, es decir, ni inicia ni ayuda a completar el caso de uso, pero puede ser notificado sobre algún aspecto de este.

Los actores pueden definirse en jerarquías de generalización, en las que una descripción de actor abstracta se comparte y se aumenta con una o más descripciones de actor específicas.



Metas

Tabla 3: Actores, objetivos y los casos de uso asociados para el sistema de control de acceso al hogar.

Aquí tienes la tabla traducida y formateada:

| Actor | Objetivo del Actor (lo que el actor intenta lograr) | Nombre del Caso de Uso |
|---------------------|--|---|
| Propietario | Desarmar la cerradura, entrar y encender las luces. | Desbloquear (UC-1) |
| Propietario | Cerrar la puerta con llave y apagar las luces (¿a veces?). | Bloquear (UC-2) |
| Propietario | Crear una nueva cuenta de usuario y permitir acceso al hogar. | AñadirUsuario (UC-3) |
| Propietario | Retirar una cuenta de usuario existente y deshabilitar acceso. | EliminarUsuario (UC-4) |
| Inquilino | Averiguar quién accedió al hogar en un intervalo de tiempo dado y potencialmente presentar quejas. | InspeccionarHistorialDeAcceso (UC-5) |
| Inquilino | Desarmar la cerradura, entrar y encender las luces. | Desbloquear (UC-1) |
| Inquilino | Cerrar la puerta con llave y apagar las luces (¿a veces?). | Bloquear (UC-2) |
| Inquilino | Configurar las preferencias de activación del dispositivo. | ConfigurarPreferenciasDelDispositivo (UC-6) |
| DispositivoDeCierre | Controlar el mecanismo físico de la cerradura. | UC-1, UC-2 |
| InterruptorDeLuz | Controlar la bombilla. | UC-1, UC-2 |
| [por identificar] | Cerrar la puerta automáticamente si se deja sin llave durante un intervalo de tiempo dado. | AutoBloqueo (UC-2) |

Relaciones

En los diagramas de casos de uso, existen varias relaciones estándar que describen cómo interactúan los diferentes casos de uso y actores. Aquí están los tipos más comunes de relaciones que se pueden encontrar en dichos diagramas:

Asociación (Association):

Es la relación más básica entre un actor y un caso de uso. Significa que el actor participa de alguna forma en el caso de uso. Se representa con una línea sólida.

Inclusión (Include): subareas requeridas del caso principal

Esta relación indica que un caso de uso (el caso de uso base) siempre incluirá el comportamiento de otro caso de uso (el caso de uso incluido). Esto se utiliza para evitar la redundancia y favorecer la reutilización. Se representa con una línea punteada con la etiqueta <<include>>.

Extensión (Extend): extensiones opcionales del caso principal

La relación de extensión indica que un caso de uso (el caso de uso de extensión) puede añadir comportamiento a otro caso de uso (el caso de uso base) bajo ciertas condiciones. Es una forma de añadir comportamiento opcional o adicional a un caso de uso existente. Se representa con una línea punteada con una flecha en el extremo y la etiqueta <<extend>>.

Generalización (Generalization):

En los casos de uso, la generalización se utiliza para representar la herencia, donde un caso de uso más generalizado es especializado por uno o más casos de uso. Se parece a la herencia en la programación orientada a objetos y se representa con una línea sólida con una flecha hueca.

En el contexto de los diagramas de casos de uso, las relaciones "**initiate**" y "**participate**" no son términos estándar definidos por UML (Unified Modeling Language), sin embargo, estas palabras podrían ser usadas para describir de manera informal cómo los actores interactúan con los casos de uso:

initiate (iniciar): Esta relación probablemente indica que el actor es el responsable de comenzar o desencadenar el caso de uso. Un actor que inicia un caso de uso es aquel que realiza la acción que causa que el sistema ejecute el caso de uso.

participate (participar): Esta relación sugiere que el actor está involucrado en el caso de uso, pero no necesariamente lo inicia. Podría ser un colaborador o un actor secundario que desempeña un rol dentro del proceso del caso de uso, posiblemente después de que haya sido iniciado por otro actor.

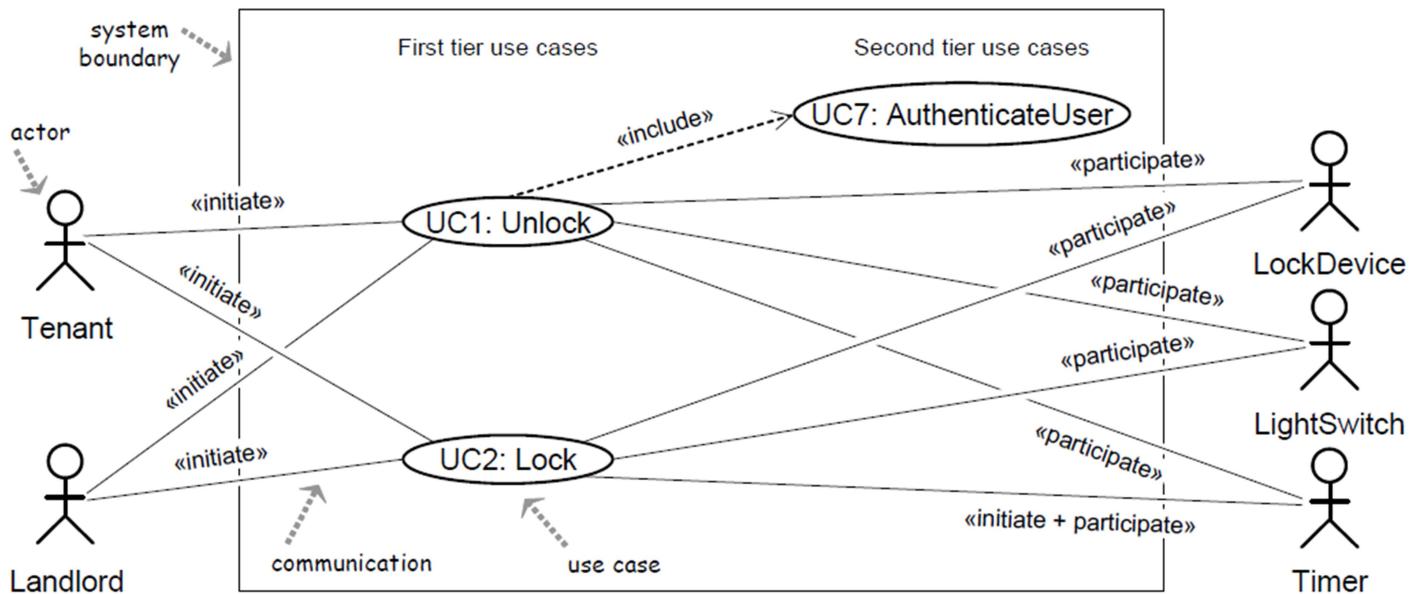
En los diagramas de casos de uso estándar de UML, la forma de modelar estas interacciones es simplemente con una línea de asociación entre el actor y el caso de uso. Sin embargo, se puede agregar notas o estereotipos personalizados para clarificar los roles específicos de los actores como "**initiate**" o "**participate**", especialmente si se considera relevante para la comprensión del diagrama.

Descripción informal de los casos de uso

Diagrama de casos de uso UML para el subsistema de control de dispositivos del sistema de acceso al hogar.

Compárese con la Tabla 3.

En la imagen proporcionada, vemos asociaciones (indicadas por líneas sólidas entre los actores y los casos de uso) y una inclusión (indicada por una línea punteada con la etiqueta <>include>>). No hay ejemplos de extensión o generalización en este diagrama en particular.



Se muestra un diagrama de casos de uso que ilustra la interacción entre actores y las funcionalidades del sistema dentro de un contexto determinado. Los casos de uso se presentan como procesos o funciones que el sistema puede realizar, y los actores son las entidades (usuarios o sistemas externos) que interactúan con el sistema. En el diagrama, podemos identificar lo siguiente:

Actores:

- Tenant (Inquilino): Inicia los casos de uso UC1: Desbloquear y UC2: Bloquear.
- Landlord (Propietario): También inicia los casos de uso UC1: Desbloquear y UC2: Bloquear.
- LockDevice (Dispositivo de Bloqueo): Participa en los casos de uso UC1 y UC2.

- LightSwitch (Interruptor de Luz): Participa en los casos de uso UC1 y UC2.
- Timer (Temporizador): Inicia y participa en el caso de uso UC2: Bloquear.

Casos de Uso:

- UC1: Unlock (Desbloquear): Este caso de uso describe el proceso que permite al Tenant (Inquilino) o al Landlord (Propietario) desbloquear algo, posiblemente una puerta o un sistema de seguridad.
- UC2: Lock (Bloquear): Este caso de uso describe el proceso opuesto al anterior, donde el Tenant o el Landlord activan el bloqueo del dispositivo o sistema.

Relaciones:

- Incluye (include): El caso de uso UC7: AuthenticateUser (Autenticar Usuario) está conectado al caso de uso UC1: Unlock (Desbloquear) con una relación "include", lo que sugiere que la autenticación del usuario es un paso necesario para desbloquear.
- Inicia (initiate): Tanto el Tenant como el Landlord tienen relaciones de iniciación hacia los casos de uso de desbloqueo y bloqueo, indicando que pueden comenzar estos procesos.
- Participa (participate): El LockDevice, LightSwitch, y el Timer tienen relaciones de participación con los casos de uso de desbloqueo y bloqueo, lo que implica que están involucrados en estos procesos.
- Comunicación: Hay una relación de comunicación entre el Tenant y el Landlord con respecto al caso de uso UC2: Lock (Bloquear).

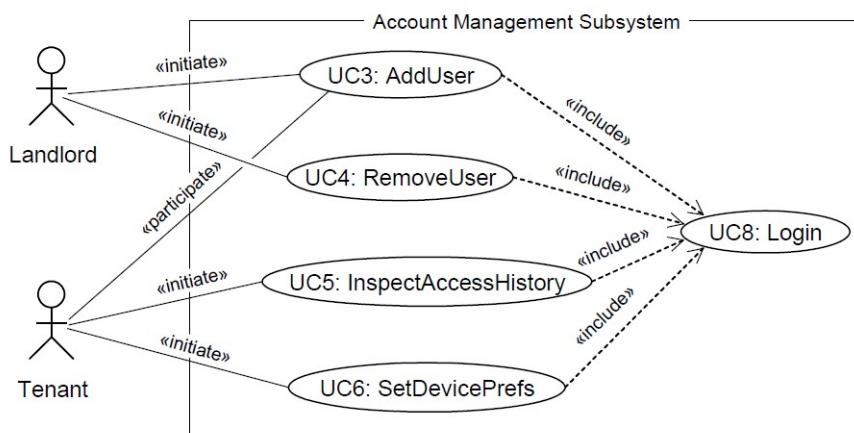
Anotaciones

- El caso de uso AuthenticateUser no es un buen candidato para los casos de uso de primera línea, porque no representa un objetivo significativo y autónomo para un actor iniciador. Sin embargo, es útil mostrarlo explícitamente como un caso de uso de segunda línea, especialmente porque revela qué casos de uso requieren autenticación de usuario.
- Por ejemplo, se podría argumentar que Lock no necesita autenticación, porque realizarlo sin autenticación no representa una amenaza de seguridad. De manera similar, Disable no debería requerir autenticación porque eso frustraría el propósito de este caso.

Frontera del Sistema (system boundary): Representada por la línea discontinua, separa lo que está dentro del sistema de lo que está fuera.

Casos de Uso de Primer y Segundo Nivel (First tier use cases and Second tier use cases): Los casos de uso están clasificados en dos niveles, indicando posiblemente una jerarquía o una secuencia en la ejecución de las funciones del sistema.

Figura Casos de uso para el subsistema de administración de cuentas del sistema de acceso al hogar.



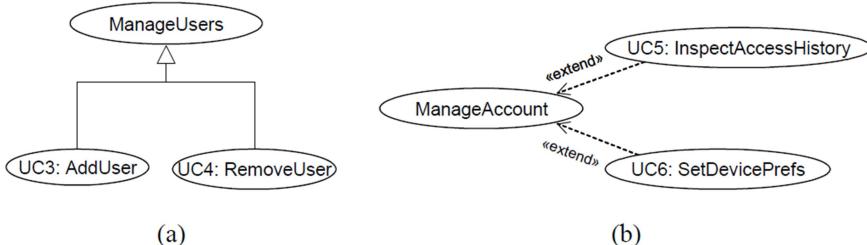
Aquí están los casos de uso representados en el diagrama:

- UC3: AddUser (Añadir Usuario)
 - El Arrendador puede iniciar este caso de uso para agregar un nuevo usuario al sistema.
- UC4: RemoveUser (Eliminar Usuario)
 - El Arrendador también puede iniciar este caso de uso para eliminar un usuario existente del sistema.
- UC5: InspectAccessHistory (Inspeccionar Historial de Accesos)
 - El Arrendatario tiene la capacidad de iniciar este caso de uso, lo que implica que pueden revisar el historial de accesos, posiblemente para verificar quién ha entrado y salido de su propiedad.
- UC6: SetDevicePrefs (Configurar Preferencias del Dispositivo)
 - El Arrendatario puede iniciar este caso de uso para configurar las preferencias de diversos dispositivos asociados con el sistema. Esto podría incluir la configuración de cerraduras inteligentes, termostatos, cámaras de seguridad, etc.

Todos estos casos de uso incluyen el caso de uso UC8: Login (Iniciar Sesión), lo que sugiere que para llevar a cabo cualquiera de las funciones de añadir o eliminar usuarios, inspeccionar el historial de accesos o establecer preferencias de dispositivos, los usuarios deben estar autenticados a través del proceso de inicio de sesión. Esto es una práctica estándar para asegurar que solo los usuarios autorizados puedan realizar acciones que afecten la seguridad y la configuración del sistema.

Las relaciones «initiate» (iniciar) y «participate» (participar) entre actores y casos de uso indican quién comienza la acción y quién está involucrado o afectado por ella. La relación «include» (incluir) indica que el inicio de sesión es una parte necesaria de todos los otros casos de uso, y cada vez que uno de esos casos de uso se activa, el caso de uso de inicio de sesión también se llevará a cabo como parte del proceso.

Más relaciones entre casos de uso: (a) Generalización de casos de uso; b) Casos de uso facultativo, denotados con el estereotipo «extendido».



A:

- El diagrama muestra una relación de generalización/especialización entre casos de uso. Aquí, "ManageUsers" actúa como un caso de uso general o base para otros dos casos de uso: "UC3: AddUser" y "UC4: RemoveUser".
- La relación se indica mediante una línea que se conecta desde los casos de uso específicos hasta el caso de uso base con un triángulo sin rellenar en el extremo de la conexión con el caso de uso base. Esto implica que "AddUser" y "RemoveUser" son especializaciones de "ManageUsers" y heredan o se relacionan estrechamente con este caso de uso más general.

B:

- El diagrama ilustra el uso de la relación «extend» en los casos de uso. "ManageAccount" es el caso de uso principal, y tiene dos casos de uso que se extienden desde él: "UC5: InspectAccessHistory" y "UC6: SetDevicePrefs".
- La relación «extend» se indica mediante una línea de puntos que se conecta de los casos de uso extendidos al caso de uso principal con la etiqueta «extend» y un triángulo abierto en el extremo de la conexión con el caso de uso principal.
- Esto sugiere que "InspectAccessHistory" y "SetDevicePrefs" son actividades opcionales que podrían realizarse como parte del "ManageAccount", pero no son obligatorias para la funcionalidad principal de "ManageAccount".

Matriz de trazabilidad

Figura: Matriz de trazabilidad de requisitos para casos de uso de acceso a hogares seguros estudiar.

| Req't | PW | UC1 | UC2 | UC3 | UC4 | UC5 | UC6 | UC7 | UC8 |
|----------|----|-----|-----|-----|-----|-----|-----|-----|-----|
| REQ1 | 5 | X | X | | | | | | |
| REQ2 | 2 | | | X | | | | | |
| REQ3 | 5 | X | | | | | | X | |
| REQ4 | 4 | X | | | | | | X | |
| REQ5 | 2 | X | X | | | | | | |
| REQ6 | 1 | | | X | X | | | | X |
| REQ7 | 2 | | | | | X | | X | |
| REQ8 | 1 | | | | X | | | X | |
| REQ9 | 1 | | | | X | | | X | |
| Max PW | | 5 | 2 | 2 | 2 | 1 | 5 | 2 | 1 |
| Total PW | | 15 | 3 | 2 | 2 | 3 | 9 | 2 | 3 |

Ponderación prioritaria (PW) indicada en el cuadro 2-1. (La trazabilidad continuó en la Figura 2-28).

La Figura muestra la **matriz de trazabilidad** que mapea los requisitos del sistema a los casos de uso. Su propósito es verificar que todos los requisitos estén cubiertos por los casos de uso y que ninguno de los casos de uso se haya creado sin motivo (es decir, sin un requisito del que se derivó).

Max PW (Peso de Prioridad Máximo):

- Esta fila indica la prioridad más alta entre los requisitos que están marcados para cada caso de uso.
- Para obtener el "Max PW" de un caso de uso, se observa cuál es el requisito marcado con la mayor prioridad (PW) en esa columna.

Total PW (Peso de Prioridad Total):

- Esta fila representa la suma de las prioridades de todos los requisitos marcados para cada caso de uso.
- Para calcular el "Total PW", se suman las prioridades (PW) de todos los requisitos marcados en esa columna.

Ficha de descripción de C.U.

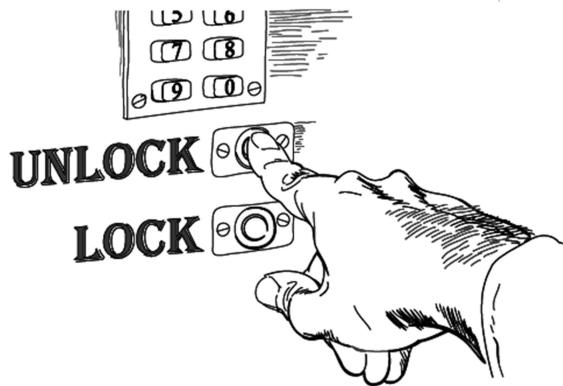
No debe confundir el diagrama de caso de uso con los casos de uso. El diagrama sirve únicamente para capturar una visión general de los servicios del sistema en una forma visual concisa. Resume las características del sistema y sus relaciones, sin detallar cómo debe operar cada característica. Los casos de uso son relatos textuales que detallan exactamente lo que sucede cuando un actor intenta obtener un servicio del sistema lo que da origen a la **Ficha de Descripción de Casos de Uso**.

Normalmente, primero elaboramos el escenario "normal", también llamado escenario principal de éxito, que asume que todo sale perfecto. Debido a que todo fluye sin complicaciones, este escenario generalmente no incluye condiciones o ramificaciones; fluye de manera lineal. Es solo una secuencia causal de pares de acción/reacción o estímulo/respuesta.

La Figura muestra el esquema del caso de uso.

| Use Case UC #: | Name / Identifier [verb phrase] |
|---|---|
| Related Requirements: | List of the requirements that are addressed by this use case |
| Initiating Actor: | Actor who <i>initiates</i> interaction with the system to accomplish a <i>goal</i> |
| Actor's Goal: | Informal description of the initiating actor's goal |
| Participating Actors: | Actors that will <i>help</i> achieve the goal or <i>need to know</i> about the outcome |
| Preconditions: | What is <i>assumed</i> about the state of the system before the interaction starts |
| Postconditions: | What are the <i>results</i> after the goal is achieved or abandoned; i.e., what must be true about the system at the time the execution of this use case is completed |
| Flow of Events for Main Success Scenario: | <p>→ 1. The initiating actor delivers an <i>action</i> or <i>stimulus</i> to the system (the arrow indicates the direction of interaction, to or from the system)</p> <p>← 2. The system's <i>reaction</i> or <i>response</i> to the stimulus; the system can also send a message to a participating actor, if any</p> <p>→ 3. ...</p> |
| Flow of Events for Extensions (Alternate Scenarios): | <p>What could go wrong? List the exceptions to the routine and describe how they are handled</p> <p>→ 1a. For example, actor enters invalid data</p> <p>← 2a. For example, power outage, network failure, or requested data unavailable</p> <p>...</p> |
| The arrows on the left indicate the direction of communication: → Actor's action; ← System's reaction | |

Escenario principal:



Los escenarios alternativos o extensiones en un caso de uso pueden resultar de:

- Entrada de datos inapropiada, como que el actor haga una elección incorrecta de elemento del menú (diferente de la que originalmente tenía la intención), o que el actor proporcione una identificación inválida.
- Incapacidad del sistema para responder como se desea, lo que puede ser una condición temporal o la información necesaria para formular la respuesta nunca puede estar disponible.

Casos de Uso Detallados

Los casos de uso detallados elaboran los casos de uso resumidos (Tabla 3).

Para el **caso de uso UC1 Desbloquear**, el escenario principal de éxito en forma abreviada puede parecer algo así:

Caso de Uso UC-1: Desbloquear

Requisitos Relacionados: REQ1, REQ3, REQ4 y REQ5 indicados en la Tabla 2-1

Actor Iniciador: Cualquiera de: Inquilino, Propietario

Objetivo del Actor: Desactivar el bloqueo para entrar y que la luz del espacio se encienda automáticamente.

Actores Participantes: LockDevice, LightSwitch, Timer

Precondiciones:

- El conjunto de claves válidas almacenadas en la base de datos y que la base de datos no debe estar vacía.
- El sistema muestra el menú de funciones disponibles; en la puerta las opciones del menú son "Bloquear" y "Desbloquear".

Postcondiciones: El timer activa el bloqueo automático y este ha comenzado la cuenta regresiva desde autoLockInterval.

Flujo de Eventos para el Escenario Principal de Éxito:

1. → El Inquilino/Propietario llega a la puerta y selecciona el ítem del menú "Desbloquear"
2. Incluir: AuthenticateUser (UC-7)
3. ← El sistema (a) señala al Inquilino/Propietario el estado del bloqueo, p.ej., "desarmado", (b) señala al DispositivoDeBloqueo para desactivar el bloqueo, y (c) señala al InterruptorDeLuz para encender la luz
4. ← El sistema señala al Temporizador para comenzar la cuenta regresiva del temporizador de bloqueo automático
5. → El Inquilino/Propietario abre la puerta, entra a la casa [y cierra la puerta y la bloquea]

- En el paso 5 anterior, la actividad de bloquear la puerta está entre corchetes porque esto se cubre bajo el caso de uso Bloquear, y no concierne a este caso de uso.
- Aunque las extensiones o escenarios alternativos no están listados en la descripción del UC-1, para cada uno de los pasos en el escenario principal de éxito debemos considerar qué podría salir mal.

Por ejemplo:

- En el Paso 1, el actor podría hacer una selección errónea del menú.
- Las excepciones durante la autenticación del actor se consideran relacionadas con el UC-7.
- En el Paso 5, el actor podría quedarse fuera por un momento, por ejemplo, saludando a un vecino.
- Por ejemplo, para abordar las excepciones en el Paso 5, podríamos considerar instalar un haz infrarrojo en la entrada que detecte cuando la persona lo cruza.

En el paso 2 del UC-1, reutilizo un caso de uso de "subrutina", AuthenticateUser, con la palabra clave "incluir", porque anticipo que esto también ocurrirá en otros casos de uso.

Aquí está el escenario principal para AuthenticateUser así como las excepciones, en caso de que algo salga mal:

Caso de Uso UC-7: AuthenticateUser (sub-caso de uso)

Requisitos Relacionados: REQ3, REQ4 indicados en la Tabla 2-1

Actor Iniciador: Cualquiera de: Inquilino, Propietario

Objetivo del Actor: Ser positivamente identificado por el sistema (en la interfaz de la puerta).

Actores Participantes: Campana de Alarma, Policía

Precondiciones:

- El conjunto de llaves válidas almacenadas en la base de datos del sistema no está vacío.
- El contador de intentos de autenticación es igual a cero.

Postcondiciones: Ninguna que valga la pena mencionar.

Flujo de Eventos para el Escenario Principal de Éxito:

- ← 1. El sistema solicita al actor su identificación, por ejemplo, una llave alfanumérica.
- → 2. Inquilino/Propietario proporciona una clave de identificación válida.
- ← 3. El sistema (a) verifica que la clave sea válida y (b) señala al actor la validez de la clave.

Flujo de Eventos para Extensiones (Escenarios Alternativos):

- 2a. Inquilino/Propietario ingresa una clave de identificación no válida
 - ← 1. El sistema (a) detecta el error, (b) marca un intento fallido y (c) señala al actor.
 - ← 1a. El sistema (a) detecta que el conteo de intentos fallidos supera el número máximo permitido, (b) señala para sonar la Campana de Alarma y (c) notifica al actor Policía de un posible allanamiento.
- → 2. Inquilino/Propietario proporciona una clave de identificación válida.
 - 3. Igual que en el paso 3 anterior.

1. La base de datos del sistema contiene al menos un conjunto de llaves válidas.
2. El contador de intentos de autenticación está en cero al inicio del proceso.
3. El sistema pide al usuario su identificación, como podría ser una clave alfanumérica.
4. El usuario proporciona la clave de identificación.
5. El sistema verifica la validez de la clave y, si es correcta, señala al usuario que la identificación ha sido exitosa.

Flujo Alternativo (Autenticación Fallida):

1. Si el usuario ingresa una clave no válida, el sistema detecta el error, registra un intento fallido y señala al usuario.
2. Si se excede el número máximo de intentos fallidos permitidos, el sistema activa la alarma y notifica a la autoridad policial de un posible intento de allanamiento.

Caso de prueba de aceptación

En el momento de escribir casos de uso detallados, también escribimos las correspondientes **pruebas de aceptación** del usuario. En el contexto de los **casos de uso**, un caso de prueba de aceptación del usuario es un procedimiento detallado que prueba:

- completamente un caso de uso o uno de sus flujos de eventos
- como el cliente debe ayudar a especificar las pruebas de aceptación
- como los casos de prueba deben diseñarse en torno a las tareas reales que el usuario deberá realizar
- la interacción de varios pasos
- la especificación de los datos de entrada y los resultados esperados.

Pasos detallados para caso de prueba del caso de uso UC-1.

| Identificador del Caso de Prueba | TC-1.01 |
|--|--|
| Caso de Uso Probado | UC-1, escenario principal de éxito y UC-7 |
| Criterios de Aprobación/Fallo | La prueba pasa si el usuario introduce una clave que está contenida en la base de datos, con menos de un número máximo permitido de intentos fallidos. |
| Datos de Entrada | Código numérico, identificador de la puerta |
| Procedimiento de Prueba | Resultado Esperado |
| Paso 1. Introduce un código incorrecto y un identificador de puerta válido | El sistema emite un bip para indicar fallo; registra el intento fallido en la base de datos; solicita al usuario que lo intente de nuevo. |
| Paso 2. Introduce el código correcto y el identificador de la puerta | El sistema parpadea una luz verde para indicar éxito; registra el acceso exitoso en la base de datos; desactiva el dispositivo de bloqueo. |

Una prueba de aceptación debe convencer al cliente de que el sistema funciona como se espera.

| | |
|---|--|
| Caso de Uso UC-2: | Bloqueo (Lock) |
| Requisitos Relacionados: | REQ1, REQ2 y REQ5 indicados en la Tabla 2-1 |
| Actor Iniciador: | Cualquiera de: Inquilino, Propietario o Temporizador |
| Meta del Actor: | Bloquear la puerta y apagar las luces automáticamente (?) |
| Actores Participantes: | Dispositivo de Bloqueo, Interruptor de Luz, Temporizador |
| Precondiciones: | El sistema siempre muestra el menú de funciones disponibles. |
| Postcondiciones: | La puerta está cerrada y el bloqueo activado & el temporizador de auto-bloqueo se reinicia. |
| Flujo de Eventos para el Escenario Principal de Éxito: | <p>→ 1. Inquilino/Propietario selecciona el ítem del menú “Bloquear”</p> <p>← 2. Sistema</p> <p>(a) señala afirmación, por ejemplo, “bloqueo activado,”</p> <p>(b) señala al Dispositivo de Bloqueo para armar el bloqueo (si aún no está armado),</p> <p>(c) señala al Temporizador para reiniciar el contador de auto-bloqueo, y</p> <p>(d) señala al Interruptor de Luz para apagar la luz (?) señala al Interruptor de Luz para apagar la luz (?)</p> |
| Flujo de Eventos para Extensiones (Escenarios Alternativos): | <p>2a. Sistema percibe que la puerta no está cerrada, así que el bloqueo no puede ser armado.</p> <p>← 1. Sistema (a) señala una advertencia de que la puerta está abierta, y (b) señala al Temporizador para iniciar el contador de alarma</p> <p>→ 2. Inquilino/Propietario cierra la puerta</p> <p>← 3. Sistema (a) percibe el cierre, (b) señala afirmación al Inquilino/Propietario, (c) señala al Dispositivo de Bloqueo para armar el bloqueo, (d) señala al Temporizador para reiniciar el contador de auto-bloqueo, y (e) señala al Temporizador para reiniciar el contador de alarma</p> |

1. El temporizador de bloqueo automático aparece como el actor iniciador y participante de este caso de uso. Esto se debe a que, si el tiempo de espera expira antes de que se reinicie el temporizador, el temporizador inicia automáticamente el caso de uso Bloquear, por lo que es un actor iniciador.
2. Como alternativa, si el usuario bloquea la puerta antes de que expire el tiempo de espera, el temporizador se reiniciará, por lo que también es un actor fuera del escenario.
3. También asumo que un solo sistema de temporizador puede manejar múltiples solicitudes simultáneas.
4. En el uso de la cerradura, es posible que el temporizador esté contando el tiempo transcurrido desde que se desarmó la cerradura.
5. Al mismo tiempo, tiempo, el sistema puede detectar que la puerta no está cerrada, por lo que puede iniciar el temporizador de alarma.
6. Si la puerta no se cierra dentro de un intervalo determinado, el sistema activa el actor AlarmBell y puede notificar al Actor Policía.
7. El temporizador es una fuente de estímulo externa en relación con el software, aunque formará parte del sistema final de hardware y software.

Caso de Uso UC-3: Añadir Usuario

Requisitos Relacionados: REQ6 indicado en la Tabla 2-1

Actor Iniciador: Propietario

Objetivo del Actor: Registrar nuevos residentes o eliminar residentes que se han ido en tiempo de ejecución.

Actores Participantes: Inquilino

Precondiciones: Ninguna que valga la pena mencionar. (Pero tener en cuenta que este caso de uso solo está disponible en el ordenador principal y no en el teclado de la puerta.)

Postcondiciones: Los datos modificados se guardan en la base de datos.

Flujo de Eventos para el Escenario de Éxito Principal:

1. → El Propietario selecciona el elemento del menú “Gestionar Usuarios”.
2. Identificación del Propietario: Incluir Inicio de Sesión (UC-8).
3. ← El Sistema: a. Muestra las opciones de actividades disponibles para el Propietario (incluyendo “Añadir Usuario” y “Eliminar Usuario”). b. Solicita al Propietario que haga una selección.
4. → El Propietario selecciona la actividad, como “Añadir Usuario”, e introduce los nuevos datos.
5. ← El Sistema: a. Guarda los nuevos datos en un almacenamiento persistente. b. Indica la finalización del proceso.

Flujo de Eventos para Extensiones (Escenarios Alternativos):

4a. La actividad seleccionada implica añadir nuevos usuarios: Incluir AñadirUsuario (UC-3).

4b. La actividad seleccionada implica eliminar usuarios: Incluir EliminarUsuario (UC-4).

1. El inquilino es un actor secundario para este caso de uso.
2. El inquilino introducirá su identificación (contraseña o una huella biométrica) durante el proceso de registro.
3. En UC-3 incluimos el caso de uso subordinado Login (UC-8), que no es lo mismo que AuthenticateUser, numerado UC-7. La razón es que UC-7 está diseñado para autenticar a las personas en la(s) entrada(s).
4. Por el contrario, la gestión de usuarios siempre se realiza desde el ordenador central, por lo que tenemos que diseñar un caso de uso completamente diferente.
5. La descripción detallada del caso de uso AddUser se dará en el Problema 2.19 y RemoveUser es similar a él. En la Tabla 3 presentamos UC-5: Inspeccionar el historial de acceso, que aborda aproximadamente REQ8 y REQ9 en la Tabla 1.

Caso de Uso UC-5: Inspeccionar Historial de Acceso

Requisitos Relacionados: REQ8 y REQ9 indicados en la Tabla 2-1

Actor Iniciador: Cualquiera de: Inquilino, Propietario

Objetivo del Actor: Examinar el historial de acceso para una puerta específica.

Actores Participantes: Base de Datos, Propietario

Precondiciones:

- El Inquilino/Propietario está actualmente registrado en el sistema y se le muestra un enlace "Ver Historial de Acceso".

Postcondiciones: Ninguna.

Flujo de Eventos para el Escenario Principal de Éxito:

1. → **Consulta del Historial:** El Inquilino/Propietario hace clic en el enlace "Ver Historial de Acceso".
2. ← **Criterios de Búsqueda:** El sistema solicita los criterios de búsqueda (por ejemplo, rango de tiempo, ubicación de la puerta, rol del actor, tipo de evento, etc.) o "Mostrar todos".
3. → **Especificación de Criterios:** El Inquilino/Propietario especifica los criterios de búsqueda y los envía.
4. ← **Consulta a la Base de Datos:** El sistema prepara una consulta a la base de datos que mejor se ajuste a los criterios de búsqueda del actor y recupera los registros de la Base de Datos.
5. → **Resultados de la Base de Datos:** La Base de Datos devuelve los registros coincidentes.
6. ← **Filtrado y Visualización:** El sistema (a) filtra adicionalmente los registros recuperados para que coincidan con los criterios de búsqueda del actor; (b) procesa los registros restantes para su visualización; y (c) muestra el resultado para la consideración del Inquilino/Propietario.
7. → **Investigación Adicional:** El Inquilino/Propietario navega, selecciona los registros "interesantes" (si los hay) y solicita una investigación más a fondo (con una descripción de la queja adjunta).
8. ← **Confirmación y Seguimiento:** El sistema (a) muestra solo los registros seleccionados y confirma la solicitud; (b) archiva la solicitud en la Base de Datos y le asigna un número de seguimiento; (c) notifica al Propietario.

1. Permite a un inquilino o propietario revisar el historial de acceso de una puerta específica.
2. Una vez que el usuario está autenticado en el sistema, puede solicitar y revisar registros de acceso basados en varios criterios de búsqueda como el rango de tiempo, la ubicación de la puerta y el tipo de evento.
3. El sistema consulta la base de datos con los criterios proporcionados, filtra y muestra los registros relevantes al usuario. El usuario puede entonces seleccionar registros específicos para una investigación más profunda, tras lo cual el sistema archiva la solicitud con un número de seguimiento para futuras referencias.

Matriz de Trazabilidad

REQ1: Mantener la puerta cerrada y autolockeo
 REQ2: Cerrar cuando se presiona "CERRAR"
 REQ3: Abrir cuando se proporciona una llave válida
 REQ4: Permitir errores pero prevenir ataques de diccionario
 REQ5: Mantener un registro histórico
 REQ6: Agregar/eliminar usuarios en tiempo de ejecución
 REQ7: Configurar las preferencias de activación del dispositivo
 REQ8: Inspeccionar el historial de acceso
 REQ9: Presentar consultas

UC1: Desbloquear
 UC2: Bloquear
 UC3: AñadirUsuario
 UC4: EliminarUsuario
 UC5: InspeccionarHistorialDeAcceso
 UC6: ConfigurarPreferenciasDelDispositivo
 UC7: AuthenticateUser
 UC8: Login

| Req't | PW | UC1 | UC2 | UC3 | UC4 | UC5 | UC6 | UC7 | UC8 |
|----------|----|-----|-----|-----|-----|-----|-----|-----|-----|
| REQ1 | 5 | X | X | | | | | | |
| REQ2 | 2 | | | X | | | | | |
| REQ3 | 5 | X | | | | | | X | |
| REQ4 | 4 | X | | | | | | X | |
| REQ5 | 2 | X | X | | | | | | |
| REQ6 | 1 | | | X | X | | | | X |
| REQ7 | 2 | | | | | X | | | X |
| REQ8 | 1 | | | | | X | | | X |
| REQ9 | 1 | | | | | X | | | X |
| Max PW | | 5 | 2 | 2 | 2 | 1 | 5 | 2 | 1 |
| Total PW | | 15 | 3 | 2 | 2 | 3 | 9 | 2 | 3 |

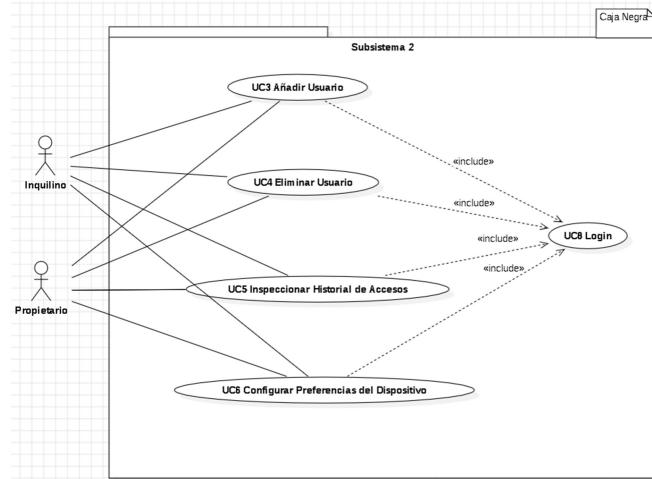
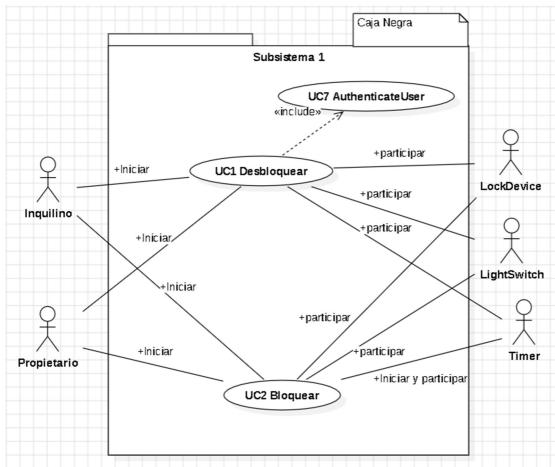
Modelo del Dominio

Es un modelo conceptual de todos los temas relacionados con un problema específico. En él se describen las distintas entidades, sus atributos, papeles y relaciones, además de las restricciones que rigen el dominio del problema.

- **Identificación de Conceptos:**
 - Una idea.
 - Es el elemento básico del pensamiento.
 - Conocimiento propio sobre una categoría de objetos o acontecimientos.
- **Atributos** de Conceptos: características o rasgos que describen una persona, una organización, un lugar o un elemento.
- **Asociaciones** de Conceptos: es una relación entre entidades, actores, y/o casos de uso, que describe la causa de la relación y las reglas que la rigen. Indica que un actor puede utilizar la funcionalidad del sistema.
- **Contratos:**
 - Precondiciones: Describe condiciones que debe cumplir el sistema para que se pueda iniciar el caso de uso.
 - Postcondiciones: Describe condiciones que debe cumplir el sistema para después que se ejecuta el caso de uso.

Generar el modelo de Dominio de UC1 Desbloquear y UC5 Inspeccionar Historial del Dispositivo

Caja negra



Caja Blanca

Generar a partir de la descripción de UC1 Desbloquear

Actores:

- Inquilino
- Propietario
- LockDevice
- LightSwitch
- Timer

Conceptos

Boundary (Entity-Boundary-Control) es un patrón arquitectónico utilizado en el diseño de software orientado a objetos, en el que los casos de uso categorizan las clases que componen el software según su responsabilidad.

Paso 1: Identificación de los conceptos límite

| Conceptos: | Responsabilidades |
|-------------------------|---|
| KeyCodeEntry | <ul style="list-style-type: none">Recibir y registrar los códigos de acceso ingresados por los usuarios. |
| StatusDisplay | <ul style="list-style-type: none">Mostrar el estado actual del sistema o resultado de las acciones como la validación de los códigos de acceso. |
| HouseholdDeviceOperator | <ul style="list-style-type: none">Operar y gestionar el estado de los dispositivos domésticos como cerraduras electrónicas y sistemas de iluminación. |

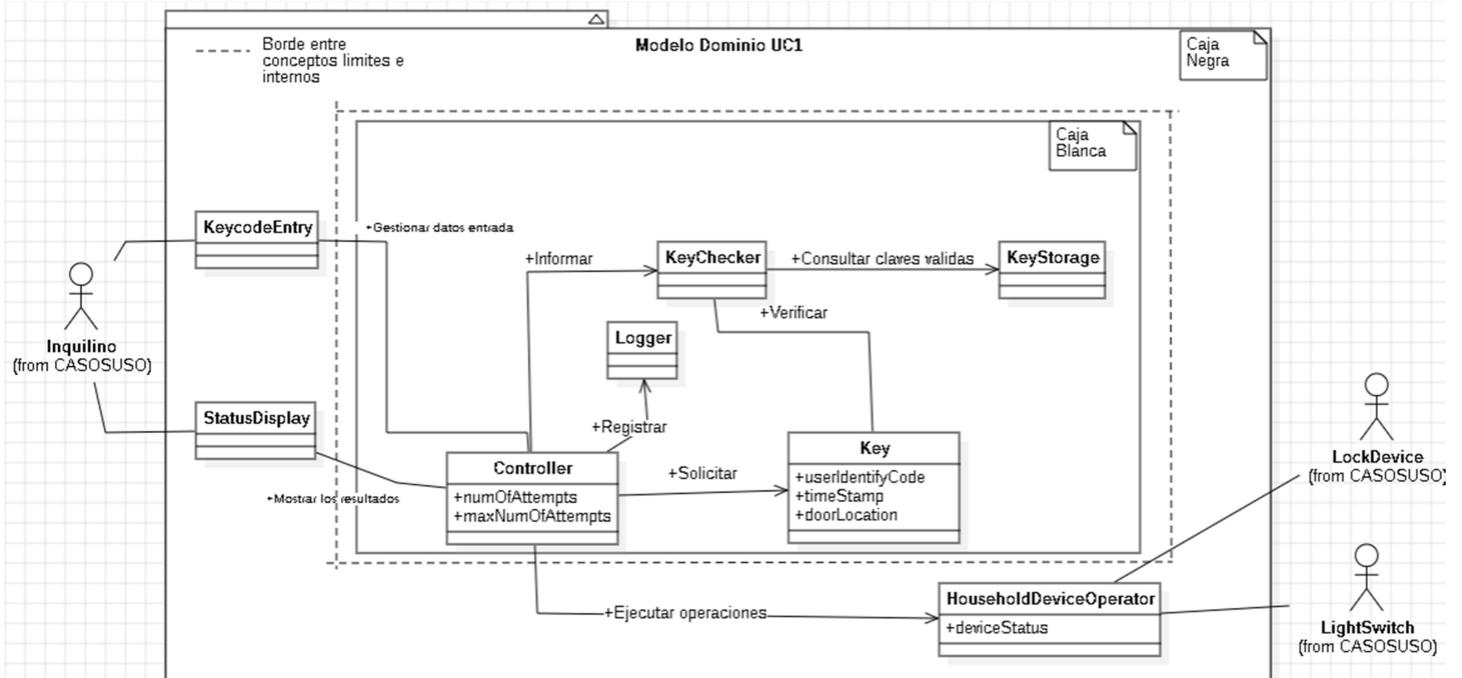
Paso 2: Identificación de los conceptos internos

| Conceptos: | Responsabilidades |
|------------|---|
| Controller | <ul style="list-style-type: none">Coordinar las acciones de todos los conceptos asociados con un caso de uso, un grupo lógico de casos de uso o todo el sistema, y delegar el trabajo a otros conceptos.Bloquear la entrada para denegar más intentos si hay demasiados intentos fallidos. |
| Key | <ul style="list-style-type: none">Contenedor para los datos de autenticación del usuario, como el código de acceso, la marca de tiempo, la identificación de la puerta, etc. |
| KeyChecker | <ul style="list-style-type: none">Verificar si el código ingresado por el usuario es válido o no. |
| KeyStorage | <ul style="list-style-type: none">Contenedor para la colección de claves válidas asociadas con puertas y usuarios. |
| Logger | <ul style="list-style-type: none">Registrar todas las interacciones con el sistema en almacenamiento persistente. |

Extraer las Asociaciones

| Par de Conceptos | Descripción de la Asociación | Nombre de la Asociación |
|--------------------------------------|---|-------------------------|
| KeyCodeEntry ↔ Controller | KeyCodeEntry recibe los códigos de los usuarios y los envía al Controller, el cual coordina las acciones necesarias según el código proporcionado. | Gestionar |
| StatusDisplay ↔ Controller | StatusDisplay muestra los resultados de las acciones del Controller, como el éxito o el fracaso de un intento de acceso. | Mostrar |
| HouseholdDeviceOperator ↔ Controller | HouseholdDeviceOperator ejecuta las operaciones sobre los dispositivos domésticos como cerraduras y luces, siguiendo las instrucciones del Controller. | Ejecutar |
| Key ↔ KeyChecker | Key proporciona los datos de autenticación necesarios para que KeyChecker verifique la validez de un código de acceso. | Verificar |
| KeyChecker ↔ KeyStorage | KeyChecker consulta a KeyStorage para verificar si el código proporcionado coincide con alguna clave válida almacenada. | Consultar |
| KeyChecker ↔ Controller | KeyChecker informa al Controller sobre la validez de un código de acceso ingresado, permitiendo al Controller tomar las acciones correspondientes. | Informar |
| Controller <-> keychecker | Controller informa a keychecker que revise los códigos y saber si son válidos o no | |
| Key ↔ Controller | El Controller solicita y utiliza la información contenida en Key para determinar la validez del acceso y registrar el intento. | Solicitar |
| Logger ↔ Controller | Logger registra las decisiones y acciones tomadas por el Controller, así como cualquier otro evento significativo, para un seguimiento y auditoría posteriores. | Registrar |

Diagrama de modelo de dominio UC1



Matriz de Trazabilidad

| | | Domain Concepts | | | | | | | | | | | | | | |
|----------|----|-----------------|---------------|--------------|-----|------------|------------|-------------------------|----------------|---------------|---------------|-----------|----------|--------------------|----------|----------------------|
| Use Case | PW | Controller-SS1 | StatusDisplay | KeycodeEntry | Key | KeyStorage | KeyChecker | HouseholdDeviceOperator | Controller-SS2 | SearchRequest | InterfacePage | PageMaker | Archiver | DatabaseConnection | Notifier | InvestigationRequest |
| UC1 | 15 | X | X | X | X | | | X | | | | | | | | |
| UC2 | 3 | X | X | | | | | X | | | | | | | | |
| UC3 | 2 | | | | | | | | X | X | X | | X | | | |
| UC4 | 2 | | | | | | | | X | X | X | | X | | | |
| UC5 | 3 | | | | | | | | X | X | X | X | X | X | X | |
| UC6 | 9 | | | | | | | | X | X | X | | X | | | |
| UC7 | 2 | X | X | | X | X | X | | | | | | | | | |
| UC8 | 3 | | | | | | | | X | X | X | | X | | | |

UC1: Desbloquear
 UC2: Bloquear
 UC3: AñadirUsuario
 UC4: EliminarUsuario
 UC5: InspeccionarHistorialDeAcceso
 UC6: ConfigurarPreferenciasDelDispositivo
 UC7: AuthenticateUser
 UC8: Login

Pruebas de Aceptación

Tipos de Pruebas:

Las **pruebas de caja blanca** explotan la estructura dentro del programa (asume que el código del programa está disponible) es un enfoque en el comportamiento interno.

Las **pruebas de caja negra** exploran el espacio de entrada de la funcionalidad definida por una especificación de interfaz y es un enfoque en el comportamiento externo.

- Prueba con la clave válida de un inquilino actual (Input data) en su propio apartamento (éxito es el Resultado esperado).
- Prueba con la clave válida de un inquilino actual en el apartamento de otra persona (fracaso)
- Prueba con una clave no válida en cualquier apartamento (fracaso)
- Prueba con la clave de un inquilino eliminado en su apartamento anterior (fracaso)
- Prueba con la clave válida de un inquilino recién agregado en su propio apartamento (éxito)

Caso de Prueba para Verificar el Acceso con Llave

Identificador del Caso de Prueba: TC-1

Caso de Uso Probado:

- UC-1 (Acceso de Usuario al Sistema)
- UC-7 (Especificación no proporcionada, pero asumiremos que está relacionado con el manejo de la base de datos de llaves)

Criterios de Éxito/Fracaso:

- Éxito: El usuario ingresa una llave válida, que está contenida en la base de datos, dentro del número máximo permitido de intentos fallidos.
- Fracaso: El usuario ingresa una clave no válida o supera el número máximo de intentos fallidos permitidos.

Input Data:

- Código numérico (clave)
- Identificador de la puerta

Procedimiento de Prueba:

Paso 1:

- Acción: Ingresar un código numérico incorrecto y un identificador de puerta válido.
- Resultado Esperado:
 - El sistema emite un pitido indicando el fallo.
 - Se registra el intento fallido en la base de datos.
 - El sistema solicita al usuario que lo intente nuevamente.

Paso 2:

- Acción: Ingresar el código numérico correcto y el identificador de puerta correspondiente.
- Resultado Esperado:
 - El sistema parpadea una luz verde para indicar éxito.
 - Se registra el acceso exitoso en la base de datos.
 - El sistema desactiva el dispositivo de bloqueo permitiendo el acceso.

Pasos Adicionales:

Paso 3:

- Acción: Intentar nuevamente con un código incorrecto después de un intento fallido.
- Resultado Esperado:
 - El sistema debería permitir un número limitado de reintentos antes de tomar una acción adicional (por ejemplo, bloquear temporalmente el acceso o alertar al personal de seguridad).

Paso 4:

- Acción: Ingresar una llave válida después de alcanzar el límite de intentos fallidos.
- Resultado Esperado:

- Dependiendo de la política de seguridad, el sistema puede negar el acceso y requerir una acción adicional para restablecer el estado o permitir el acceso si la llave es válida.

Casos de prueba

Este caso de prueba se diseñó para validar tanto la funcionalidad de reconocimiento de llaves válidas como la seguridad en términos de manejo de intentos fallidos, abarcando las condiciones normales de uso y posibles escenarios de seguridad.

Cobertura de Pruebas mide el grado en que la especificación o el código de un programa de software ha sido ejercitado por pruebas.

- **Pruebas de caja negra**
 - pruebas de equivalencia
 - pruebas de límites
- **Pruebas de caja blanca**
 - pruebas de flujo de control
 - pruebas basadas en el estado

Caso de Prueba de Equivalencia

Para crear un caso de prueba de equivalencia que abarque el Caso de Uso 1 (Acceso de Usuario al Sistema) como el Caso de Uso 7 (Manejo de la base de datos de claves), se dividen los posibles valores de entrada en grupos o clases de equivalencia, se selecciona al menos un valor representativo de cada grupo para las pruebas. Este caso de prueba se asegura de que el sistema maneja adecuadamente las llaves válidas y las no válidas, así como las cuentas de usuarios activos frente a los deshabilitados, garantizando que el acceso solo se concede bajo las condiciones correctas.

Caso de Prueba de Equivalencia: Acceso con Llave

Identificador del Caso de Prueba: TC-2

Caso de Uso Probado:

- UC-1 (Acceso de Usuario al Sistema)
- UC-7 (Manejo de la base de datos de claves)

Criterios de Éxito/Fracaso:

- Éxito: El sistema concede acceso cuando se ingresa una llave válida contenida en la base de datos.
- Fracaso: El sistema no concede acceso cuando se ingresa una llave no válida o no contenida en la base de datos.

Clases de Equivalencia:

- Llaves válidas contenidas en la base de datos (deberían conceder acceso).
- Llaves no válidas no contenidas en la base de datos (no deberían conceder acceso).
- Llaves válidas de usuarios deshabilitados o eliminados (no deberían conceder acceso).

Input Data:

- Código numérico (clave)
- Identificador de puerta

Procedimiento de Prueba:

Paso 1 (Clase de Equivalencia Válida):

- Acción: Ingresar una llave válida y un identificador de puerta válido para un usuario activo.
- Resultado Esperado:

- El sistema parpadea una luz verde para indicar éxito.
- Se registra el acceso exitoso en la base de datos.
- El sistema desactiva el dispositivo de bloqueo permitiendo el acceso.

Paso 2 (Clase de Equivalencia Inválida - Llave No Válida):

- **Acción:** Ingresar una llave no válida y un identificador de puerta válido.
- **Resultado Esperado:**
 - El sistema emite un pitido para indicar fallo.
 - No se registra el acceso en la base de datos.
 - El dispositivo de bloqueo permanece activo.

Paso 3 (Clase de Equivalencia Inválida - Usuario Deshabilitado):

- **Acción:** Ingresar una llave válida para un usuario deshabilitado y un identificador de puerta válido.
- **Resultado Esperado:**
 - El sistema emite un pitido para indicar fallo.
 - Se registra el intento fallido en la base de datos.
 - El dispositivo de bloqueo permanece activo.

Caso de Prueba de Límites

Este caso de prueba verifica específicamente el comportamiento del sistema cuando se ingresa una llave válida justo en el límite del número de intentos permitidos, así como cuando se excede este límite. Es crucial para confirmar que las políticas de seguridad relacionadas con los intentos fallidos de acceso se implementan y funcionan como se espera.

Caso de Prueba de Límites: Acceso con Llave en el Borde del Límite de Intentos

Identificador del Caso de Prueba: TC-3

Caso de Uso Probado:

UC-1 (Acceso de Usuario al Sistema)

UC-7 (Manejo de la base de datos de llaves y control de acceso)

Criterios de Éxito/Fracaso:

- **Éxito:** El usuario ingresa una llave válida en el último intento permitido, y el sistema concede acceso.
- **Fracaso:** El usuario ingresa una llave válida después de superar el número máximo de intentos fallidos, y el sistema no concede acceso.

Input Data:

- Código numérico (clave válida)
- Identificador de puerta correcto
- Número de intentos fallidos previos igual al límite menos uno.

Procedimiento de Prueba:

Paso 1:

- **Acción:** Ingresar repetidamente un código numérico incorrecto hasta alcanzar el número máximo permitido de intentos fallidos menos uno.
- **Resultado Esperado:**
 - El sistema emite un pitido en cada intento fallido.
 - Se registra cada intento fallido en la base de datos.

Paso 2:

- **Acción:** En el último intento permitido, ingresar el código numérico correcto y el identificador de puerta correspondiente.
- **Resultado Esperado:**
 - El sistema parpadea una luz verde para indicar éxito.
 - Se registra el acceso exitoso en la base de datos.

- El sistema desactiva el dispositivo de bloqueo permitiendo el acceso.

Paso 3 (Prueba de Límites Adicional):

- **Acción:** Después de un bloqueo temporal debido al exceso de intentos fallidos, intentar acceder nuevamente con la llave válida.
- **Resultado Esperado:**
 - El sistema puede requerir una acción de restablecimiento antes de permitir más intentos.
 - Si se permite el intento, el sistema debería conceder acceso con una llave válida.

Pasos Adicionales:

Paso 4:

- **Acción:** Superar intencionalmente el número máximo de intentos fallidos y luego intentar acceder con una llave válida.
- **Resultado Esperado:**
 - El sistema no concede acceso y requiere intervención para restablecer el estado de acceso o activar un protocolo de seguridad.

Caso de Prueba de flujo de control

Se debe analizar la lógica interna del código y crear pruebas que cubran todas las posibles rutas de ejecución relevantes para esos casos de uso. Estos pasos cubren el flujo de control principal y las condiciones alternativas para ambos casos de uso, garantizando que todas las rutas lógicas se prueben. Estos casos de prueba ayudan a identificar problemas en la lógica de flujo de control que podrían no ser evidentes a través de pruebas funcionales o de interfaz de usuario.

Caso de Prueba de Flujo de Control: Proceso de Acceso y Gestión de Llaves

Identificador del Caso de Prueba: TC-4

Casos de Uso Probados:

- UC-1 (Acceso de Usuario al Sistema)
- UC-7 (Manejo de la base de datos de claves)

Criterios de Éxito/Fracaso:

- **Éxito:** Todas las rutas de ejecución posibles son ejercitadas, y el código se comporta como se espera en cada ruta.
- **Fracaso:** Una ruta de ejecución no se comporta como se espera, indicando la presencia de un defecto.

Procedimiento de Prueba:

Para UC-1:

Paso 1 (Ruta de Éxito):

- **Acción:** Ingresar un ID de usuario y contraseña válidos.
- **Resultado Esperado:** El sistema valida las credenciales y concede acceso a la biblioteca personal.

Paso 2 (Ruta de Fallo):

- **Acción:** Ingresar un ID de usuario y contraseña inválidos.
- **Resultado Esperado:** El sistema no valida las credenciales y deniega el acceso, posiblemente con un mensaje de error.

Paso 3 (Ruta de Intentos Múltiples):

- **Acción:** Ingresar credenciales inválidas varias veces hasta alcanzar el límite de intentos fallidos.
- **Resultado Esperado:** El sistema deniega el acceso y aplica una restricción adicional, como un bloqueo temporal.

Para UC-7:

Paso 1 (Ruta de Éxito):

- **Acción:** Ingresar un código de llave válido para una puerta específica.
- **Resultado Esperado:** El sistema valida la llave, actualiza la base de datos y permite el acceso.

Paso 2 (Ruta de Llave no Reconocida):

- **Acción:** Ingresar un código de llave que no está en la base de datos.

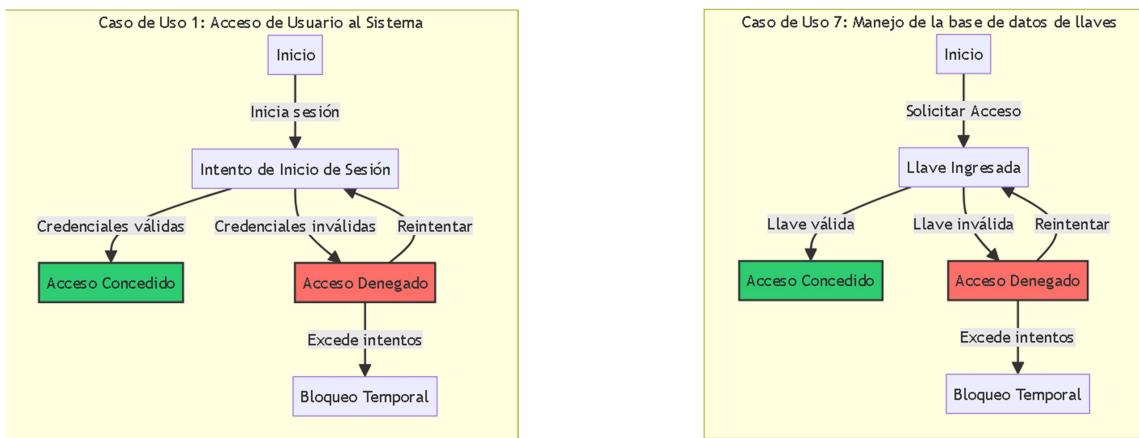
Resultado Esperado: El sistema no reconoce la llave, registra el intento fallido y deniega el acceso.

Paso 3 (Ruta de Usuario Deshabilitado):

- **Acción:** Ingresar un código de llave válido para un usuario deshabilitado.
- **Resultado Esperado:** El sistema reconoce que el usuario está deshabilitado, registra el intento y deniega el acceso.

Paso 4 (Ruta de Llave Válida Post-Límite de Intentos):

- **Acción:** Intentar acceder con una llave válida después de exceder el número de intentos permitidos.
- **Resultado Esperado:** El sistema registra el acceso válido, pero requiere una acción adicional debido a los intentos fallidos previos.



Caso de Prueba de estado

Definen un conjunto de estados abstractos que una unidad de software (objeto) puede tomar y prueban el comportamiento de la unidad comparando sus estados reales con los estados esperados.

Caso de Prueba de Estado de Control para el Caso de Uso UC-1 y UC-7

Caso de Uso UC-1: Desbloquear

Identificador del Caso de Prueba: TC-StateControl-UC1-01

Objetivo del Caso de Prueba: Confirmar que el sistema gestiona correctamente el estado de desbloqueo, y la luz se enciende automáticamente después de la autenticación exitosa.

Precondiciones:

- El conjunto de claves válidas almacenadas en la base de datos del sistema no está vacío.
- El sistema muestra el menú de funciones disponibles.

Pasos de Prueba:

1. El usuario selecciona la opción "Desbloquear".
2. El sistema realiza el sub-caso de uso "AuthenticateUser" (UC-7).
3. Tras la autenticación exitosa, el sistema desactiva el bloqueo y enciende la luz.

Resultados Esperados:

- El sistema transita del estado "Locked" al estado "Accepting" y finalmente al estado "Unlocked" después de la autenticación exitosa.
- El sistema inicia la cuenta regresiva del temporizador de bloqueo automático después del desbloqueo.

Caso de Uso UC-7: AuthenticateUser

Identificador del Caso de Prueba: TC-StateControl-UC7-01

Objetivo del Caso de Prueba: Verificar que el sistema gestiona correctamente los estados de autenticación del usuario.

Precondiciones:

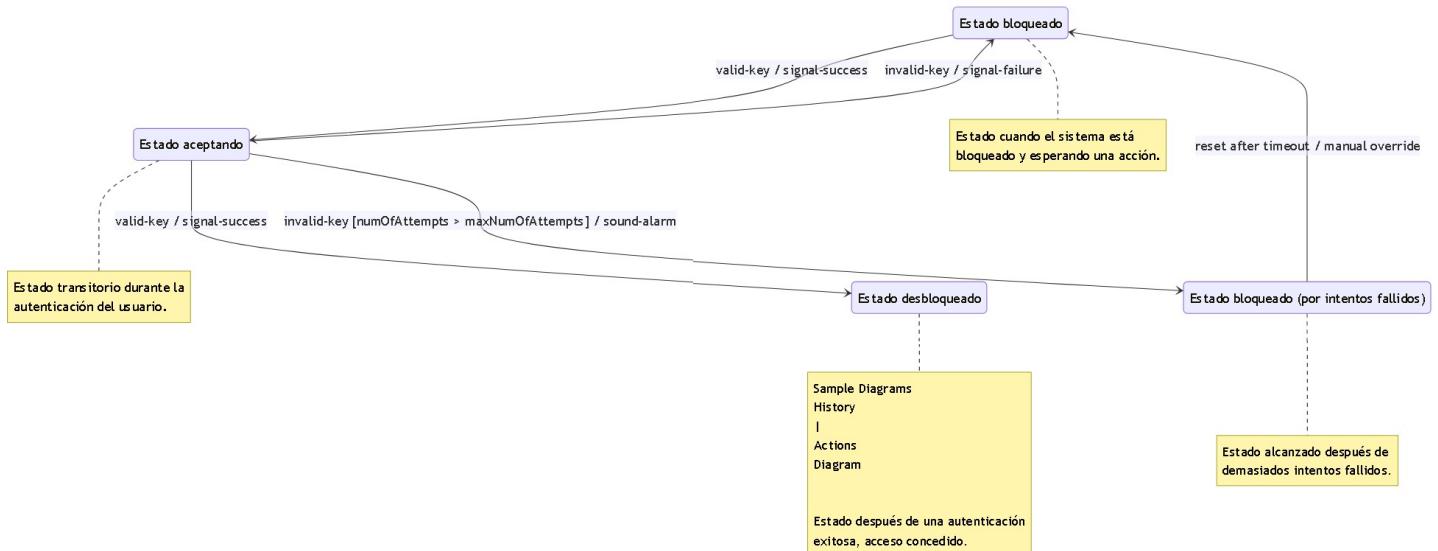
- El conjunto de claves válidas almacenadas en la base de datos del sistema no está vacío.
- El contador de intentos de autenticación es igual a cero.

Pasos de Prueba:

1. El sistema solicita al usuario su identificación.
2. El usuario proporciona una clave de identificación.
3. El sistema verifica la validez de la clave.

Resultados Esperados:

- Si la clave es válida, el sistema permanece en el estado "Accepting".
- Si la clave es inválida y se supera el número máximo de intentos, el sistema transita al estado "Blocked".



Pruebas unitarias

Se centran en la funcionalidad aislada de cada componente (por ejemplo, un método o clase). Los objetos ficticios (mocks) o sustitutos de prueba (stubs) se utilizan para simular la base de datos o cualquier otro componente con el que la unidad bajo prueba deba interactuar. Esto permite verificar el comportamiento esperado sin la necesidad de dependencias externas, lo cual es esencial para validar la lógica de negocio y el manejo de estados dentro del software. Estos casos de prueba están diseñados para cubrir tanto el flujo de éxito como el flujo de error en los procesos de desbloqueo y autenticación. Se asume que se utilizarán objetos ficticios o sustitutos de prueba para simular la base de datos de claves y otros componentes externos, permitiendo así aislar la lógica de los casos de uso para las pruebas.

UC-1: Desbloquear y UC-7: AuthenticateUser.

Pruebas Unitarias para UC-1: Desbloquear

Caso de Prueba Unitaria: Desbloqueo Exitoso

Identificador del Caso de Prueba: PU-UC1-01

Caso de Uso Probado: UC-1: Desbloquear

Criterios de Éxito/Fracaso:

Éxito: El sistema desbloquea correctamente y permite el acceso.

Fracaso: El sistema no desbloquea a pesar de una autenticación exitosa.

Input Data:

- Clave válida de usuario.

Procedimiento de Prueba:

- Inicializar el sistema en estado "bloqueado".
- Simular la entrada de una clave válida de usuario.

- Confirmar que el sistema transita al estado "desbloqueado".
- Verificar que el sistema permite el acceso y enciende la luz automáticamente.
- Caso de Prueba Unitaria: Fallo en Desbloqueo por Clave Inválida

Identificador del Caso de Prueba: PU-UC1-02

Procedimiento de Prueba:

- Inicializar el sistema en estado "bloqueado".
- Simular la entrada de una clave inválida de usuario.
- Verificar que el sistema permanece en estado "bloqueado".

Pruebas Unitarias para UC-7: AuthenticateUser

Caso de Prueba Unitaria: Autenticación Exitosa

Identificador del Caso de Prueba: PU-UC7-01

Caso de Uso Probado: UC-7: AuthenticateUser

Criterios de Éxito/Fracaso:

Éxito: El usuario es autenticado exitosamente.

Fracaso: El sistema rechaza una clave válida.

Input Data:

- Clave válida de usuario.

Procedimiento de Prueba:

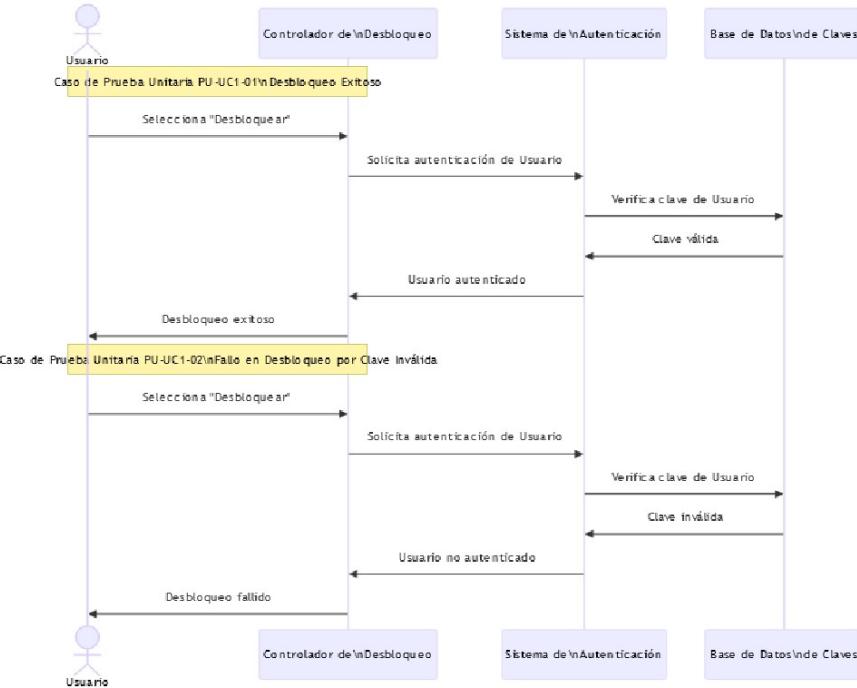
- Inicializar el sistema con contador de intentos en cero.
- Simular la entrada de una clave válida de usuario.
- Verificar que el sistema autentica positivamente al usuario.

Caso de Prueba Unitaria: Autenticación Fallida y Bloqueo

Identificador del Caso de Prueba: PU-UC7-02

Procedimiento de Prueba:

- Inicializar el sistema con contador de intentos en cero.
- Simular entradas consecutivas de claves inválidas hasta superar el número máximo de intentos permitidos.
- Hay que confirmar que el sistema incrementa el contador de intentos con cada entrada inválida.
- Verificar que el sistema activa el estado "bloqueado" al superar el número máximo de intentos.



Pruebas de Integración

- Fase en el desarrollo del software en la que se combinan y prueban unidades individuales de código como un grupo.
- Estas pruebas se centran en la interfaz y el flujo de datos entre las unidades para garantizar que trabajen juntas correctamente.

Pruebas de Integración Vertical

Se combinan y prueban unidades de forma secuencial de abajo hacia arriba o de arriba hacia abajo siguiendo la arquitectura del sistema.

Por ejemplo, en un enfoque de abajo hacia arriba, se comienza por las unidades de menor nivel (como funciones o clases individuales), y luego se agregan gradualmente las unidades de nivel superior (como módulos o grupos de clases) que dependen de estas.

Este método permite la detección temprana de defectos en las interfaces y en la interacción entre unidades y subsistemas. Es especialmente útil en situaciones en las que el software se desarrolla en capas y se quiere probar la funcionalidad incrementalmente.

Pruebas de Integración Horizontal

Se combinan y prueban unidades que operan en el mismo nivel de la jerarquía del sistema.

Este enfoque se utiliza a menudo para sistemas que tienen una arquitectura bien definida en capas o módulos, donde cada capa o módulo tiene una serie de unidades que trabajan juntas en el mismo nivel de abstracción.

Las pruebas se centran en asegurarse de que todos los módulos dentro de una capa interactúen correctamente entre sí.

Se utilizan "drivers" y "stubs" (**JUnit**) para simular las unidades de nivel superior e inferior que aún no se han integrado o desarrollado.

Pruebas de Integración Horizontal para UC-1 y UC-7 probar la interacción entre el sistema de autenticación y el mecanismo de desbloqueo.

Ejemplo de Caso de Prueba de Integración Horizontal:

Prueba de Integración del Sistema de Autenticación y Desbloqueo

Precondiciones: El sistema tiene una base de datos de claves con claves válidas e inválidas.

Proceso de Prueba:

- Verificar que el sistema de autenticación valida las claves correctamente cuando se intenta desbloquear.
- Hay que confirmar que después de una autenticación exitosa, el sistema de desbloqueo permite el acceso.

Resultados Esperados:

- El flujo de trabajo entre la autenticación y el desbloqueo funciona sin problemas y todas las transiciones de estado son como se esperan.

Ejemplo de Caso de Prueba de Integración Vertical:

Prueba de Integración del Flujo Completo de Autenticación y Desbloqueo

Precondiciones: El sistema tiene una base de datos de claves con claves válidas e inválidas y está en estado inicial bloqueado.

Proceso de Prueba:

- Un usuario intenta desbloquear utilizando la interfaz de usuario proporcionando una clave.
- El sistema procesa la solicitud pasando por las capas de controlador, lógica de negocio, y llegando hasta la base de datos para verificar la clave.
- Después de la autenticación, el sistema realiza el desbloqueo si la clave es válida.

Resultados Esperados:

- El flujo completo desde la interfaz de usuario hasta la base de datos funciona como se espera, y el sistema maneja correctamente los estados de bloqueo/desbloqueo.
- Para cada uno de estos casos de prueba de integración, usarías una mezcla de pruebas unitarias, mocks y stubs para simular las partes del sistema que aún no se han desarrollado o que no están disponibles durante la fase de prueba.

Revisiones de código usando Git

Clonar el Repositorio: Primero, clonas el repositorio de Git a tu máquina local.

`git clone [URL del repositorio]`

Crear una Nueva Rama: Antes de hacer cambios, creas una nueva rama para tu característica o corrección de errores.

`git checkout -b [nombre-de-la-nueva-rama]`

Hacer Cambios: Realizas los cambios en tu código localmente y los confirmas en tu rama.

`git add . git commit -m "Descripción de los cambios realizados"`

Subir la Rama al Repositorio Remoto: Empujas tu rama y los cambios al repositorio remoto.

`git push origin [nombre-de-la-nueva-rama]`

Crear una Pull Request o Merge Request: En la interfaz de usuario de GitLab, GitHub o Bitbucket, creas una solicitud de merge/pull request desde tu rama hacia la rama principal o la rama de desarrollo.

Revisión por Parte de Colegas: Otros desarrolladores revisarán tu código, dejando comentarios y sugerencias. En GitLab, esto se hace a través de la funcionalidad de Merge Requests.

Realizar Cambios Basados en Comentarios: Si es necesario, harás cambios adicionales en tu código basados en la retroalimentación.

`git add . git commit -m "Cambios basados en la revisión de código" git push origin [nombre-de-la-nueva-rama]`

Aprobación y Fusión: Una vez que tu código ha sido revisado y aprobado, un mantenedor del repositorio o alguien con permiso de merge puede fusionar tu rama al branch principal.

Cerrar la Merge Request: Después de la fusión, la Merge Request asociada se puede cerrar.

Eliminar la Rama Remota: Si es una política del equipo, la rama de características puede ser eliminada del repositorio remoto después de la fusión.

`git push origin --delete [nombre-de-la-nueva-rama]`

Plantilla

Caso de uso UC-#: Nombre / Identificador [frase verbal]

Requisitos relacionados: Lista de los requisitos que aborda este caso de uso

Actor iniciador: Actor que inicia la interacción con el sistema para lograr un objetivo

Objetivo del actor: Descripción informal del objetivo del actor iniciador

Actores participantes: Actores que ayudarán a lograr el objetivo o necesitan conocer el resultado

Condiciones previas: Lo que se supone sobre el estado del sistema antes de que comience la interacción.

Postcondiciones: ¿Cuáles son los resultados después de que se alcanza o abandona el objetivo? es decir, lo que debe ser verdadero sobre el sistema en el momento en que se completa la ejecución de este caso de uso

Flujo de eventos para el escenario de éxito principal:

1. El actor iniciador entrega una acción o estímulo al sistema (la flecha indica la dirección de la interacción, hacia o desde el sistema)
2. La reacción o respuesta del sistema al estímulo; El sistema también puede enviar un mensaje a un actor participante, si lo hay
3. ...

Flujo de eventos para extensiones (escenarios alternativos):

¿Qué podría salir mal? Enumere las excepciones a la rutina y describa cómo se manejan

- 1a. Por ejemplo, el actor introduce datos no válidos
- 2a. Por ejemplo, corte de energía, falla de red o datos solicitados no disponibles

...

Las flechas de la izquierda indican la dirección de la comunicación: ->Acción del actor; <-Reacción del sistema

Proceso de negocio

1. Autenticación del Usuario (Use Case UC-7 "AuthenticateUser"):

Inicio: El actor (inquilino o propietario) inicia el proceso de autenticación en la interfaz del sistema ubicada en la puerta.

Entrada del Actor: El sistema solicita al actor que proporcione una forma de identificación, como una clave alfanumérica.

Validación: El sistema verifica si la clave proporcionada es válida.

Feedback al Actor: Si la clave es válida, el sistema notifica al actor de la validez.

2. Manejo de Errores de Autenticación:

Intentos Fallidos: Si el actor introduce una clave incorrecta, el sistema detecta el error, marca el intento fallido y notifica al actor.

Excesivos Intentos Fallidos: Si el número de intentos fallidos supera el máximo permitido, el sistema activa una alarma y notifica a la policía de un posible intento de intrusión.

3. Escenario de Bloqueo Automático (No especificado pero mencionado en el contexto):

Temporizador de Auto-bloqueo: Un temporizador de bloqueo automático asegura que la puerta se bloquee tras un intervalo de tiempo definido, independientemente de la actividad del actor.

4. Escenarios de Extensión:

Desbloqueo Manual: Se especifican los comportamientos del sistema si la puerta se desbloquea manualmente con una llave física.

Interacciones Externas: El sistema puede manejar situaciones donde el actor se retarda al ingresar, por ejemplo, al detenerse a hablar con un vecino.

5. Desarrollo del Caso de Uso Principal (Use Case UC-1 "Lock"):

Este proceso de negocio implica la interacción con el sistema para bloquear la puerta, pero los detalles específicos no se proporcionaron en la información anterior.

6. Pruebas de Aceptación del Usuario:

Preparación de Pruebas: Se redactan pruebas de aceptación detalladas que incluyen todos los flujos de eventos de un caso de uso para asegurar que el sistema funciona según lo requerido.

Enfoque del Usuario: Las pruebas se centran en las acciones del usuario en lugar de en los procesos internos del sistema.

Escenarios de sistema

Caso de prueba es una elección particular de valores de entrada que se utilizarán en la prueba de un programa y los valores de salida esperados. Una prueba es una colección finita de casos de prueba.

Por ejemplo, para el requisito REQ3, el cliente puede sugerir estos casos de prueba:

- Prueba con la clave válida de un inquilino actual en su propio apartamento (aprueba)
- Prueba con la clave válida de un inquilino actual en el apartamento de otra persona (falla)
- Prueba con una clave inválida en cualquier apartamento (falla)
- Prueba con la clave de un inquilino que ha sido retirado en su anterior apartamento (falla)
- Prueba con la clave válida de un inquilino recién agregado en su apartamento (aprueba)

El requisito REQ7 permite al usuario configurar las preferencias para activar varios dispositivos domésticos en respuesta a diferentes eventos. Las preferencias se establecerían utilizando una interfaz de usuario Figura 2. Esto no es para abogar por el diseño de la interfaz de usuario en esta etapa temprana del desarrollo del proyecto.

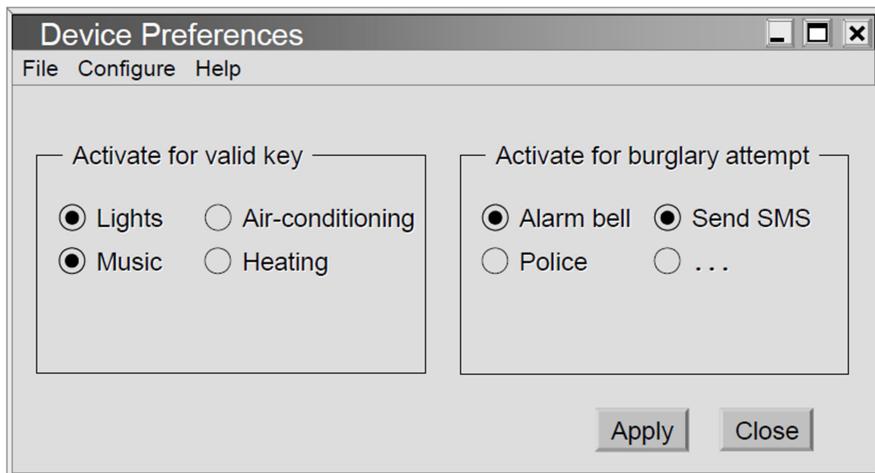
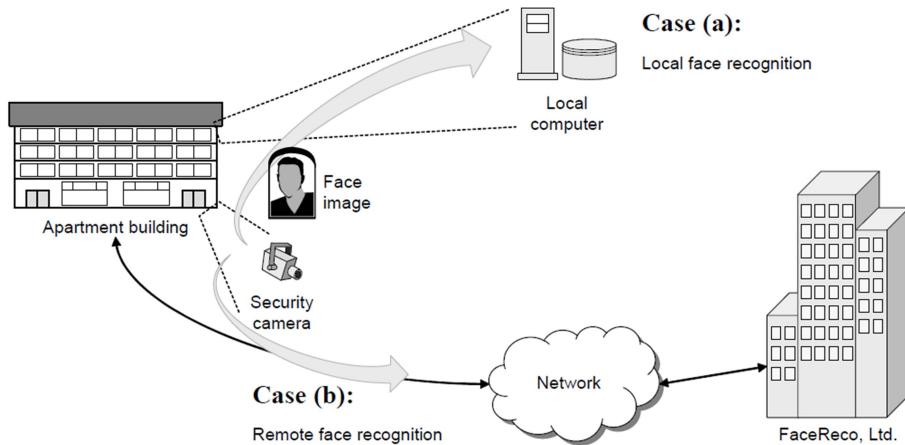


Figura 2: Visualización de la configuración de preferencias para el control de dispositivos domésticos.

Modificaciones:

Considera una variación del sistema de control de acceso domiciliario que se utilizará en un edificio de apartamentos o una comunidad de edificios de apartamentos, en lugar de una casa unifamiliar. La administración exige la identificación del usuario basada en el reconocimiento facial, en lugar de las claves de contraseña alfanuméricas. Hablando en términos generales, un sistema de reconocimiento facial funciona tomando una imagen del rostro de una persona ("foto de frente"), compara esa imagen con las caras conocidas y produce un resultado booleano: usuario "autorizado" o "no autorizado". Aquí hay dos variaciones (ver Figura 2-16):

Figura: Casos alternativos de reconocimiento facial para el sistema de acceso seguro al hogar.



(a) Adquieres un software de reconocimiento facial, lo instalas en tu computadora local y lo vinculas con una base de datos relacional/SQL estándar para memorizar las caras de los usuarios legítimos.

(b) Después de un estudio preliminar, encuentras que mantener la base de datos de caras legítimas, junto con entrenar al sistema de reconocimiento sobre nuevas caras y desaprender las caras de los residentes que se han ido, es excesivamente complejo y costoso. Decides que el procesamiento del reconocimiento facial debe ser externalizado a una compañía de seguridad especializada, FaceReco, Ltd. Esta empresa se especializa en autenticación de usuarios, pero no proporciona ningún servicio específico para aplicaciones. Por lo tanto, aún necesitas desarrollar el resto del sistema de control de acceso.

La primera tarea es identificar los actores, entonces la cuestión es: ¿Las nuevas herramientas (software de reconocimiento facial y base de datos relacional) son nuevos actores o son parte del sistema y no deben distinguirse del sistema? En el caso (a), no vale la pena distinguirlos, por lo que son parte del sistema planificado. Aunque cada uno de estos es un complejo sistema de software desarrollado por una gran organización, en lo que a nosotros (los desarrolladores) respecta, son solo módulos que proporcionan almacenamiento de datos y autenticación de usuarios. Lo más importante es que están bajo nuestro control, por lo que no hay nada especial en ellos en comparación con cualquier otro módulo del sistema planificado.

Por lo tanto, para el caso (a), todo permanece igual que en el diseño original.

Para el caso (b), una parte del nuevo diagrama de casos de uso se necesita distinguir un nuevo actor, la Compañía FaceReco, que proporciona servicios de autenticación. Es necesario saber que son parte del proceso de cumplimiento de

algunos objetivos(s) de los actores iniciadores.

