

DESEÑO DUN SISTEMA DE CONFIGURACIÓN DUN FOGAR INTELIXENTE

Víctor Xesús Barreiro Domínguez
Manuel Bendaña Gómez
David Carracedo Montero
Daniel Chenel Cea

Deseño de Software
2º Grao en Enxeñaría Informática
Universidade de Santiago de Compostela
Curso 2019/2020

Índice

1. Introducción - Descripción do problema	5
1.1. Introducción	5
1.1.1. Obxectos Intelixentes	7
1.1.2. Desenvolvemento de usuario final (EUD)	8
1.2. Sistema de configuración dun fogar Intelixente (SCFI)	9
1.2.1. Rede de sensores de propósito xeral	10
1.3. Arquitectura do SCFI	11
1.3.1. Conexións	11
1.3.2. Xestión	12
1.3.3. Control Interno	12
2. Casos de uso	14
2.1. Diagrama de casos de uso	14
2.2. Descripción detallada dos casos de uso	15
2.2.1. Caso de uso 1: Rexistrar SO	15
2.2.2. Caso de uso 2: Eliminar SO	16
2.2.3. Caso de uso 3: Seleccionar SO	16
2.2.4. Caso de uso 4: Configurar SO	17
2.2.5. Caso de uso 5: Configurar SO por formulario	17
2.2.6. Caso de uso 6: Configurar SO mediante un editor	18
2.2.7. Caso de uso 7: Configurar SO no modo avanzado	19
2.2.8. Caso de uso 8: Planificar actuación	19
2.2.9. Caso de uso 9: Visualizar información	20
3. Diagramas de Actividade	20
3.1. Unha ferramenta para describir accións	20
3.2. Diagrama de Actividade 1: rexistrar SO	20
3.3. Diagrama de Actividade 2: eliminar SO	22
3.4. Diagrama de Actividade 3: seleccionar SO	22
3.5. Diagrama de Actividade 4: configurar SO	22
3.6. Diagrama de Actividade 5: planificar actuación	26
3.7. Diagrama de Actividade 6: visualizar información	27

4. Interface de Usuario	28
4.1. Pantalla principal	28
4.2. Pantalla de información	28
4.3. Pantalla de xestión de obxectos	28
4.3.1. Pantalla Inserción	29
4.3.2. Pantalla Eliminar	29
4.3.3. Pantalla de Selección	29
4.4. Pantalla de configuración	30
5. Casos de Proba	32
5.1. Caso de Proba 1	32
5.2. Caso de Proba 2	32
5.3. Caso de Proba 3	33
5.4. Caso de Proba 4	33
5.5. Caso de Proba 5	33
5.6. Caso de Proba 6	33
5.7. Caso de Proba 7	34
5.8. Caso de Proba 8	34
5.9. Caso de Proba 9	34
5.10. Caso de Proba 10	34
5.11. Caso de Proba 11	35
5.12. Caso de Proba 12	35
5.13. Caso de Proba 13	35
5.14. Caso de Proba 14	35
5.15. Caso de Proba 15	35
5.16. Caso de Proba 16	36
6. Diagrama de Componentes	36
6.1. GPSN: Rede de comunicación	37
6.2. SCFI: Sistema de Control do Fogar Intelixente	37
7. Conceptos susceptibles de ser clases	39
8. Diagramas de Secuencia	40
8.1. Recuperar SO	41

8.2. Registrar SO	41
8.3. Eliminar SO	42
8.4. Seleccionar SO	42
8.4.1. Activar SO	42
8.4.2. Desactivar SO	43
8.5. Configurar SO	44
8.5.1. Configurar SO por Formulario	44
8.5.2. Configurar SO por Editor	44
8.5.3. Configurar SO no modo avanzado	45
8.6. Planificar actuación	45
8.7. Visualizar información	45
9. Diagrama de clases inicial	47

Abstract. Neste proxecto adicarémonos ao proceso de deseño dun Sistema de Configuración dun Fogar Intelixente (SCFI), a través de diferentes fases. Dividiremos o documento en varias partes: primeiro, incluiremos unha descrición completa do problema, na que se amosa como se abordou o problema; a continuación, comezaremos co desenvolvemento das actividades propostas ata o de agora: na primeira delas desenvolvimos os casos de uso, os diagramas de actividade, algúns casos de proba e o deseño da interface gráfica. Na segunda, tratamos os diagramas de compoñentes e os de secuencia, ademais dunha primeira aproximación ao de clases.

Palabras Clave: IoT, Obxectos Intelixentes, EUD, GPSN, SCFI, UML, Casos de Uso, diagramas, Deseño.

1. Introducción - Descrición do problema

1.1. Introducción

Comezamos este proxecto presentando o problema de deseño que se aborda no mesmo, o deseño dun Sistema de Configuración dun Fogar Intelixente.

A difusión da Internet das cousas (IoT), dirixida a conectar todos os obxectos do mundo físico, está afectando a diferentes dominios como a asistencia sanitaria, a automatización industrial, o control intelixente de enerxía, a asistencia de persoas maiores, a seguridade pública, a xestión urbana, a construción de infraestruturas, ou as casas intelixentes. Por iso, nos próximos anos espérase que se conecten mil millóns de obxectos intelixentes grazas ás tecnoloxías IoT, aínda que hai que afrontar varios retos sobre a fiabilidade dos datos, e as necesidades de interoperación e eficiencia.

A IoT permite conectar obxectos intelixentes e controlalos de xeito remoto mediante aplicacións instaladas nos dispositivos intelixentes. O principal problema das aplicacións de detección baseadas en sensores IoT é que as medidas individuais dos sensores dun único dispositivo IoT son en xeral insuficientes para tarefas de detección complexas e fiables. Para xestionar isto, introdúcense sensores virtuais (VS) para operar na nube de sensores como abstracción dos dispositivos físicos.

En lugar de analizar as medicións dun único sensor, o VS pode usar medicións de sensores independentes de varios dispositivos e executar un algoritmo de estimación / predición para cumprir de xeito eficiente coa súa tarefa. Por exemplo, un VS pode usar un sistema de posicionamento global (GPS) para detectar a situación do dispositivo, ou pode estimar a situación en función das posicións doutros sensores de localización próximos cando se atopa en interiores. Os VS tamén se poden ver como sensores software que proporcionan medicións indirectas de condicións abstractas, combinando datos captados por sensores físicos heteroxéneos. Polo tanto, os VS poden realizar un amplo conxunto de operacións, como a fusión de datos, a agregación de datos, ou o manexo de modelos de predición.

Os principais beneficios dos VS son os seguintes: (1) a reutilización de redes multi-obxectivo despregadas para dar soporte a diferentes aplicacións, de xeito que poden proporcionar diferentes tipos de información dependendo das aplicacións particulares; (2) a heteroxeneidade, e (3) o enmascarado das fontes de datos, ao actuar como mediadores/.../proxies entre sensores físicos e aplicacións clientes.

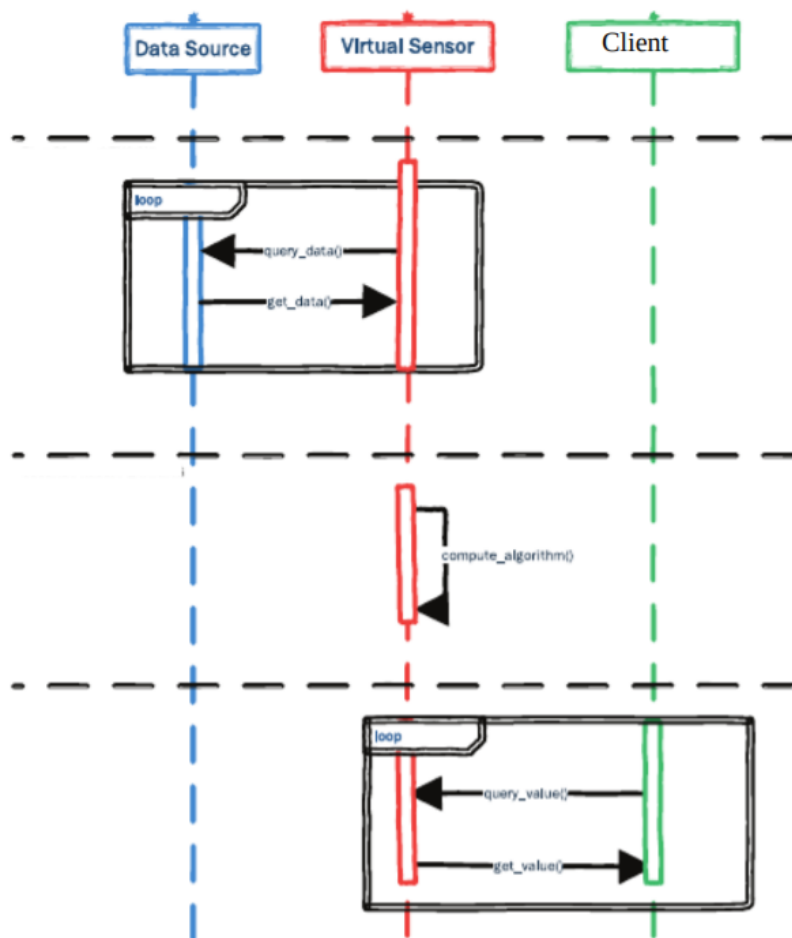


Figura 1. Diagrama de secuencia que presenta a interacción dun sensor virtual (VS)

Neste proxecto, interézanos a aplicación da IoT aos fogares intelixentes. O termo fogar intelixente evolucionou desde a referencia exclusiva ao control centralizado e semiautomatizado dos sistemas ambientais, como o calor e as luces, ao uso de tecnoloxía para controlar calquera obxecto compatible co ambiente doméstico.

Normalmente, o obxectivo dos sistemas domésticos intelixentes é proporcionar servizos de comodidade, asistencia sanitaria, seguridade e consumo de enerxía.

Comezaremos describindo antes de nada algúns conceptos relevantes para a IoT especialmente no ámbito dos fogares intelixentes:

A figura 1 describe o diagrama de secuencia UML do acceso dun obxecto cliente aos datos proporcionados por un sensor virtual: no eixe horizontal están marcados os obxectos implicados, ou sexa, controlador de fonte de datos externa (Data Source), cliente (Client) e VS (Virtual Sensor); e no eixe vertical amósase a secuencia de mensaxes intercambiadas e o tempo.

A figura 2 amosa un exemplo dun diagrama de clases para un sistema de mobilidade eléctrica, que conta con varios servizos, e cada un deles manexa datos de diferentes sensores (físicos ou virtuais). Neste caso o “Physical Sensor” (ou Data Source) e o “Vir-

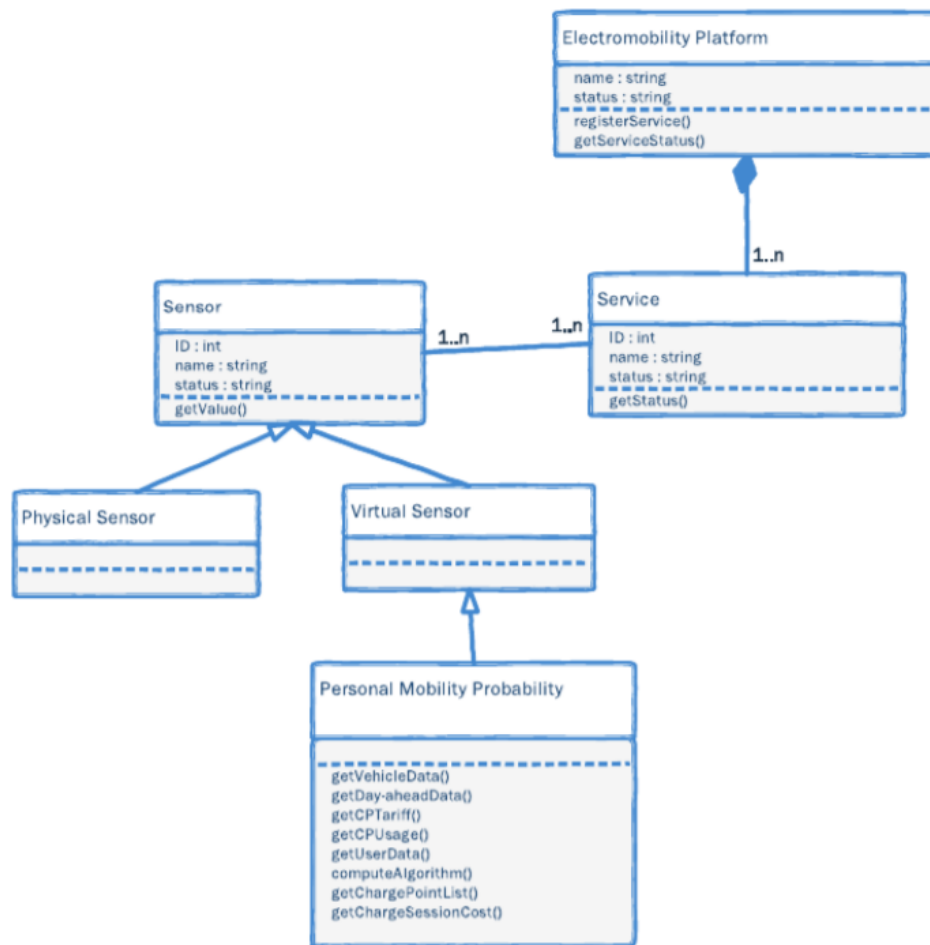


Figura 2. Diagrama de clases coas relacións entre servizos, sensores e sensores virtuais para un sistema de mobilidade eléctrica

tual Sensor” poderían actuar como clases abstractas das que se poden derivar outras máis específicas.

O patrón operativo proposto permite tratar por separado diferentes aspectos dun VS (por exemplo, aspectos técnicos e comerciais) que poden ser útiles tanto desde o punto de vista do desenvolvedor como do punto de vista xurídico/comercial.

1.1.1. Obxectos Intelixentes

Os obxectos intelixentes (SO) son os bloques fundamentais da IoT e combinan fontes de datos conectadas a obxectos físicos e os sensores virtuais asociados, con outros elementos software que permiten configurar o comportamento do obxecto e enviar comandos aos controladores dos seus actuadores.

Un obxecto intelixente pódese implementar empregando un amplo abano de metodoloxías. Por exemplo, a metodoloxía de implementación inspirada no estándar Sense-Plan-Act clásico usado normalmente na robótica. De feito, a metodoloxía Sense-Plan-Act permite deseñar as tres capacidades críticas que deben ter os SO para poder funcionar con

eficacia: (1) adquisición de datos a partir de sensores (Sense); (2) execución do algoritmo para ofrecer unha resposta axeitada, de acordo con algunha estratexia predefinida (Plan); e (3) accións de saída (Act) segundo o plan.

O patrón de operacións dun obxecto intelixente comeza na fase de recoller datos de fontes de datos externas (Sense) por mediación dos sensores físicos. Cos datos recollidos actualízase a medición indirecta por parte do VS, no caso de que este combine os datos proporcionados por varios sensores. A nova información elaborada basearase na aplicación de métodos estatísticos e / ou técnicas de intelixencia artificial, sobre os datos en bruto. No caso dun único sensor físico conectado, o resultado podería ser o mesmo que o recollido do sensor físico.

A segunda fase (Plan) é a fase onde se usan o estado interno do obxecto xunto ca información proporcionada polo VS. Nesta fase, o SO aplica un algoritmo coa estratexia que define o plan de acción.

A terceira e última fase é a fase de acción (Act), onde se pon en marcha o plan de actuación definido previamente, e que en xeral consistirá, por exemplo, no envío de comandos aos controladores dos actuadores físicos do obxecto virtual, pero tamén pode consistir en proporcionar servizos a outros subsistemas.

1.1.2. Desenvolvemento de usuario final (EUD)

Hoxe en día, a lóxica do control dos SO en fogares intelixentes resulta inaccesible á maioría dos usuarios finais. Non obstante, a entrada destes dispositivos tecnolóxicos na vida cotiá obriga a darlle aos usuarios un papel activo na súa configuración para que estes sistemas se poidan axustar dinamicamente ás súas necesidades individuais. Este fenómeno foi acumado na literatura como Desenvolvemento de Usuario Final (EUD). EUD refírese a un conxunto de actividades, técnicas e ferramentas que permiten aos usuarios finais configurar, modificar e controlar artefactos de software e hardware.

En xeral, un SO é un subsistema (hardware/software) que pode interactuar con outros subsistemas ou humanos do seu contorno. Os SO poden empregarse para axustar automaticamente a temperatura en cada habitación, medir os niveis de monóxido de carbono ou controlar luces a través do teléfono móbil.

A maioría dos obxectos do noso contorno non están dentro da definición anterior dun SO. De aí que existan placas de sensores e actuadores de propósito xeral que se poden acoplar a estes obxectos e así compatibilizar estes obxectos coas casas intelixentes. As placas incorporan un procesador e un transceptor inalámbrico, e diferentes tipos de sensores (luz, son, presión, aceleración, temperatura, etc.) segundo as necesidades. Como exemplo, poden engadirse sensores a unha cama para rastrexar o movemento e a posición do usuario, a outros obxectos (cadeiras, lámpadas, mesas, despertadores, etc.) para habilitar a interacción entre os obxectos e o usuario, acelerómetros no tirador da porta e no pulso do usuario para identificar a persoa é detectada medindo a correlación entre os dous sinais do acelerómetro, etc.

O EUD permite aos usuarios finais deseñar visualmente aplicacións domésticas intelixentes mediante unha interface táctil. Permite aos usuarios especificar dispositivos de entrada e saída e a configuración da lóxica de comportamento empregando regras "If-Then",

que describen unha condición e a súa acción asociada. Por exemplo, unha interface formada por palabras que se poden agrupar para formar regras, combinando catro elementos de información: quen, que, onde e cando. Estes elementos describen a acción que o sistema enviará en resposta a un estímulo dado e en que momento.

Para responder ás limitacións da lóxica definida por programa, propuxéronse varios sistemas EUD que admiten a descrición do comportamento baseado en regras. As regras definen principalmente unha condición e unha acción. As condicións son probas sobre os valores do sensor e as accións son comandos que se deben enviar aos actuadores. As normas EUD utilizan os valores de sensores e actuadores para definir condicións e accións.

1.2. Sistema de configuración dun fogar Intelixente (SCFI)

Neste proxecto imos desenvolver un sistema composto por compoñentes de hardware e software, que permita aos usuarios finais deseñar e configurar un sistema doméstico intelixente que responda ás súas necesidades. O sistema debe deseñarse para soportar dúas clases de usuarios: informáticos e usuarios non informáticos. Os usuarios non informáticos son os que non recibiron ningunha formación en desenvolvemento de software e hardware. Para soportar usuarios non informáticos, propoñemos a linguaxe de control simple (SCL) para o control central dos SO nun fogar intelixente.

Un fogar intelixente típico está composto por varios SO que se comunican cun sistema central denominado sistema de Control e Visualización Central (CVC). As funcións principais deste sistema pódense resumir do seguinte xeito:

1. Recibe información sensorial dos SO implementados no ambiente doméstico intelixente.
2. Controla o ambiente intelixente mediante comandos enviados a un subconxunto de SO despregados nese contorno. Estes comandos envíanse normalmente en resposta á información sensorial obtida do ambiente, un comando de usuario, ou un temporizador preconfigurado.
3. Permite ao usuario visualizar a información sensorial recollida en varios niveis de granularidade.

O SCFI interactúa con dous tipos de entidades: usuarios e SOs. Definimos a un usuario como un individuo que crea SOs usando clústers de transdutores (sensores ou actuadores) de propósito xeral que aparelha a obxectos do entorno formando unha rede (GPTN), configura un entorno intelixente, pode enviar comandos aos SO e/ ou visualiza a información producida polos SO usando o CVC. Para configurar un entorno intelixente, o usuario especifica que debe facer o CVC en resposta aos datos recibidos dos sensores do SO. Por exemplo, considerando o seguinte SO: unha cadeira de oficina equipada cun sensor de presión e un actuador de vibración, ambos integrados na almofada do asento. O usuario pode configurar o CVC para enviar un comando ao actuador para que vibre se se detecta que a presión é alta (é dicir, unha persoa está sentada na cadeira) durante un par de horas. A vibración recordarlle á persoa sentada que debe dar un paseo. Para configurar un ambiente intelixente, o usuario realiza o seguinte:

1. Constrúe SO usando GPTN e implantaos no contorno.
2. Usa o CVC para:
 - Rexistrar os SO dispoñibles subministrando os seus nomes e enderezos IP.










Transducer	Prototype	Remarks
Wireless Node's Core (Mainboard)		Data collection, processing and communication
Pressure Sensor		I ² C IDs 1-10 are reserved for pressure sensors (FSRs)
Temperature Sensor		I ² C IDs 72-75 are reserved for Temperature Sensors (TMP102)
Accelerometer		I ² C IDs 83-84 are reserved for accelerometer (ADXL345)
Light Sensor		I ² C IDs 41, 57 are reserved for Light sensors (TSL2561)
Vibro-tactile Actuator		I ² C IDs 11-20 are reserved for vibro-tactile actuators
On/Off Actuator		I ² C IDs 21-30 are reserved for on/off actuators
Dimming Actuator		I ² C IDs 31-40 are reserved for dimming actuators
Extension		Connected to the I ² C bus when needed

Figura 3. Algúns sensores e actuadores

- Configura os SO rexistrados definindo regras para controlalos a través do CVC.
- Visualiza información dos sensores dos SO.

1.2.1. Rede de sensores de propósito xeral

A GPSN permite a creación de agrupacións de sensores en rede aparelados con obxectos, para formar SO. Cada clúster fórmanse mediante un mecanismo de plug-&-play que os conecta a unha placa principal a través de fíos mediante unha comunicación en serie, de xeito que a tarxeta principal pode recoñecer os sensores engadidos e eliminalos dinámicamente. A rede soporta sensores de uso común como temperatura, presión, luz, aceleración, etc, coma os mostrados na figura 3.

Asociado a cada clúster de sensores haberá un obxecto controlador que mantén a lista

dos sensores conectados en cada momento e que pode devolver os datos recollidos por cada un deles.

O controlador organiza os datos dos sensores e transmíteos no formato Language Model Sensor (sensorML).

Consideraremos que cada VS se conecta cunha única rede GPSN a través do seu controlador. A súa vez o SO pode conectarse con un ou con varios VS dos que recibe os datos, e tamén con un ou varios actuadores a través dos controladores respectivos.

Os SO envían datos de control no formato Actuator Model Language (actuadorML).

1.3. Arquitectura do SCFI

O apartado máis importante desta introdución é esta descrición da arquitectura do SCFI, que a faremos dende unha perspectiva algo diferente á achegada no guión inicial. Definiremos as compoñentes dividíndoas en tres segmentos diferentes: un de conexións (para rexistro e selección), outro de xestión de SOs (principalmente para a configuración) e un terceiro de control interno (esa parte que está máis oculta para o usuario):

1.3.1. Conexións

1. Rexistrador de SO:

Os SO dispoñibles son rexistrados polo usuario a través do Rexistrador de SO. Durante o rexistro, o usuario especifica o nome do SO e o enderezo IP. Toda a información recollida é almacenada na base de datos de información estática (SID).

Ademáis desa tarefa de rexistro, consideramos que nesta compoñente se recolle tamén a posibilidade de eliminar un SO que non esté activo, xunto con toda a súa información almacenada na base de datos.

2. Selector de SO:

O Selector permite que o usuario seleccione un subconxunto de SO rexistrados cuxa información está almacenada no SID para estar activos no sistema. Os datos recadados dos sensores dos SOs seleccionados poden usarse para especificar as regras de actuación dos SO a través do Configurador, datos que se enviarían a través do transmisor de datos, consultando os SO seleccionados sobre a súa información dinámica. A información dinámica refírese ao número e tipo de sensores e actuadores de cada SO (xa que os transdutores poden engadirse ou eliminarse dinamicamente mediante un mecanismo plug-&-play), e as medidas do sensor en tempo real. Tendo un SO seleccionado, poderase escoller entre activalo ou desactivalo.

- Se se escolle activar un SO, entón solicitaráselle a información descrita previamente e almacenarase na base de datos de información dinámica (DID).
- Pola contra, se o usuario escolle desactivar o SO, o sistema reflexará na DID que o SO se desactivou.

1.3.2. Xestión

1. **Configurador de SO:** A definición de control dos SO baséase na lóxica difusa (fuzzy). A teoría de conxuntos difusos foi deseñada para imitar o mecanismo de razoamento humano. Por exemplo, é moito máis doado para os humanos pensar na temperatura ambiente en termos cualificativos como calor ou frío, no canto de especificar limiares nítidos que describen o estado térmico da habitación. Polo tanto, desde a perspectiva de usabilidade, empregando a lóxica difusa, faise máis intuitiva a configuración dun entorno intelixente, especialmente para usuarios con pouca ou ningunha competencia en programación informática.

Hai unha gran cantidade de linguaxes de control difuso, con todo, estes controladores son adecuados para aplicacións de enxeñaría e requiren experiencia técnica. Non obstante, o SCFI debe dar soporte a usuarios sen formación técnica formal. Por iso, introducimos o SCL, unha linguaxe baseada en regras que permite aos usuarios definir accións que os actuadores realizan en resposta aos datos do sensor ou aos comandos do usuario. É dicir, definen as estratexias a seguir polos SO. O usuario pode configurar regras de control (Plans) dun obxecto seleccionado usando un dos tres modos: **baseado en formularios**, **baseado en editor**, e **avanzado**. Os usuarios que posúen coñecementos de programación poden aprender SCL ou a crear regras de lóxica difusa moito máis rápido que outros usuarios. Así, o modo avanzado proporciona ao usuario un control de axuste fino.

- No modo baseado en formulario, o usuario define as regras de control usando SCL. Non obstante, en lugar de proporcionar ao usuario un editor de texto para escribir SCL, emprégase un editor gráfico. Este editor permítelle ao usuario crear regras SCL ao cubrir un formulario. Polo tanto, os usuarios non teñen que aprender a estrutura de SCL.
 - No modo baseado no editor, o usuario especifica SCL usando un editor de texto. O controlador do SCFI está baseado completamente en lóxica difusa. De aí que, como último paso, tanto no método baseado en formulario como no baseado no editor, as regras SCL se traduzan automaticamente en funcións difusas a través do Xerador Fuzzy.
 - No modo Avanzado, o usuario especifica funcións de membresía directamente sen usar SCL. Polo tanto, o Xerador Fuzzy non se usa neste último modo. En calquera caso, ao final da configuración, as regras difusas e as funcións de membresía gárdanse na base de datos Fuzzy (FD).
2. **Visualizador:** Este compoñente envía unha solicitude de lectura aos SO que se atopan activados polo usuario a través do transmisor de datos e mostra na pantalla as medidas en tempo real. Tamén mostra información do sensor, como o tipo, o fabricante e a taxa de mostraxe.

1.3.3. Control Interno

1. **Verificador Sintaxe SCL:** Este compoñente é o responsable de verificar a sintaxe das regras SCL antes de que as traduza o Xerador Fuzzy converténdooas en regras difusas e funcións de membresía. As regras SCL son procesadas a través de dúas etapas polos módulos:

- Analizador léxico: para converter a serie de caracteres SCL en expresións regulares (*tokens*).
 - Analizador de sintaxe: para asegurar que as regras concordan coa gramática SCL, e construír unha árbore de análise.
2. **Xerador Fuzzy:** Este compoñente é o responsable de xerar as funcións de membresía e as regras difusas baseadas nas regras SCL. Recibe dúas entradas: regras SCL organizadas nunha árbore de análise do verificador de sintaxe SCL, e a lista de SO dispoñibles, que inclúe os seus sensores e actuadores, obtidos do DID. O Xerador Fuzzy produce funcións de membresía difusa para os sensores, os comandos do usuario e os actuadores referenciados no SCL, e logo produce regras difusas que coinciden coa lóxica descrita no SCL.
 3. **Planificador:** Este compoñente, recibe datos dos SOs, borrosifícaos utilizando as funcións de membresía asociadas, xera a saída difusa empregando algunha das regras almacenadas na base de datos FD e desborrosifica a saída con resultados cuantitativos. Os resultados cuantitativos xuntos cos comandos correspondentes son enviados ao actuador no formato *actuadorML*. Ademais de actuadores tradicionais como vibradores ou interruptores eléctricos, o sistema soporta actuadores en forma de xeradores de mensaxes. Polo tanto, estes actuadores poden enviar avisos ou mensaxes informativas ao usuario en forma de correo electrónico ou SMS.

Este compoñente carga as regras difusas da FD e solicita datos de medición de sensores en tempo real dos SO. Unha vez que recibe os datos solicitados, procésao e xera comandos de actuación pertinentes sempre que sexa necesario (segundo as regras difusas). O compoñente do transmisor de datos transmite estes comandos (en caso de haber algo que transmitir) aos SO asignados.
 4. **Transmisor e Receptor de datos:** Estes compoñentes son os responsables de intercambiar información entre o sistema, o usuario e os SO. A información intercambiada está empaketada en mensaxes *SensorML* e *ActuatorML*. Os comandos do usuario tamén se empaketan en mensaxes *SensorML* onde o campo de valor conterá o ID do comando. O sistema envía tres tipos de solicitudes aos SO: ler información do SO, ler medicións dos sensores ou enviar un sinal de acción. Cando se reciben datos dos SO, son enviados polo receptor de datos ao DID.
 5. **Bases de Datos:** O sistema dispón de tres bases de datos para almacenar datos relevantes sobre os obxectos intelixentes, medidas de sensores e regras do controlador. Estas bases de datos detállanse a continuación:
 - Base de datos de información estática (SID): o SID almacena información sobre o rexistro dos SO, en particular, os nomes e os enderezos IP.
 - Base de datos de información dinámica (DID): esta base de datos almacena a información dinámica recibida dos obxectos intelixentes a través do receptor de datos. Esta información inclúe o número e tipo de sensores e actuadores de cada SO, e medicións de datos en bruto en tempo real dos sensores do SO. Tamén consideramos que é necesario que se recolla se o sensor está activo ou non (posto que puido ser activado inicialmente e logo desactivado, e ter xa almacenada na DID información sobre el).
 - Base de datos difusa (FD): esta base de datos é a responsable de almacenar as funcións e regras de membresía xeradas como un ficheiro difuso durante a fase de configuración. O controlador recupera esta información para

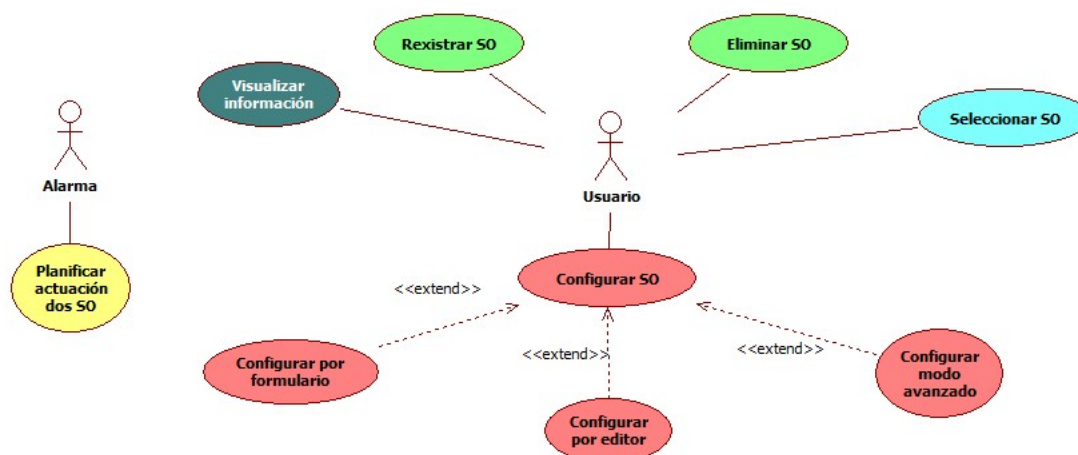


Figura 4. Diagrama de casos de uso do SCFI

poder difundir os datos brutos dos sensores.

Nos tres casos, entendemos que os SOs almacenados se poden identificar en calquera das bases de datos polo seu nome, que consideraremos a clave primaria.

2. Casos de uso

2.1. Diagrama de casos de uso

Trala especificación inicial do problema, procedemos a mostrar o diagrama de casos de uso que deseñamos para o sistema. Na figura 17 amósase o diagrama, no que se pensamos que se recollen todos os casos de uso que involucran directamente ao usuario.

Consideramos que é necesario realizar unha pequena xustificación das decisións tomadas, para que así o lector poida interpretar mellor o resultado.

Para comezar, decidimos incluír dous actores, que son o Usuario que accede ao sistema (o máis importante de todos, posto que o que estamos a deseñar vai encamiñado á utilización do sistema por parte del), e unha Alarma, un actor de tipo temporal que representa a frecuencia coa que se realizará a planificación dun SO, e que nos parece igualmente interesante que se vexa reflexada neste diagrama dado que, inda que non sexa algo co que o usuario vai interactuar directamente, pensamos que é un comportamento derivado doutras accións do usuario o suficientemente importante como para recollelo aquí dalgún xeito (non faremos o mesmo con outras partes do sistema).

Vamos a comentar o contido deste diagrama en tres partes:

- A parte superior reflexa as transaccións que pode realizar o usuario para o que englobaremos como a xestión dos SOs: rexistrar, eliminar, seleccionar SOs e visualizar a información dos SOs seleccionados.
- A parte inferior (cor vermella) reflexa a parte de configuración do SO, que modelamos mediante un caso de uso "principal" que representa a parte común da configuración, e varios caso de uso que estenden ao principal, un por cada posible modo de configuración (formulario, por editor e avanzado). Esta idea poderíase

modelar tamén por unha xerarquía, parécenos que son dúas representacións equivalentes, pero tras varias discusións chegamos a que resulta mellor e máis intuitivo entender que se trata de extensións da configuración (dependendo do modo).

- A parte esquerda representa unha parte do sistema que non está vinculada directamente co usuario, pero que evidentemente é provocada pola configuración realizada: o planificador. Ao caso de uso da planificación dos SO asóciase ese actor alarma que representamos previamente.

Distinguímos xa dúas partes importantes do sistema: unha parte de xestión de SOs e outra adicada á configuración e planificación. Estas partes son as que percibimos que, para a descrición dos casos de uso, deben estar presentes. No sistema evidentemente hai máis cuestións, como as bases de datos (que irán aparecendo mencionadas en cada un dos casos de uso), pero que non nos ten sentido modelar como casos de uso ao non ter ningún tipo de interacción con ningún actor.

2.2. Descrición detallada dos casos de uso

A continuación levaremos a cabo unha descrición en detalle de cada un dos casos de uso, para aclarar o seu funcionamento. En cada caso iremos especificando o seu propósito, os actores que interveñen e as precondicións e poscondicións dos mesmos, así como os posibles cursos dos eventos (o normal e os alternativos) e as relacións que teñan con outros casos de uso seguindo o diagrama presentado previamente.

Para distinguir o comportamento do actor do do sistema, empregaremos cores diferentes para describir os pasos realizados por cada unha das partes: en verde os pasos dos actores e en azul os pasos realizados polo sistema.

2.2.1. Caso de uso 1: Rexistrar SO

- **Propósito:** Dar de alta un obxecto intelixente no sistema.
- **Actores:** Usuario
- **Precondicións:** -
- **Poscondicións:** o sistema rexistrará un novo SO cos datos introducidos polo usuario (nome e IP) e almacenará a información na base de datos estática (SID), ou devolverá un erro se corresponde.
- **Curso Normal dos Eventos:**
 1. O usuario introduce nome do obxecto intelixente e a IP.
 2. O sistema validará os datos introducidos.
 3. O sistema accede á base de datos estática (SID) para almacenar os datos do novo SO.
 4. O sistema informa de que o proceso se executou correctamente.
- **Curso alternativo 1**
 2. Falla a validación (Por mor dunha introdución de parámetros incorrecta).
 3. O sistema informa ao usuario do erro producido.
- **Relacións con outros casos de uso:** non existe ningunha relación con outros casos de uso que sinalar.

2.2.2. Caso de uso 2: Eliminar SO

- **Propósito:** Retirar un obxecto intelixente do sistema.
- **Actores:** Usuario
- **Precondicións:** Hai algún SO rexistrado e inactivo no sistema.
- **Poscondicións:** O sistema actualizará a Base de Datos Estática (SID), a Dinámica (DID) e a Difusa (FD) eliminando toda a información sobre o obxecto intelixente que se decidiu eliminar.
- **Curso Normal dos Eventos:**
 1. O sistema accede á SID para tomar a información de todos os SOs rexistrados.
 2. Recollidos os SOs, o sistema selecciona aqueles que non están activos e amósaos ao usuario.
 3. O usuario selecciona un SO dos amosados.
 4. O sistema accede á FD para borrar as regras difusas que poida haber asociadas a ese SO.
 5. O sistema accede á DID para borrar a información dinámica que poida estar rexistrada do SO.
 6. O sistema accede á SID para borrar a información estática rexistrada do SO.
 7. O sistema informa de que o proceso rematou correctamente.
- **Relacións con outros casos de uso:** non existe ningunha relación con outros casos de uso que sinalar.



2.2.3. Caso de uso 3: Seleccionar SO

- **Propósito:** Permite escoller un obxecto intelixente dos rexistrados no sistema para o perfil de configuración actual, e activalo/desactivalo (dependendo do caso).
- **Actores:** Usuario
- **Precondicións:** É necesario que esté activa a conexión coa rede de sensores (GPSN) para posibles comunicacións cos SOs.
- **Poscondicións:** O sistema escollerá un obxecto dos rexistrados para activalo ou desactivalo e, no primeiro caso, traerá a súa información da DID. No segundo caso, almacenará que se desactiva.
- **Curso Normal dos Eventos:**
 1. O sistema accede á SID para recuperar a información de todos os SOs rexistrados.
 2. O sistema amosa os SOs rexistrados ao usuario.
 3. O usuario selecciona un obxecto intelixente rexistrado no sistema.
 4. O usuario escolleu un SO para activalo.
 5. O sistema envía unha mensaxe ActuatorML ao SO para pedir información do mesmo.
 6. O sistema accede á DID para almacenar os datos recibidos.
 7. Sistema informa ao usuario de que o proceso se executou correctamente.
- **Curso alternativo 1**
 1. Non hai SOs rexistrados.

2. O sistema avisa ao usuario da situación.
- **Curso alternativo 2**
 4. O usuario escolleu un SO para desactivalo.
 5. O sistema garda na DID que o SO seleccionado pasa a estar inactivo.
 6. O sistema informa de que o proceso se realizou correctamente.
 - **Relacións con outros casos de uso:** non existe ningunha relación con outros casos de uso que sinalar.

2.2.4. Caso de uso 4: Configurar SO

- **Propósito:** Xerar regras de control para un SO activo a partir das preferencias do usuario. Tras xerarse as regras correspondentes, almacenaranse na base de datos.
- **Actores:** Usuario
- **Precondicións:** -
- **Poscondicións:** Produciranse unha serie de regras difusas e funcións de membresía sobre algún SO activo que se almacenarán na FD, ou devolverase algún erro se é preciso.
- **Curso Normal dos Eventos:**
 1. O usuario escolle un dos modos (editor, formulario ou avanzado).
 2. O sistema recupera a información dos SO activos que polo tanto poden ser configurados.
 3. O sistema amosa os SOs seleccionados para estar activos ao usuario.
 4. O usuario escolle un dos SOs seleccionados no perfil actual para ser configurado.
 5. **Punto de extensión:** usuario seleccionou cubrir formulario. Executar o caso de uso **Configurar SO por formulario**.
 6. **Punto de extensión:** usuario seleccionou introducir directamente regras SCL. Executar o caso de uso **Configurar SO por editor**.
 7. **Punto de extensión:** usuario seleccionou o modo avanzado. Executar o caso de uso **Configurar SO no modo avanzado**.
 8. O sistema accede á base de datos FD para almacenar as regras xeradas a través dunha das configuracións.
 9. O sistema informa que se completou a configuración correctamente.
- **Curso alternativo 1**
 2. O sistema non ten SOs seleccionados como activos para poder configuralos.
 3. O sistema avisa ao usuario para que active antes algún SO.
- **Relacións con outros casos de uso:** este caso de uso está relacionado con outros tres mediante extensións: eses tres casos de uso equivalen a configuracións do SO por parte do usuario de diferentes xeitos (por formulario, por editor e avanzado).

2.2.5. Caso de uso 5: Configurar SO por formulario

- **Propósito:** Xerar regras de control para un SO mediante a introducción por parte do usuario de información nos campos dun formulario.
- **Actores:** Usuario

- **Precondicións:** Comezou a executarse o caso de uso 'Configurar SO', e seleccionouse unha configuración por formulario.
- **Poscondicións:** O configurador do sistema xerará a partir dos datos introducidos unha serie de regras difusas, ou devolverá un erro.
- **Curso Normal dos Eventos:**
 1. O sistema accede á DID para recuperar a información do SO que foi seleccionado para configurar.
 2. Grazas aos datos recuperados, o sistema amosa o formulario a cubrir ao usuario.
 3. O usuario cubre os diversos campos do formulario presentado.
 4. O sistema valida que a información dos campos sexa correcta.
 5. O sistema xera regras SCL segundo os valores introducidos.
 6. O sistema xera unha árbore de análise ao verificar as regras SCL xeradas.
 7. O sistema accede á DID para recuperar a información dinámica do SO.
 8. O sistema xera a lóxica difusa (regras difusas e funcións de membresía).
- **Curso alternativo 1**
 4. Hai algún fallo nos campos introducidos.
 5. O sistema informa dos problemas atopados.
- **Relacións con outros casos de uso:** este caso de uso é unha extensión de **Configurar SO**.

2.2.6. Caso de uso 6: Configurar SO mediante un editor

- **Propósito:** Xerar regras de control para os SO mediante a introducción por parte do usuario de sentenzas na linguaxe SCL.
- **Actores:** Usuario
- **Precondicións:** Comezou a executarse o caso de uso 'Configurar SO', e seleccionouse unha configuración por editor.
- **Poscondicións:** O configurador do sistema xerará coas regras introducidas polo usuario unha serie de regras difusas, en caso de ser correctas. Se non, devolverá un erro.
- **Curso Normal dos Eventos:**
 1. O usuario escribe regras en SCL para configurar SOs.
 2. O sistema xera unha árbore de análise ao verificar as regras SCL introducidas polo usuario.
 3. O sistema accede á DID para recuperar a información dinámica do SO.
 4. O sistema xera a lóxica difusa (regras difusas e funcións de membresía).
- **Curso alternativo 1**
 2. As regras SCL introducidas son incorrectas.
 3. O sistema informa dos problemas ao usuario.
- **Relacións con outros casos de uso:** este é un caso de uso que estende a **Configurar SO**.

2.2.7. Caso de uso 7: Configurar SO no modo avanzado

- **Propósito:** Introducción de lóxica difusa (regras difusas e funcións de membresía) directamente por parte do usuario para configurar un SO.
- **Actores:** Usuario
- **Precondicións:** Comezou a executarse o caso de uso 'Configurar SO', e seleccionouse unha configuración avanzada.
- **Poscondicións:** O sistema recolle as regras difusas do usuario e válidas, ou devolve un erro se houbo problemas.
- **Curso Normal dos Eventos:**
 1. O usuario introduce funcións de membresía e regras difusas para un SO.
 2. O sistema comproba que a lóxica introducida polo usuario sexa válida.
- **Curso alternativo 1**
 2. A lóxica introducida polo usuario non é válida.
 3. O sistema informa dos problemas atopados ao usuario.
- **Relacións con outros casos de uso:** este caso de uso é unha extensión de **Configurar SO**

2.2.8. Caso de uso 8: Planificar actuación

- **Propósito:** Xerar comandos de actuación axeitados para os SO en función das súas medicións e a lóxica difusa creada polo usuario, cando así o indique a alarma.
- **Actores:** Alarma
- **Precondicións:** Debe estar activa a conexión coa rede de sensores GPSN para as comunicacións co SO, e planifícase un SO que se encontra activo no SCFI.
- **Poscondicións:** Xeraranse comandos de actuación que se enviarán aos SO para que configuren os seus actuadores (se é preciso).
- **Curso Normal dos Eventos:**
 1. A alarma avisa ao planificador para que se leve a cabo a planificación dun SO.
 2. O sistema accede á FD para recuperar as regras difusas asociadas a un SO.
 3. O sistema envía unha mensaxe ActuatorML ao SO para solicitar datos de medicións.
 4. O sistema almacena a información recibida na DID.
 5. En función das regras difusas e os datos do SO, xéranse comandos de actuación para el.
 6. Envíase unha mensaxe ActuatorML ao SO cos comandos de actuación.
- **Curso alternativo 1**
 5. En función das regras difusas e os datos do SO, non se xerou ningún tipo de comando de actuación.
 6. O sistema non realiza ningún envío, e remata a execución do caso de uso.
- **Relacións con outros casos de uso:** non existe ningunha relación con outros casos de uso que sinalar.

2.2.9. Caso de uso 9: Visualizar información

- **Propósito:** Amosar información dos SOs que foron seleccionados polo usuario.
- **Actores:** Usuario
- **Precondicións:** Debe estar a conexión coa rede GPSN activa para a comunicación cos SOs.
- **Poscondicións:** Amosarase información dos SOs activos nese momento ao usuario.
- **Curso Normal dos Eventos:**
 1. O sistema accede á Base de Datos para recuperar información dos SOs activos.
 2. O sistema envía mensaxes ActuatorML a todos os SOs para solicitar información e datos de medicións.
 3. O sistema recolle a información recibida dos SOs e almacénna na DID.
 4. O sistema organiza a información recibida e amósalla ao usuario.
- **Curso Alternativo 1**
 1. Non hai SOs activos.
 2. O sistema non amosa ningún tipo de información, e indica nunha mensaxe a forma de actuación.
- **Relacións con outros casos de uso:** non existe ningunha relación con outros casos de uso que sinalar.

3. Diagramas de Actividade

3.1. Unha ferramenta para describir accións

Presentado xa todo o relativo aos casos de uso, neste apartado centrarémonos nos diagramas de actividade que deseñamos.

A descrición feita previamente dos casos de uso co texto plano pode resultar máis incómoda para o usuario. Alternativamente, podemos modelala mediante un diagrama de actividade, que nos permite reflexar o fluxo procedural de accións que son parte dunha actividade maior.

Amosamos os diferentes diagramas deseñados. A idea que tivemos foi intentar ter un por cada funcionalidade contemplada nos casos de uso, se ben englobamos algúns deles nun só diagrama (concretamente a configuración do SO a través dos diferentes métodos dispoñíbeis). Nos seguintes puntos, engadiremos cada un dos diagramas xunto con algún razoamento acerca das decisións que se foron tomando en cada caso.

3.2. Diagrama de Actividade 1: rexistrar SO

Modelamos o rexistro dun SO a través da introdución dos datos por parte do usuario. Na figura 5 pódese observar o diagrama asociado.

Este diagrama resulta sinxelo, comezando coa introdución dos datos dun SO por parte do usuario, que logo o sistema tentará verificar. Tras isto, temos unha decisión, posto que dependendo de se os datos son correctos ou non permitimos camiños diferentes:

- Se os datos son correctos, continúaase co fluxo principal, que levará a que o proceso remate correctamente cun aviso do remate por parte do sistema.

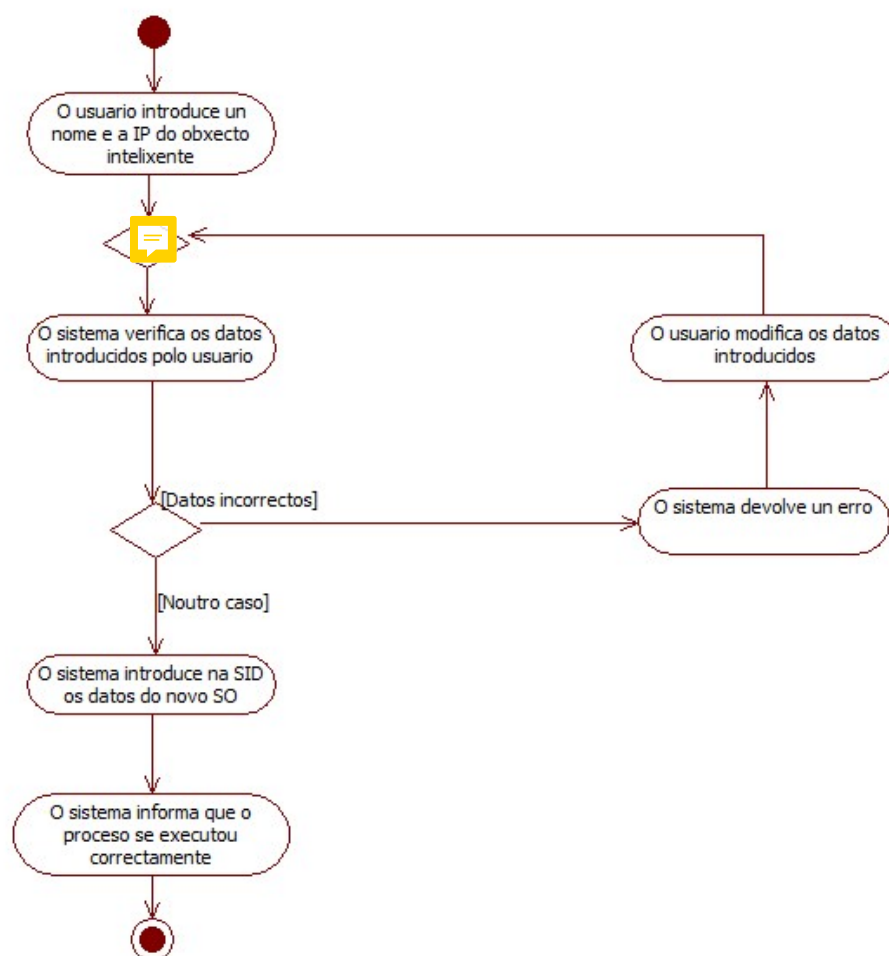


Figura 5. Diagrama de actividade que modela o rexistro dun novo SO por parte do usuario

- Se son incorrectos, consideramos que o axeitado é dar opción a que o usuario poida volver a introducir os datos introducidos e regresar á verificación dos datos por parte do sistema. Estimamos máis axeitado modelar este tipo de bucles neste diagrama (cousa que non consideramos na descrición dos casos de uso para evitar maior complexidade).

3.3. Diagrama de Actividade 2: eliminar SO

Eliminar un Obxecto Intelixente será unha acción que tamén poderá facer o usuario a partir de ter obxectos rexistrados. Na figura 5 pódese consultar o diagrama de actividade que representa esa situación.

Algo que destacar neste diagrama respecto do anterior é a introdución de estados de sincronización, que empregamos para modelar que hai dúas tarefas que cremos que se poden realizar simultaneamente ou ben en calquera orde, trátase do borrado dos datos do SO da DID, da SID e da FD, posto que pode haber información deles en calquera deses puntos.

Neste caso, supuxemos (como precondition) que tiña que haber algún SO rexistrado, posto que esta operación é algo adicional e non quixemos amosar tanta complexidade. Noutros casos sí que contemplaremos a posibilidade de que non haxa SOs rexistrados como unha alternativa dentro do caso de uso ou no diagrama.

3.4. Diagrama de Actividade 3: seleccionar SO

Pasamos a outra parte do SCFI como é a selección dos SO para seren activados ou desactivados. Na figura 7 amósase o diagrama de actividade correspondente.

Neste diagrama hai varios puntos de toma de decisións para representar por unha parte a situación na que non hai SOs rexistrados (o que leva a amosar unha mensaxe axeitada á situación) e pola outra a activación/desactivación do SO, posto que se se activa hai que realizar algunha tarefa máis.

Observar que hai dúas posibles vías de finalización:

- Que non exista ningún SO rexistrado (polo que non se pode seleccionar).
- Que se escolla un SO e se decida sobre a súa activación ou desactivación correctamente. Se se chega a esta situación, temos outro punto de decisión no que se opera de maneira distinta se se activa ou se desactiva.

3.5. Diagrama de Actividade 4: configurar SO

Este é o diagrama máis complexo dos desenvoltos no proxecto, empregado para amosar a configuración dun SO, posto que hai que ter en conta que neste diagrama englobamos catro casos de uso (se ben tres deles son extensións, o que nos levou a contemplalo todo no diagrama):

- Configurar SO.
- Configurar SO por formulario.
- Configurar SO mediante un editor.
- Configurar SO no modo avanzado.

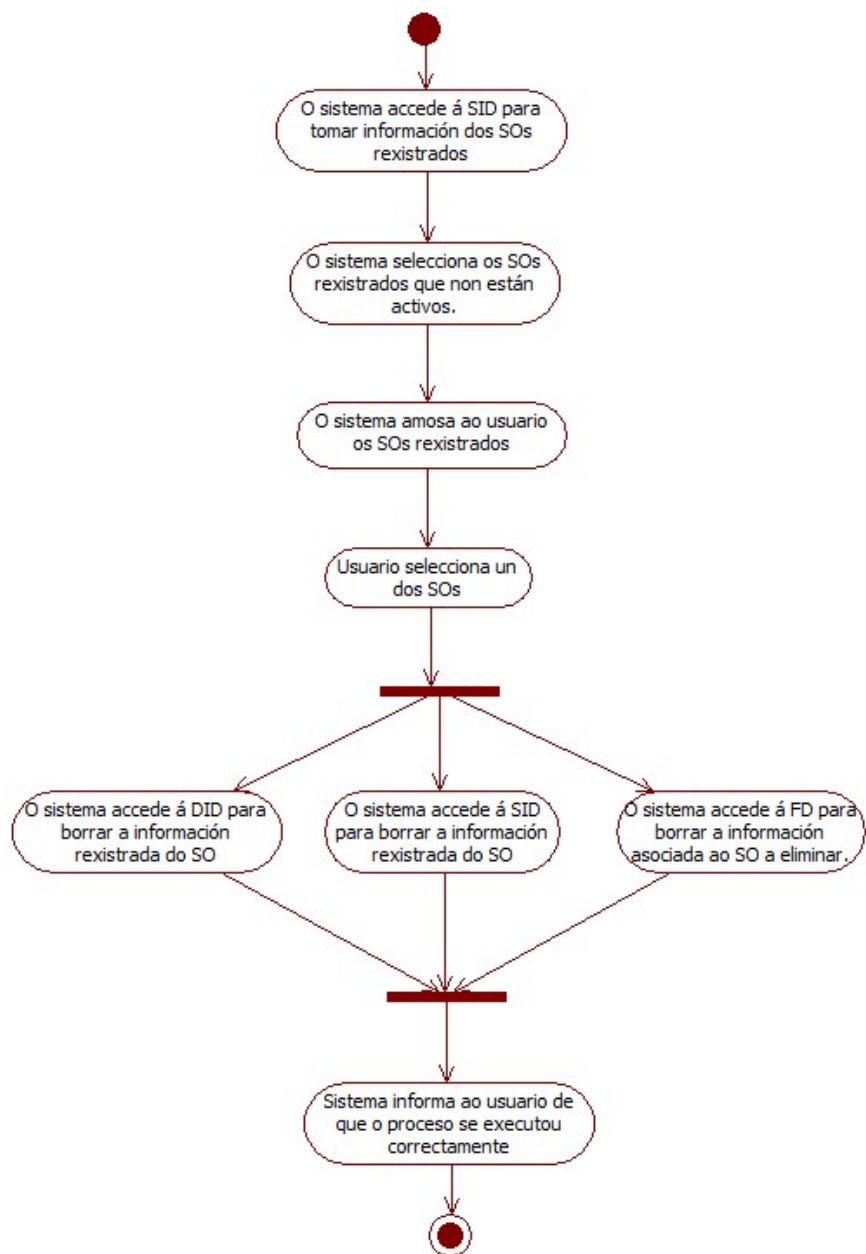


Figura 6. Diagrama de actividade que modela a eliminación de SOs

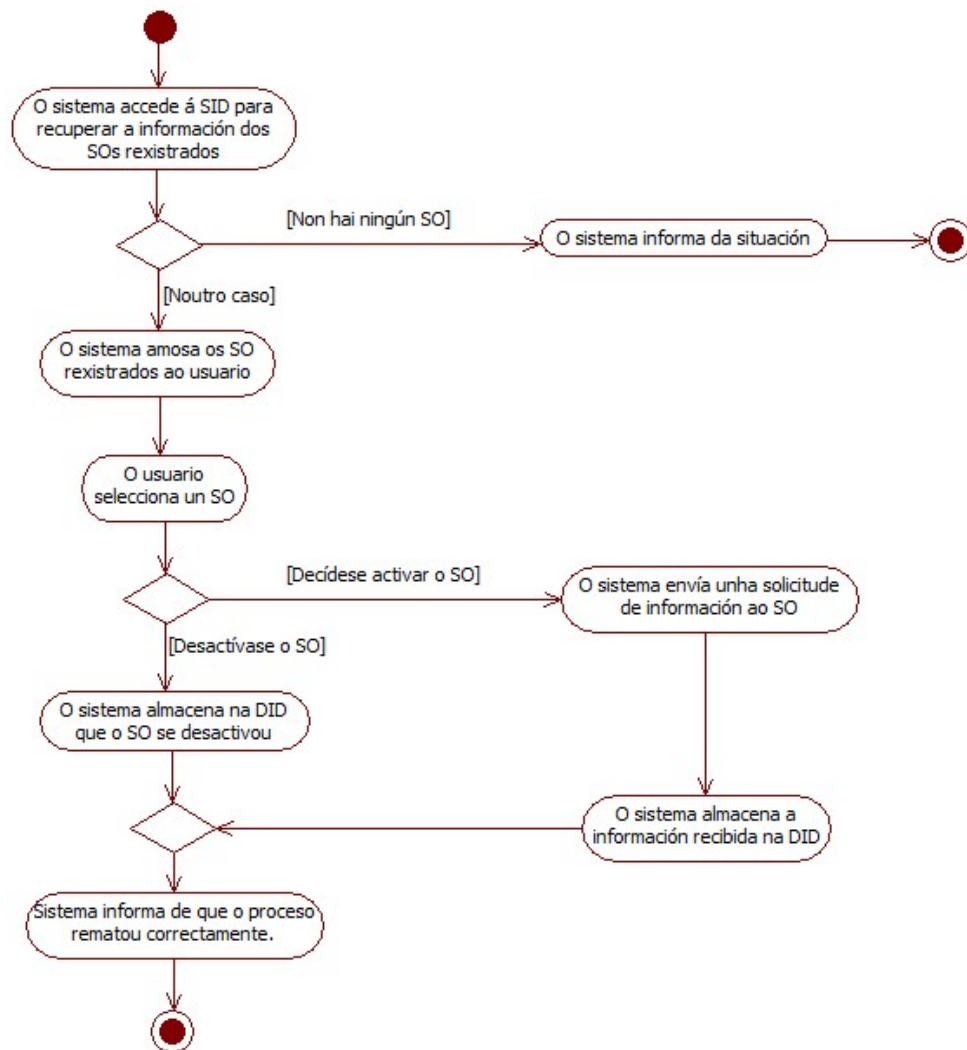


Figura 7. Diagrama de actividade que representa o proceso de selección dun SO

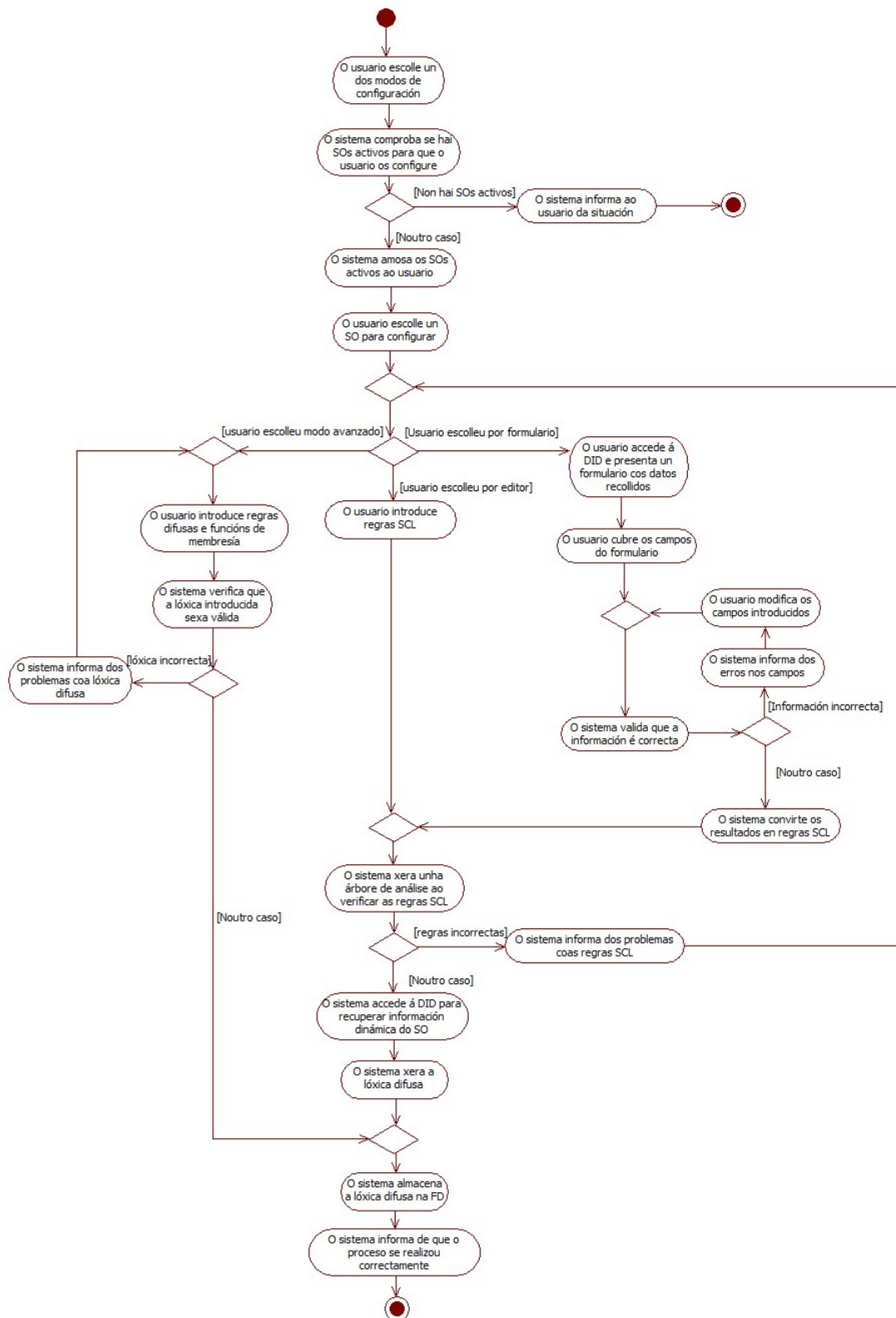


Figura 8. Diagrama de actividade que representa o proceso de configuración dun SO

Modélase a actividade completa de configuración nun só diagrama, o que se ve reflexado no seu tamaño e nos grandes puntos de decisión e de unión que se teñen ao longo do diagrama, que se pode observar na Figura 8.

Neste diagrama o punto de decisión máis importante atópase no paso de elección do modo de configuración, posto que en función diso hai partes propias de cada modo. Hai que sinalar que se pode ver como ao final os tres pasos alternativos acaban converxendo nun punto común, que é o almacenamento da lóxica difusa na FD. Tamén se unen antes os casos de configurar por formulario e por editor, cando xa se teñen as regras SCL.

3.6. Diagrama de Actividade 5: planificar actuación

Este diagrama de actividade representa as accións a realizar dentro do caso de uso 'planificar actuación', que tamén incluímos dentro do desenvolvemento deste sistema, inda que non haxa interacción directa co usuario. Na Figura 9 amósase o mencionado diagrama.

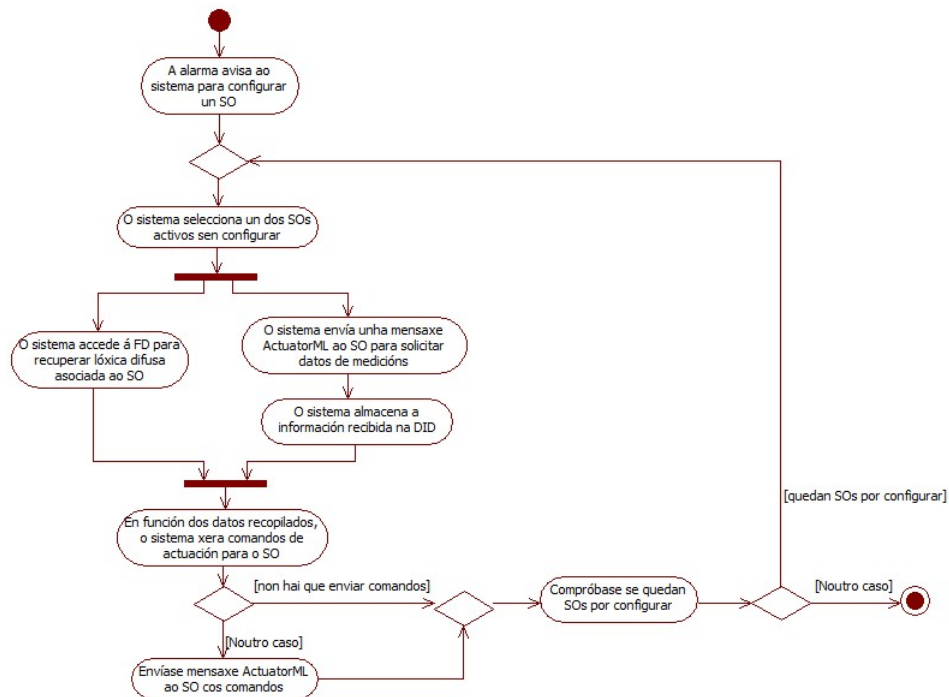


Figura 9. Diagrama de actividade que representa o proceso de planificación dun SO

Neste caso o proceso de planificación é activado por unha alarma, e a partir de entón realízase a mesma tarefa para cada un dos SOs que estén activos. Decidimos por iso reflexalo coma un bucle neste diagrama (algo que non recolleemos no caso de uso por simplicidade).

Dentro desa tarefa, introdúcese un estado de sincronización no punto no que se accede á base de datos e se envían mensaxes aos SOs, posto que son dúas tarefas de recuperación de información que entendemos que se poden realizar simultaneamente ou en calquera orde.

Polo demais, pode haber que mandar comandos ou non, o que reflexamos no diagrama mediante un punto de decisión.

3.7. Diagrama de Actividade 6: visualizar información

Quedaría con isto unha situación sen modelar, que é a visualización de información recopilada dos SOs. Podémolo consultar na figura 10.

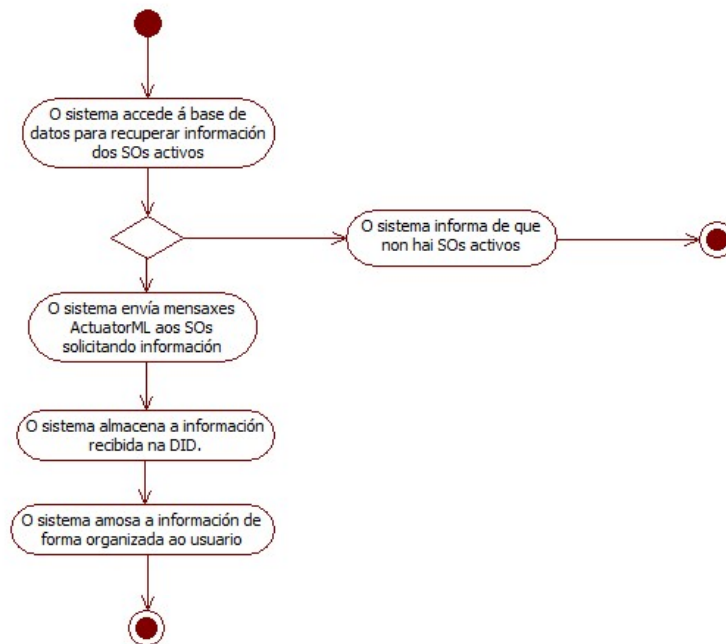


Figura 10. Diagrama de actividade que representa o proceso de visualización da información

O diagrama que se amosa non ten grande complexidade, posto que non existen moitas tarefas a realizar no marco deste caso de uso, mais que a consulta de información aos SOs activos e un punto de decisión por se non houbera ningún seleccionado.

4. Interface de Usuario

A continuación describiremos o diseño da interface da aplicación para a xestión do fogar intelixente xunto coas súas funcionalidades que consideramos. Comentar que realizamos este deseño pensando en que esta aplicación se poida desenvolver nun entorno móbil, dado o avance das tecnoloxías e a comodidade que pode supoñer dispor desa aplicación no teléfono.

4.1. Pantalla principal

Esta pantalla corresponde á vista inicial cando se inicia a aplicación. Nela mostrarase información como a hora e a temperatura a parte dunha mensaxe de benvida. Na zona inferior da pantalla aparecerá un menú que nos permitirá acceder o resto de funcionalidades da aplicación (Figura 11a)

4.2. Pantalla de información

Esta pantalla mostrará os dispositivos dispoñibles, así como unha pequena información acerca do seu estado, como por exemplo se unhas luces están encendidas ou un sistema de audio está reproducindo algún sonido. O usuario terá a posibilidade de premer enriba dos iconos dos dispositivos que aparezan en pantalla. Ao premer aparecerá outra pantalla con toda a información dispoñible sobre o obxecto en cuestión. Esta información adaptarse en función do tipo de dispositivo (Figura 11).

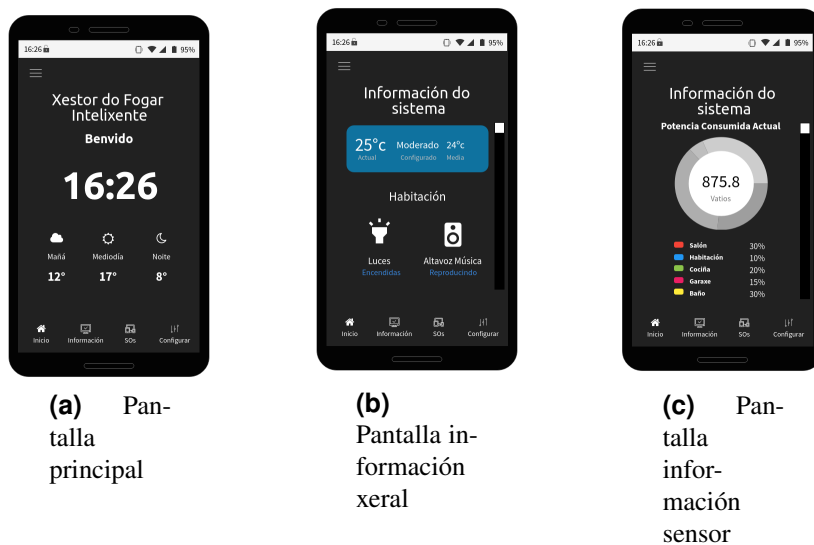


Figura 11. Pantallas de información

4.3. Pantalla de xestión de obxectos

Esta pantalla da interface permitirá ao usuario dar de alta obxectos intelixentes, eliminalos do sistema ou seleccionar cales se queren que estean activos. A interfaz mostrará un pequeno menú no que se poderá elixir a acción a realizar. Cada un destes botóns abrirá outra pantalla, as cales se describen a continuación (Figura 12a):

4.3.1. Pantalla Inserción

Nesta pantalla haberá dous campos que permitan ao usuario introducir o nome e a dirección IP do novo SO que se está dando de alta. Unha vez cubertos, mediante o botón gardar poderase rexistrar o obxecto intelixente (Figura 12b).

4.3.2. Pantalla Eliminar

Nesta pantalla haberá un item despregable no que se mostrarán todos os obxectos rexistrados no sistema e inactivos. O usuario pode seleccionar un deles para ser eliminado. Haberá un botón que fará que se realice a acción de eliminar (Figura 13a).

4.3.3. Pantalla de Selección

Nesta pantalla haberá un selector para activar e desactivar os obxectos intelixentes. Aparecerán unha serie de botóns con iconas que corresponderán ao respectivo obxecto intelixente. Pulsando cada botón, o usuario pode activar ou desactivar os SO, dependendo do seu estado actual (Figura 13b).



Figura 12. Pantallas de Xestión dun SO (I)



Figura 13. Pantallas de Xestión dun SO (II)

4.4. Pantalla de configuración

Esta pantalla permitirá a configuración dos SOs por parte do usuario en tres pasos. Primeiro mostrarase unha pantalla na que se lle permitirá ao usuario seleccionar o modo de configuración: mediante formulario, editor ou modo avanzado. A continuación aparecerá un item despregable onde o usuario seleccionará un obxecto de entre aqueles que estean activos no sistema (Figura 14b). Finalmente aparecerá unha pantalla en función do modo de configuración:

1. **Formulario** Na pantalla mostrarase unha lista coas condicións do obxecto e outra lista coas accións a realizar. Ademais haberá tres botóns: un para engadir novas regras, outro para engadir novas actuacións e outro botón para gardar os cambios (Figura 15b).

Ao pulsar no botón de engadir regra, mostrarase unha serie de items despregables nos que o usuario poderá seleccionar distintos parámetros e opcións para configurar a regra desexada. Despois o usuario terá a opción de gardar a regra mediante un botón situado na parte inferior (Figura 14c).

Se o usuario preme o botón de engadir actuación, aparecerá unha pantalla con diversos items nos que se lle permitirá seleccionar distintas opcións que permitan configurar a acción desexada. Despois poderá gardarse dita actuación mediante un botón (Figura 15b).



Figura 14. Pantallas de configuración (I)

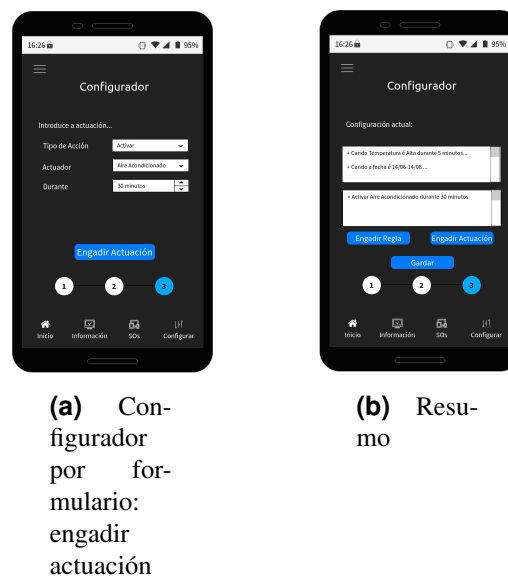


Figura 15. Pantallas de configuración (II)

2. **Editor** O usuario poderá introducir as regras SCL que desexe mediante un campo de texto na que se poderá escribir. Tamén haberá un botón que permitirá gardar as sentenzas que escriba no sistema (Figura 16a).
3. **Avanzado** O usuario poderá introducir as funcións que desexe escribindo no campo de texto dispoñible. Ademáis haberá un botón que permita gardar toda a lóxica introducida no sistema (Figura 16b).

Destacar que na parte inferior da sección de configuración haberá un ítem interactivo a modo de guía que lle permita ao usuario saber en que fase da configuración se atopa.



Figura 16. Pantallas de configuración (III)

5. Casos de Proba

Co obxectivo de comprobar que a implementación realizada é correcta, resulta conveniente ter definidos algúns casos de proba, é dicir, situacións que se utilizarán para comprobar que a funcionalidade que fomos describindo ata o de agora está presente no sistema. Deste xeito, presentamos a continuación os casos de proba que nos pareceron axeitados para cada un dos caso de uso que describimos previamente.

5.1. Caso de Proba 1

- **Caso de Uso:** Rexistrar SO.
- **Funcionalidade a probar:** O sistema introduce os datos do SO na SID ao ser válidos.
- **Estado Inicial:** Escollemos a opción para rexistrar novos SOs e estamos en condicións de introducir os datos dun novo SO.
- **Entrada:** Insertamos un nome correcto e unha dirección IP válida.
- **Saída Esperada:** O sistema comproba que se introduciu algo correcto no campo do nome e da IP, polo que se gardarán os datos e se informará de que todo rematou correctamente.

5.2. Caso de Proba 2

- **Caso de Uso:** Rexistrar SO.
- **Funcionalidade a probar:** O sistema non introduce os datos do SO na SID porque se introduciu a **información** de xeito incorrecto.
- **Estado Inicial:** Escollemos a opción para rexistrar novos SOs e estamos en condicións de introducir os datos dun novo SO.
- **Entrada:** Insertamos incorrectamente o nome ou a IP (non se pon nada, introdúcese IP no formato incorrecto...).

- **Saída Esperada:** O sistema atopa de que hai erros nos campos postos polo usuario e informa do erro, rematando así a operación.

5.3. Caso de Proba 3

- **Caso de Uso:** Eliminar SO.
- **Funcionalidade a probar:** O sistema elimina a información do SO de todas as bases de datos.
- **Estado Inicial:** Escollemos a opción do menú para eliminar un SO e estaremos en condicións de seleccionar un deles para eliminalo.
- **Entrada:** Escollemos un dos SOs amosados para eliminar.
- **Saída Esperada:** O sistema accede a **todas as bases de datos** e elimina a información do SO escollido, rematando así esta operación.

5.4. Caso de Proba 4

- **Caso de Uso:** Seleccionar SO.
- **Funcionalidade a probar:** O usuario ten SOs rexistrados e decide activar un, polo que o sistema envía a dito SO unha petición de información que se gardará na DID.
- **Estado Inicial:** Escollemos a opción do menú para seleccionar un SO, e estaremos en condicións de escoller un deles.
- **Entrada:** Escollemos un SO e activámolo.
- **Saída Esperada:** O sistema comproba que se ten que activar o SO, enviando unha solicitude de información ao mesmo que almacena na DID, e avisando de que rematou correctamente.

5.5. Caso de Proba 5

- **Caso de Uso:** Seleccionar SO
- **Funcionalidade a probar:** O usuario ten SOs rexistrados e decide desactivar un, polo que o sistema simplemente deixa de consideralo activo.
- **Estado Inicial:** O usuario escolle a opción de seleccionar dende o menú da aplicación e ten a posibilidade de escoller un dos SO.
- **Entrada:** Escollemos un SO e desactivámolo.
- **Saída Esperada:** O sistema comproba que se ten que desactivar o SO e así o fai, informando que se desactivou correctamente.

5.6. Caso de Proba 6

- **Caso de Uso:** Seleccionar SO
- **Funcionalidade a probar:** O sistema comproba que non hai SO rexistrados, polo que non permite a selección.
- **Estado Inicial:** O usuario selecciona a opción de seleccionar dende o menú da aplicación e ten a posibilidade de escoller un dos SO.
- **Entrada:** O usuario entrou nesta opción sen ter rexistrado ningún SO previamente.
- **Saída Esperada:** Infórmase ao usuario de que non hai SOs rexistrados, polo que non se pode seleccionar.

5.7. Caso de Proba 7

- **Caso de Uso:** Configurar SO (con calquera extensión)
- **Funcionalidade a probar:** O sistema non permite configurar nada posto que non hai SOs activos para poder configurar.
- **Estado Inicial:** Escollemos a opción de configurar dende o menú da aplicación e poderemos elixir o modo de configuración.
- **Entrada:** Escollemos un modo de configuración e un SO para configurar sen ter activado ningún SO previamente.
- **Saída Esperada:** O sistema dase de conta de que non hai ningún SO activo, polo que avisa ao usuario desta situación e impide a configuración.

5.8. Caso de Proba 8

- **Caso de Uso:** Configurar SO por formulario
- **Funcionalidade a probar:** O sistema procesa correctamente os campos introducidos polo usuario ao ser correctos e almacena a lóxica difusa xerada na FD.
- **Estado Inicial:** Escollemos a opción de configurar un SO, e seleccionamos o modo por formulario.
- **Entrada:** Dentro do modo por formulario, o usuario introduce os campos de forma correcta.
- **Saída Esperada:** O sistema comprobará que a información é válida, xera a lóxica difusa e almacénala na FD.

5.9. Caso de Proba 9

- **Caso de Uso:** Configurar SO por formulario
- **Funcionalidade a probar:** O sistema non xera lóxica algunha, posto que se introduciu información de forma incorrecta.
- **Estado Inicial:** Escollemos a opción de configurar un SO, e seleccionamos o modo por formulario.
- **Entrada:** Introducimos algún dos campos do formulario de forma incorrecta.
- **Saída Esperada:** O sistema comprobará que hai erros nos campos introducidos polo usuario e informa deles, rematando a operación.

5.10. Caso de Proba 10

- **Caso de Uso:** Configurar SO mediante un editor
- **Funcionalidade a probar:** O sistema procesa correctamente as regras introducidas ao ser correctas e almacena a lóxica difusa xerada na FD.
- **Estado Inicial:** Escollemos a opción de configurar un SO, e seleccionamos o modo por editor.
- **Entrada:** Insertamos o código SCL de maneira correcta.
- **Saída Esperada:** O sistema comprobará que a información e a sintaxe sexan correctas, xerará a lóxica difusa, e almacenará dita lóxica na base de datos, informando do éxito da operación.

5.11. Caso de Proba 11

- **Caso de Uso:** Configurar SO mediante un editor
- **Funcionalidade a probar:** O sistema non xera lóxica difusa nin a almacena posto que non se introduciron sentenzas SCL válidas.
- **Estado Inicial:** Escollemos a opción de configurar un SO, e seleccionamos o modo por editor.
- **Entrada:** Insertamos un código SCL incorrectamente (con erros de sintaxe).
- **Saída Esperada:** O sistema comprobará que hai erros na sintaxe do código SCL escrito polo usuario e informa do erro, rematando así a operación.

5.12. Caso de Proba 12

- **Caso de Uso:** Configurar SO no modo avanzado
- **Funcionalidade a probar:** O sistema garda a lóxica difusa introducida porque é correcta.
- **Estado Inicial:** Escollemos a opción de configurar un SO, e seleccionamos o modo avanzado.
- **Entrada:** Introducimos lóxica difusa correctamente.
- **Saída Esperada:** O sistema comprobará que todo é correcto e gardará a información na base de datos FD, indicando que todo rematou correctamente.

5.13. Caso de Proba 13

- **Caso de Uso:** Configurar SO no modo avanzado
- **Funcionalidade a probar:** O sistema non garda a lóxica difusa introducida porque é incorrecta.
- **Estado Inicial:** Escollemos a opción de configurar un SO, e seleccionamos o modo avanzado.
- **Entrada:** O usuario introduce lóxica difusa incorrectamente, con algún erro.
- **Saída Esperada:** O sistema comprobará que a lóxica introducida polo usuario non é correcta e informa do erro, rematando así a operación.

5.14. Caso de Proba 14

- **Caso de Uso:** Planificar actuación
- **Funcionalidade a probar:** O sistema prepara todos os SOs por activación da alarma.
- **Estado Inicial:** A alarma actívase e avisa ao sistema para comezar a configuración.
- **Entrada:** -
- **Saída Esperada:** O sistema recolle toda a lóxica difusa e información dos SOs para xerar os comandos de actuación que a continuación enviará aos SOs (se procede).

5.15. Caso de Proba 15

- **Caso de Uso:** Visualizar información
- **Funcionalidade a probar:** O sistema amosa información xeral de todos os SOs activos correctamente.

- ### 5.16. Caso de Proba 16

- ## 6. Diagrama de Componentes



O diagrama de compoñentes é unha ferramenta que permite representar de forma estática o sistema de información. Este diagrama proporciona unha vista de alto nivel dos compoñentes do sistema, que poden ser dende software, unha base de datos, unha interfaz de usuario ou ata unha compoñente hardware como un circuito ou un dispositivo.

No noso proxecto faremos un diagrama de compoñentes que represente unha visión completa do sistema (maioritariamente a nivel do software), incluíndo tanto o propio sistema de control do fogar intelixente, o SCFI, como unha visión da rede de comunicación dos obxectos intelixentes, a GPSN.

Decidimos facer unha análise destes dous subsistemas xa que nos permite ter unha visión máis ampla de como o sistema principal interactúa co exterior mediante a comunicación cos obxectos intelixentes, e estes, á súa vez, cos seus sensores físicos e os seus actuadores.

Desta maneira, a continuación expoñemos os dous subsistemas e faremos un razoamento das decisións tomadas en canto ao por qué do deseño escollido.

6.1. GPSN: Rede de comunicación

Neste subsistema atópanse, principalmente, os obxectos intelixentes e os seus sensores virtuais. Este subsistema será o responsable de recoller os datos do exterior, do medio físico, ou enviar ordes aos actuadores. Ademais fará de ponte co SCFI transmitindo os datos entre o exterior e o sistema principal.

A continuación describimos os compoñentes do subsistema GPSN:

- **Sensor Virtual:** Encargado de recibir a información do sensores físicos. Este sensor comunícase co SO enviando os datos das medicións a través da interface para a lectura de datos. Isto permítelle aos dous compoñentes comunicarse, ofrecendo o sensor a funcionalidade de transmitir os datos obtidos dos sensores físicos e facendo que o SO poida obter ditos datos.
- **SO:** Encárgase da comunicación cos sensores e actuadores do sistema. Esta compoñente recibe as medicións do sensor virtual, envía comandos de actuación aos actuadores e comunícase bidireccionalmente co SCFI, ao cal lle envía os datos recollidos dos sensores e recibe os comandos de actuación.

Desta forma reflexamos o que consideramos como unha compoñente necesaria para o sistema, xa que dalgunha forma é a base sobre a que se cimenta a funcionalidade do SCFI.

6.2. SCFI: Sistema de Control do Fogar Intelixente

Este subsistema supón a parte máis importante do diagrama, posto que dentro deste teremos todos os compoñentes necesarios para controlar os obxectos intelixentes así como a interacción co usuario, ben sexa proporcionando interfaces para a entrada de datos ou interfaces para a saída de información por pantalla para que o usuario interactúe ou consulte, respectivamente.

A continuación describimos as distintas compoñentes deste subsistema:

- **Rexistrador:** Esta compoñente encárgase de rexistrar os datos dos SO que introduce o usuario (que serán o nome e a IP do SO que se quere rexistrar). A SID ofrece a funcionalidade de introducir datos na propia base, e reflexamos isto mediante a interface de rexistro dun SO.
- **Selector:** O selector encárgase de activar ou desactivar os SO. Comunícase ca SID para listar os SO dispoñibles mediante o uso da interface que permite consultar os

obxectos gardados na mesma. Tamén se comunica co Transmisor-Receptor, para que este lle indique ao SO seleccionado que se necesita obter a información. Reflexamos que esta información recibida, xunto coa confirmación de que o sensor se atopa activo, se almacena na DID mediante a interface de almacenar datos ofrecida pola mencionada base de datos.

- **Transmisor/Receptor:** Encárgase da comunicación cos SOs, transmítelle comandos de actuación no formato ActuadorML e recibe os datos recollidos en SensorML. É empregado polo Visualizador, polo Selector e polo Planificador para recibir datos dos SOs, e polo Planificador tamén para enviar datos directamente aos SOs. Esta compoñente ofrece a funcionalidade de transmitir ordes e solicitar medidas dos SOs en tempo real mediante mensaxes ActuadorML, representadas coas súas respectivas interfaces.
- **Planificador:** Encárgase de enviar os plans de actuación aos SO. Para iso comunícase co Transmisor/Receptor, ao cal lle envía ditas instrucións. Por outro lado tamén precisa de ler datos da Base de Datos Difusa, para coñecer as funcións de membresía e a lóxica difusa que permiten controlar os SOs, e faino co interface que esta ofrece para ler. Anotar neste punto que tivemos que duplicar esa compoñente do sistema para poder evitar cruces de liñas que fixesen o diagrama totalmente ilexible.
- **Visualizador:** Encárgase de amosar os datos dos SOs aos usuarios. Para iso precisa ler os datos recollidos na SID e tamén obter medidas dos SOs. Esta comunicación faise a través das interfaces xa mencionadas anteriormente, como a ofrecida polo receptor para obter medidas. Ademáis o visualizador comunícase coa SID para poder amosar a información básica dos SOs. Tamén destacar que o visualizador ofrece o interface de saída de información para o usuario.
- **Bases de datos:** No caso das bases de datos pensamos que é máis adecuado ter unha compoñente para cada base en lugar de un subsistema que agrupe todas elas, posto que nos permite visualizar a interacción coas demais compoñentes dunha forma máis clara. Deste xeito diferenciamos:
 - **SID:** Esta compoñente do sistema almacena a información estática (IP,nome) dos SO. O Rexistrador envíalle a información de rexistro, a cal é almacenada nesta base de datos. Ofrece a funcionalidade de ler e escribir os datos estáticos dos SO.
 - **DID:** Esta compoñente almacena a información dinámica dos SOs. Ofrece a funcionalidade a través das interfaces de ler e escribir na propia base.
 - **FD:** Esta compoñente almacena as funcións de membresía e regras difusas, e ofrece coas súas interfaces, tanto ao configurador como ao planificador, a funcionalidade de escribir e ler nela. Destacar que para evitar o cruce de relacións no diagrama decidimos que era mellor representar dúas veces a compoñente FD, de forma que ambas representan o mesmo e simplemente permiten unha mellor lectura do diagrama.

Poderíase tamén considerar a opción de eliminar na base de datos, pero dada a complexidade do diagrama e para evitar introducir demasiadas relacións e sobrecargar o diagrama, pensamos que é mellor deixala aparte nesta descrición de compoñentes.

- **Configurador:** Pensamos que é conveniente representar esta compoñente como un subsistema do propio SCFI, xa que dentro dela se levan a cabo outras accións

que caben ser destacadas. A función principal desta compoñente será recibir os datos introducidos polo usuario para a configuración dun SO e producir as regras difusas e funcións de membresía derivadas.

Dentro desta compoñente teremos outras dúas, que serán o verificador de sintaxe de SCL e o xerador Fuzzy. A continuación describimos estas dúas compoñentes:

- **Verificador de sintaxe:** Esta compoñente recibe o código SCL introducido polo usuario. Este código procésase e verifícase se a sintaxe do mesmo é correcta. Se é correcta, xérase un unha árbore de expresión, que será ofrecida ao xerador Fuzzy, o que representamos mediante a interface 'Xerar Árbore'.
- **Xerador Fuzzy:** O xerador fuzzy encárgase de xerar as funcións de membresía e o regras difusas que serán gardadas na Base de datos Fuzzy. Primeiro obtén a funcionalidade do verificador, o cal xera a árbore que esta compoñente convertirá en lóxica difusa. Despois, ofrece a funcionalidade de xerar a lóxica difusa, que será almacenada polo configurador na base de datos correspondente grazas a outra interfaz ofertada pola FD: 'InsertarDifusa'.

7. Conceptos susceptibles de ser clases

O seguinte paso no noso proceso de deseño centrouse en dúas vertentes: por un lado, ir identificando algúns conceptos que foran susceptibles de converterse en clases para elaborar un primeiro diagrama e, polo outro, desenvolver diagramas de secuencia para cada un dos casos de uso.

Dado que estas dúas tarefas se foron levando a cabo dun xeito paralelo, presentaremos primeiro os conceptos que consideramos susceptibles de seren clases (xa que así o lector poderá entender as clases que se foron empregando nos diagramas de secuencia), logo amosaremos os diagramas de secuencia asociados e, finalmente, presentaremos o diagrama de clases que construímos á par que fomos desenvolvendo os de secuencia.

Comentar que as clases que se van a amosar a continuación non se definiron nunha primeira iteración, senón que moitas delas foron xurdindo a medida que avanzamos no proceso de deseño. Tamén, tivemos en conta as ideas presentadas na clase e que se recollen no campus virtual sobre patróns de deseño. Concretamente:

- Tentamos realizar unha separación entre a parte de interacción co usuario, as respostas aos seus comandos e o control interno do sistema seguindo o patrón MVC.
- Decidimos representar as clases que modelan as bases de datos como clases DAO (*Data Access Object*).
- Finalmente, tamén tentamos na medida do posible seguir o patrón Fachada, incluíndo clases que conectan con outras e realizan diferentes xestións.

Deste xeito, imos amosar unha pequena descrición das clases que consideramos que, de entrada, deberíamos ter no SCFI, para o que distinguiremos algúns grupos:

- Por unha banda, diferenciamos claramente a parte da interfaz gráfica. Teremos, ademáis dunha `FachadaGUI`, outras tres clases que se corresponden ademais cos tres grandes grupos de pantallas amosados no Apartado 4:
 - `VentaXestion`: para a parte de rexistro, eliminación e selección de SOs.

- `VentaConfiguracion`: para a parte de configuración de SOs.
- `VentaVisualizacion`: para a parte de visualización da información.
- Por outro lado, temos claro tamén que hai que distinguir a base de datos do demais. Isto levámolo a cabo introducindo unha clase `FachadaBD` que se comunicará cos DAO:
 - `DAOSID`: para acceso á SID (Información estática).
 - `DAODID`: para acceso á DID (Información dinámica).
 - `DAOFD`: para acceso á FD (Lóxica difusa).
- Temos outra fachada, chamada `FachadaXestion`, que se irá comunicando con outras compoñentes xa descritas e que se atopará máis enfocada na parte de xestión de SOs (de aí o seu nome).
- A última fachada das que consideramos inicialmente é a `FachadaConfiguracion` e, como o seu propio nome indica, estará enfocada na parte de configuración e planificación interna. Consideramos conveniente facer esta distinción para non sobrecargar unha única clase con toda a información. Ademais, estará vinculada con outras dúas clases:
 - `Configurador`: é o que se adicará á parte de configuración. Nesa clase atoparanse métodos propios da compoñente homónima: verificar a sintaxe SCL e xerar as regras difusas.
 - `Planificador`: estará exclusivamente enfocado na tarefa de planificación, a partir das regras difusas e datos do SO a planificar.
- Finalmente, decidimos distinguir unha clase pola importancia da compoñente que representa, o SO, que será empregada dende outras para o almacenamento de información como o nome ou a IP, e para a comunicación cos SOs.

Recalcar finalmente que estas clases foron obtidas á par que os diagramas de secuencia e, en iteracións posteriores, seguiremos redefinido e ampliando para adaptar o noso sistema a novos patróns.

8. Diagramas de Secuencia

Pasamos agora a amosar os diagramas de secuencia. Para comezar con este apartado, pensamos que é importante ter en conta as seguintes consideracións:

- É relevante a lectura dos textos asociados a cada un dos diagramas, xa que se incorpora tanto a explicación das decisións tomadas coma as omisións realizadas, igualmente argumentadas.
- Incorporouse un diagrama non existente coma caso de uso, isto pretende xeralizar parte dos deseños evitando redundancias.
- Do mesmo xeito, decidimos que os diagramas comezan a partir do estado no que os actores indicaron xa o que desexaban facer, o que representamos como un sinal que se envía do actor á primeira clase de cada diagrama.
- É habitual tamén que se reflexe un retorno ao final de todo cando o actor é o usuario a modo de mensaxe de confirmación. Cremos que é preciso indicar que se realiza unha confirmación da operación ao usuario, por iso o poñemos.

8.1. Recuperar SO

Este diagrama de secuencia empregáremolo de forma recorrente coma parte doutros diagramas asociados a distintos casos de uso, dada a frecuencia coa que se van a querer realizar estas operacións. Na Figura 18, pódese consultar sobre el.

Salientar que o que se reflexa neste caso é un proceso mediante o cal se recuperan os SO da base de datos SID. Con eses datos vanse creando instancias de SOs e consúltase se están activos ou non (accedendo á DID), o que se almacena tamén en cada unha das instancias creadas.

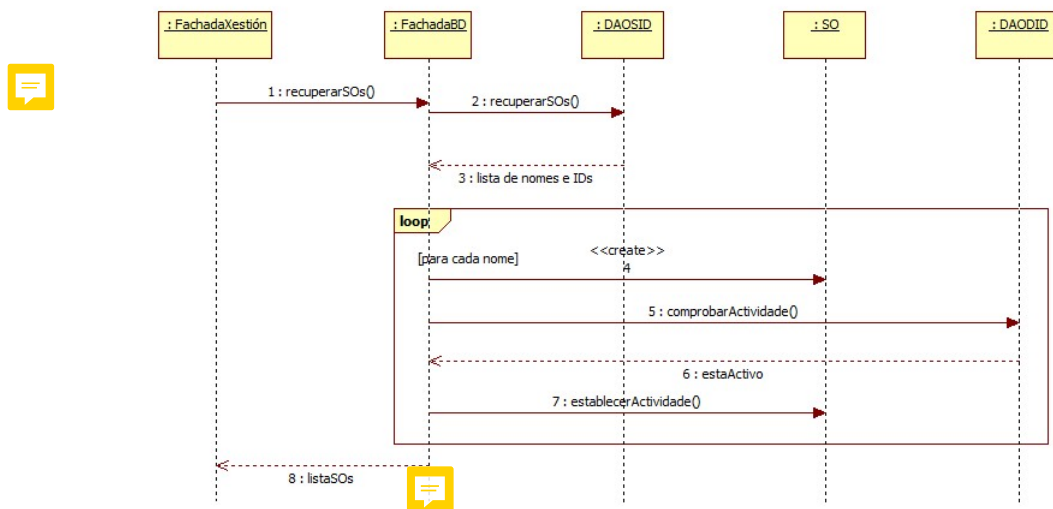


Figura 18. Diagrama de secuencia auxiliar 'Recuperar SO'

8.2. Rexistrar SO

Pódese consultar na Figura 19 o diagrama asociado ao caso de uso homónimo. Destacar algunhas consideracións tomadas:

- Na FachadaGUI realizase unha primeira comprobación xenérica, como sería o caso de que non se pasen parámetros baleiros. Esta ten unha escasa importancia polo que se decidiu omitila.
- O marco empregado para indicar o caso no que os datos introducidos sexan incorrectos e polo tanto pedilos de novo ao usuario. Neste sentido habería a posibilidade de indicar un bucle xa que na medida na que a información obtida de novo non sexa correcta reiterariase a acción, porén considerase que debe haber máis énfase no feito da alternatividade respecto da situación esperada. De feito, noutros diagramas non se incluíron comprobacións deste tipo para evitar demasiada complexidade.
- Dun xeito máis xenérico, comentar que este tipo de cuestións de control e de casos alternativos provocados por erros intentamos recollelos só nos casos nos que o diagrama non resultase demasiado complexo. Por iso, veranse outros casos nos que non se reflexan as validacións, para evitar sobrecargar os diagramas.

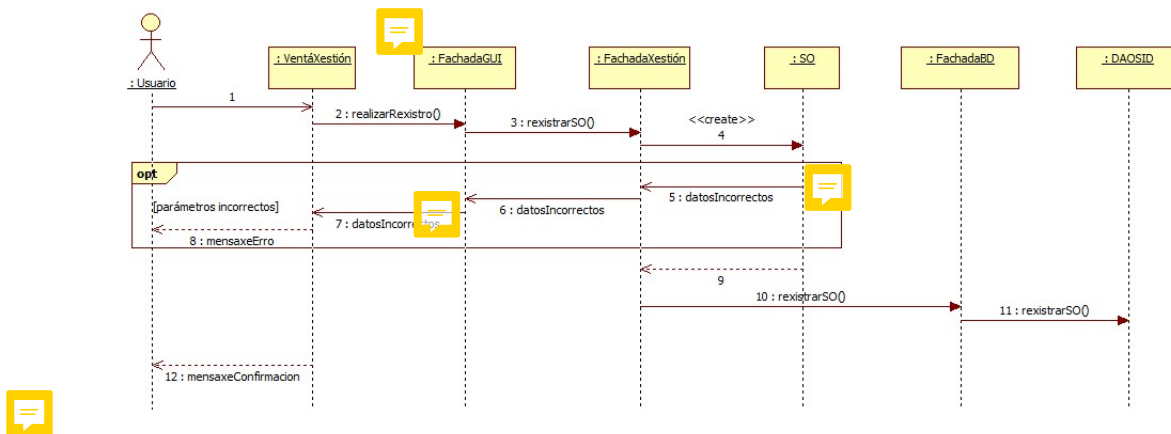


Figura 19. Diagrama de secuencia do caso de uso 'Registrar SO'

8.3. Eliminar SO

Entre as suposicións feitas no diagrama representado na Figura 20 está o feito de que consideramos o nome coma a clave primaria do obxecto (é dicir, que na base de datos poderemos identificar a información dun obxecto intelixente polo nome de dito obxecto). Nel, obsérvase como se fai referencia a 'Recuperar SO', o diagrama exposto ao comezo desta sección (posto que se necesita coñecer todos os SOs e seleccionar deles os inactivos).

Resaltar que neste diagrama observamos a xustificación do uso da clase `FachadaBD`. Mediante a operación desa clase `eliminarSO()`, chámanse a cada unha das clases que representan as bases de datos para levar a cabo a eliminación de todos os datos dun SO do sistema.

Finalmente, resulta preciso indicar que co primeiro sinal do usuario simbolizamos que este accedeu á opción de eliminar un SO na parte de xestión, e co segundo que selecciona un SO para ser desactivado. Usaremos normalmente ese tipo de sinais para reflexar que o usuario intervén dalgunha maneira no caso de uso.

8.4. Seleccionar SO

Neste caso empregaremos dous diagramas de secuencia: "Activar SO" e "Desactivar SO", por entender que se atopan suficientemente diferenciados (e así facilitar a súa comprensión) e por tratarse de dúas alternativas que consideramos realmente preciso representar, para.

8.4.1. Activar SO

No diagrama da Figura 25, debemos destacar dúas decisións:

- Dous accesos a base de datos, en primeiro lugar para recuperar toda a información e en segundo lugar para obter a IP coa que crear a instancia do SO e executar o resto da activación. Esta aparente redundancia vén dada polo feito de que calquera alternativa aumentaba a complexidade ou comprometía fortemente a reutilización de código e algunhas das ideas sobre patróns de deseño introducidas previamente.

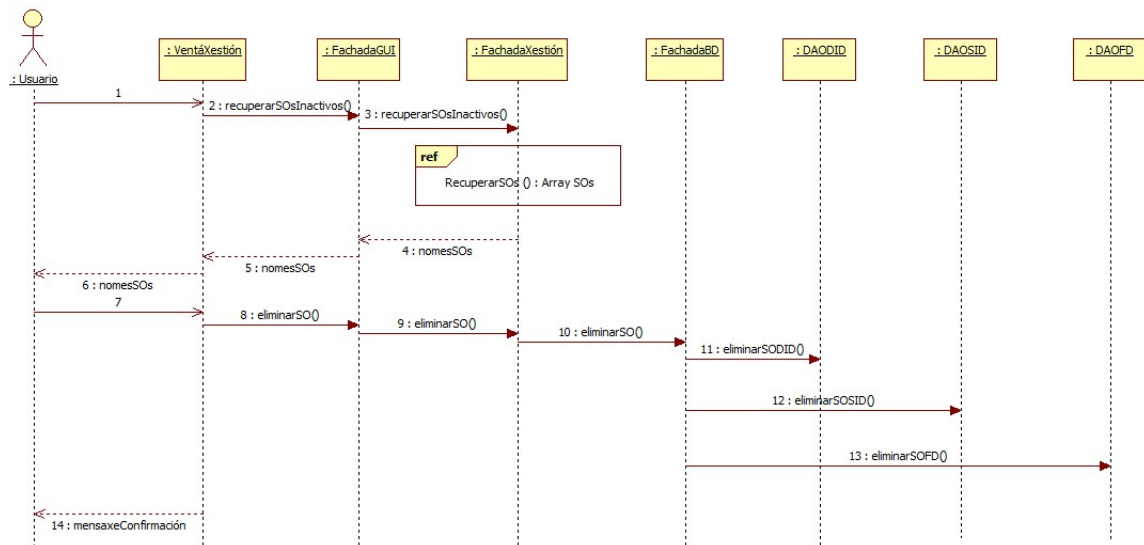


Figura 20. Diagrama de secuencia do caso de uso 'Eliminar SO'

- Matizar que a información devolta polo SO recollerase na aplicación con algún tipo de dato axeitado para o seu correcto manexo. Dadas estas circunstancias, simplemente se recolle que esa información se almacenará nalgún contedor de datos (array) durante a súa estancia na aplicación.

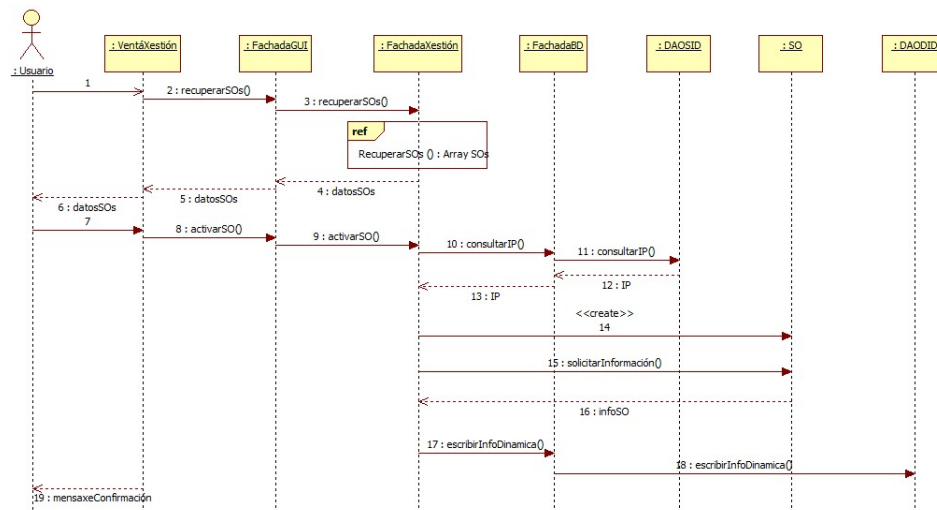


Figura 21. Diagrama de secuencia 'Activar SO' dentro do caso de uso 'Seleccionar SO'

8.4.2. Desactivar SO

O único que convén salientar respecto do diagrama da Figura 22, é que só se almacena o feito de que o SO deixe de estar activo na DID. Polo demais, hai algunha similitude co caso anterior no comezo, e ideas que xa se foron comentando previamente, polo que non se redundará nelas.

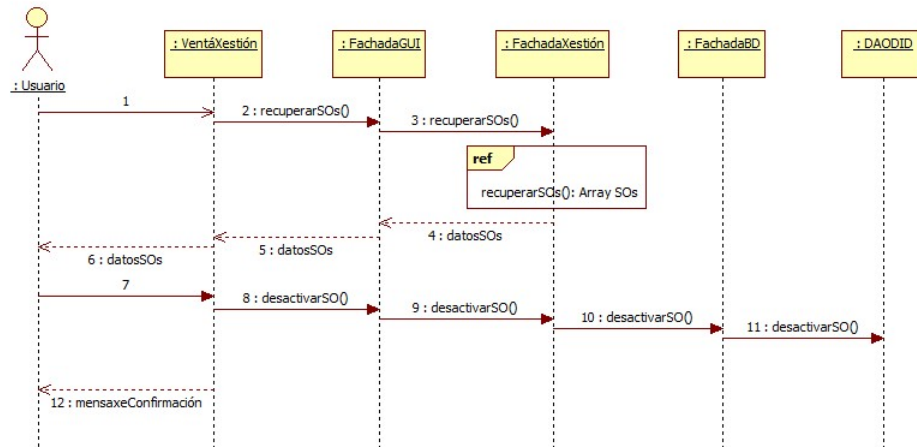


Figura 22. Diagrama de secuencia 'Desactivar SO' dentro do caso de uso 'Seleccionar SO'

8.5. Configurar SO

Para modelar a configuración dos SO empréganse tres diagramas de secuencia, un por cada extensión do caso de uso correspondente: "Configurar Formulario", "Configurar Editor" e "Configurar Avanzado", dado que albergan suficiente complexidade como para non ser viable unha representación en conxunto (si abordada no diagrama de actividade).

8.5.1. Configurar SO por Formulario

Este caso recollido na Figura 23 precisa aclarar que non consideramos nel a obtención de información acerca dos SO (que suporemos que xa se realizou con anterioridade) para centrarnos no desenvolvemento da configuración en específico e evitar un diagrama máis complexo. Incorporamos algunhas anotacións sobre a Figura 23.

- Os datos que poderían implicar unha maior complexidade de xestión coma tokens ou arbores decidimos xestionalos coma strings ou arrays de strings atendendo a cada situación. Non queremos entrar, de momento, en cuestións moi concretas sobre implementación.
- Realizáronse dous accesos á base de datos: un primeiro cando comeza o caso de uso para amosar ao usuario os campos do formulario (que posto que é unha única consulta pensamos que é coherente que pase pola *FachadaXestión*), e un segundo xa cando nos atopamos na *FachadaConfiguracion*, no que se colle dita información para o proceso de xeración da lóxica difusa (que o precisa). Tal e como acontecía nun apartado previo no que se realizaban dous accesos que poderían parecer 'redundantes', preferimos deixalo así para evitar dependencias entre diferentes partes do sistema.

8.5.2. Configurar SO por Editor

Este caso que se pode ver na Figura 24, é moi semellante ao anterior, aínda que non se xera a sintaxe SCL, pois xa é introducida directamente polo usuario, nin se fai a recupe-

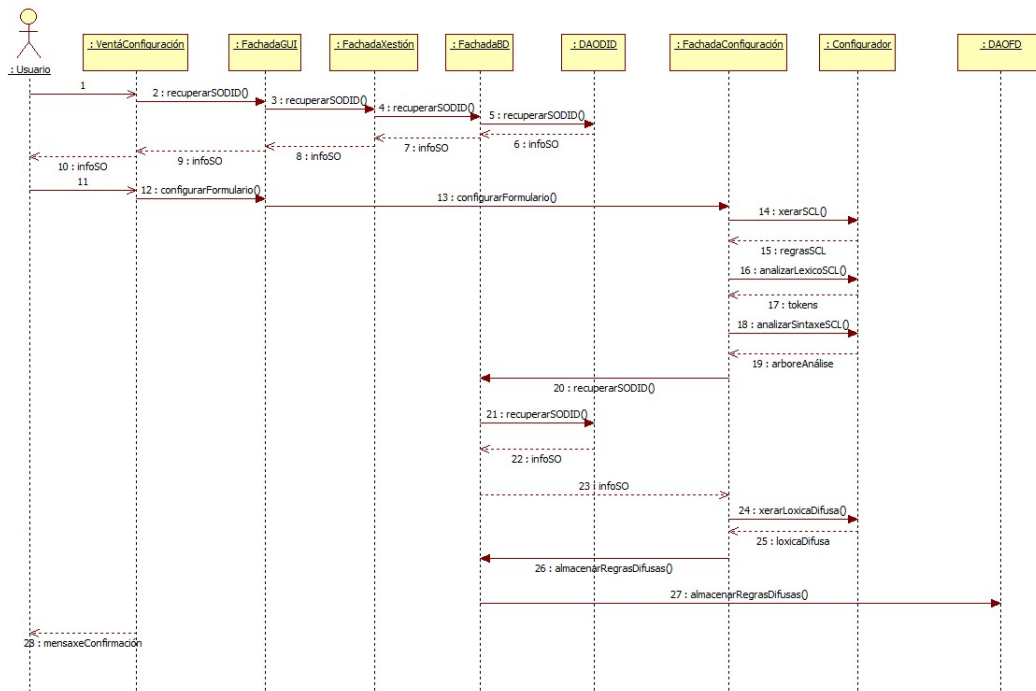


Figura 23. Diagrama de secuencia do caso de uso 'Configurar SO por Formulario'

ración inicial de datos do SO porque non hai que elaborar ningún formulario.

8.5.3. Configurar SO no modo avanzado

Neste caso recollido na Figura 25 é salientable que desaparece a maior parte da complexidade respecto dos anteriores dado que non hai que converter a entrada do usuario (trátase directamente da lóxica difusa).

25.

8.6. Planificar actuación

Representado na Figura 26, expónse para un SO concreto, aínda que en realidade faríase de todos os SO activos, o que foi motivado pola complexidade da propia secuencia para un só caso e tendo en conta que sería intuitivo realizalo con todos eles.

O único que se pode resaltar a maiores é o feito de que consideramos a opción de ter que enviar (ou non) comandos de actuación ao SO, dúas alternativas do caso de uso reflexado, por iso engadimos ese pequeno marco ao final.

Tampouco enviamos mensaxe de confirmación nin participación de ningunha compoñente da interfaz posto que neste caso de uso non existe participación algunha por parte do usuario.

8.7. Visualizar información

O único salientable dentro do diagrama recollido na Figura 27, é o bucle de petición de información aos SO e o almacenamento na base de datos da mesma (motivado pola

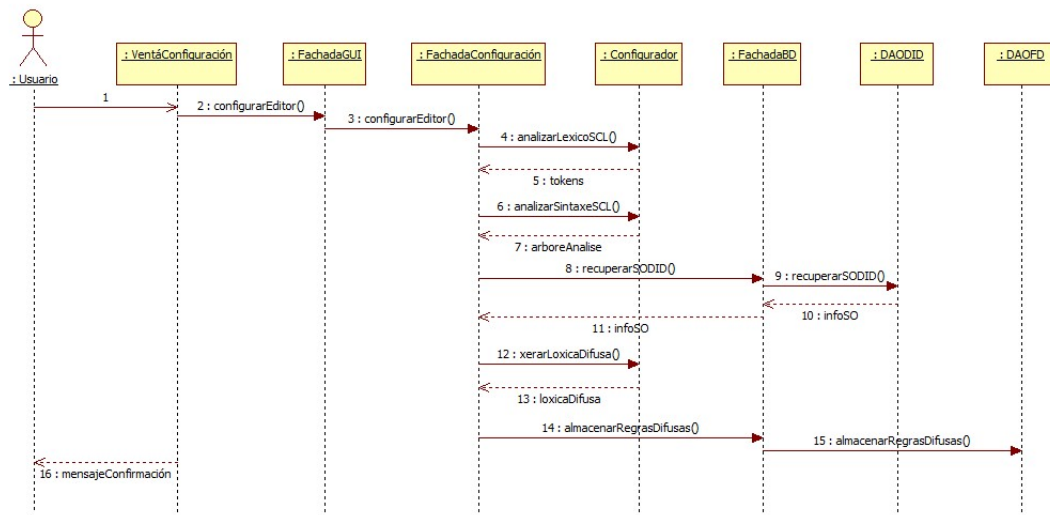


Figura 24. Diagrama de secuencia do caso de uso 'Configurar SO por Editor'

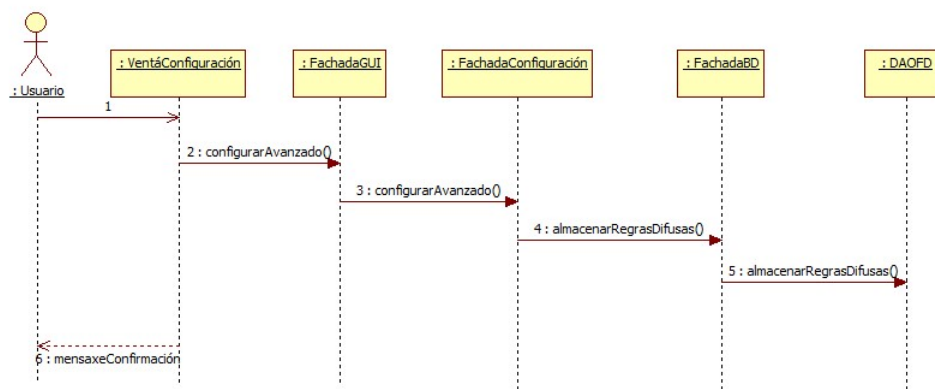


Figura 25. Diagrama de secuencia do caso de uso 'Configurar SO no modo avanzado'

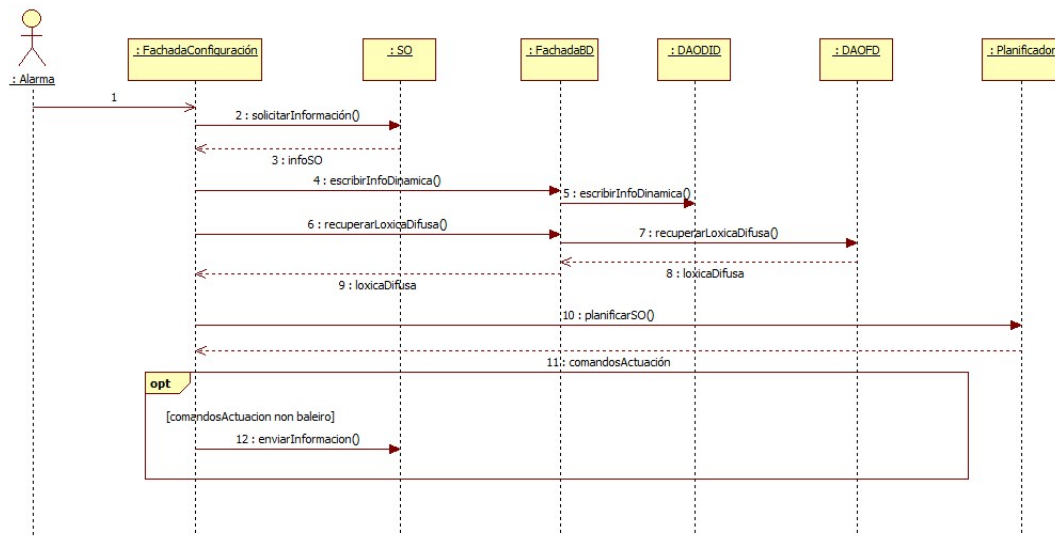


Figura 26. Diagrama de secuencia do caso de uso 'Planificar Actuación'

recepción de información).

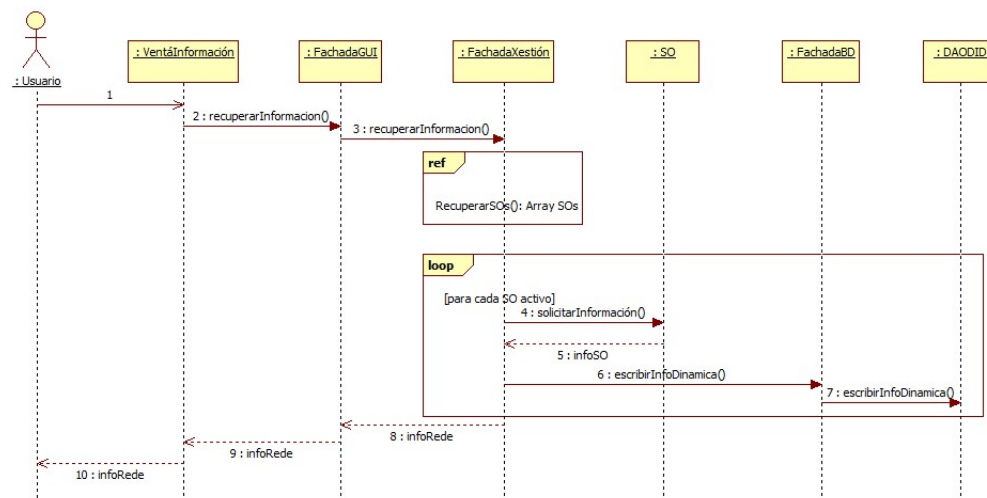


Figura 27. Diagrama de secuencia do caso de uso 'Visualizar información'

9. Diagrama de clases inicial

O diagrama de clase amosado na Figura 28 é resultado de avaliar as clases que van sendo precisas a través de cada un dos diagramas de secuencia realizados, e as relacións que existen entre as mesmas (co que se foron modelando as asociacións e dependencias que se amosan). Para o deseño do mesmo seguimos un modelo MVC aínda que nalgún caso algunha das clases situese entre dúas das partes do modelo. Empregouse ademais cores afines para as clases que teñen unha maior afinidade.

Destacar por enriba de todo o demais a clase SO, que relacionamos con varias fachadas mediante dependencias. Isto modelámolo así porque esa clase é empregada dende dife-

rentes operacións das fachadas (non cremos que o vínculo sexa a un nivel estrutural como para representalo mediante unha asociación).

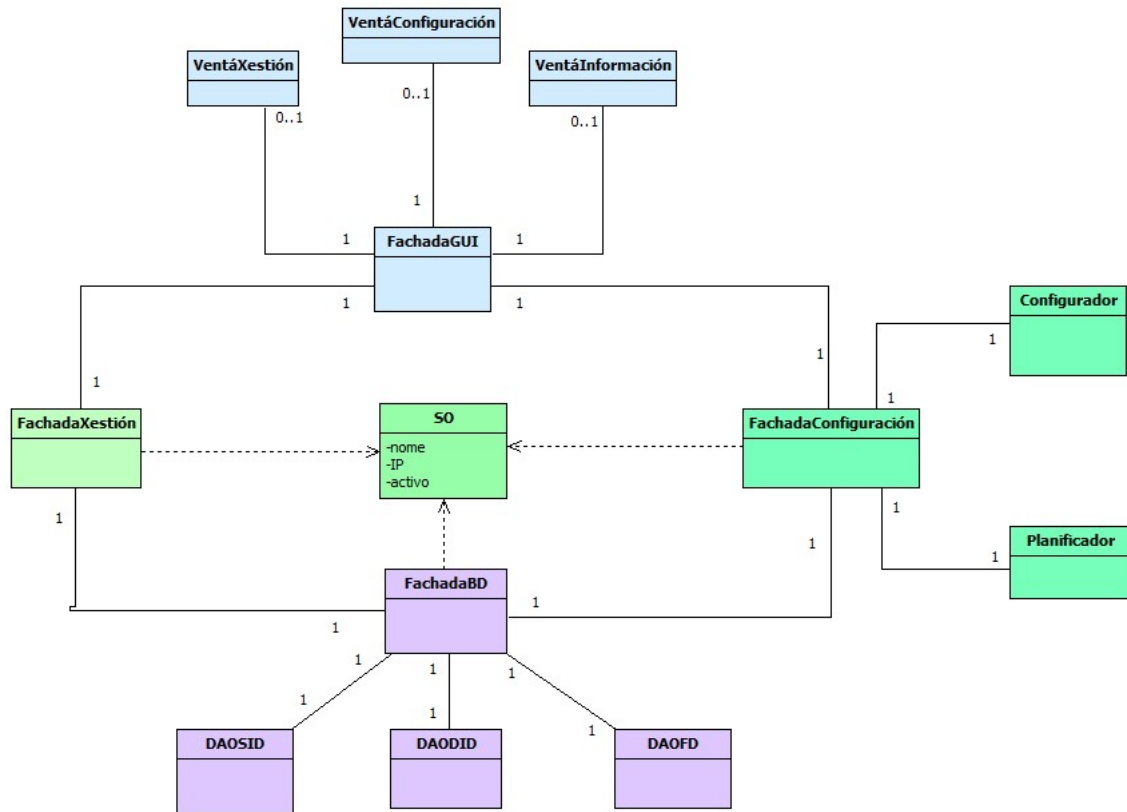


Figura 28. Diagrama de Clases

Referencias

- [1] IEEE, “SITE: The Simple Internet of Things Enabler for Smart Homes“ [en liña].
Dispoñible en: <https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=7831376> [data de consulta 20/02/2020].
- [2] Microsoft, “Model-View-Controller“ [en liña].
Dispoñible en: <https://docs.microsoft.com/en-us/previous-versions/msp-n-p/ff649643> [data de consulta 17/03/2020].
- [3] Best Practice Software Engineering, “Facade Pattern“ [en liña].
Dispoñible en: <http://best-practice-software-engineering.ifs.tuwien.ac.at/patterns/facade.html> [data de consulta 17/03/2020].
- [4] Oracle, “Core J2EE Patterns - Data Access Object“ [en liña].
Dispoñible en: <https://www.oracle.com/technetwork/java/dataaccessobject-138824.html> [data de consulta 17/03/2020].
- [5] Visual Paradigm, “What is Component Diagram“ [en liña].
Dispoñible en: <https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-component-diagram/> [data de consulta 22/03/2020].
- [6] IBM, “The sequence diagram“ [en liña].
Dispoñible en: <https://developer.ibm.com/articles/the-sequence-diagram/> [data de consulta 27/03/2020].
- [7] IBM, “Tips for writing good use cases.“ [en liña].
Dispoñible en: <ftp://ftp.software.ibm.com/software/rational/web/whitepapers/RAW14023-USEN-00.pdf> [data de consulta 17/03/2020].
- [8] Fowler, M. (2004). UML distilled: A brief guide to the standard object modeling language. Boston: Addison-Wesley. [data de consulta 02/03/2020].