ESCUELA SUPERIOR DE INGENIERÍA INFORMÁTICA

INGENIERÍA INFORMÁTICA

CURSO ACADÉMICO 2010/2011

Proyecto de Fin de Carrera

Diseño de una Institución Electrónica para la coordinación de traslados inter-hospitalarios

Autor: Diego Fradejas Muñoz

Tutor: Holger Billhardt

Resumen

En este trabajo presentaremos el análisis de requisitos y el diseño de un prototipo de estructura organizativa para un sistema de coordinación de traslados inter-hospitalarios. Dicho diseño utilizará el paradigma de las Instituciones Electrónicas, en el contexto del demostrador mHealth del Proyecto Agreement Technologies (AT - *Tecnologías del acuerdo*).

El campo de la coordinación de traslados inter-hospitalarios se refiere a los traslados de pacientes entre hospitales. Las razones para tales traslados incluyen el que un hospital no pueda aplicar un determinado tratamiento, por ejemplo. Uno de los aspectos más importantes de un traslado es la decisión de qué hospital es el más adecuado para trasladar a él al paciente. Esta decisión debería ser tomada como un acuerdo entre los Hospitales involucrados, por lo que podría incluir procesos de negociación y argumentación.

Debido a este problema, en este trabajo se propone una aplicación que, mediante un sistema multiagente, aspira a automatizar este proceso de decisión, ahorrando costes tanto de tiempo como personales, los cuales podrían ser mejor invertidos en otras tareas.

Esta aplicación, una vez finalizada, será capaz de agilizar enormemente el proceso de traslado, reduciendo el tiempo empleado y teniendo un impacto en la calidad del servicio del sistema de sanidad, con los pacientes siendo trasladados al hospital más adecuado, mejorando su calidad de vida.

Este trabajo contiene las siguientes secciones principales:

Introducción: En el capítulo 1, se presentarán todos los conceptos necesarios para comprender el dominio y el contexto de la aplicación.

Objetivos: En el capítulo 2 se describirán los objetivos de la aplicación, así como las dos diferentes alternativas para implementarlo planteadas, dando las razones de la alternativa empleada finalmente.

Instituciones Electrónicas: En el capítulo 3, se describirá el paradigma de las Instituciones Electrónicas usado en este trabajo.

Descripción Informática: Esta sección está dividida en tres capítulos. En ellos se describirá la aplicación desde un punto de vista de ingeniería del software: primero, en el capítulo 4, presentaremos un análisis detallado de los requisitos y un diseño de alto nivel, seguido del diseño detallado en el capítulo 5, y la descripción de las pruebas realizadas para asegurar su correcto funcionamiento en el capítulo 6.

Conclusiones: Finalmente, en el capítulo 7 se presentarán los logros alcanzados y los posibles trabajos futuros a realizar expandiendo el presente proyecto.

Índice general

Capítulo 1

Introducción

En esta sección, se presentarán los conceptos necesarios para comprender el dominio de la aplicación.

1.1. Coordinación de traslados inter-hospitalarios

Este trabajo trata sobre traslados de pacientes entre hospitales. Las razones para tales traslados son variadas, incluyendo:

- que el tratamiento necesario para un paciente no esté disponible en el hospital en el que se encuentra actualmente,
- que el paciente prefiera ser tratado en otro hospital que, por ejemplo, esté más cerca de su casa, o que tenga buena reputación,
- que un hospital decida trasladar al paciente para evitar la saturación de su servicio.

En tales casos, el hospital en el que se encuentra el paciente hace una petición a las autoridades pertinentes, y son ellas las que deciden si el traslado se acepta o rechaza. Si se acepta, se decide a qué hospital se debe trasladar al paciente, basándose en el tratamiento requerido, el nivel de prioridad, etc. Finalmente, el paciente es trasladado a su destino por medio de un vehículo médico equipado con los medios necesarios para atenderle apropiadamente durante el traslado.

En la Comunidad Autónoma de Madrid, estos traslados son coordinados por el SUMMA-112. El SUMMA112 fue creado a partir de la unión de dos servicios sanitarios preexistentes: el SERCAM y el 061. Su misión es proporcionar asistencia sanitaria en urgencias, emergencias, catástrofes y situaciones especiales en la Comunidad de Madrid. Además, es responsable de gestionar la coordinación funcional entre diferentes niveles de asistencia. Actualmente, los servicios proporcionados por el SUMMA112 incluyen la recepción y

gestión de llamadas médicas y la gestión de las peticiones de asistencia, entre otros. Este trabajo está relacionado directamente con dos de estos servicios, el de traslado de pacientes con ambulancias convencionales, y el traslado de pacientes que requieran soporte vital avanzado.

Estos dos servicios están llevados a cabo por la mesa de enfermería, la cual es responsable de planear y desarrollar un plan de gestión que integre a los profesionales de la enfermería en el servicio de coordinación de emergencias del SUMMA112 [mdedlpwdS]. Sus servicios incluyen la repatriación de pacientes de otras comunidades, y la coordinación de los traslados inter-hospitalarios, el servicio en el cual se enfoca este trabajo.

En la actualidad, cuando un hospital decide que un paciente debe ser trasladado, realiza una petición telefónica a la mesa de enfermería del SUMMA112, la cual incluye el nivel de prioridad y posiblemente el tratamiento que se requiere. La tarea del centro de emergencias consiste en determinar a qué hospital debería ser trasladado el paciente. Para ello, la mesa de enfermería llama a varios hospitales (seleccionados porque prestan el servicio requerido) y les pregunta por su capacidad disponible (el número de plazas libres que tienen). Tras ello, se decide el hospital que recibirá al paciente, y se efectúa el traslado.

1.2. Contexto de la aplicación

1.2.1. Proyecto Agreement Technologies

Este trabajo se encuentra englobado en el proyecto Agreement Technologies (en adelante referido como el proyecto AT), que busca el desarrollo de modelos, marcos de desarrollo (*frameworks*), métodos y algoritmos para construir sistemas distribuidos abiertos a gran escala.

Este proyecto es parte de la línea estratégica CONSOLIDER del programa INGENIO del gobierno de España. Este programa intenta conseguir el objetivo de ayudar a España a converger con la Unión Europea en términos de renta per cápita, empleo y sociedad del conocimiento [wdpI].

Como parte de él, el proyecto AT trata de proponer un nuevo paradigma para sistemas distribuidos de nueva generación, estructurados en torno al concepto de acuerdo entre agentes computacionales. Estos acuerdos deben ser consistentes con el contexto normativo en el que se establecen y permitirán, una vez aceptados, que los agentes llamen a servicios mutuos y los cumplan. Autonomía, mobilidad, interacción y apertura son las características que este paradigma cubrirá desde una perspectiva teórica y práctica [dlpwdpA].

Este proyecto está siendo llevado a cabo por tres organizaciones:

- Universidad Rey Juan Carlos (URJC), A través del Centro para las Tecnologías

Inteligentes de la Información y sus Aplicaciones (CETINIA);

- Universitat Politècnica de València (UPV), a través del Departament de Sistemes Informàtics i Computació;
- Institut d'Investigació en Intelligència Artificial (IIIA), el cual es parte del Consejo Español de Investicación Científica (CSIC). Esta organización actúa no solamente como un equipo de investigación, sino como coordinador del proyecto.

1.2.2. Demostrador mHealth

Los avances realizados en el proyecto AT en el campo teórico son probados mediante tres demostradores, aplicaciones software que aplican dichos avances a problemas del mundo real: eProcurement, mWater and mHealth. Este trabajo es parte de este último, el cual trata situaciones relacionadas con la asistencia sanitaria fuera del hospital en urgencias y emergencias médicas.

En el dominio de la salud, la gestión de las urgencias médicas tiene un gran impacto social dado la amenaza inminente a la vida o salud prolongada del paciente. Tales circunstancias extremas requieren el uso de los recursos apropiados en un tiempo de respuesta limitado para poder proporcionar una asistencia eficiente. El empleo de los avances en Informática para mejorar la prestación de servicios sanitarios es un área que tiene un potencial significativo [LIB08, CFB+09, CFBO09]. Recientemente, la noción del mHealth se ha vuelto prominente, enfocándose en aplicaciones que proporcionan asistencia sanitaria en tiempo real a personas en cualquier lugar y en cualquier momento mediante comunicaciones inalámbricas y de banda ancha, así como dispositivos móviles [CFO05].

El demostrador mHealth trata con la prestación de servicios sanitarios de alta calidad para apoyar la asistencia sanitaria regular en caso de urgencias. En particular, intenta proporcionar valor añadido tanto a pacientes como a los profesionales médicos. Dentro del ámbito de aplicación del proyecto AT, tiene dos propósitos fundamentales:

- La utilización y evaluación de resultados de investigación –algoritmos, técnicas, métodos y modelos– producidos en el proyecto AT en una situacion real.
- Construir prototipos de aplicaciones que aborden problemas reales y, así, tengan un gran potencial de evolucionar eventualmente a una aplicación comercial.

Desde el punto de vista de los profesionales médicos, podrían ser liberados de ciertas tareas rutinarias de decisión y negociación que actualmente son desempeñadas por humanos pero podrían ser delegadas a agentes artificiales. Tales tareas incluirían, por ejemplo, encontrar un hospital apropiado para un paciente de urgencias. Finalmente, el aumentar las posibilidades de intercambio de datos médicos bajo demanda entre diferentes partes (por ejemplo, los historiales médicos) será beneficioso para el sistema sanitario en su

conjunto, dado que permite una atención más personalizada y, por tanto, más efectiva, pudiendo reducir costes y consumo de recursos evitando duplicados de pruebas médicas.

Se identificaron tres posibles aplicaciones en el demostrador mHealth: *Transporte médico de urgencia*, *Coordinación de traslados inter-hospitalarios*, y *autoayuda para los pacientes*. Estas posibles aplicaciones corresponden a tareas que son conseguidas por el SUMMA112 en condiciones de operación normales (lo que excluye situaciones extraordinarias como castástrofes) y que podrían ser potencialmente mejoradas mediante la aplicación de las tecnologías que serán estudiadas en el proyecto AT.

De dichas aplicaciones identificadas en el demostrador mHealth, este trabajo está centrado en el segundo de ellos, la *Coordinación de traslados inter-hospitalarios*.

Capítulo 2

Objetivos

En este capítulo se describirá el problema a resolver, las diferentes alternativas posibles para desarrollar la aplicación que lo resuelve, y las razones de la elección realizada.

2.1. Descripción del problema

El problema presentado en este trabajo consistía en diseñar una estructura organizativa para un sistema multiagente que automatizase la coordinación de traslados interhospitalarios, presentada en la sección 1.1.

Los requisitos del sistema eran los siguientes:

- Debía ser abierto, esto es, debía permitir actuar dentro de él a agentes que no necesariamente estarían implementados por los mismos desarrolladores de la estructura organizativa. Por tanto, cualquier agente, implementado de forma arbitraria, sería capaz de interactuar con este sistema, siempre que siguiera las normas establecidas en él.
- La estructura se limitaría a definir las normas utilizadas por los agentes para interactuar, y permitiría toda interacción que siguiera dichas normas. En otro caso, el sistema debería impedir que ocurrieran dichas interacciones.
- La estructura solo debía describir el proceso general, permitiendo intercambiar los procesos de decisión, concretamente, el de la decisión del hospital de destino del paciente, a fin de permitir futuras investigaciones sobre la adecuación de diferentes alternativas de dicho proceso al problema, permitiendo clasificarlas en función de su velocidad o su eficiencia.
- El sistema debía tener en cuenta las preferencias (si las hubiera) tanto del paciente como de los hospitales, intentando lograr la satisfacción de todas las partes, asegurando a su vez cierto nivel de equidad en la distribución de pacientes, de manera

que, de ser necesario, los pacientes serían transferidos a un hospital adecuado, aunque dicho hospital no tuviera especial preferencia por él, evitando a su vez que dicho hospital sólo recibiera pacientes que no quisiera.

Debido a estos requisitos, se decidió que la estructura organizativa a diseñar tomaría forma de Institución Electrónica, paradigma para el desarrollo de sistemas multiagente desarrollado por el IIIA, que se describirá en detalle en el capítulo 3.

2.2. Alternativas

Había dos alternativas en cuestión de frameworks para implementar la estructura organizativa:

- JADE, Framework de desarrollo de sistemas multiagente basado en java,
- EIDE (*Electronic Institutions Development Environment*), entorno de desarrollo concebido para soportar el paradigma de las Instituciones Electrónicas, creado por el IIIA.

JADE tenía la ventaja de ser interoperable en la mayoría de las plataformas, dado su base de java, así como ser capaz de comunicarse con otras aplicaciones java. También estaba el hecho de que había mucha más documentación sobre JADE (tutoriales, el JavaDoc de todas las librerías y clases, y manuales de diversas fuentes, siendo más sencillo encontrar la mejor solución a un problema dado) que sobre EIDE (que contaba sólo con el JavaDoc y la ayuda creada por los desarrolladores –en ocasiones, escasa–, con lo que no había demasiadas alternativas a la hora de buscar la solución a un problema dado). Por último, JADE tiene la ventaja de que se puede desarrollar directamente mediante un IDE conocido como puede ser Eclipse, mientras que para usar EIDE, las herramientas eran desconocidas.

Sin embargo, se decidió utilizar el EIDE por los siguientes motivos:

- Siendo un entorno realizado específicamente para desarrollar Instituciones Electrónicas, el EIDE dispone de varias utilidades realmente útiles para tal efecto, como un editor gráfico (ISLANDER) con herramientas de verificación del diseño, mientras que JADE es más adecuado para el desarrollo de sistemas multiagente sin estructura organizativa.
- El EIDE también se engloba dentro del proyecto AT, por lo que también era una oportunidad más de demostrar su utilidad.
- Mediante EIDE, se puede probar el diseño antes incluso de implementarlo, mediante el dúo Ameli-Dummy Agent. El Ameli es una plataforma para ejecutar Instituciones Electrónicas que únicamente necesita la especificación realizada con el

ISLANDER, mientras que el Dummy Agent proporciona una interfaz para interactuar con dicha Institución. Por tanto, el ciclo de pruebas sería más rápido, pudiendo detectar problemas en el diseño en menos tiempo (algo útil teniendo en cuenta la novedad del paradigma utilizado).

- El EIDE da la oportunidad de generar esqueletos de agentes mediante ABuilder, para implementar los procesos de decisión del agente a posteriori. Esto permitiría que, cuando fuera necesario, la implementación sería sencilla, y el código generado sería estándar respecto al de otras Instituciones Electrónicas desarrolladas con EIDE.

- Otro aspecto interesante del EIDE es que permite cambiar el comportamiento de cada una de las etapas del protocolo *(escenas)*, pudiendo utilizar uno de los varios creados en la especificación de la Institución. Dado que uno de los requisitos críticos era el añadir dicha modularidad en el proceso de decisión, fue un factor decisivo en seleccionar al EIDE sobre JADE.

2.3. Metodología utilizada

Al ser los requisitos muy estables, se optó por un modelo en cascada, primero analizando los requisitos hasta tener una primera propuesta del flujo de trabajo que debía seguir la estructura organizativa, para después realizar el diseño en dos partes: el diseño en sí mismo, y las pruebas sobre dicho diseño.

En la primera parte del proceso de diseño, el objetivo era conseguir una especificación completa de la Institución Electrónica requerida, y verificada formalmente mediante las herramientas de verificación del ISLANDER (solucionando así lo que en otros paradigmas o lenguajes, se considerarían ”Errores sintácticos”). De detectar algún error, se corregiría y ser volvería a verificar, dando lugar a un proceso cíclico que terminaría cuando la verificación concluyese sin errores.

En la segunda parte, el objetivo era comprobar la especificación con el Ameli y el Dummy Agent (tratando de detectar y solucionar los ”Errores lógicos” del diseño). La mayoría de dichos errores impedían que terminase la simulación, por lo que debían ser corregidos inmediatamente. En este caso, por tanto, las pruebas terminarían cuando la especificación entera (es decir, el flujo de trabajo de la Institución Electrónica) fuera navegado completamente sin errores por los agentes correspondientes.

Capítulo 3

Instituciones Electrónicas

En este capítulo describiremos el paradigma software utilizado en este trabajo, es decir, los sistemas multiagente y las Instituciones Electrónicas.

3.1. Sistemas Multi-Agente

El concepto de *agente* proporciona un método conveniente y potente de describir una entidad software compleja que es capaz de actuar con un cierto grado de autonomía para poder cumplir tareas en nombre de su usuario [WJ95].

Por tanto, un enfoque de sistema multiagente permite la gestión inteligente de un sistema complejo, coordinando los diferentes subsistemas que son parte de él e integrando sus metas particulares en un objetivo global. Este tipo de sistemas son especialmente útiles en varias situaciones [Mas05]:

- Cuando el sistema está físicamente distribuido.
- Cuando la solución requiere de experiencia heterogénea.
- Cuando el problema está definido sobre una red de ordenadores.

En muchos casos hoy en día, un sistema multiagente puede ser aconsejable, dado que proporciona una solución distribuida y capaz de adaptarse a los cambios en el entorno de la aplicación, así como una metodología de desarrollo software que permite la construcción de un sistema completo partiendo de varias unidades autónomas [Fer].

Como la aplicación presentada en este trabajo se encuentra muy distribuida (después de todo, involucra a agentes que pueden estar en los diferentes hospitales, en ocasiones con kilómetros de distancia entre ellos), este paradigma es el más adecuado.

3.2. El paradigma de las Instituciones Electrónicas

El prototipo propuesto en este trabajo está construido mediante el paradigma de las *Instituciones Electrónicas*.

Este paradigma es una propuesta del IIIA (Instituto de Investigación en Inteligencia Artificial). Su meta es proponer principios para el diseño y desarrollo de sistemas multi-agente abiertos teniendo como base un entorno distribuido, abierto y computacionalmente dinámico:

> La idea detrás de las Instituciones Electrónicas es imitar el papel que las instituciones tradicionales juegan en el establecimiento de ”las reglas del juego” –un conjunto de convenciones que articulan las interacciones de los agentes– pero en nuestro caso aplicados a agentes (humanos o software) que interactúan a través de mensajes cuyos efectos (socialmente relevantes) son conocidos para las partes que interactúan.
>
> [...] Las IE, como artefactos, constituyen un marco conceptual para describir interacciones entre agentes así como un marco ingenieril para especificar y desplegar auténticos entornos de interacción [JAS05].

La propuesta de las Instituciones Electrónicas comprende [wdeE]:

- La exploración de algoritmos flexibles para alcanzar acuerdos y aplicarlos.
- El estudio del uso de estructuras ”institucionales” aplicadas al modelado de sistemas sociales.
- El desarrollo de una capa de software ”institucional” para especificar, activar y probar Instituciones Electrónicas.
- Experimentos en Instituciones Electrónicas reales.

3.2.1. Componentes de una Institución Electrónica

Una especificación de una Institución Electrónica comprende varios conceptos, los cuales se detallan a continuación. Se muestran asimismo los nombres en inglés, para mejor comprensión de los diagramas presentes en este trabajo. Esta información está sacada en su mayor parte de las experiencias recopiladas en el transcurso de este trabajo, así como de la ayuda proporcionada por el EIDE [Dev]:

Estructura performativa (*Performative Structure*)

Ésta es la estructura principal dentro de una especificación de Institución Electrónica. Puede ser vista a grandes rasgos como un diagrama de estados, en el cual varios agentes pueden entrar a través de un nodo ”inicial”, y salir a través de un nodo ”final”.

Se compone de los siguientes elementos:

- Un **nodo inicial**, por el cual los agentes entran en la estructura performativa. Siempre debe haber uno por cada estructura performativa.
- Varios **nodos de escena**, situados entre el nodo inicial y el final, cada uno de los cuales contiene una *escena.* Pueden marcarse como *lista*, significando que puede haber varias instancias activas de dicha escena simultáneamente.
- Varias **transiciones**, las cuales sirven como puntos de unión entre los nodos de escena. Pueden ser:
  - De tipo *or*, en los cuales el agente puede elegir a cuál (o cuales) de las escenas subsiguientes a la transición quiere entrar, pudiendo elegir varias a la vez.
  - De tipo *orx*, en los cuales el agente solamente puede entrar en una de las escenas subsiguientes.
  - De tipo *and*, en los cuales el agente debe entrar en todas las escenas subsiguientes.
- Varios **arcos de transición**, a través de los cuales el agente navega de una escena a una transición, o de dicha transición a la siguiente escena. Estos arcos pueden ser marcados con precondiciones a cumplir para poder transitar por ellos, acciones que realizar despues de dicha navegación, o por ambos, permitiendo controlar los caminos posibles del agente de acuerdo a cambios en el estado del agente o del entorno. Si estos arcos van de una transición a una escena, pueden estar marcados como:
  - ***one*** (*una*): El agente puede entrar a sólo una instancia de la escena destino.
  - ***some*** (*algunas*): El agente puede entrar a varias instancias de la escena destino, con un número indeterminado de ellas.
  - ***all*** (*todas*): El agente debe entrar a todas las instancias de la escena destino.
  - ***new*** (*nueva*): El agente debe crear una nueva instancia de la escena destino.
- Un **nodo final**, a través del cual los agentes pueden salir de la estructura performativa. Como ocurre con el nodo inicial, debe haber uno por cada estructura performativa.

Escena (*Scene*)

Las escenas son las etapas por las que un agente tiene que pasar mientras navega por una estructura performativa, y representan cada una de las interacciones que el agente debe realizar. Como tales, las escenas deben tener un *tipo*, esto es, deben tener asignado una descripción de la interacción que representa la escena.

Estas descripciones pueden ser de dos tipos:

- Un **Tipo de escena**. Este elemento se explica más adelante.
- Otra **Estructura performativa**, Lo que permite anidarlas, siendo capaz de definir varios niveles de detalle en la especificación del sistema.

Tipos de escena (*Scene Types*)

Los tipos de escena representan las interacciones entre agentes dentro de una escena. Al contrario que las estructuras performativas, no pueden ser anidados. Además, deben tener especificado la cantidad de agentes (así como sus roles) que deben haber entrado en ella para que la interacción pueda comenzar.

Son representados como diagramas de estados clásicos, teniendo por tanto dos componentes:

- **Estados** de la interacción, los cuales pueden ser marcados como puntos de entrada, salida o *stay and go* para agentes que desempeñen determinados **roles** en el sistema.
- **Transiciones** entre estados, similares a los *arcos de transición* en las estructuras performativas. Estas transiciones representan los mensajes interacambiados entre agentes, cuyos elementos (el emisor, el receptor, el contenido, etc.) se definen en la especificación de dicha transición.

Cabe destacar aquí el concepto de *stay and go* mencionado anteriormente. Al igual que los estados de un tipo de escena se pueden marcar como puntos de entrada o de salida de la escena (los cuales son autoexplicativos), el concepto de *stay and go* conlleva el que un agente se divida en dos caminos diferentes de ejecución o *alteroides*, a fin de poder dejar una escena para seguir recorriendo la estructura performativa correspondiente, pero a su vez quedarse en ella, siendo por tanto capaz de recibir nuevos mensajes en dicha escena.

Esto permite modelar comportamientos parecidos a, por ejemplo, un operador telefónico, el cual atiende peticiones y, mientras comunica la nueva petición al siguiente proceso de la organizacón, no abandona su puesto, a fin de recibir la siguiente petición. Sin embargo, al no ser agentes independientes, todos los alteroides de un mismo agente comparten las mismas propiedades, lo cual se debe tener en cuenta a la hora de diseñar

la estructura performativa y los elementos de control que en ella se necesiten, pues no se puede considerar a cada alteroide como un agente independiente.

Mensajes (*messages*)

Los mensajes describen la información intercambiada entre los agentes, la cual pueden ser datos o notificaciones. Un mensaje en una Institución Electrónica comprende cuatro elementos:

1. La **partícula ilocutoria (*illocutory particle*)**, la cual sirve como método de distinción de varios tipos de mensajes.
2. El **origen**, especificando qué rol debe estar desempeñando un agente para poder emitir el mensaje. Además del rol, también se debe especificar un nombre de variable, el cual podrá servir para enviar respuestas a dicho mensaje. Si a la variable le antecede un '?', significa que se utiliza un valor nuevo para esa variable, creándola si fuera necesario. Si en cambio es un '!', significa que se utiliza el valor anterior de dicha variable.
3. El **destino**, especificando qué rol debe estar desempeñando un agente para poder recibir el mensaje. Está formado de manera análoga al origen. Si el nombre de la variable es *all* (todos), entonces todos los agentes presentes en esa escena con dicho rol recibiran el mensaje.
4. El **contenido del mensaje**, el cual se suele modelar como una llamada a función.

Marco dialógico (*Dialogical Framework*)

Un marco dialógico es una especificación de los diferentes roles y partículas ilocutorias que existen en una Institución Electrónica.

- Un **rol**, en el contexto de este trabajo, es un conjunto de comportamientos que asume un agente como parte de una Institución Electrónica. Un agente dado puede desempeñar más de un rol simultáneamente, permitiendo por tanto más de un comportamiento posible, excepto si dichos roles se definen expresamente como excluyentes (Mediante una relación *SSD –Static Separation of Duties–*, *Separación estricta de deberes*).

  Todos los componentes de la Institucuón Electrónica son discriminantes en cuanto al rol, por lo que agentes con diferentes roles seguirán diferentes caminos dentro de la institución, y se evitarán de manera automática problemas derivados de agentes siguiendo caminos que no les corresponden.

- Las **partículas ilocutorias** representan los diferentes tipos de mensajes que un agente puede enviar (como *peticiones*, *respuestas*, *notificaciones*, etc.).

Ontología (*Ontology*)

La ontología de una Institución Electrónica describe los tipos de datos extendidos presentes en la especificación, y las funciones presentes como contenidos de mensaje, junto con sus parámetros. Esto permite crear tipos de datos complejos que sirvan a nuestras necesidades, como por ejemplo, un tipo de datos con la información de un hospital, o de un paciente.

3.2.2. Herramientas requeridas

Siendo una propuesta teórica reciente, una Institución Electrónica no puede ser desarrollada con facilidad con los métodos y entornos convencionales. Debido a ello, el IIIA ha creado un entorno de desarrollo junto con la propuesta teórica: el EIDE (*Electronic Institution Development Enviroment –Entorno de Desarrollo de Instituciones Electrónicas–*), el cual es descrito por sus creadores como sigue:

> El EIDE es un conjunto de herramientas que aspira a soportar la creación ingenieril de aplicaciones inteligentes distribuidas de manera que sean Ins-tituciones Electrónicas. Los agentes software aparecen como la tecnología clave detrás de la visión de las Instituciones Electrónicas. Por tanto, las Instituciones Electrónicas encapsulan los mecanismos de coordinación que median en las interacciones entre agentes software representando a diferentes partes.
>
> El EIDE permite crear tanto Instituciones Electrónicas como sus agentes participantes. Notablemente, el EIDE se alega de las visiones orientadas-a-máquinas de la programación y se acerca a conceptos organizacionales que reflejan más de cerca la forma en la que podríamos entender las aplicaciones distribuidas. Soporta un enfoque de diseño top-down: primero la organización, después los individuos. El EIDE se compone de:
>
> - ISLANDER: Una herramienta gráfica para especificar las reglas y protocolos en una Institución Electrónica. Su aspecto se muestra en la figura 3.1.
> - SIMDEI: Herramienta de simulación para animar y analizar especificaciones creadas con ISLANDER antes de la etapa de despliegue.
> - Agent Builder: Herramienta de creación de agentes. Esta herramienta permite generar esqueletos de código Java de los agentes intervinientes, quedando bajo responsabilidad del programador el definir sus procesos de decisión.
> - AMELI: Plataforma software para ejecutar Instituciones Electrónicas. Una vez que se especifica una Institución Electrónica con el ISLANDER está preparada para ser ejecutada con el AMELI sin necesidad de programar nada más. Su aspecto se muestra en la figura 3.2 [Dev].

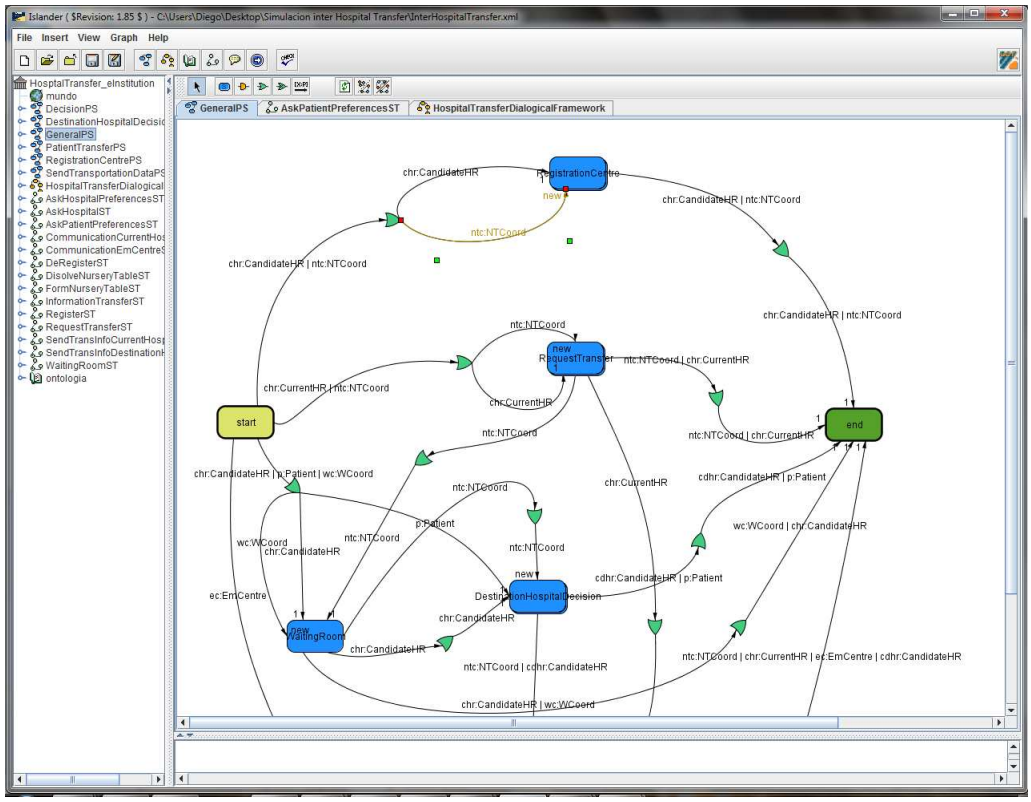

Figura 3.1: Imagen del islander

Asimismo, con el entorno EIDE viene otro programa, llamado *Dummy Agent*, el cual proporciona a un usuario humando una interfaz de agente para poder interactuar con la Institución Electrónica como si fueran agentes software, decidiendo en cada paso que acción debe realizar el Dummy Agent. Esto permite (Como se verá en el capítulo 6) probar una Institución Electrónica sin necesidad de programar los agentes, a fin de detectar fallos más rápidamente. Su aspecto se puede observar en la figura 3.3.

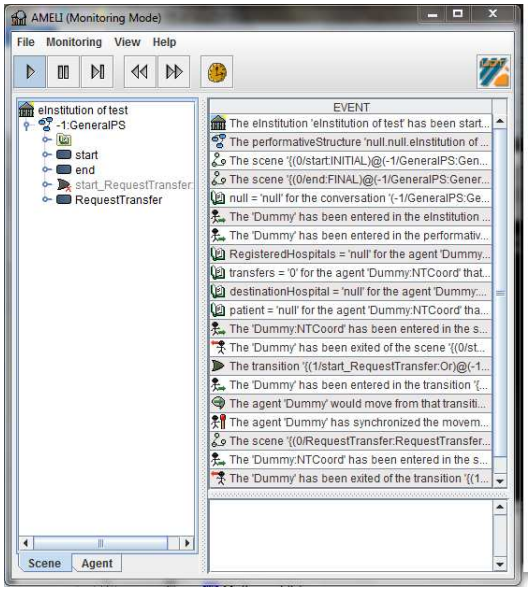

Figura 3.2: Imagen del Ameli

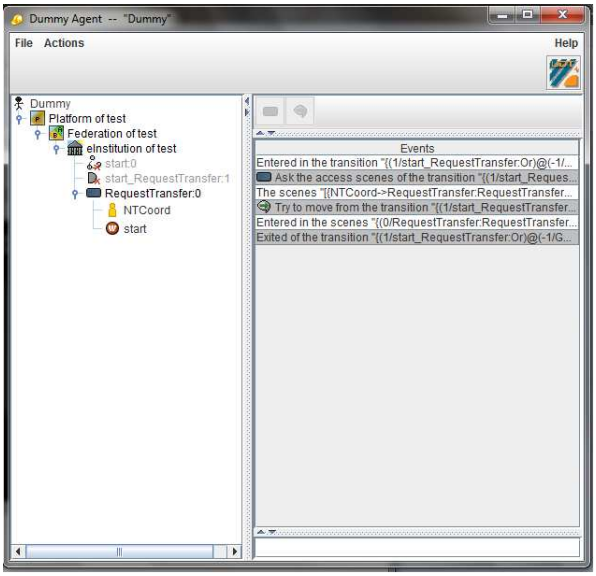

Figura 3.3: Imagen del Dummy Agent

Capítulo 4

Análisis

En este capítulo, vamos a presentar el análisis de requisitos y el diseño de alto nivel del sistema de coordinación de traslados inter-hospitalarios. Dicho análisis está fuertemente basado en el análisis preliminar del mHealth realizado para el proyecto AT [BFD$^+$].

Para ello, identificaremos los aspectos de un sistema multiagente aplicados al problema de la coordinación de traslados inter-hospitalarios. Dichos aspectos son:

1. Las *entidades* presentes en el dominio de la aplicación;
2. Los *objetivos* a conseguir por cada una de las entidades;
3. Las *tareas* asociadas a las entidades para conseguir ciertos objetivos globales;
4. Las *interacciones* entre entidades para cumplir tareas;
5. Los *procesos de decisión* llevados a cabo por las entidades (individuales o en grupo);

El caso de estudio propuesto se concentra principalmente en el problema de encontrar un acuerdo entre los hospitales sobre a qué hospital debería ser trasladado cada paciente. Por tanto, se dejaría fuera del ámbito de esta aplicación la tarea de coordinar el traslado real una vez que la decisión sobre el destino se ha realizado. Esta tarea es muy similar a las realizadas en el caso de uso *transporte médico de urgencia*, también perteneciente al demostrador mHealth, con lo que podría ser más facilmente incorporada a las tareas tratadas en él.

4.1. Entidades, objetivos y tareas

El objetivo general de la aplicación de Coordinación de Traslados Inter-Hospitalarios es:

> Resolver el problema de asignar un hospital de destino adecuado para un paciente que ya está siendo tratado en un hospital pero requiere un traslado. Las razones para que dicho traslado sea necesario incluyen la falta de recursos para tratar al paciente, o que el paciente prefiera ser tratado en otro hospital.

La decisión de pedir un traslado es tomada por el hospital que ha recibido inicialmente al paciente (el *hospital actual* del paciente). Esta decisión podría ser tanto una acción reactiva, originada como respuesta directa a una situación crítica en el hospital (por ejemplo, no tener más camas), o una acción proactiva que busca impedir que ocurran dichas situaciones críticas en el futuro.

Las entidades principales involucradas en el problema de la coordinación de traslados inter-hospitalarios son: el *paciente*, la *mesa de enfermería*, diferentes *hospitales* y el *centro de emergencias*.

Paciente: Una persona que está siendo tratada en un hospital.

Objetivo #1.1: Recibir la atención médica necesaria para su caso de acuerdo a sus preferencias personales.

Tarea #1.1: *Pedir que le atiendan cerca de casa:* Un paciente debería ser capaz de pedir ser trasladado, cuando sea médicamente posible, a un centro cercano a su lugar de residencia.

Coordinador de la Mesa de Enfermería: Un mediador que coordina el proceso de encontrar una solución para una petición de traslado, e inicia dicho traslado. El proceso es conseguido de tal manera que es parcialmente transparente para los hospitales.

Objetivo #2.1: Ofrecer una mediación entre hospitales para resolver los traslados que sean necesarios entre hospitales.

Tarea #2.1: *Mediar entre hospitales:* El coordinador de la mesa de enfermería tiene que proporcionar la infraestructura necesaria para poder coordinar peticiones de traslado. Esto incluye:

1. La recepción de dicha petición y la iniciación del proceso de decisión,
2. La coordinación de dicho proceso, a fin de alcanzar un acuerdo entre los hospitales en cuanto a qué hospital trasladar al paciente,
3. Cerrar el proceso de decisión e iniciar el proceso real de traslado.

Hospital: Un centro médico donde se atiende a los pacientes.

Objetivo #3.1: Proporcionar el mejor tratamiento médico posible a los pacientes.

Tarea #3.1: *Proporcionar atención médica:* Los pacientes son atendidos de la mejor manera posible (de acuerdo a la dolencia que sufran). De no ser posible, un paciente debería ser trasladado a otro hospital que sea capaz de proporcionarle el tratamiento necesario.

Objetivo #3.2: Mantener la ocupación de camas por debajo de los límites.

Tarea #3.2: *Liberar camas:* Si la ocupación de un hospital (o uno de sus departamentos) supera un determinado nivel umbral, específico para cada hospital, el hospital puede tratar de liberar camas. Esto se puede realizar trasladando algunos pacientes a otros hospitales.

Centro de Emergencias: Es la entidad que coordina el transporte de pacientes fuera del hospital, en concreto, en los traslados inter-hospitalarios.

Objetivo #4.1: Trasladar pacientes de su hospital actual al hospital de destino si dicho traslado ha sido aceptado.

Tarea #4.1: *Traslado de pacientes:* Los pacientes deberían ser trasladados asegurando el nivel de asistencia adecuado durante el traslado y en el menor tiempo posible.

4.2. Interacciones

El problema de la coordinación de traslados inter-hospitalarios puede ser dividido en dos partes:

1. El proceso general de traslado;
2. El proceso de acuerdo que decide a qué hospital debería ser trasladado un paciente dado.

Mientras que el proceso general de traslado es fijo, la forma en la que el acuerdo es llevado a cabo puede variar, utilizando cada vez uno de los mecanismos reales que se utilizan para llegar a dichos acuerdos. Lo mismo se puede decir de las interacciones entre las entidades

y el flujo de trabajo de dichas interacciones, los cuales dependen del mecanismo concreto de decisión que se esté utilizando.

En esta sección y en la siguiente nos concentraremos en el proceso general de traslado, ya que es el objetivo principal de este trabajo. Por otra parte, dado que para evaluar dicho diseño se necesita algún proceso de decisión, en la sección 4.5 presentaremos un análisis somero de dicho proceso, con unas pautas generales a cumplir por él, y propondremos un proceso de decisión de hospital de destino simple que cumpla con dichas pautas y que permita evaluar el diseño de la estructura del proceso general de traslado.

Con respecto al proceso general de coordinación de traslados, las principales interacciones son las siguientes:

Interacción #1: *Petición de traslado para un paciente:* El hospital que está tratando a un paciente emite una petición de traslado al coordinador de la mesa de enfermería. La petición incluye el tratamiento requerido, la prioridad y el nivel de asistencia requerido, y las razones por las que se considera necesario el traslado.

Interacción #2: *Confirmar la recepción de la petición:* Tras recibir la petición, el coordinador envía una confirmación al hospital que la emitió, comunicándole si ha sido aceptada o rechazada.

Interacción #3: *Convocar a la mesa de enfermería:* Si el traslado ha sido aceptado, el coordinador convoca a algunos hospitales a la mesa de enfermería, y les informa del caso. Dichos hospitales deberán haber sido seleccionados en base al tratamiento requerido y a los servicios disponibles en cada hospital.

Interacción #4: *Cerrar la mesa de enfermería:* Después del proceso de decisión, el coordinador de la mesa cierra la mesa de enfermería e informa a todos los hospitales involucrados sobre la decisión adoptada.

Interacción #5: *Enviar datos médicos:* El hospital actual del paciente envia la información médica del paciente al hospital de destino.

Interacción #6: *Petición de transporte:* Una vez que la decisión ha sido adoptada, la mesa de enfermería envía una petición al centro de emergencias, solicitando la asignación de los recursos necesarios y la planificación del transporte del paciente del hospital actual al hospital de destino.

Interacción #7: *Envío de datos de transporte:* Tras planificar el transporte del paciente (asignación de vehículos, planificación temporal, etc.), el centro de emergencias informa al hospital actual del paciente sobre la hora a la que se ha planificado el traslado, y al hospital de destino sobre la hora estimada de llegada del vehículo asignado.

En el caso de la coordinación de traslados inter-hospitalarios, consideramos a los pacientes como entidades principalmente pasivas. Esto es así porque, en cierto modo, sus intereses están representados por los hospitales en los que se encuentran, dado que éstos deciden sobre la necesidad de trasladarlo a fin de que pueda obtener el mejor tratamiento posible. Sin embargo, los pacientes podrían iniciar un proceso de traslado para conseguir su objetivo #1.1, por ejemplo, pidiendo explícitamente recibir el tratamiento cerca de su lugar de residencia. En este caso, el paciente le comentaría dicha situación a su médico, y el personal del hospital decidiría si realmente un traslado es la mejor opción. Más tarde, estas preferencias podrían ser tenidas en cuenta en el proceso de acuerdo entre hospitales, preguntando al paciente sobre sus preferencias personales.

Con respecto al centro de emergencias, en este caso, esta entidad solamente recibe la petición de transporte para el paciente, a fin de coordinar su traslado entre los hospitales. Como se planifica y se lleva a cabo dicho transporte, recordemos, no se tiene en cuenta en este trabajo, dado que se puede considerar parte del caso de uso de Transporte médico de urgencia.

4.3. Procesos de decisión

En el sistema de coordinación de traslados inter-hospitalarios se han identificado los siguientes procesos de decisión:

Decisión #1 *Decidir si trasladar al paciente o no:* Esta decisión es tomada por un hospital, en el caso de que alguien de su personal (o el propio paciente) considere que es necesario un traslado del paciente a otro hospital. Esto puede ocurrir por varias razones:

- Los recursos disponibles y los servicios no permiten proporcionar el tratamiento adecuado a un paciente.
- El hospital no tiene suficientes recursos como para tratar al paciente en un período de tiempo adecuado (por ejemplo, no hay camas disponibles, hay otros casos con mayor prioridad que deben ser atendidos primero, no hay especialistas apropiados para el caso disponibles en ese momento, etc.).
- El paciente prefiere ser trasladado a otro hospital que, por ejemplo, se encuentre más cercano a su residencia o tenga mejor reputación.

Esta decisión incluye la asignación de un nivel de prioridad para la petición de traslado que indica la urgencia de dicho traslado. Dicho nivel de prioridad es un número que va de 0 (pacientes muy críticos cuyo traslado es vital) a 3 (pacientes estables cuyo traslado no es muy urgente).

Decisión #2 *Aceptar o posponer la petición de traslado:* El coordinador de la mesa de enfermería tiene que decidir si un traslado pendiente debe ser pospuesto (con lo que el hospital debería volver a solicitar el traslado más tarde) o decidido de inmediato. Esta decisión depende del nivel de prioridad de la petición y de si otras posibles peticiones que queden pendientes.

Decisión #3 *Selección del hospital de destino:* Esta decisión es la más importante en la coordinación de traslados inter-hospitalarios, y consiste en el proceso de acuerdo mediante el cual se decide a qué hospital debería ser trasladado un paciente dado. Esta decisión involucra a diferentes entidades: principalmente, al coordinador de la mesa de enfermería, a los hospitales invitados a dicha mesa, y, en algunos mecanismos de decisión, también al paciente. La forma en que estas entidades deciden el hospital de destino más adecuado para el paciente depende del mecanismo de argumentación/negociación particular que se implemente. En la sección 4.5 presentaremos un análisis específico de este proceso.

Decisión #4 *Determinar las preferencias de los hospitales:* Esta decisión es realizada por los hospitales que se encuentran en el proceso de alcanzar el acuerdo sobre el hospital de destino del paciente. Un hospital debería decidir si es capaz de proporcionar el tratamiento adecuado en el intervalo de tiempo dado (por tanto cuantificando sus *capacidades* de recibir más pacientes) y asimismo debería, considerando sus preferencias personales, ser capaz de informar sobre si tiene algún interés especial en un paciente determinado (o si no lo tiene en absoluto), cuantificando su interés en dicho paciente.

Decisión #5 *Determinar las preferencias del paciente:* Adicionalmente a la decisión anterior, un paciente, en determinados mecanismos de decisión, debería ser capaz también de expresar sus preferencias respecto a qué hospitales preferiría ser trasladado. Dichos hospitales podrían ser cercanos a su lugar de residencia (cumpliendo su objetivo #1.1), o simplemente preferirlos debido a su buena reputación.

Decisión #6 *Planificar el transporte:* Esta decisión es realizada por el centro de emergencias. Conlleva la asignación de un vehículo de transporte médico adecuado al caso y posiblemente la asignación de recursos adicionales (tales como medicamentos, personal médico, etc.) así como la decisión de la planificación del horario del traslado.

4.4. Flujo de trabajo del proceso general de coordinación de traslados

La figura 4.2 Muestra el flujo de trabajo del proceso general de coordinación de traslados. Muestra las diferentes *entidades*, *decisiones*, e *interacciones* que dicho proceso supone:

1. El proceso comienza cuando un hospital decide solicitar el traslado de un paciente (Decisión #1) y envia una petición de traslado de paciente al coordinador de la mesa de enfermería (Interacción #1).

2. El coordinador acepta la petición (Interacción #2) y entonces decide si la petición debería ser decidida en el momento o pospuesta (Decisión #2).

3. Una vez que una petición va a comenzar a ser atendida, el coordinador convoca a la mesa de enfermería e informa a los hospitales invitados a ella del caso de traslado en curso (Interacción #3).

4. La mesa de enfermería utilizará un proceso de decisión específico (ver sección 4.5 para un ejemplo de tal proceso) para alcanzar un acuerdo respecto a la selección de un hosptial adecuado para el paciente (Decisiones #3, #4, #5).

5. Una vez que se ha llegado a un acuerdo, el coordinador cierra la mesa de enfermería e informa a todos los hospitales de la decisión que ha sido adoptada (Interacción #4).

6. Finalmente, se prepara el traslado propiamente dicho. El hospital actual del paciente envía sus datos médicos al hospital de destino (Interacción #5). Mientras, la mesa de enfermería envía una petición al centro de emergencias, solicitando un transporte para el paciente adecuado al tratamiento requerido (Interacción #6).

7. El caso de uso finaliza cuando el centro de emergencias ha planificado el transporte e informa de dicha planificación tanto al hospital actual del paciente como al hospital de destino. Dicha información incluirá la hora estimada de llegada a cada uno de ellos del vehículo asignado para transportar al paciente (Interacción #7).

4.5. Procesos de decisión

En esta sección, analizaremos un poco más en detalle cómo la mesa de enfermería alcanza el acuerdo respecto a encontrar un hospital de destino adecuado para el paciente. Las entidades involucradas en dicha decisión son el coordinador de la mesa de enfermería y algunos de los hospitales registrados, los cuales han sido seleccionados para participar

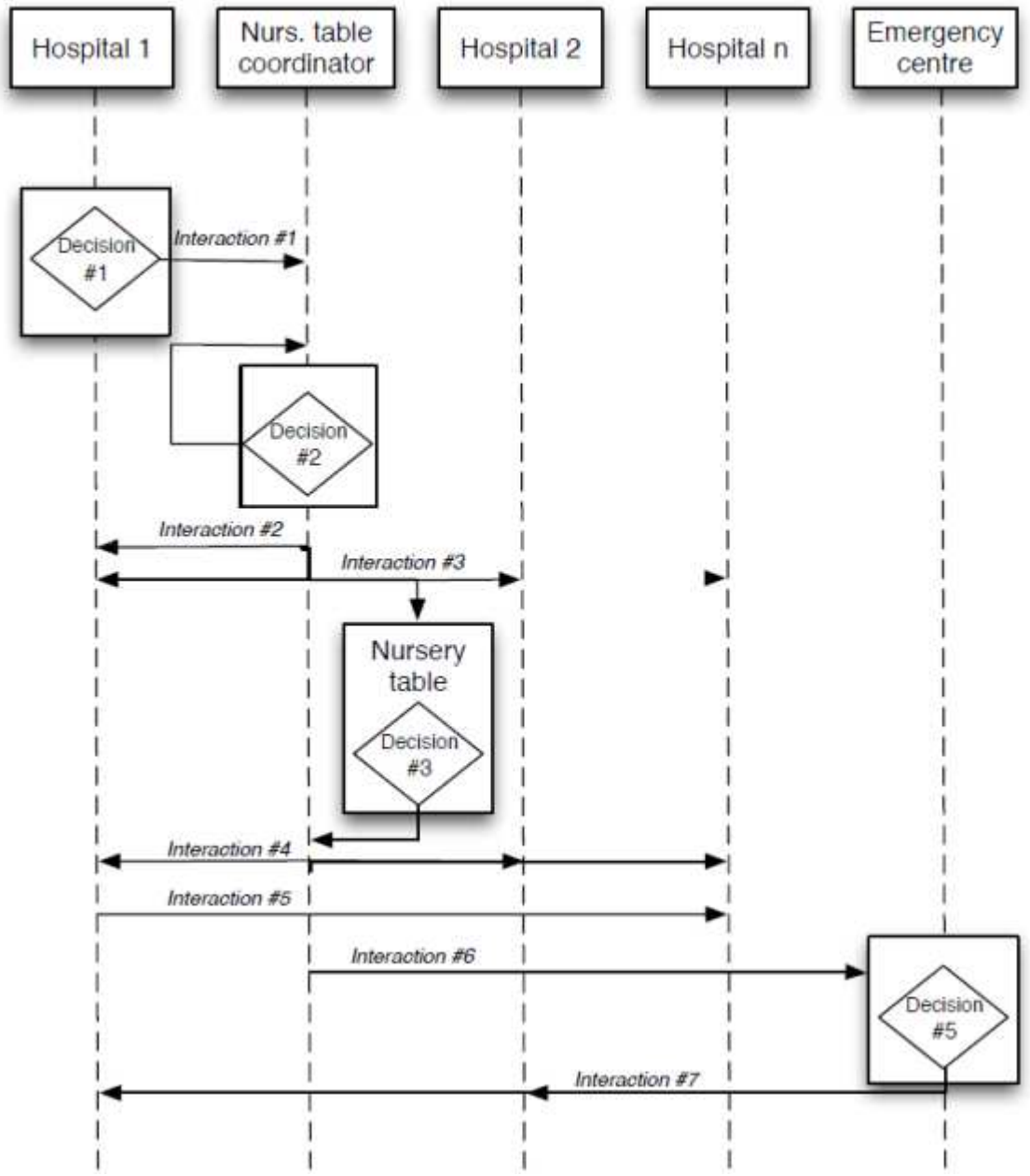

Figura 4.1: Flujo de trabajo del proceso general de traslado

en esa mesa concreta debido a que son capaces de proporcionar los servicios acordes al tratamiento requerido por el paciente. El proceso de decisión comienza después de que el coordinador de la mesa de enfermería haya informado a los hospitales invitados del caso, y finaliza después de que la decisión haya sido alcanzada, con el coordinador de la mesa de enfermería notificando de la decisión adoptada a los hospitales.

En esta sección, en primer lugar, estableceremos las propiedades específicas y requisitos generales que un determinado mecanismo de decisión para este dominio debería cumplir. Después, propondremos un ejemplo de tal mecanismo.

Para extraer dichos requisitos, nos basaremos en las características específicas y requisitos de tales decisiones en el modelo de referencia del SUMMA112. Los tres primeros son requisitos obligatorios (*hard constraints*) que son vitales para la eficacia del proceso de decisión, mientras que los otros tres son requisitos que, aunque opcionales, mejorarían enormemente la calidad de la decisión adoptada en términos de satisfacción de las partes involucradas:

- *Una solución para cada caso*: El primer y principal requisito es que se debe proporcionar una solución para cada petición de traslado, incluso si tal decisión es no trasladar al paciente, y que éste permanezca en el hospital que le está tratando actualmente.

- *Servicios de los hospitales*: Existe un ”catálogo de servicios” que describe los recursos técnicos disponibles en cada hospital y los tratamientos que es capaz de ofrecer. Dicho catálogo puede convertirse fácilmente en uno de los criterios de selección del proceso de decisión.

- *Capacidades de los hospitales*: El término ”capacidades de un hospital” se refiere a los recursos disponibles en dicho hospital en un momento dado, y podrían (y deberían) tenerse en cuenta en el proceso de decisión.

- *Preferencias de los hospitales*: Los hospitales podrían tener preferencias en cuanto a recibir determinados tipos de pacientes, las cuales podrían ser tenidas en cuenta al decidir.

- *Preferencias del paciente*: Los pacientes podrían tener preferencias en cuanto al hospital en el que quieren ser tratados. Dichas preferencias también podrían ser tenidas en cuenta en la decisión de la mesa de enfermería.

- *Acuerdo implícito entre hospitales*: Existe un acuerdo implícito entre los hospitales que participan en la mesa de enfermería. Dicho acuerdo establece que todos los hospitales aceptarán la decisión tomada por la mesa, incluso si tal decisión va en contra de sus preferencias individuales.

Una propiedad particular del modelo del SUMMA112 es que la información de los servicios de cada hospital es estática y conocida por el coordinador de la mesa de enfer-

mería. Esta información debería ser utilizada para minimizar el número de hospitales que participan en la mesa de un caso de traslado dado.

Otra propiedad importante de nuestro modelo de referencia es que los tiempos de traslado pueden ser ignorados (con la excepción de pacientes muy críticos). Esto ocurre así porque, en la Comunidad de Madrid, no hay diferencias significativas en las distancias entre hospitales.

Proceso de decisión: propuesta inicial

Aunque el objetivo de este trabajo es diseñar una estructura para el proceso general de traslado (la cual pudiese contener diferentes procesos de decisión para alcanzar el acuerdo respecto al hospital de destino del paciente, a fin de probarlos y compararlos) es necesario tener, al menos, uno de dichos procesos dentro de dicha estructura, a fin de comprobar su correcto funcionamiento.

Debido a esto, en este trabajo propondremos como ejemplo de dicho proceso de decisión una toma de decisiones simple y puramente no colaborativa, en la cual el coordinador de la mesa de enfermería decide el hospital de destino para un caso determinado basándose en la información recibida de los hospitales invitados a la mesa y del paciente. Tal información comprende las capacidades de cada hospital, y las preferencias tanto del paciente como de los hospitales.

El flujo de trabajo de tal toma de decisiones puede verse en la figura **??**. Comprende los siguientes pasos:

1. Basándose en el tratamiento requerido, y el catálogo de servicios de los hospitales registrados, el coordinador de la mesa de enfermería selecciona a los hospitales que proporcionan los servicios requeridos acordes con el tratamiento que necesita el paciente (obteniendo el conjunto H1). Nótese que esta selección se realiza antes del comienzo del proceso de decisión propiamente dicho.

2. El coordinador pregunta por las capacidades disponibles de todos los potenciales hospitales de destino, esto es, de todos los hospitales que podrían proporcionar el servicio requerido, y han sido invitados por tanto a la mesa de enfermería. Tras recibir sus respuestas, el coordinador selecciona a aquellos que tiene la capacidad suficiente (obteniendo el conjunto H2 a partir de H1).

3. Entonces, el coordinador pide las preferencias del paciente respecto al hospital en el que quiere ser tratado, a fin de que este lo más comodo posible. Tras recibirlas, selecciona los hospitales de H2 que están en las preferencias del paciente (o todos si no hay preferencias o no coincide ninguno). De esta manera se obtiene el conjunto H3.

4. Por último el coordinador pide cualquier preferencia de los hospitales de H3 en cuanto a los pacientes que preferirían tratar. Tras recibir su respuesta, se reduce el

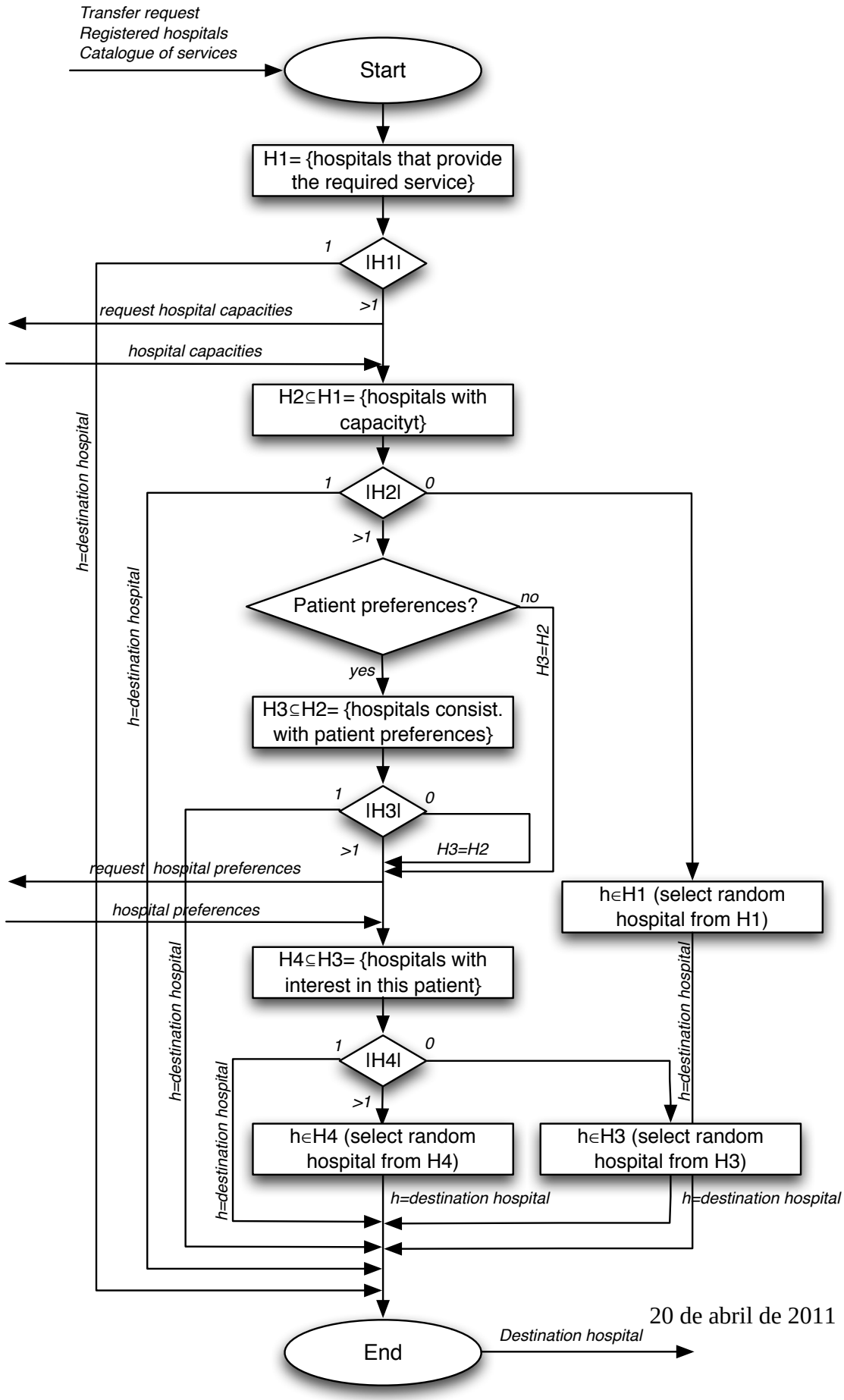

Figura 4.2: Diagrama de flujo del proceso de decisión de ejemplo

conjunto a los hospitales que muestran interés por el paciente en cuestión (conjunto H4).

5. Si, en alguna etapa del proceso, o después de todos los pasos anteriores, solamente hay un hospital en el conjunto actual, se selecciona y el proceso termina. Si no hay ninguno, se selecciona uno aleatoriamente de los hospitales del conjunto anterior. Si hay varios, se seleccionará uno de ellos aleatoriamente.

Como podemos ver, siempre se seleccionará un hospital, el cual puede ser uno diferente del actual, o el hospital actual (si se considera necesario, siempre y cuando dicho hospital pueda proporcionar el servicio requerido), cumpliendo de esta manera el requisito de *una solución para cada caso* descrito anteriormente.

Capítulo 5

Diseño de la institución electrónica

En este capítulo, describiremos detalladamente el diseño de la Institución Electrónica que se utiliza en nuestra aplicación de coordinación de traslados inter-hospitalarios, basado en el análisis descrito en la sección 4.

Primero describiremos los supuestos realizados y, después, procederemos a detallar la especifiación de la Institución Eletrónica, realizada mediante ISLANDER, en el siguiente orden:

- El *marco dialógico* de la institución.
- Las *estructuras performativas* presentes en el diseño.
- Los *tipos de escena* presentes en el diseño.
- Las *restricciones* planteadas.

5.1. Supuestos

Para este trabajo, se han supuesto los siguientes aspectos, a fin de simplificar el diseño de la primera versión de esta aplicación, dado que algunos de ellos añadirían demasiada complejidad:

- Dos hospitales que proporcionan un servicio determinado son iguales para el paciente en términos de calidad, resultando irrelevante por tanto a cuál de ellos se traslada al paciente, aparte de las preferencias que pueda tener el paciente.
- No hay un límite de tiempo para los traslados, esto es, no se considera (por el momento) el nivel de urgencia del paciente a la hora de tomar una decisión sobre a qué hospital se traslada al paciente (por ejemplo, trasladando a pacientes más

urgentes a hospitales más cercanos). Por el momento, el nivel de urgencia sólo se tendrá en cuenta a la hora de aceptar o rechazar una petición.

- Hay un coste fijo para los traslados independiente de la distancia, esto es, tampoco se consideran, a efectos de decidir el hospital de destino, aspectos como el combustible, los medicamentos aplicados al paciente para mantenerlo estable durante el transporte, etc.

- Por cada diagnóstico, hay al menos un hospital disponible que proporcione el servicio requerido. Esto es consecuencia de que solamente se consideran aquí dolencias comunes. Los tratamientos especiales, por tanto, quedan fuera del ámbito de este trabajo.

- Si un hospital proporciona un determinado servicio, debe tomar parte en la mesa de enfermería, dado que, incluso si no tiene camas libres, su experiencia con una dolencia determinada podría ser de ayuda en algunos procesos de decisión.

5.2. Especificación de ISLANDER

5.2.1. Marco dialógico

Roles

Hay un total de seis roles participando en la Coordinación de traslados inter–hospitalarios. Aquí presentaremos su significado, propósito dentro de la Institución Electrónica, y las propiedades que se le confieren, a fin de poder (como se verá en el apartado 5.2.4), restringir el camino a seguir por dicho agente. En negrita aparece el nombre en inglés del rol, a fin de que se puedan comprender mejor los nombres de cada rol en los diagramas de este capítulo:

- Un *Representante del hospital Actual* (**Current Hospital Representative - CurrentHR**) es un representante del hospital en el que se encuentra el paciente actualmente. Su tarea es emitir la petición de traslado y, de ser ésta aceptada, enviar la información del paciente al hospital determinado por la mesa de enfermería. Sus propiedades son:

  – *accepted*: Un booleano que representa si la petición de traslado que dicho representante ha emitido ha sido aceptada o no.

  – *noTransfer*: Un booleano que representa si el paciente va a ser trasladado o, por el contrario, la mesa de enfermería ha decidido que es mejor no trasladarlo.

- Un *Representante del hospital candidato* (**Candidate Hospital Representative** - **CandidateHR**), por otra parte, es un representante de uno de los hospitales registrados en la mesa de enfermería, o que se quiere registrar en ella. Su tarea es participar en la mesa de enfermería y, quizás, recibir traslados.

  Sus propiedades son:

  - *invitations*: Un entero con el número de invitaciones que este hospital ha recibido para participar en una mesa de enfermería.
  - *newPatients*: Un entero con el número de nuevos pacientes que este hospital va a recibir.

- Un *Paciente* (**Patient**) es el paciente que debe ser trasladado, y es consultado en el proceso sobre si tiene alguna preferencia respecto a qué hospital desea ser trasladado. Dado que sólo tiene esta tarea, no tiene ninguna propiedad.

- Un *Coordinador de mesa de enfermería* (**Nursery Table Coordinator** - **NTCoord**) es el rol de los agentes responsables de recibir y atender peticiones de traslado. Asimismo, también son responsables de, en el caso de que acepten una de dichas peticiones, convocar y disolver la mesa de enfermería, realizar la selección inicial de los hospitales capaces de recibir al paciente, comunicar al hospital del paciente la decisión de la mesa, y comunicar al centro de emergencias que se requiere un nuevo transporte médico.

  Sus propiedades son:

  - *transfers*: Un entero con el número de traslados que este coordinador está atendiendo en ese momento.
  - *noTransfer*: Un entero con el número de traslados que han sido procesados, pero cuyo resultado ha sido el de no transferir al paciente.

- Un *Coordinador de sala de espera* (**Waiting Coordinator** - **WCoord**) es el responsable de crear y gestionar una sala de espera en la cual los hospitales candidatos esperarán a que se les invite a una mesa de enfermería. Este rol no tiene propiedades.

- Por último, el *Centro de emergencias* (**Emergency Centre** - **EmCentre**) es la entidad responsable de asignar un vehículo médico adecuado al caso para el traslado del paciente, y comunicar la planificación de dicho traslado al hospital actual del paciente y al de destino. No tiene propiedades.

Todos estos roles tienen una relación de exclusión mutua (SSD - *Static Separation of Duties*), impidiendo que un agente dado tenga dos roles a la vez, como se puede ver en la figura 5.1. Esto es así para evitar que el diseño sea demasiado complejo. También se puede observar que solamente hay dos roles internos en la Institución, **NTCoord**, y **WCoord**. El resto son externos al proceso general de traslado.

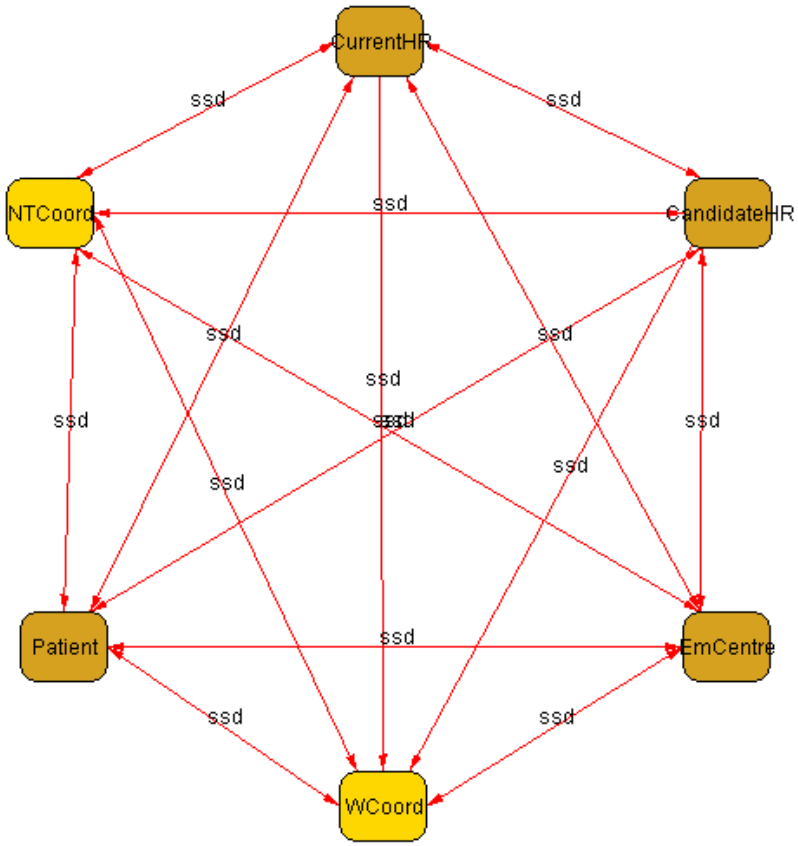

Figura 5.1: Roles del marco dialógico y sus relaciones

Partículas ilocutorias

Las partículas ilocutorias definen los tipos de mensajes que se pueden intercambiar en la institución:

- **accept:** El emisor informa al receptor de que su petición va a ser llevada a cabo, ya sea una petición de registro/deregistro en la mesa de enfermería, o una petición de traslado de paciente.
- **reject:** El emisor informa al receptor de que su petición (de los tipos expuestos anteriormente) no va a ser llevada a cabo.
- **ask**: El emisor pide al receptor alguna información, ya sea la capacidad de un hospital, las preferencias del paciente respecto a los hospitales que pueden atenderle, o las preferencias de los hospitales respecto a los tipos de pacientes que desean atender.
- **inform:** El emisor da al receptor una información concreta. Dicha información puede haber sido previamente solicitada por el receptor, o ser una notificación de parte del emisor.
- **request:** El emisor pide al receptor que lleve a cabo una acción determinada. El contenido del mensaje comprenderá una descripción de la acción a realizar.

Funciones

Ahora, definiremos las funciones que se utilizarán en las escenas de nuestra estructura organizativa. Incluiremos una descripción de cada función y las escenas en las que se utiliza (entre corchetes):

- **acceptDecision():** El representante del hospital candidato acepta la decisión de la mesa de enfermería. [DissolveNurseryTableST]
- **acceptInvitation():** El representante del hospital candidato acepta una invitación para participar en la mesa de enfermería. [WaitingRoomST]
- **answer(String reason):** El coordinador de la mesa de enfermería responde a una petición realizada por uno de los hospitales, dando junto con dicha respuesta la razón de tal decisión. [RequestTransferST, RegisterST, DeregisterST]

- **askCapacity():** El coordinador de la mesa de enfermería pide a un hospital que le informe sobre su capacidad. [AskHospitalST]

- **askForDeregistration(String hospitalName):** Un hospital pide al coordinador su deregistro de la mesa de enfermería. [DeRegisterST]

- **askForRegistration(HospitalInfo h):** Un hospital pide al coordinador su registro en la mesa de enfermería. [RegisterST]

- **askHospitalPreferences():** El coordinador de la mesa de enfermería pide a uno de los hospitales participantes sus preferencias en cuanto a qué pacientes prefiere tener en su servicio. [AskHospitalPreferencesST]

- **askPreferences():** El coordinador de la mesa de enfermería pide al paciente sobre sus preferencias respecto a qué hospital prefiere ser trasladado. [AskPatientPreferencesST]

- **closeScene():** El coordinador de la mesa de enfermería informa a todos los hospitales de que está dejando su puesto como encargado de la escena en la que se encuentra. Tambin lo utiliza el coordinador de la sala de espera para notificar a los hospitales que dicha sala de espera se va a cerrar. [RegisterST, DeRegisterST, RequestTransferST, WaitingRoomST]

- **decisionTaken():** El coordinador informa a un hospital o a un paciente (según la escena) de que el número de hospitales se ha reducido a uno o cero, por lo que sus preferencias ya no son necesarias para tomar una decisión. [AskHospitalPreferencesST, AskHospitalST, AskPatientPreferencesST]

- **destinationDecision(String h):** El coordinador de la mesa de enfermería informa al hospital actual del paciente de qué hospital ha sido el seleccionado para trasladar al paciente. [CommunicationCurrentHospitalST]

- **finished():** El coordinador de la mesa de enfermería informa a todos los hospitales de que el proceso en curso (preguntar capacidades o preguntar preferencias) ha finalizado. También se utiliza para informar de que las notificaciones de disolución de la mesa han finalizado. [AskHospitalPreferencesST, AskHospitalST, DissolveNurseryTableST]

- **giveCapacity(int capacity):** Un hospital al que se le ha pedido su capacidad informa de ella al coordinador de la mesa de enfermería. [AskHospitalST]

- **giveHospitalPreferences(PatientProfileInfo[] c):** Un hospital al que se le han pedido sus preferencias informa de ellas al coordinador de la mesa de enfermería (dichas preferencias consistiendo en una serie de *perfiles de paciente* que prefiere que sean transferidos a su servicio). [AskHospitalPreferencesST]

- **givePreferences(String[] preferences):** Un paciente al que se le ha preguntado por sus preferencias informa de ellas al coordinador de la mesa de enfermería (siendo estas preferencias un conjunto de nombres de hospitales a los cuales preferiría ser transferido). [AskPatientPreferencesST]

- **invite():** El coordinador de la mesa de enfermería invita a un representante de hospital candidato a participar en la mesa de enfermería. [WaitingRoomST]

- **newCase(HospitalInfo[] hospitalList, PatientInfo p):** El coordinador de la mesa de enfermería forma la mesa e informa a todos los miembros del caso que se va a tratar en ella, así como de la lista de hospitales inicial. [FormNurseryTableST]

- **noTransfer(String reason):** El coordinador de la mesa de enfermería informa a todos los hospitales candidatos de que la decisión tomada por la mesa es la de no transferir al paciente, y las razones para tal decisión. [DissolveNurseryTableST]

- **notSelected(String decisionInfo):** El coordinador de la mesa de enfermería informa a un representante de hospital candidato de que no ha sido seleccionado para recibir al paciente cuyo caso se trataba en dicha mesa de enfermería, dándole el nombre del hospital seleccionado finalmente. [DissolveNurseryTableST]

- **patientInfoTransfer(PatientInfo p):** El hospital del paciente transfiere su información al hospital de destino, para que se realicen las preparaciones necesarias. [InformationTransferST]

- **requestTransfer(PatientInfo p, integer priority, string reason):** Un hospital pide un traslado para un paciente que no puede atender apropiadamente, proporcionando la información relevante del paciente, el nivel de prioridad asignado por el hospital y las razones para que dicho traslado sea necesario. El coordinador utilizará esta información para decidir si acepta o rechaza la petición. [RequestTransferST]

- **selected():** El coordinador de la mesa de enfermería informa a un representante de un hospital candidato de que ha sido seleccionado para recibir al paciente. [DissolveNurseryTableST]

- **sendSchedule(ScheduleTime s):** El centro de emergencias informa a los representantes del hospital actual y del hospital de destino de la hora estimada de llegada del vehículo médico asignado al traslado, bien para recoger al paciente o para dejarlo en el hospital de destino. [SendTransInfoCurrentHospitalST, SendTransInfoDestinationHospitalST]

- **transfer(TransferInfo t):** El coordinador de la mesa de enfermería informa al centro de emergencias de que hay un nuevo traslado que debe ser procesado, y le proporciona la información tanto del paciente como de los hospitales actual y de destino, a fin de que el centro de emergencias asigne un vehículo apropiado con el equipamiento necesario. [CommunicationEmCentreST]

Componentes ontológicos

La ontología se completa con las siguientes entidades. Dichas entidades estarán modeladas mediante tipos de datos creados por el usuario, y son la base de la ontología del dominio:

- **HospitalInfo:** Esta entidad contiene toda la información relevante para la mesa de enfermería concerniente a un hospital en concreto. Dicha información es:
  - Un string con el nombre del hospital.
  - Un string con la dirección del hospital.
  - Un entero con la capacidad total del hospital.
  - Una lista de los servicios proporcionados por ese hospital en particular.
  - Una lista de objetos PatientProfileInfo con los perfiles de pacientes que se prefieren en ese hospital.
- **PatientInfo:** Esta entidad contiene toda la información relevante del paciente en términos de un traslado. Esta información es:
  - Un string con el nombre del paciente.
  - Un entero con la edad del paciente.
  - Un string con el diagnóstico del paciente.
  - Una lista de los servicios requeridos para atender al paciente.
  - Un entero con el nivel de urgencia del paciente (de 0 a 3, de menor a mayor).
  - Un String con el tratamiento requerido.
- **PatientProfileInfo:** Esta entidad contiene toda la información de un perfil de paciente que un hospital querría recibir. Esta información es:
  - Un string describiendo el perfil: sexo, dolencia, etc.
  - Un entero con la edad mínima.
  - Un entero con la edad máxima.
- **ScheduleTime:** Esta entidad contiene informacion sobre la hora estimada de llegada de un vehículo médico a un hospital, determinada en la planificación de un traslado determinado. Esta información es:
  - Un entero con las horas.
  - Un entero con los minutos
- **TransferInfo:** Esta entidad contiene la información relevante de un traslado a atender por el centro de emergencias. Esta información es:

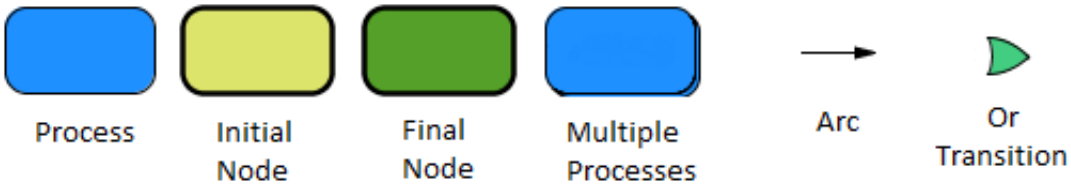

Figura 5.2: Notación utilizada en los diagramas de estructuras performativas

- Un objeto PatientInfo con la información del paciente.
- Un objeto HospitalInfo con la información del hospital actual del paciente.
- Un objeto HospitalInfo con la información del hospital de destino.

5.2.2. Estructuras performativas

En esta sección describiremos con detalle las diferentes estructuras performativas presentes en la aplicación de coordinación de traslados inter-hospitalarios. La notación utilizada en estos diagramas puede verse en la figura 5.2.

Es conveniente señalar aquí que, por conveniencia, las estructuras performativas y tipos de escena descritas en las secciones 5.2.2 y 5.2.3 están nombradas a partir de los nodos de la estructura superior que están implementando, añadiendo un ”PS” extra en el caso de estructuras performativas, y un ”ST” en el caso de tipos de escena.

GeneralPS

GeneralPS es la estructura performativa principal. Es por la que los agentes entran y salen de la institución electrónica. En algunas escenas esta estructura incluye, en lugar de un tipo de escena, una estructura performativa anidada, en aras a proporcionar flexibilidad a los diseñadores de sucesivas versiones de la aplicación a la hora de modificar o ampliar el diseño, consiguiendo a su vez que éste no sea excesivamente complejo. El diagrama de esta estructura puede verse en la figura 5.3.

Esta estructura performativa incluye en sus nodos tres estructuras performativas anidadas (RegistrationCentrePS, DestinationHospitalDecisionPS y PatientTransferPS) y dos escenas simples (es decir, descritas mediante tipos de escena): RequestTransferST y WaitingRoomST. Esta estructura implementa el proceso general de traslado de un paciente:

- Un hospital solicita el traslado de un paciente [escena *RequestTransfer*].

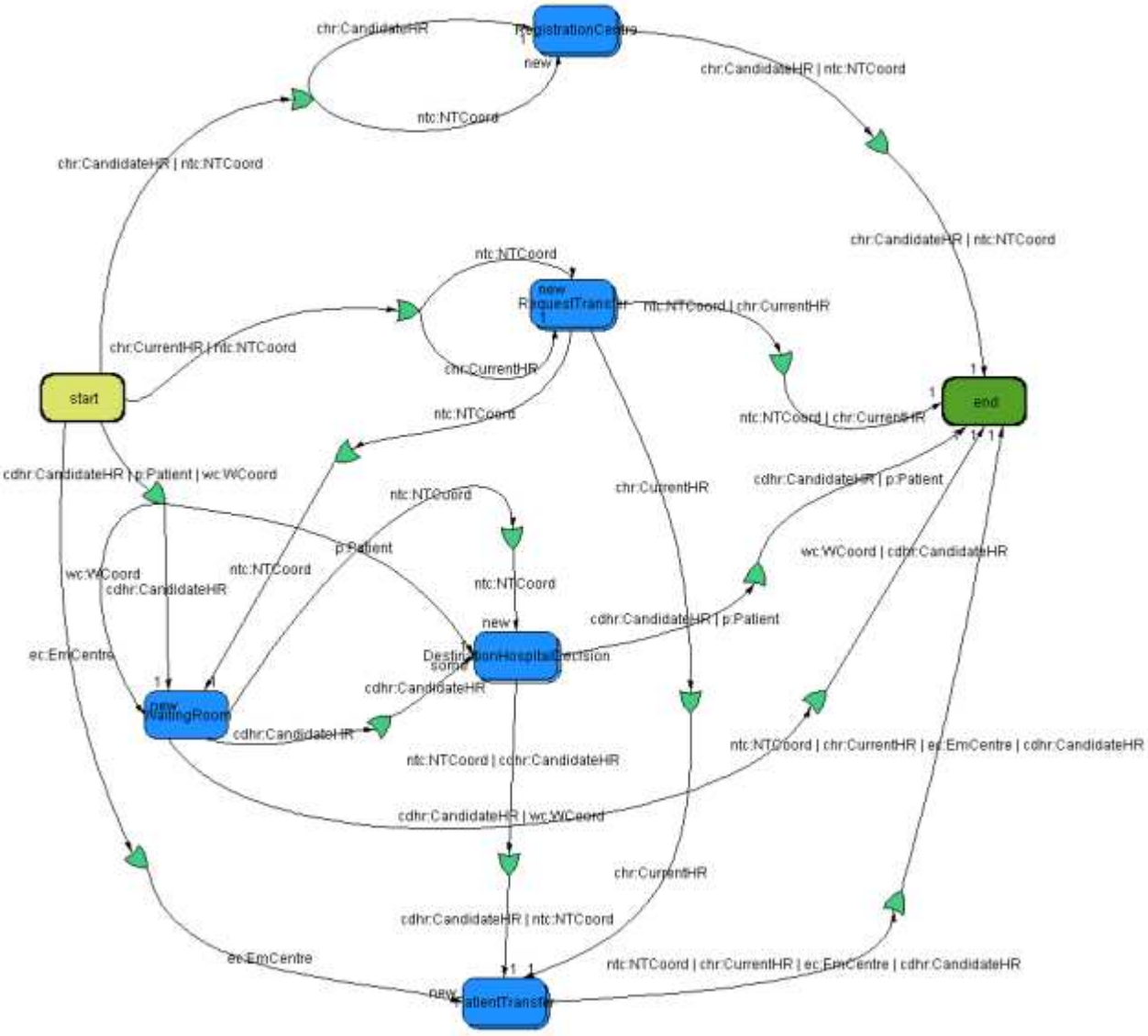

Figura 5.3: La estructura performativa GeneralPS

- Si se acepta, el coordinador de la mesa de enfermería invita a los hospitales que proporcionan los servicios requeridos a que participen en la mesa [escena *WaitingRoom*].

- La mesa de enfermería decide a qué hospital trasladar al paciente [escena *DestinationHospitalDecision*].

- El paciente es trasladado al hospital de destino [escena *PatientTransfer*].

Asimismo, en esta estructura performativa general vemos una escena adicional, *RegistrationCentre*. En dicha escena los hospitales pueden pedir su registro en la mesa de enfermería (si quieren tomar parte en ella) o su deregistro (si quieren dejar de ser invitados a las mesas de negociación de los traslados). Le corresponderá al coordinador de la mesa de enfermería decidir si acepta o rechaza tales peticiones.

Como podemos ver, puede haber simultáneamente varias instancias de cada una de estas escenas (al igual que en el caso real, donde puede haber varios procesos de traslado o registro procesándose simultáneamente) excepto de la escena *WaitingRoom* la cual es única por la conveniencia de tener en el mismo lugar a todos los hospitales candidatos para poder invitarles a participar en los procesos de decisión de las distintas mesas de enfermería.

RegistrationCentrePS

Esta estructura performativa puede verse en la figura 5.4. Extiende el nodo *RegistrationCentre* en la estructura performativa *GeneralPS*. Esta estructura performativa incluye solamente dos escenas, *Register* y *Deregister*. *Register* es la escena en la que los hospitales pueden solicitar su registro en la mesa de enfermería, mientras que *Deregister* sirve para solicitar su deregistro, si ése fuera el caso. Nótese que un coordinador de la mesa de enfermería debe entrar a las dos escenas, para que un hospital pueda entrar en cualquiera de las instancias de RegistrationCentre y solicitar un registro o deregistro.

DestinationHospitalDecisionPS

Esta estructura performativa puede verse en la figura 5.5. Extiende el nodo *DestinationHospitalDecision* en la estructura performativa *GeneralPS*.Esta estructura performativa comprende dos escenas (*FormNurseryTable*, y *DissolveNurseryTable*) y una estructura performativa anidada en la escena *Decision*. Representa el proceso general de decidir un hospital de destino para el paciente:

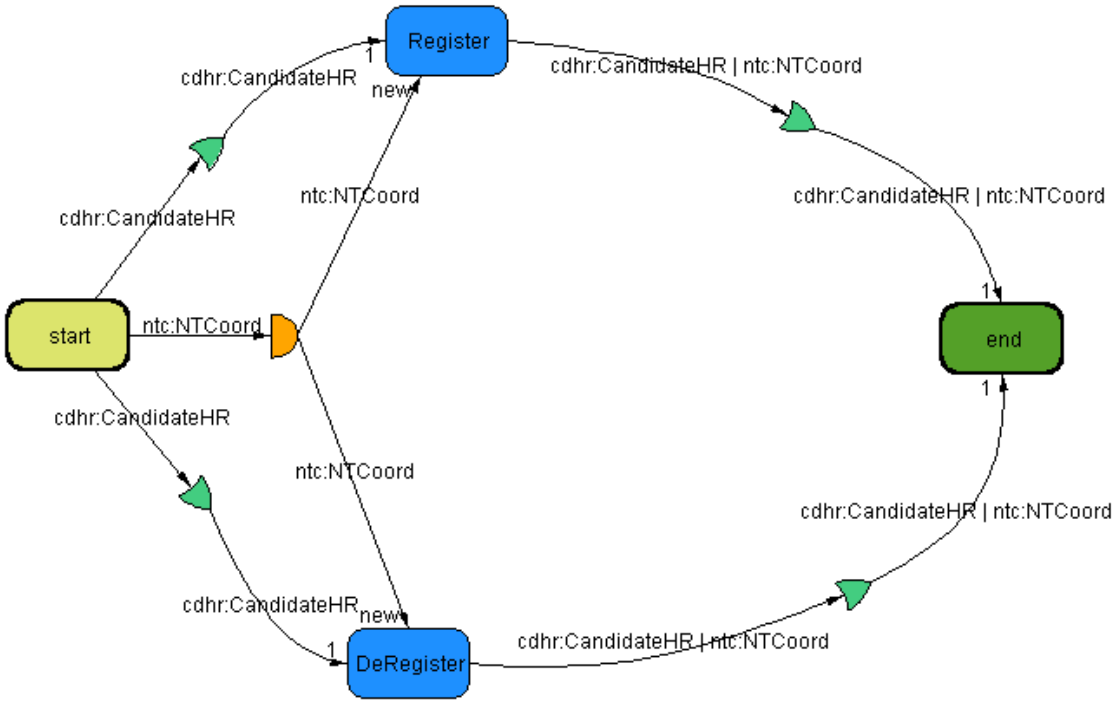

Figura 5.4: La estructura performativa RegistrationCentrePS

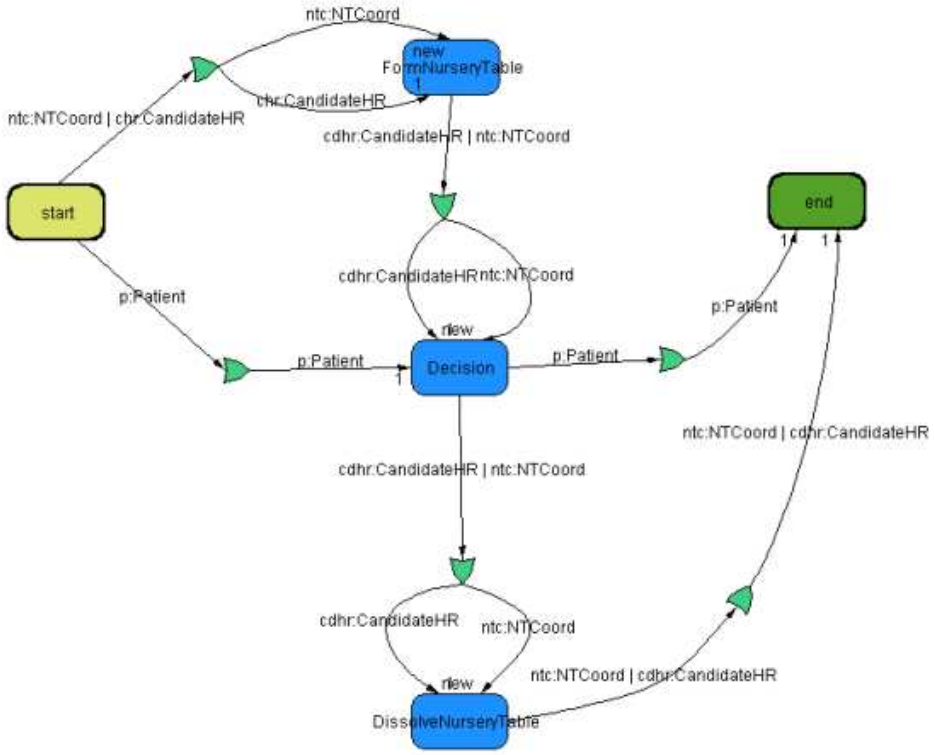

Figura 5.5: La estructura performativa DestinationHospitalDecisionPS

1. Se forma la mesa de enfermería [escena *FormNurseryTable*].
2. Se toma la decisión [escena *Decision*].
3. Se disuelve la mesa de enfermería [escena *DissolveNurseryTable*].

Estas escenas son todas únicas, por lo que, en una determinada instancia de DestinationHospitalDecisionPS sólo puede haber una instancia de cada una de ellas, aunque de la propia estructura performativa pueda haber varias instancias.

PatientTransferPS

Esta estructura performativa puede verse en la figura 5.6. Extiende el nodo *PatientTransfer* en la estructura performativa *GeneralPS*. Esta estructura performativa comprende tres escenas (*CommunicationCurrentHospital*, *InformationTransfer*, y *CommunicationEmCentre*) y una estructura performativa (en la escena *SendTransportationData*). Representa el proceso de traslado de un paciente:

1. El coordinador de la mesa de enfermería informa al hospital que pidió el traslado del paciente sobre la decisión tomada por la mesa de enfermería [escena *CommunicationCurrentHospital*].

2. El hospital del paciente envía sus datos médicos al hospital de destino [escena *InformationTransfer*].

3. El coordinador de la mesa de enfermería informa al centro de emergencias de que es necesario un nuevo traslado, proporcionando la información necesaria sobre el paciente y los hospitales involucrados [escena *CommunicationEmCentre*].

4. El centro de emergencias informa a los dos hospitales de la planificación del traslado. [escena *SendTransportationData*].

Si se decidiese no trasladar al paciente, el hospital actual y el coordinador de la mesa de enfermería dejarían la escena, dejando esa instancia disponible para otro traslado, ya que la crea el centro de emergencias. Como ocurre con la estructura *DestinationHospitalDecisionPS*, todas estas escenas son únicas.

SendTransportationDataPS

Esta estructura performativa puede verse en la figura 5.7. Extiende el nodo *SendTransportationData* en la estructura performativa *PatientTransferPS*. Esta estructura comprende solamente dos escenas, *SendTransInfoCurrentHospital* y *SendTransInfoDestinationHospital*. Representa el proceso de informar a los hospitales involucrados en un traslado (el actual y el de destino) sobre la planificación del mismo:

1. El centro de emergencias informa al hospital actual de la hora de llegada del vehículo asignado para recoger al paciente [escena *SendTransInfoCurrentHospital*].

2. El centro de emergencias informa al hospital de destino de la hora estimada de llegada del paciente a sus instalaciones [escena *SendTransInfoDestinationHospital*].

Como en los casos anteriores, estas escenas son únicas.

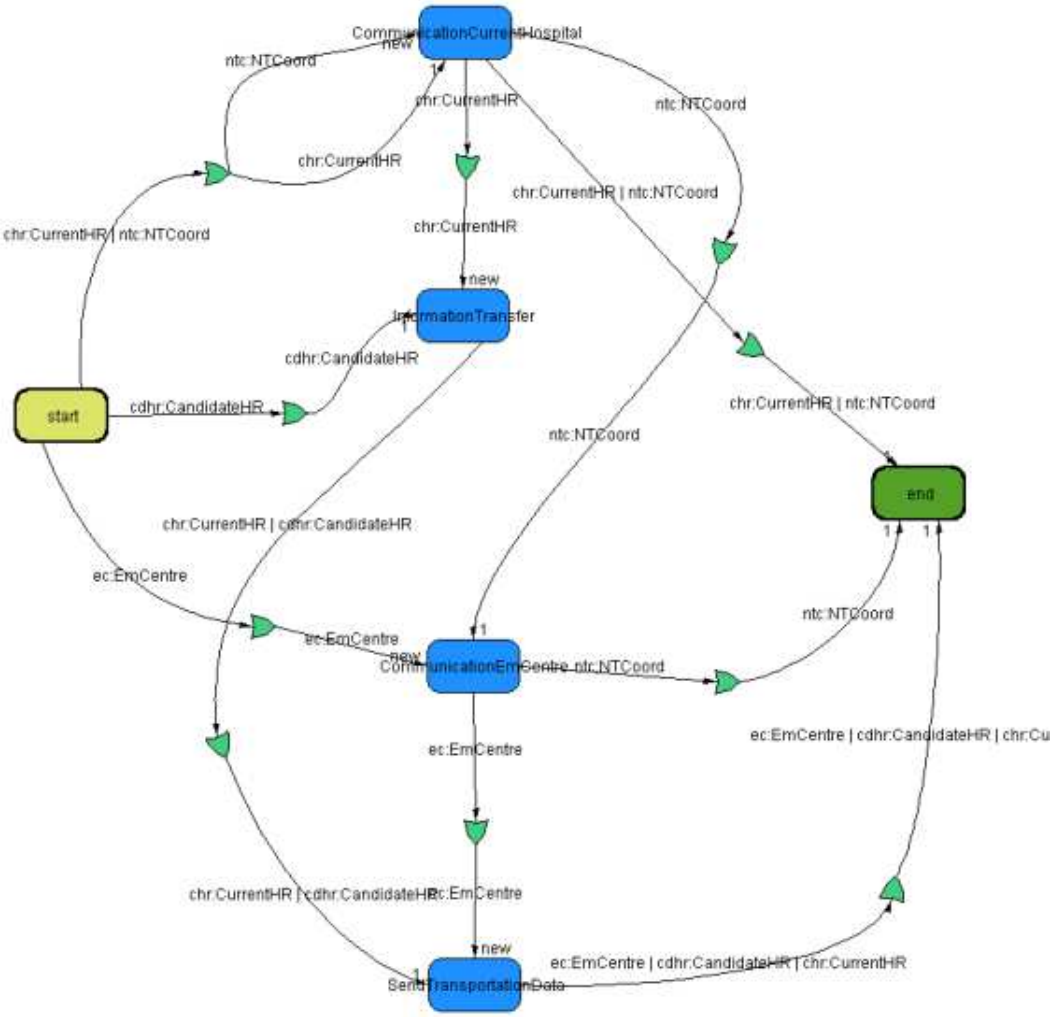

Figura 5.6: La estructura performativa PatientTransferPS

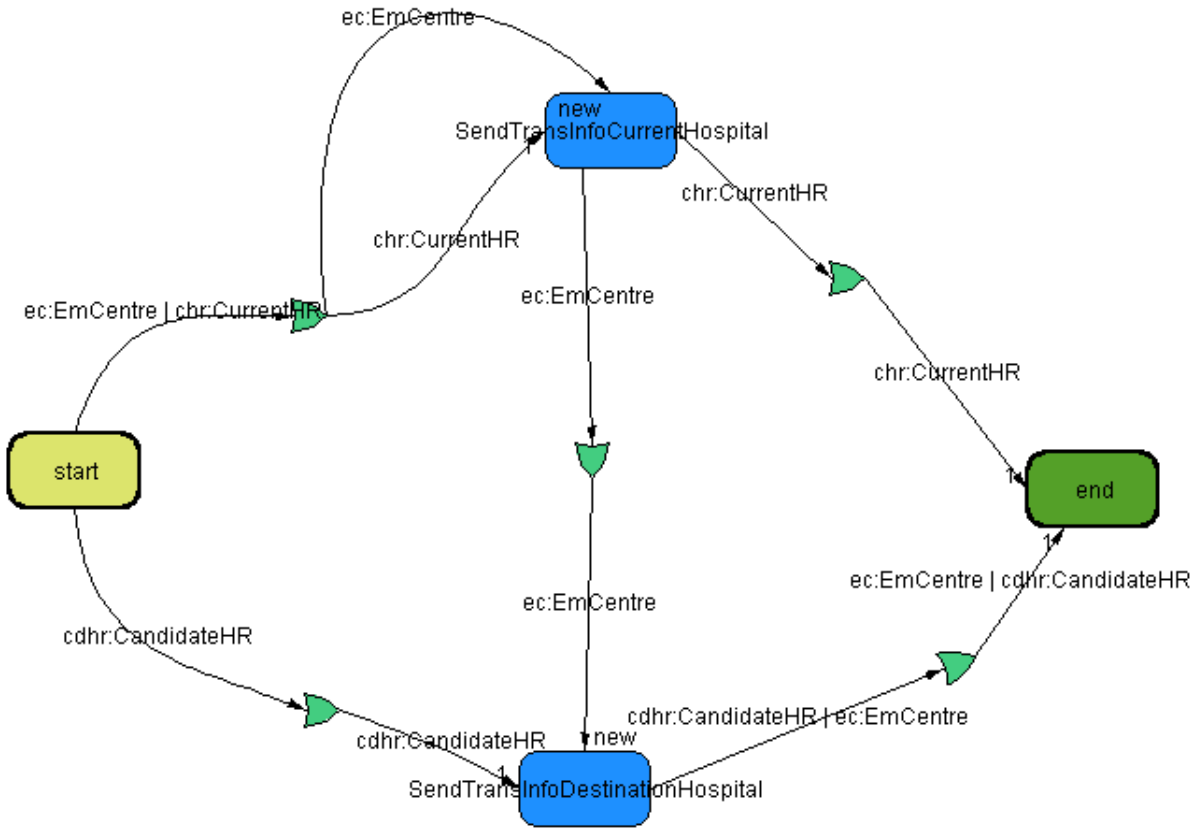

Figura 5.7: La estructura performativa SendTransportationDataPS

DecisionPS

Esta estructura performativa puede verse en la figura 5.8. Extiende el nodo *Decision* en la estructura performativa *DestinationHospitalDecisionPS*. Esta estructura comprende tres escenas (*AskHospital*, *AskHospitalPreferences*, y *AskPatientPreferences*). Representa el proceso de decisión del hospital de destino una vez formada la mesa de enfermería, siguiendo el ejemplo de proceso de decisión visto en la sección 4.5:

1. El coordinador de la mesa de enfermería pregunta a los hospitales que prestan los servicios requeridos por su capacidad [escena *AskHospital*].

2. El coordinador de la mesa de enfermería pregunta al paciente por sus prioridades (o le informa de que dicha información ya no es necesaria, al haber ya solamente un hospital disponible, o debido a que ninguno de ellos haya pasado la criba anterior) [escena *AskPatientPreferences*].

3. El coordinador de la mesa de enfermería pregunta a los hospitales restantes por sus preferencias (o les informa de que dicha información ya no es necesaria, al haber ya solamente un hospital disponible, o debido a que ninguno de ellos haya pasado la criba anterior) [escena *AskPatientPreferences*].

Si en algún punto del proceso, el número de hospitales disminuyese a cero, entonces la decisión se toma seleccionando un hospital al azar del último conjunto de candidatos no vacío. De haber más de uno, se toma uno aleatoriamente del conjunto de hospitales actual. Si solamente quedase uno, ése sería el seleccionado. Esta acción ocurre en el arco de transición desde *AskHospitalPreferences* hasta el nodo final.

Asimismo, dado que la intención de este diseño es dar cabida a diferentes procesos de decisión, esta escena involucra todos los roles relevantes (los hospitales, el coordinador, y el paciente). Será por tanto responsabilidad del diseñador decidir si utiliza dichos roles o no.

5.2.3. Tipos de escena

En esta sección describiremos los tipos de escena que encontramos en el diseño. Recordemos que un tipo de escena es una descripción de las interacciones que se realizan en una escena, definida mediante estados y transiciones entre ellos. La notación utilizada para dichos diagramas se puede observar en la figura 5.9.

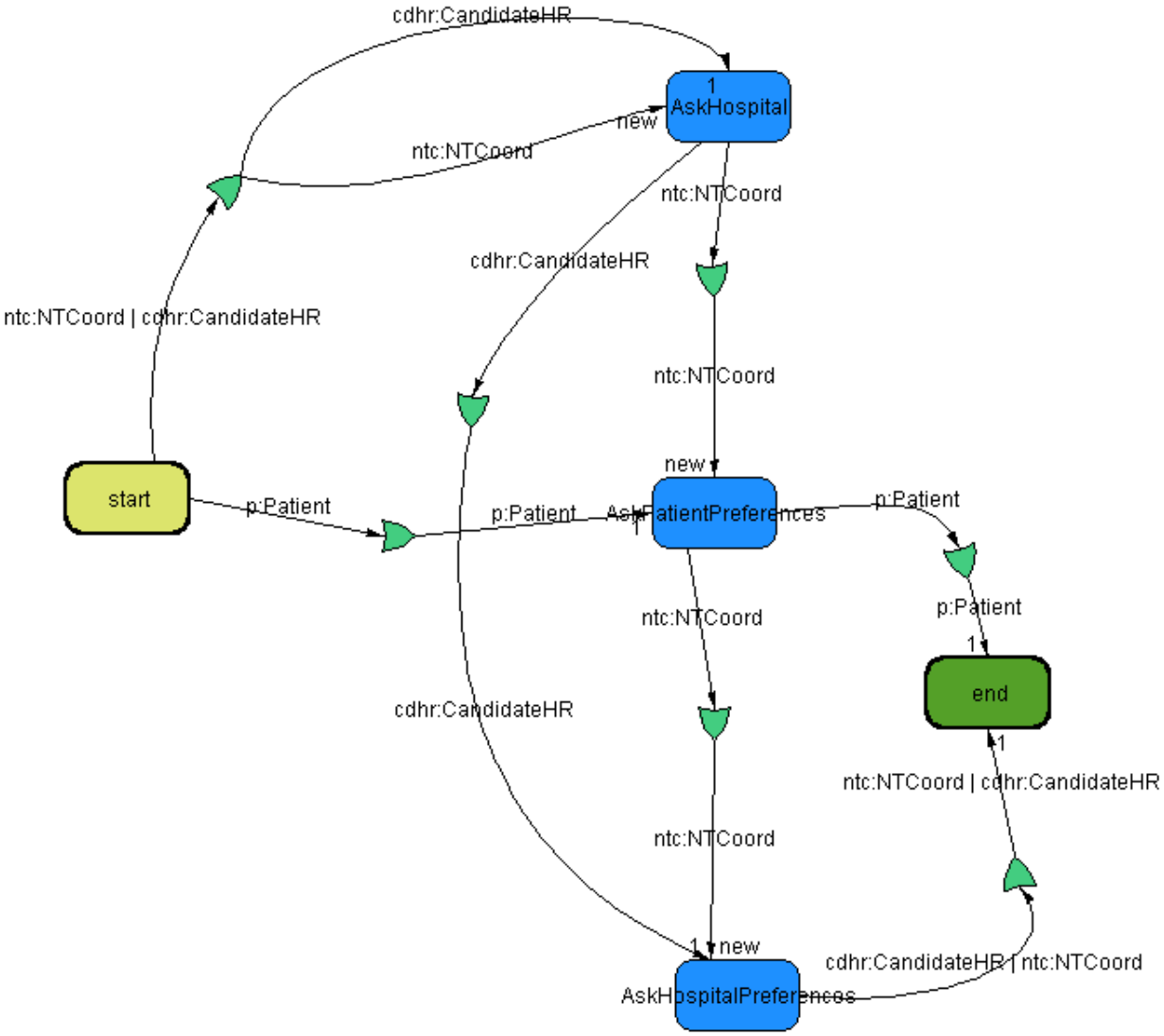

Figura 5.8: La estructura performativa DecisionPS

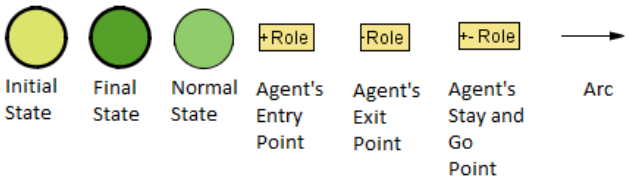

Figura 5.9: Notación utilizada en los diagramas de los tipos de escena

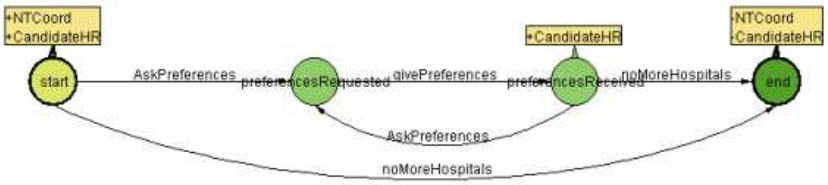

Figura 5.10: La escena AskHospitalPreferencesST

AskHospitalPreferencesST

En esta escena, como podemos observar en la figura 5.10, el coordinador de la mesa de enfermería pregunta a los diferentes hospitales canditatos sobre sus preferencias respecto al tipo de pacientes que prefieren recibir. Cuando un hospital ha dado sus preferencias, el coordinador decide si continúa con las preguntas o envía una notificación de salir de la escena. Dicha decisión será responsabilidad de la implementación del agente, por lo que no la trataremos aquí. Por otra parte, al iniciar la escena, el coordinador tiene la posibilidad de enviar un mensaje diciendo que ya no se necesita mas información, el cual se utilizará típicamente cuando sólo haya un hospital que pueda recibir al paciente. En esta escena, además de los estados inicial y final, podemos encontrar los siguientes:

- **preferencesRequested**: En este estado, las preferencias del hospital al que se consulta han sido solicitadas, y el coordinador está esperando por la respuesta. Debido a ello, en este estado ningún agente puede salir de la escena.
- **preferencesReceived**: En este estado, las preferencias del hospital han sido recibidas por el coordinador de la mesa. En este estado es en el que aparece la decisión comentada anteriormente sobre si seguir preguntando o salir de la escena. Dado que este estado es relativamente estable, los hospitales también pueden entrar por él a la escena.

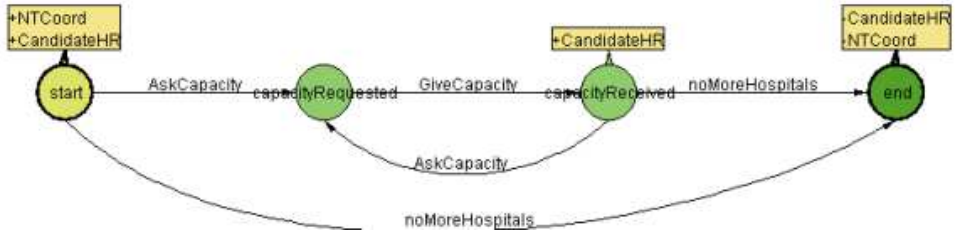

Figura 5.11: El tipo de escena AskHospitalST

AskHospitalST

En esta escena, como podemos observar en la figura 5.11, el coordinador de la mesa de enfermería pregunta a los diferentes hospitales por sus capacidades, a fin de decidir si son capaces de recibir al paciente o no. Cuando un hospital informa de su capacidad, el coordinador, al igual que en la escena anterior, puede decidir si continuar con las consultas o terminar el proceso. Por otra parte, en esta escena el coordinador tiene también al inicio la posibilidad de enviar un mensaje diciendo que ya no se necesita mas información, el cual se utilizará típicamente cuando sólo haya un hospital que pueda recibir al paciente. En esta escena, además de los estados inicial y final, podemos encontrar los siguientes:

- **capacityRequested**: En este estado, la capacidad de un hospital de la lista ha sido solicitada, y el coordinador de la mesa de enfermería está esperando por la respuesta. Debido a ello, en este estado ningún agente puede salir de la escena.
- **capacityReceived**: En este estado, la información sobre las capacidades del hospital ha sido recibida por el coordinador de la mesa. En este estado es en el que aparece la decisión comentada anteriormente sobre si seguir preguntando o salir de la escena. Dado que este estado es relativamente estable, los hospitales también pueden entrar por él a la escena.

AskPatientPreferencesST

En esta escena, como podemos observar en la figura 5.12, el coordinador de la mesa de enfermería pregunta al paciente sobre sus preferencias respecto a qué hospital prefiere ser transferido. Una vez que el paciente informa de dichas preferencias, la escena se termina. Sin embargo, si el coordinador considera que la información ya no es necesaria, puede enviar un mensaje notificando de ello al paciente. Junto al estado inicial encontramos los siguientes:

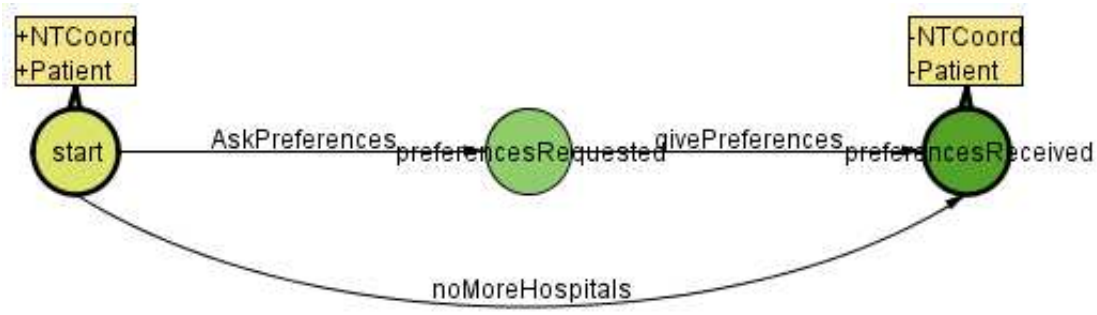

Figura 5.12: El tipo de escena AskPatientPreferencesST

Figura 5.13: El tipo de escena CommunicationCurrentHospitalST

- **preferencesRequested**: En este estado, el coordinador de la mesa ha solicitado las preferencias del paciente pero no las ha recibido, por lo que ninguno de los dos agentes puede dejar la escena.

- **preferencesReceived**: En este estado, las preferencias han sido recibidas por el coordinador, por lo que ambos agentes pueden dejar ya la escena.

CommunicationCurrentHospitalST

En esta escena, como podemos observar en la figura 5.13, el coordinador de la mesa de enfermería informa al hospital actual del paciente sobre la decisión adoptada

Figura 5.14: El tipo de escena CommunicationEmCentreST

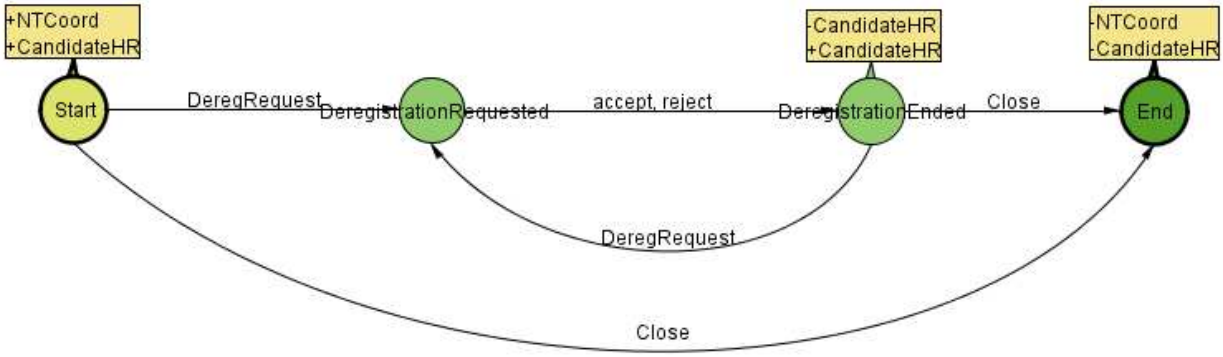

Figura 5.15: El tipo de escena DeRegisterST

por la mesa de enfermería. Asimismo, si la decisión final es no transferir al paciente, el coordinador informa a los hospitales de tal circunstancia.

Esta escena, al igual que otras en el diseño, es muy simple y puede ser considerada superflua, pero es importante desde el punto de vista del diseño, dado que así se ven de manera clara las etapas del proceso de traslado.

CommunicationEmCentreST

En esta escena, como podemos observar en la figura 5.14, el coordinador de la mesa de enfermería informa al centro de emergencias de que hay un nuevo traslado que realizar, dándole la información necesaria sobre el paciente y los dos hospitales involucrados, a fin de que se realicen los preparativos necesarios.

DeRegisterST

En esta escena, como podemos observar en la figura 5.15, un hospital registrado pide su deregistro al coordinador de la mesa de enfermería, el cual puede aceptar dicha

petición o rechazarla, dando al hospital una razón de tal decisión. Una vez que el hospital ha recibido respuesta, puede dejar la escena. El coordinador, sin embargo, se queda atendiendo nuevas peticiones. En esta escena, aparte de los nodos inicial y final, encontramos los siguientes:

- **DeregistrationRequested:** En este estado, la petición de deregistro ha sido enviada al coordinador, pero no se ha recibido respeusta. Al igual que en otras escenas parecidas, en este estado ningún agente puede salir o entrar en la escena.
- **DeregistrationEnded:** En este estado, la decisión (ya sea aceptar o rechazar la petición) ya ha sido tomada y comunicada al hospital junto con las razones para tal decisión. Es en este estado, por tanto, en el que el hospital puede dejar la escena, mientras que el coordinador se queda en ella esperando nuevas peticiones, procedentes de hospitales que entren en la escena en este estado.

DissolveNurseryTableST

En esta escena, como podemos observar en la figura 5.16, el coordinador de la mesa de enfermería informa a todos los hospitales del resultado del proceso de decisión (es decir, si han sido seleccionados o no para recibir al paciente) antes de disolver la mesa de enfermería. Cada hospital debe aceptar la decisión a fin de confirmar la recepción del mensaje.

Por otra parte, si la decisión final es no transferir al paciente, el coordinador informa a todos los hospitales de tal circunstancia.

FormNurseryTableST

En esta escena, como podemos observar en la figura 5.17, el coordinador de la mesa de enfermería informa a los hospitales del caso que se va a tratar, así como de la lista de los hospitales que son capaces de recibir a dicho paciente. Aunque esta lista de hospitales no se utiliza en el proceso de decisión de ejemplo, se ha incluido en el diseño final a fin de incrementar la extensibilidad del diseño, dado que hay algunos procesos de decisión (consenso, votación) podrían requerir que los hospitales conocieran la información del resto de hospitales invitados a la hora de decidir.

InformationTransferST

En esta escena, como podemos observar en la figura 5.18, el hospital actual del paciente envía la información relevante del paciente al hospital de destino, para que éste pueda realizar los preparativos necesarios para recibirlo.

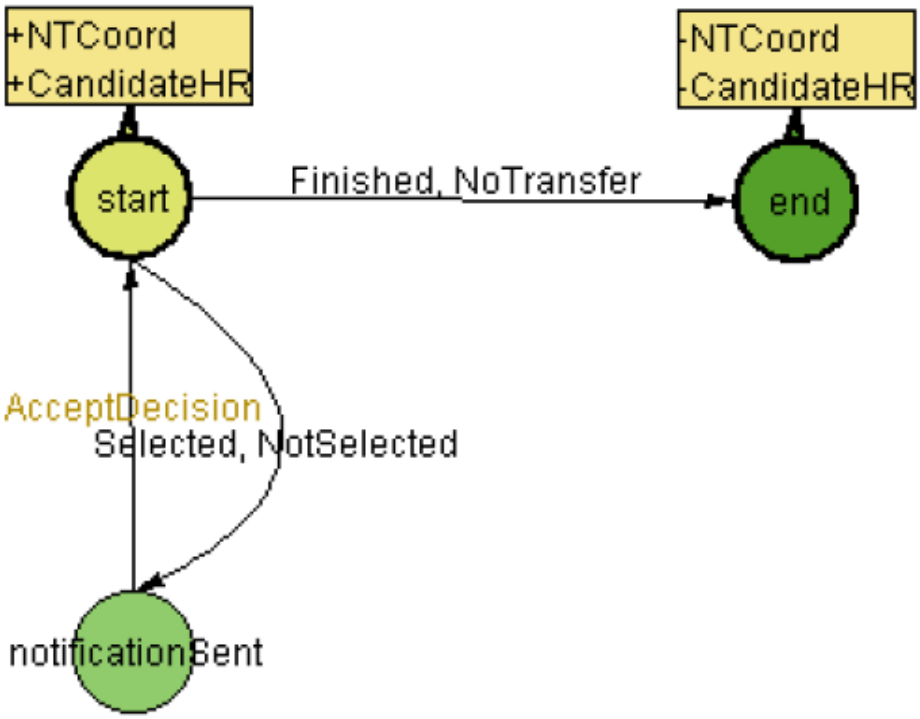

Figura 5.16: El tipo de escena DissolveNurseryTableST

Figura 5.17: El tipo de escena FormNurseryTableST

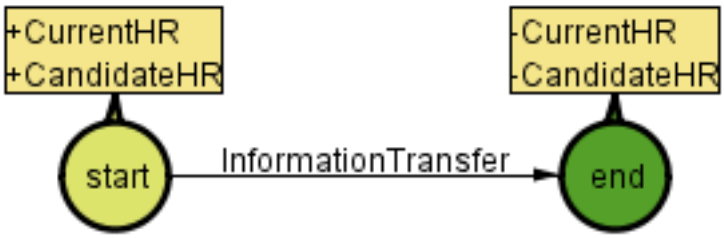

Figura 5.18: El tipo de escena InformationTransferST

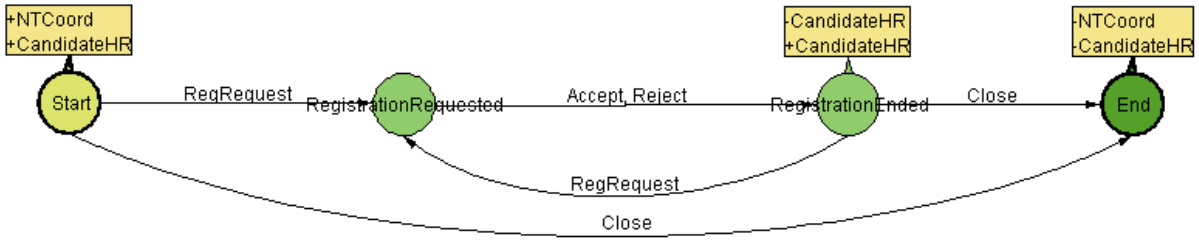

Figura 5.19: El tipo de escena RegisterST

RegisterST

En esta escena, como podemos observar en la figura 5.19, un hospital no registrado solicita su registro al coordinador de la mesa de enfermería, el cual puede decidir aceptar o rechazar dicha petición, dando al hospital las razones de tal decisión. Una vez que el hospital ha sido notificado de la decisión, puede dejar la escena, mientras que el coordinador permanece en la escena para atender nuevas peticiones. Aparte de los estados inicial y final, se pueden encontrar los siguientes:

- **RegistrationRequested**: En este estado, la petición de registro se ha enviado pero no ha recibido respuesta, por lo que en este estado ningún agente puede puede entrar o salir de la escena.
- **RegistrationEnded**: En este estado, la decisión (ya sea aceptar o rechazar la petición de registro) ya ha sido tomada, y el hospital ha recibido la respuesta a su petición. Por tanto, en este estado el hospital solicitante ya puede dejar la escena, y otros hospitales podrán entrar en ella para realizar sus peticiones de registro, ya que el coordinador permanece en la escena.

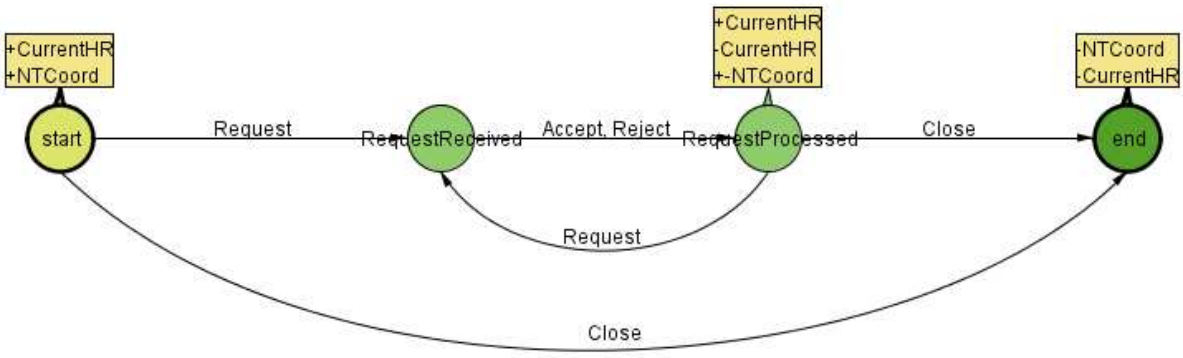

Figura 5.20: El tipo de escena RequestTransferST

RequestTransferST

En esta escena, como podemos observar en la figura 5.20, un hospital solicita un traslado al coordinador de la mesa de enfermería. Por su parte, el coordinador puede decidir aceptar o rechazar dicha petición, dando al hospital una razón para ello. Una vez que el hospital ha sido notificado respecto a la decisión tomada, puede dejar la escena. Asimismo, si la petición ha sido aceptada, el coordinador realiza un *stay and go* a fin de poder permanecer en la escena atendiendo nuevas peticiones y, simultáneamente, activar el proceso de traslado. En esta escena, adicionalmente a los nodos inicial y final, podemos encontrar los siguientes:

- **RequestReceived**: En este estado, el hospital ha enviado su petición al coordinador y está esperando respuesta. En este estado, por tanto, ningún agente puede entrar o salir de la escena.
- **RequestProcessed**: En este estado, la decisión de aceptar o rechazar la petición ha sido tomada por el coordinador, y el hospital ha recibido su respuesta con la decisión y las razones para ella. De este modo, el hospital puede salir de la escena. El coordinador, por otra parte, puede elegir entre permanecer a la espera de nuevas peticiones (dado que en este estado pueden entrar nuevos hospitales solicitantes), cerrar la escena, o realizar un *stay and go*, para activar el proceso de traslado y a su vez permanecer a la espera de nuevas peticiones de traslado.

SendTransInfoCurrentHospitalST

En esta escena, como podemos observar en la figura 5.21, el centro de emergencias envía al hospital actual del paciente la hora a la que el vehículo asignado en la planificación del traslado llegará para recoger al paciente.

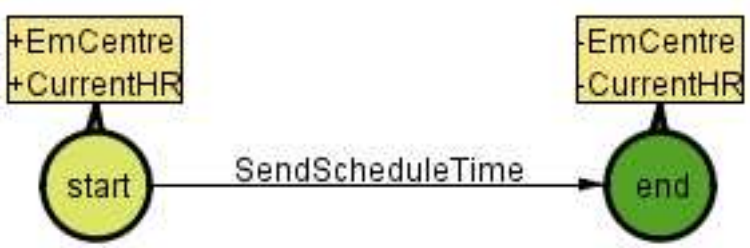

Figura 5.21: El tipo de escena SendTransInfoCurrentHospitalST

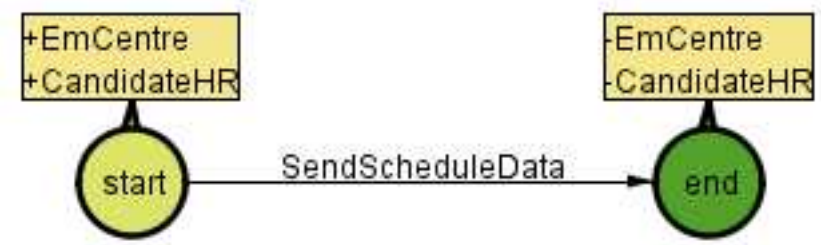

Figura 5.22: El tipo de escena SendTransInfoDestinationHospitalST

SendTransInfoDestinationHospitalST

En esta escena, como podemos observar en la figura 5.22, el centro de emergencias envia al hospital de destino la hora a la que llegará el vehículo asignado para el traslado del paciente. De esta manera, el hospital de destino podrá realizar los preparativos necesarios para la llegada del paciente.

WaitingRoomST

En esta escena, como podemos observar en la figura 5.23, el coordinador de la mesa de enfermería, tras recibir una petición de traslado, invita a cada uno de los hospitales que proporcionan los servicios requeridos por el caso a la formación de la mesa de enfermería. Tras confirmar la recepción de la invitación, cada uno de los hospitales invitados realiza un *stay and go* para poder tanto recibir nuevas invitaciones como tomar parte en la mesa de enfermería.

En esta escena, además de los estados inicial y final, encontramos los siguientes:

- **InvitationSent**: En este estado, el coordinador ha enviado una invitación, y está esperando a la confirmación por parte del hospital.
- **InvitationProcessed**: En este estado, se ha confirmado la recepción de la invitación. En este punto, el hospital que ha recibido la invitación puede realizar un *stay and*

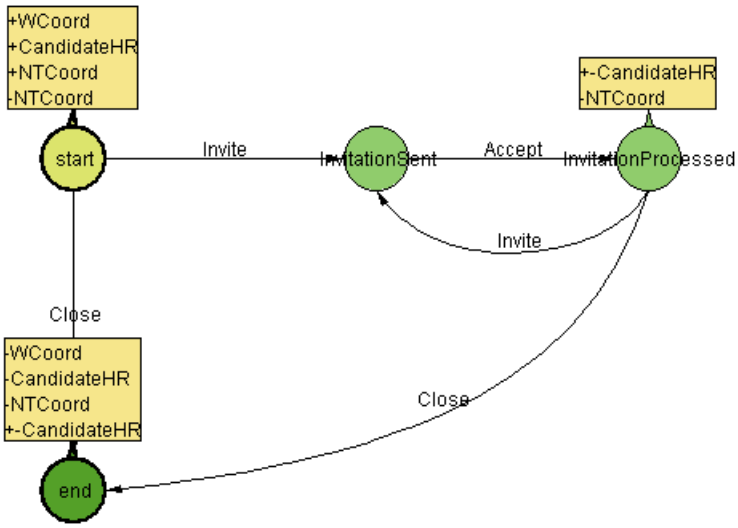

Figura 5.23: El tipo de escena WaitingRoomST

go, para ir al proceso de decisión y a su vez permanecer a la espera de nuevas invitaciones.

- También cabe destacar que, en el estado final, los hospitales también pueden realizar un *stay and go*. Esto es así para evitar que algún hospital candidato se quede sin poder realizar el *stay and go* necesario.

5.2.4. Restricciones en la actuación de los agentes

Como hemos visto en la sección 5.2.2, los agentes, dependiendo del rol que interpreten en la institución electrónica, se encuentran con una bifurcación en alguna parte del proceso, en la cual deben elegir qué camino tomar. Sin embargo, algunas de estas decisiones pueden acarrear situaciones problemáticas, como un coordinador saliendo de la institución electrónica tras haber aceptado una petición de traslado que nunca llega a procesarse, o un hospital candidato tomando parte en procesos de decisión que no le corresponden.

Debido a esto, ha sido necesario implementar algunas restricciones dentro de la especificación respecto a cúando uno de dichos agentes debería tomar un camino u otro en dichas bifurcaciones. Estas restricciones se han definido mediante la capacidad de ISLANDER de especificar precondiciones en los arcos de transición, los cuales permiten limitar cuándo se puede transitar por dicho arco.

De esta forma, se impide que ocurran tales situaciones problemáticas aunque los agentes intenten actuar de forma diferente a la que dicta su comportamiento esperado, asegurando de este modo el cumplimiento de los requisitos expuestos en la sección 4.5.

En esta subsección presentaremos las circunstancias en las que se producen dichas bifurcaciones, los roles que pueden acceder a ellas, las restricciones y propiedades utilizadas a fin de anular las decisiones problemáticas, y la generación y gestión de dichas propiedades:

- La primera bifurcación ocurre en la estructura perfomativa *GeneralPS*, cuando el coordinador de la mesa de enfermería (con el rol **NTCoord**), al salir de la escena *RequestTransfer*, puede o bien salir de la institución electrónica (lo que llamaremos el ”camino de salida”) o seguir el camino a la escena *WaitingRoom*, en la que invitará a los hospitales que proporcionen los servicios requeridos por el traslado solicitado (lo que llamaremos el ”camino de traslado”).

  Para evitar que esta bifurcación tenga efectos colaterales problemáticos, se utiliza la propiedad ***transfers*** del rol **NTCoord**, para que si ***transfers*** es mayor que cero, el agente podrá seguir únicamente el camino de traslado, lo que restará una unidad a ***transfers***. Por contra, si no hay traslados pendientes (o se ha realizado un número adecuado de *stay and go* de manera que todas las peticiones estén atendidas) solamente podrá seguir el camino de salida, para evitar que haya más procesos de traslado en curso de los necesarios.

Esto también evita los problemas derivados de realizar *stay and go* indiscriminadamente, o salir de la escena *RequestTransfer* sin haber realizado un *stay and go* tras aceptar una petición de traslado. En el primer caso, solo se permitirá seguir el camino de traslado al número exacto de agentes, mientras los demás se verán obligados a salir de la institución electrónica. En el otro caso, el agente **NTCoord** que salga antes de tiempo solo sería capaz de seguir el camino de traslado, asegurando así que se procese la petición aceptada.

Relacionada con la anterior, también hemos añadido una precondición en la transición que lleva al NTCood a cerrar la escena, de manera que solamente pueda salir si le queda una petición por atender, o si no le queda ninguna. De este modo, se evita que un NTCoord realice *menos* traslados de los que ha aceptado.

Esta propiedad ***transfers*** tiene valor 0 por defecto, significando que todo agente **NTCoord** es creado con ***transfers*** igual a cero. Tras su creación, esta variable es manejada a través de la escena *RequestTransferST*. En esta escena, aceptar una petición añade una unidad a ***transfers***, mientras que sólo se decrementa en el momento en el que se decide seguir el camino de traslado.

- La segunda bifurcación ocurre en la estructura performativa *GeneralPS*, cuando el representante de un hospital candidato (**CandidateHR**), al salir de la escena *WaitingRoom*, puede o salir de la institución electrónica (”camino de salida”) o continuar a la escena *DestinationHospitalDecision* (”camino de decision”), en el que tomará parte en la mesa de enfermería.

  Como vemos, este caso es similar al descrito anteriormente con el **NTCood**, y tiene una solución similar. Esta vez, cada CandidateHR tiene una propiedad ***invitations*** que lleva la cuenta del número de invitaciones que el agente ha recibido, pero aún no ha procesado. De ser mayor que cero, solamente permitirá seguir el camino de decisión, mientras que de ser igual a cero, solo permitirá seguir el camino de salida.

  De este modo, se evita que los agentes con el rol CandidateHR participen en más procesos de decisión de los que son necesarios, incluso aunque haga más *stay and go* de los que necesita. Desafortunadamente, solamente se puede controlar el número de procesos de decisión en los que participa, pero no en cuáles, por lo que puede ocurrir que un CandidateHR participe en procesos que no le corresponden. Esto es, por tanto, responsabilidad de la implementación del agente.

  Adicionalmente, de nuevo de forma análoga al caso anterior, se han implementado restricciones adicionales en la escena *WaitingRoom*, para que un agente CandidateHR sólo pueda salir totalmente de la escena si no tiene invitaciones pendientes.

  Esta propiedad, al igual que ***transfers*** en el caso anterior, tiene valor cero por defecto, significando que todo agente CandidateHR comienza con su propiedad ***invitations*** igual a cero. Tras ello, esta variable es gestionada mediante la escena *WaitingRoom*, añadiendo una unidad cada vez que se confirma una invitación, y sustrayendo una unidad cuando se sigue el camino de decisión, con los efectos beneficiosos descritos anteriormente.

- La tercera bifurcación ocurre en la estructura performativa *GeneralPS*, cuando el representante de un hospital candidato (**CandidateHR**), saliendo de la escena *DestinationHospitalDecision*, puede o bien salir de la institución electrónica (”camino de salida”) o seguir el camino a la escena *PatientTransfer*, si es que es el hospital de destino seleccionado para el paciente (”camino de traslado”).

  Como se puede observar, este caso es similar a los anteriores, salvo porque no nos tenemos que preocupar de que algún agente haga más *stay and go* de la cuenta. Por tanto, la resolución es similar, o, incluso, más sencilla de comprender. Para ello, utilizamos una propiedad ***newPatients***, la cual, de ser mayor que cero, permite proseguir al camino de traslado, mientras que ser igual a cero obliga a seguir el camino de salida.

  Esto se realiza de esta manera porque, aunque el **CandidateHR** no puede realizar un *stay and go* en la escena *DestinationHospitalDecision*, podría haberlo hecho en la escena *WaitingRoom* para participar en varias mesas diferentes, por lo que podría haber sido seleccionado varias veces.

  Teniendo una propiedad ***newPatients***, y controlando los caminos a seguir con esta propiedad, impide que un hospital que ha sido seleccionado no participe en el proceso de traslado y recepción del nuevo paciente, y también impide que participe en más procesos de traslado de los que le corresponden.

  Esta propiedad tiene valor cero por defecto, y se gestiona mediante la escena *DissolveNurseryTable*. En esta escena, cada notificación de haber sido seleccionado añade una unidad a ***newPatients***, mientras que el contrario no tiene efecto. Por otra parte, ***newPatients*** se decrementará cuando se sigue el camino de traslado, con los efectos descritos anteriormente.

- La cuarta y última bifurcación ocurre cuando el coordinador de la mesa (**NTCoord**) y el representante del hospital actual (**CurrentHR**), saliendo de la escena *CommunicationCurrentHospital* de la estructura performativa *PatientTransfer*, pueden o bien salir de la institución electrónica (”camino de salida”) o bien continuar con el proceso de traslado, siguiendo el camino a las escenas *CommunicationCurrentHospital*, que llevan al traslado del paciente (”camino de traslado”).

  Esta bifurcación ha sido introducida para poder tratar con casos en los que la decisión es no transferir al paciente. En tales casos, no es necesario comunicar nada al centro de emergencias, dado que el origen y el destino del paciente son el mismo hospital.

  Para esto, usamos una propiedad ***noTransfer*** en cada uno de los roles, para controlar su camino y evitar que los agentes salgan de la institución después de decidir que se necesita un traslado. Esta propiedad será un booleano en el caso del **CurrentHR** (dado que los agentes con dicho rol sólo tiene un traslado en curso simultáneamente) y un entero en el caso del **NTCoord** (con el número de traslados en los que la decisión ha sido finalmente no trasladar al paciente).

Esta propiedad se gestiona en la escena *DissolveNurseryTable* para el **NTCoord**, y en la escena *CommunicationCurrentHospitalST* para el **CurrentHR**. En el primer caso, si el **NTCoord** informa a los hospitales de que se ha decidido no trasladar al paciente, se suma una unidad a su propiedad ***noTransfer***, que se decrementará al seguir el camino de salida. Por otra parte, el **CurrentHR** verá cambiada su ***noTransfer*** a *true* cuando el **NTCoord** le notifique que el traslado no tiene lugar, cambiando a *false* cuando siga el camino de salida.

No obstante, cabe destacar que, de elegir una instancia equivocada de la estructura performativa *PatientTransfer*, puede tener efectos colaterales indeseados, como que un **CurrentHR** se meta en un proceso de traslado que no le corresponda. No obstante, en el diseño de la Institución Electrónica no podemos tratar estos problemas, ya que en la especificación solamente se describe las escenas a las que se puede ir, mas no las instancias disponibles de dichas escenas.

Capítulo 6

Pruebas del prototipo

En este capítulo describiremos las pruebas realizadas a la especificación del ISLANDER definida en el capítulo 5.

Estas pruebas fueron llevadas a cabo mediante las herramientas del EIDE AMELI y Dummy Agent, creando varios agentes de diferentes roles y haciéndoles realizar diferentes acciones en el entorno de ejecución AMELI. Aunque el proceso era lento (debido al enfoque de ”Control total sobre las acciones del agente” del Dummy Agent, el cual obligaba a realizar decisiones cuando solamente había realmente una opción disponible), las pruebas fueron más controlables e incluso dieron lugar a situaciones interesantes e inesperadas, las cuales descubrieron fallos no contemplados y ayudaron a mejorar el diseño (como, por ejemplo, descubrir que en un momento dado se podían realizar más acciones de las esperadas, lo que desembocaría en las restricciones descritas en la sección 5.2.4).

Primero en la sección 6.1 centraremos nuestra atención en la estructura performativa RegisterCentrePS, dado que es una parte semi-independiente del proceso general. Después, en la sección 6.2 describiremos las interacciones que ocurren en un proceso de traslado típico.

Cabe destacar que, aunque aquí se muestra uno solo de los posibles flujos de ejecución en cada sección, en el proceso de pruebas se comprobó cada una de las acciones posibles de cada rol en cada momento dado, asegurando por tanto el buen desempeño de la Institución Electrónica, excepto por casos que dependiesen de la buena voluntad de los agentes (por tanto, los casos de agentes mintiendo no son contemplaron en estas pruebas).

Dado que el proceso de traslado se describe con detalle en los capítulos 4 y 5, aquí nos centraremos en describir las imagenes obtenidas durante las pruebas, y relacionar dichas imágenes (y los mensajes que muestran) con el proceso que está teniendo lugar en la Institución Electrónica.

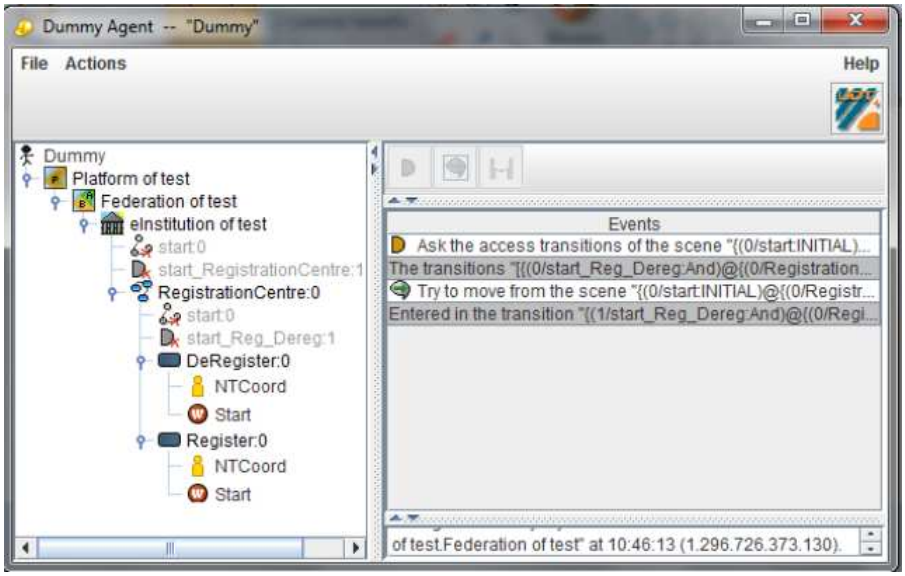

Figura 6.1: El estado del NTCoord antes de recibir una petición de registro

6.1. Centro de Registro

Esta sección describe un posible caso de estudio del Centro de Registro, en la cual un agente CandidateHR del Hospital Universitario de Móstoles entra a la Institución Electrónica para ser registrado en la mesa de enfermería. El agente NTCoord, habiendo creado la escena previamente, acepta su petición y le da la bienvenida a la mesa de enfermería.

6.1.1. Creación de la escena e inicio de la conversación

En las figuras 6.1 y 6.2 podemos ver cómo, tras haber entrado en la Institución Electrónica y en la estructura performativa **RegistrationCentrePS**, el NTCoord crea la escena **Register** y el CandidateHR entra en dicha escena. Como se ha observado en el diseño, el NTCoord está en dos escenas a la vez, Register y DeRegister.

6.1.2. Petición de registro y aceptación de dicha petición

Aquí, en las figuras 6.3 y 6.4, podemos ver, marcado en rojo, los mensajes que son consecuencia del CandidateHR enviando una petición de registro (mediante un mensaje *request*, el cual consiste en la función *askForRegistration()* con el contenido ”Hospital de Móstoles, C/ Rio Jucar S/N, 300, Pediatry, Operating Room, ER, Clinic Test on Appendix Inflamation”). El contenido del mensaje es, por tanto, la información relevante para el sistema sobre el hospital al que representa.

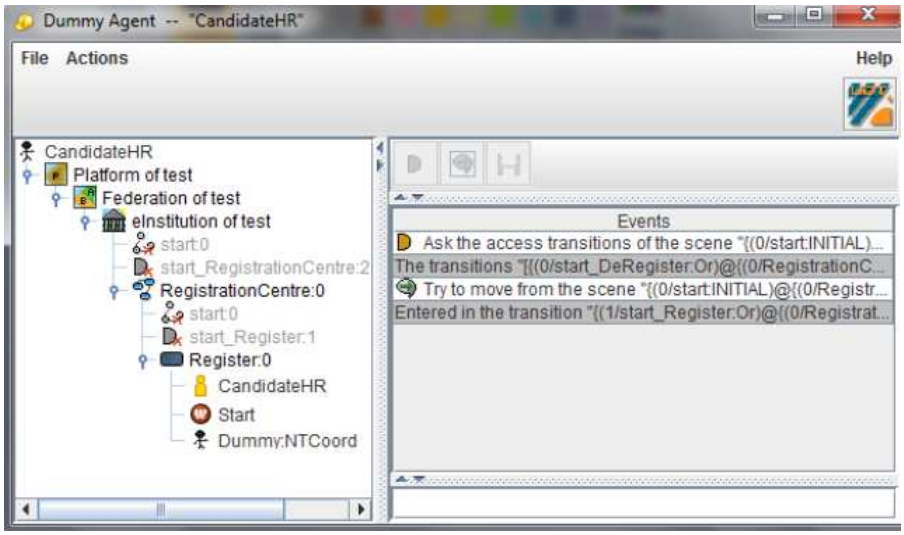

Figura 6.2: El estado del CandidateHR antes de emitir una petición de registro

El NTCoord, después, acepta la petición, enviando un mensaje *accept* consistente en la función *answer()*, que contiene un mensaje de bienvenida (este último mensaje está marcado en verde).

Tras ello, el CandidateHR sale de la institución, dado que su objetivo (registrar al hospital en la mesa de enfermería) ya se ha cumplido. El NTCoord, en cambio, permanece en la escena, esperando a que se realice la siguiente petición.

6.2. Patient Transfer

Esta sección describe un posible caso de estudio de traslado típico:

> Un CurrentHR del Hospital Universitario de Alcorcón entra en la Institución Electrónica para pedir un traslado al agente NTCoord, el coordinador de la mesa de enfermería. Dicho traslado es para el paciente Pepe Pérez, de 20 años, que ha sufrido una perforación en el apéndice. Dado el alto riesgo potencial de infección que puede sufrir, y las pocas camas disponibles en ese momento, se le ha marcado con un nivel de urgencia 1, es decir, debe ser trasladado cuanto antes, aunque no es vital, dado que, por el momento, se le puede estabilizar con los recursos disponibles en el hospital de Alcorcón mientras se decide sobre la petición de traslado.
>
> El coordinador de la mesa de enfermería nota que se necesita un servicio de urgencias en el hospital de destino, así que llama a participar en la mesa de enfermería a los dos hospitales que tienen el servicio requerido: el Hospital Universitario de Móstoles y el Hospital Universitario La Paz. Dado que

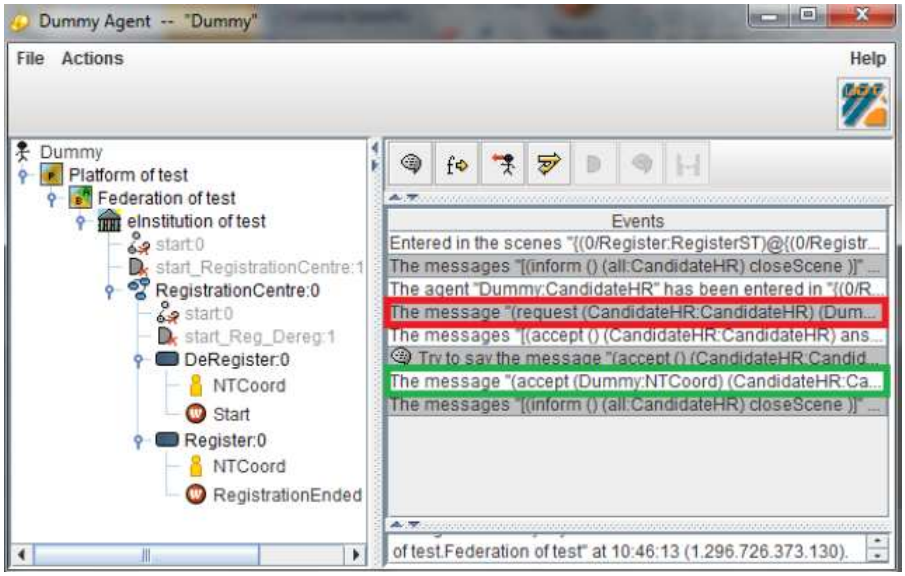

Figura 6.3: El estado del NTCoord después de procesar una petición de registro

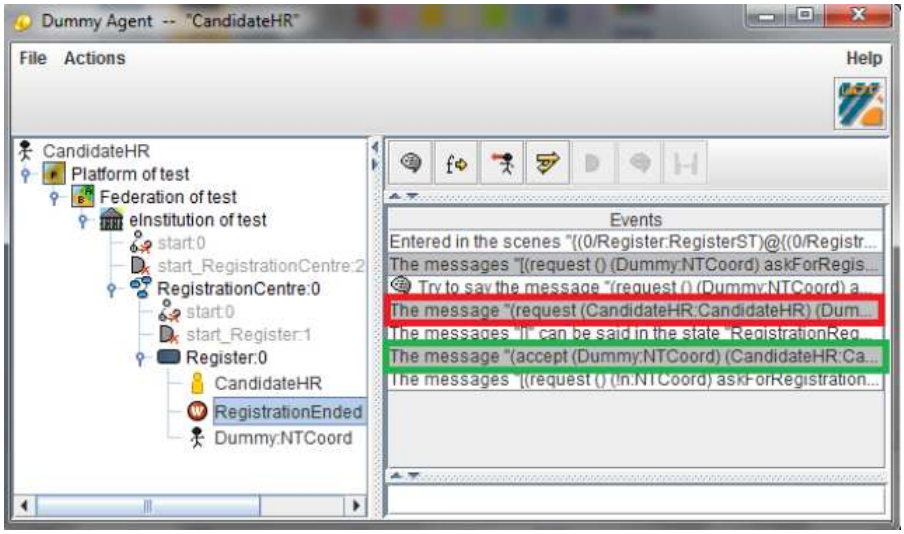

Figura 6.4: El estado del CandidateHR después de que su petición de registro haya sido procesada

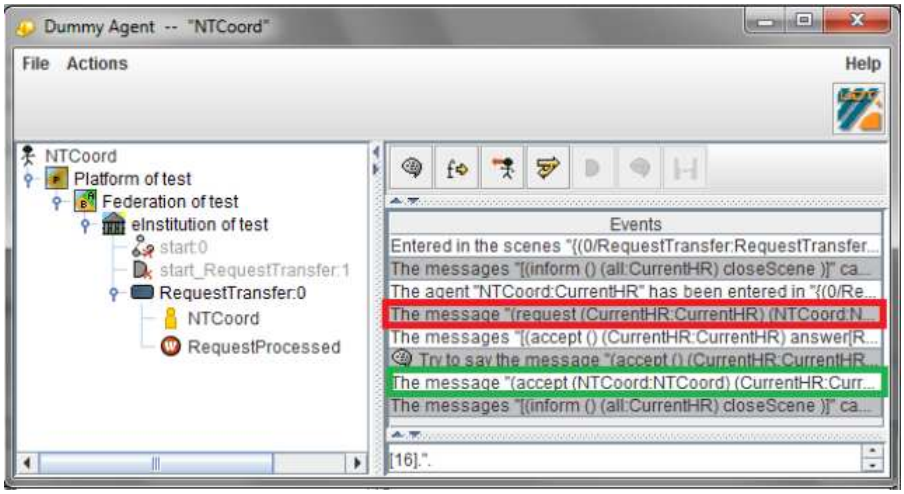

Figura 6.5: El estado de NTCoord tras procesar una petición de traslado

la Aplicación de Coordinación de Traslados Inter-Hospitalarios se encuentra aún en fase de pruebas, de momento no hay más hospitales registrados con ese servicio.

Tras ser invitados a la mesa de enfermería e informar del caso a los hospitales, el proceso de decisión comienza. Ambos hospitales son capaces de recibir al paciente en términos de capacidad y recursos disponibles, y al paciente le gustan ambos, dado que uno esta cerca de su lugar de residencia (el de Móstoles), mientras que el otro tiene buena reputación (La Paz). Sin embargo, el Hospital de Móstoles tiene un interés particular en recibir a este paciente, dado que está llevando a cabo un ensayo clínico relacionado con la inflamación del apéndice y sus consecuencias, así que se decide que el paciente será trasladado al hospital de Móstoles.

El NTCoord entonces pide el traslado del paciente al agente EmCentre del SUMMA112, el cual envía la planificación del traslado al Hospital Universitario de Alcorcón (informando de que la ambulancia llegará al hospital a las 14:30 para recoger al paciente), y al Hospital Universitario de Móstoles (notificando la llegada del paciente a las 15:30).

Tras eso, la situación de traslado contemplada aquí ha finalizado, dado que el proceso de traslado en sí no es relevante para esta aplicación.

6.2.1. Petición de traslado y aceptación

Aquí, en las figuras 6.5 y 6.6, podemos ver la interacción de mensajes entre el CurrentHR y el NTCoord en la escena **RequestTransfer**, en la estructura performativa **General**. El CurrentHR envía un mensaje *request* consistente en la función *requestTrans-*

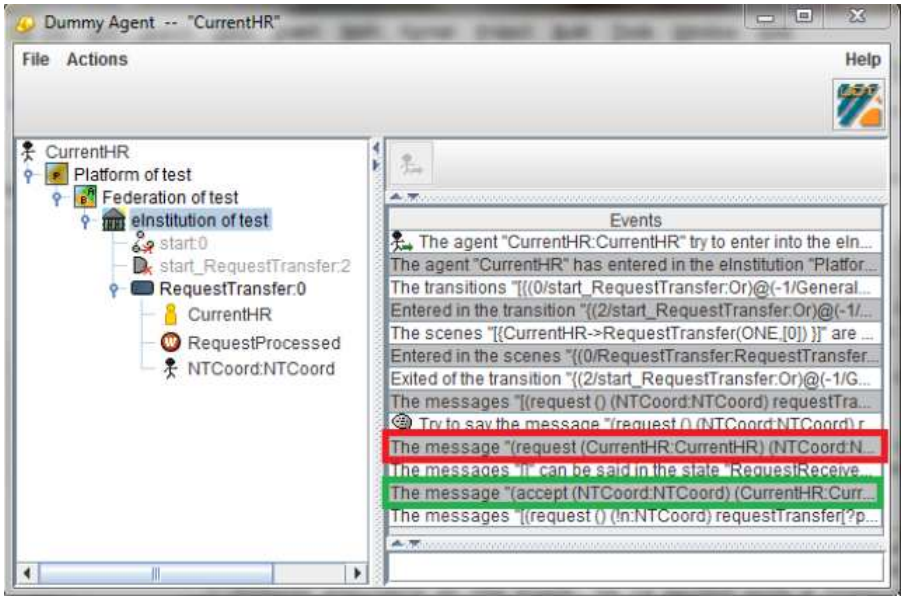

Figura 6.6: El estado de CurrentHR tras solicitar un traslado

fer() con el contenido *"Pepe Perez, Age 20, Appendix Perforation, Urgency Level 1, Not Enough beds"*, y el NTCoord responde con un mensaje *accept*, aceptando así la petición.

Tras eso, NTCoord hace un *stay and go* para poder iniciar el proceso de transferencia, mientras que el CurrentHR espera a que finalice la decisión de la mesa de enfermería.

6.2.2. Formación de la mesa de enfermería

Aquí, podemos ver como se forma la mesa de enfermería. En las figuras 6.7 y 6.8 vemos cómo el NTCoord invita a dos de los hospitales que están en la sala de espera (escena **WaitingRoom**) mediante un mensaje *inform* conteniendo la función *invite()* (dado que los dos hospitales se comportan exactamente igual, solo se muestra la imagen de uno de ellos). Es el NTCoord quien decide qué hospitales invitar a la mesa de enfermería, siendo selección competencia de la programación de la lógica de los agentes. Las invitaciones están marcadas en rojo, mientras que las respuestas de los hospitales aceptando la invitación (mediante un mensaje *accept* con la función *acceptInvitation()* como contenido) están marcadas en verde. Ambos mensajes no tienen contenido más allá de la función *invite()* y *acceptInvitation()*, dado que son solamente señales enviadas entre agentes.

Tras recibir las respuestas de los agentes, la mesa de enfermería es formada en la escena **FormNurseryTable** (figuras 6.9 y 6.10), ya dentro por tanto de la estructura performativa **DestinationHospitalDecision**. El NTCoord informa a todos los hospitales invitados del nuevo caso (mediante un mensaje *inform* cuyo contenido es una función *newCase()* con la información del paciente obtenida de su hospital actual, y la lista de los hospi-

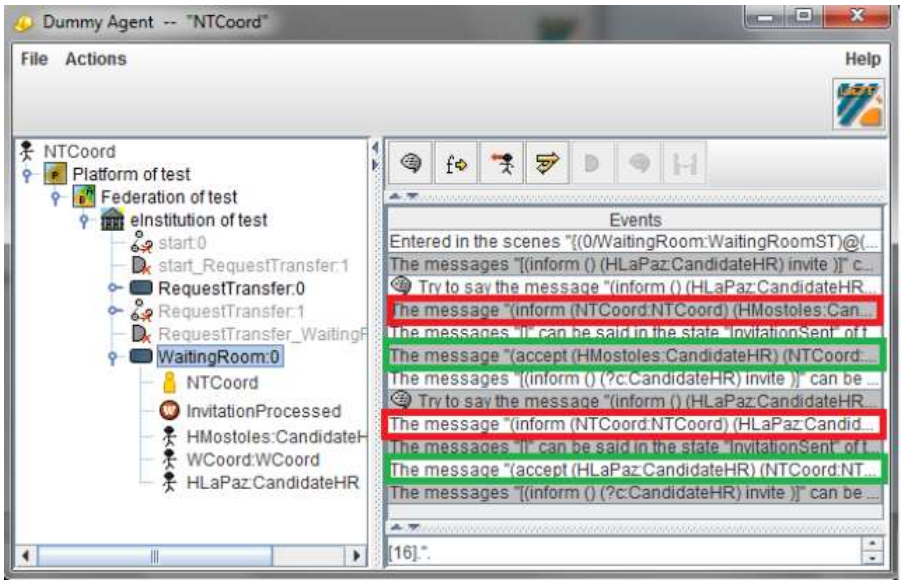

Figura 6.7: El estado de NTCoord tras invitar a los hospitales a la mesa de enfermería

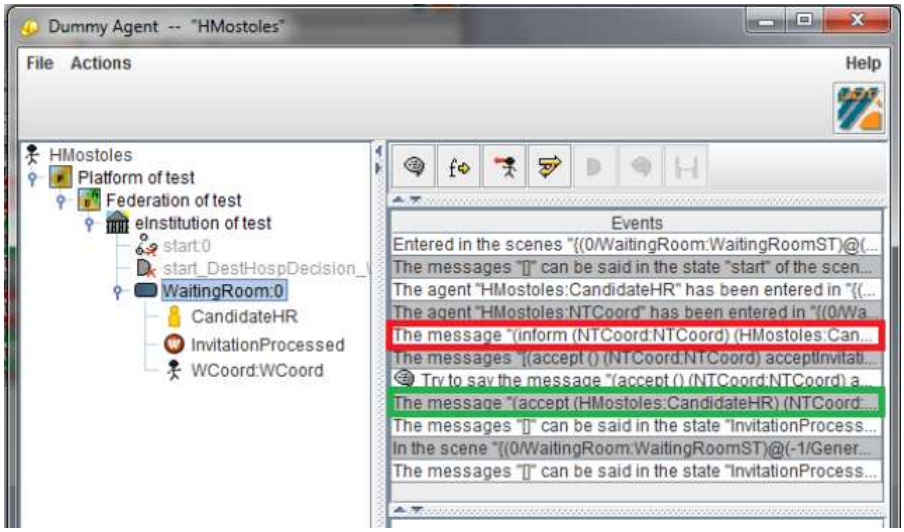

Figura 6.8: El estado de CandidateHR tras ser invitado a la mesa de enfermería

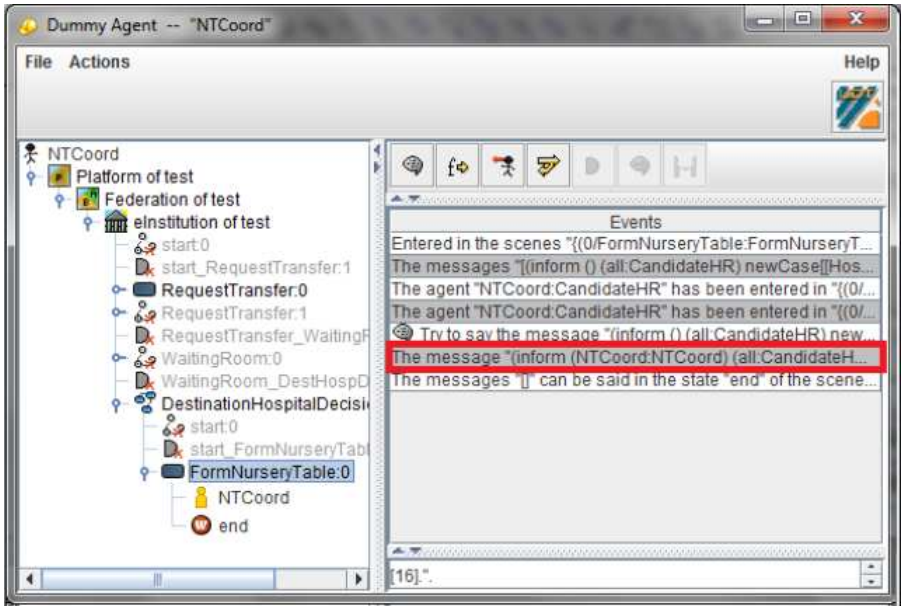

Figura 6.9: El estado de NTCoord tras informar a los hospitales del nuevo caso

tales con los servicios requeridos –e invitados a la mesa de enfermería–). Este mensaje (marcado en rojo) está destinado a todos los agentes CandidateHR de la escena.

6.2.3. Decidir el Hospital de destino

Aqui podemos ver cómo se toma la decisión de a qué hospital trasladar al paciente, en la estructura performativa **Decision**. En las figuras 6.11 y 6.12, correspondientes a la escena **AskHospital**, podemos ver al NTCoord preguntando a los hospitales sobre su capacidad disponible en ese momento, enviándoles un mensaje *ask* con la función *askCapacity()* como contenido (marcado en rojo), y a los hospitales informando de dicha capacidad mediante un mensaje *inform*, con la función *giveCapacity()* conteniendo la capacidad de ese hospital en cuestión (marcado en verde). El hospital de Móstoles responde que podría recibir a 40 pacientes más, mientras que La Paz puede recibir 30 más, por lo que el NTCoord les considera a ambos adecuados para recibir al paciente.

Tras preguntar a todos los hospitales, el NTCoord les informa de que esa etapa del proceso ha finalizado (mediante un mensaje *inform* con la función *finished()* –marcado en naranja–) y avanza al siguiente paso. Mientras, los hospitales esperan a la tercera etapa del proceso en la transición siguiente a esta escena.

En el siguiente paso, mostrado en las figuras 6.13 y 6.14 (correspondientes a la escena **AskPatientPreferences**) el NTCoord, teniendo aún a los dos hospitales como candidatos posibles a recibir al paciente, pregunta al agente *Patient* del paciente (mediante un mensaje *ask* con la función *askPreferences()*) sobre sus preferencias entre los dos hospitales que han superado la criba anterior. En este caso de estudio, recordemos, ambos hospitales son

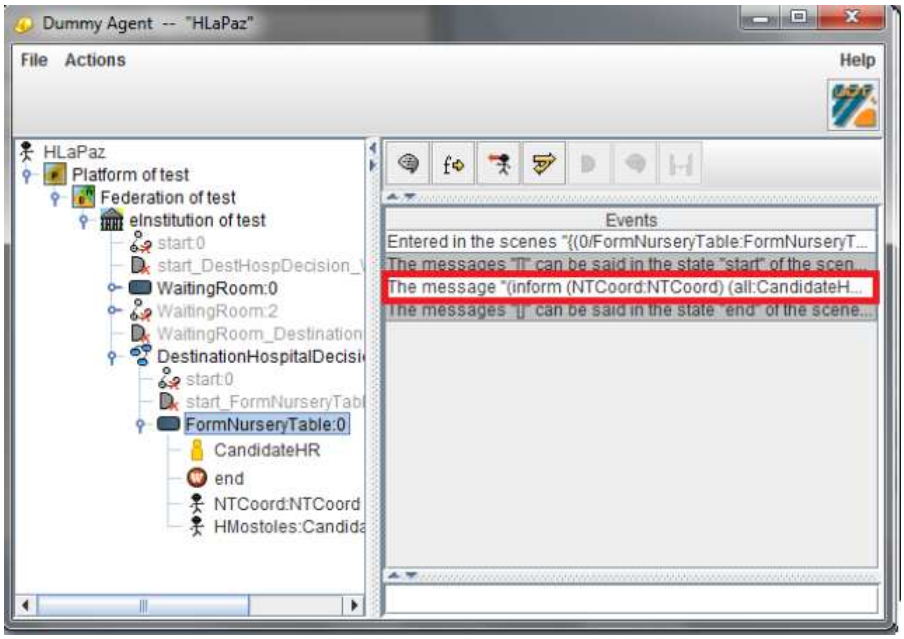

Figura 6.10: El estado de CandidateHR tras ser informado del nuevo caso

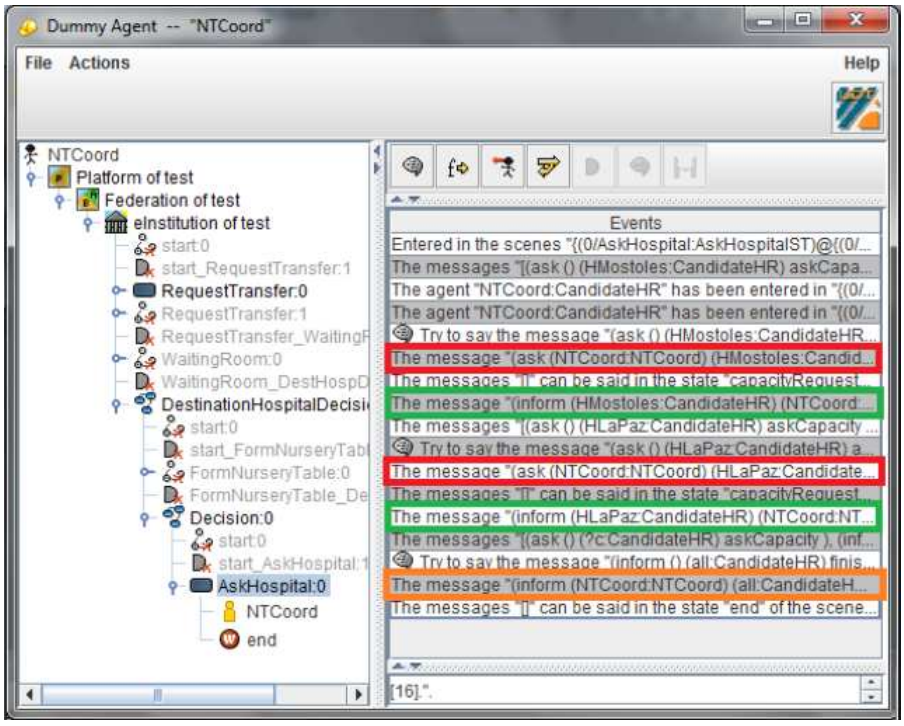

Figura 6.11: El estado de NTCoord después de preguntar la capacidad de cada hospital

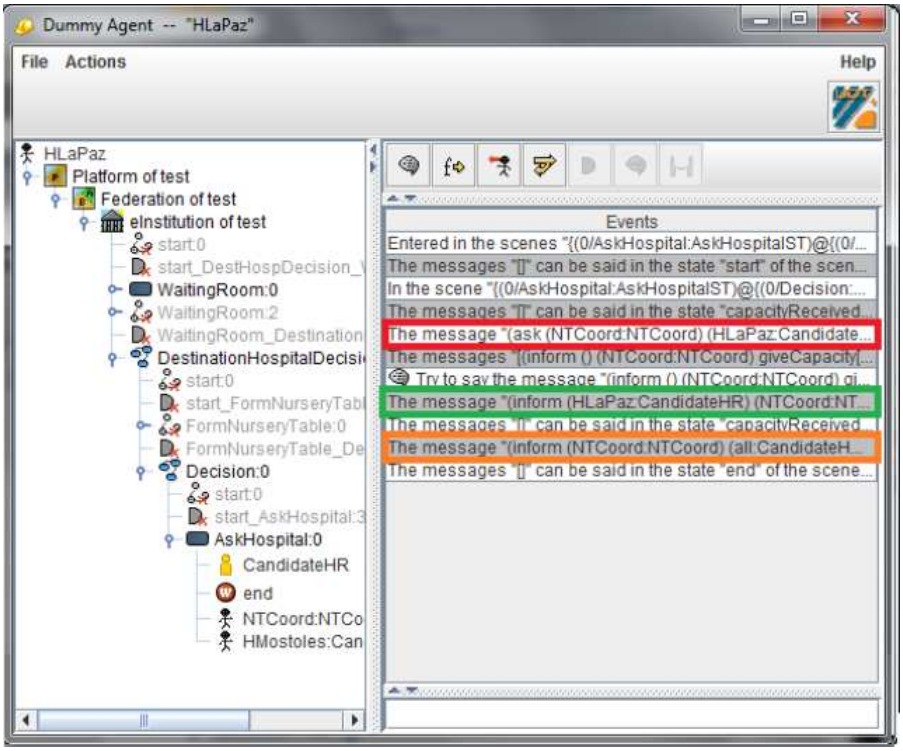

Figura 6.12: El estado del CandidateHR después de ser preguntado por su capacidad

del agrado del paciente, dado que el Hospital de Móstoles esta cerca de su casa, y el de La Paz tiene buena reputación, por lo que envía de vuelta un mensaje *inform* con la función *givePreferences()*, conteniendo ésta una lista que los incluye a los dos. El mensaje del NTCoord está marcado en rojo y la respuesta en verde. Tras ello, el agente Patient puede salir de la aplicación, dado que ya no es necesario.

Por último, dado que las preferencias del paciente no han ayudado a reducir el conjunto de posibles candidatos, en las figuras 6.15 y 6.16 podemos ver cómo el NTCoord pregunta a los hospitales (mediante un mensaje *ask* con la función *askHospitalPreferences()* –marcado en rojo–) sobre sus propias preferencias en cuanto a determinados perfiles de pacientes. El Hospital de Móstoles responde que prefiere pacientes con afecciones del apéndice, ya que está actualmente llevando a cabo un ensayo relacionado con la inflamación del apéndice y sus posteriores consecuencias. El hospital de La Paz, por el contrario, preferiría pacientes con osteoporosis, dado que está llevando a cabo un ensayo de un medicamente para paliar o prevenir sus efectos. Las respuestas de los hospitales (marcadas en verde) son mensajes *inform* con la función *giveHospitalPreferences()*, la cual contiene una lista de perfiles de paciente.

Al igual que cuando se preguntan las capacidades disponibles, el NTCoord, una vez ha finalizado, informa a todos los hospitales de que deben avanzar a la siguiente etapa mediante un mensaje *inform* con la función *finished()*, el cual está marcado en naranja. Tras esto, finaliza la estructura performativa **Decision**.

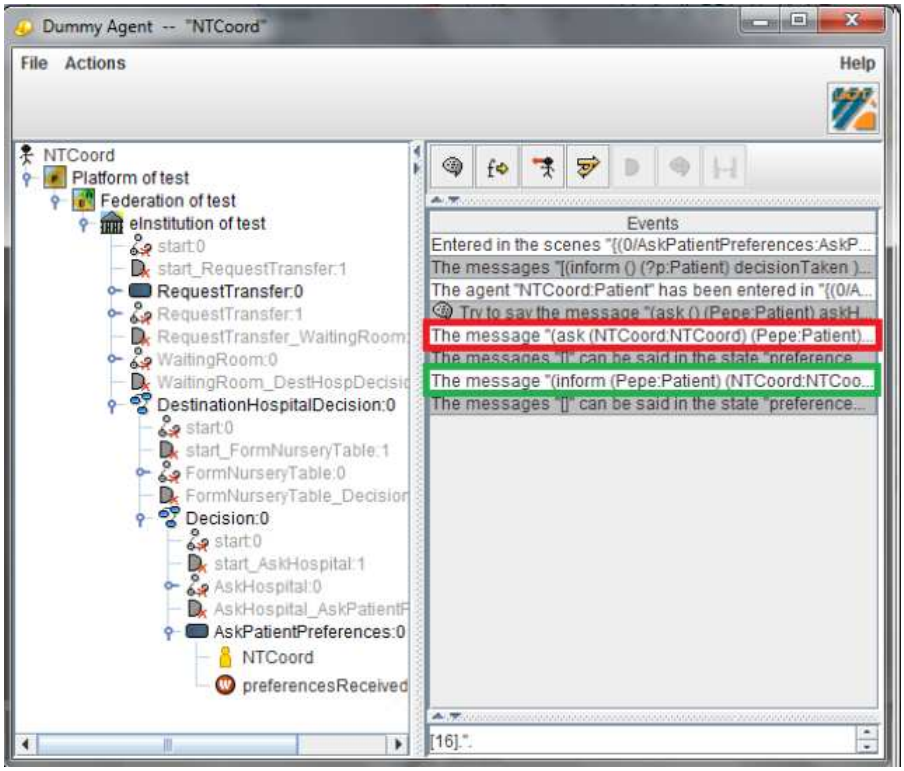

Figura 6.13: El estado de NTCoord después de preguntar las preferencias del paciente

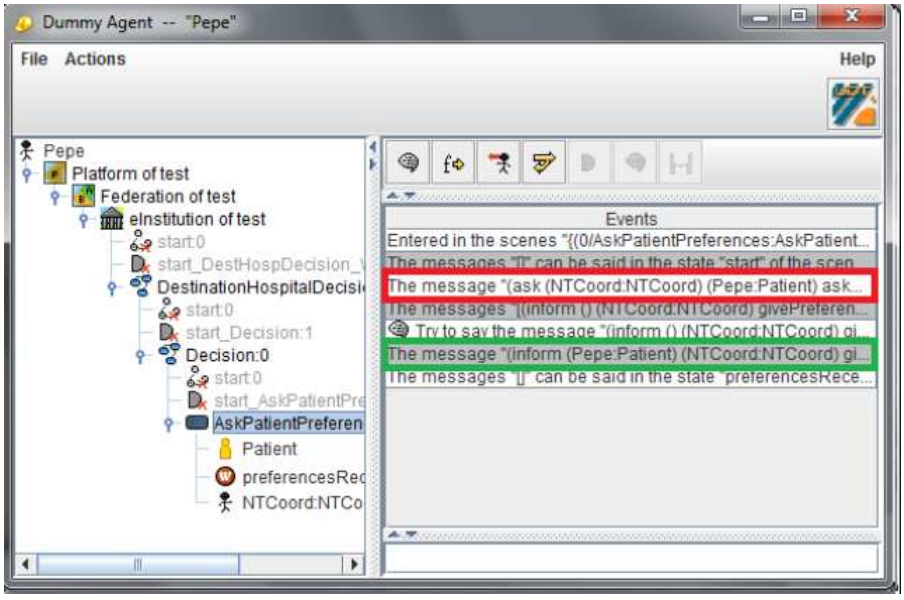

Figura 6.14: El estado del paciente después de que se le pregunten sus preferencias

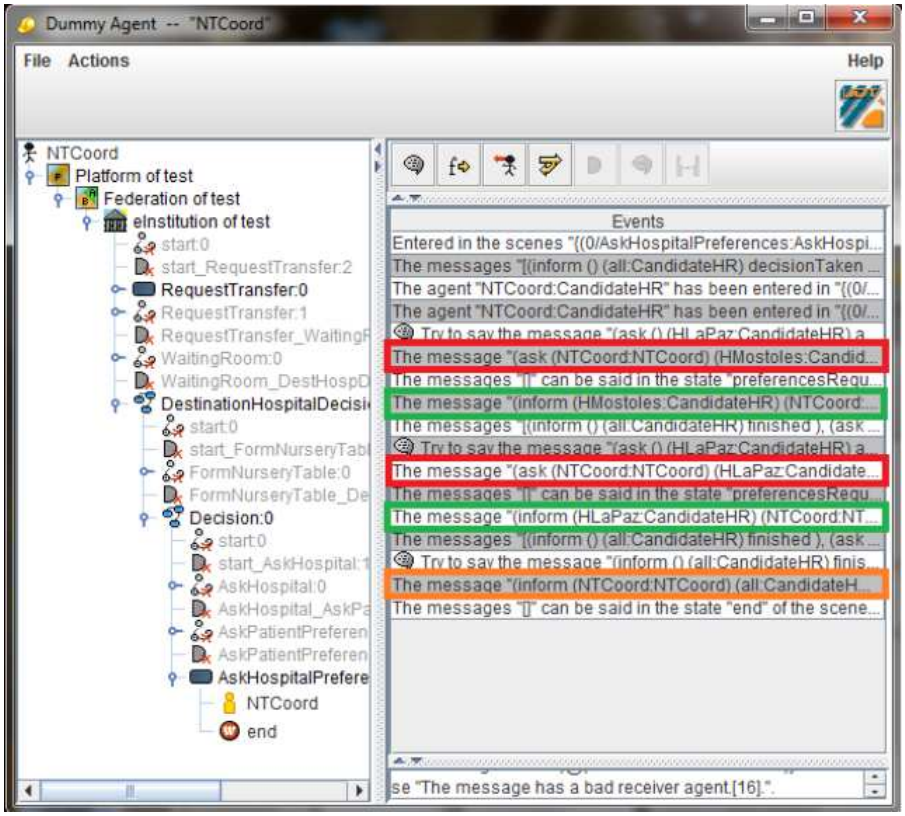

Figura 6.15: El estado de NTCoord después de preguntar las preferencias de los hospitales

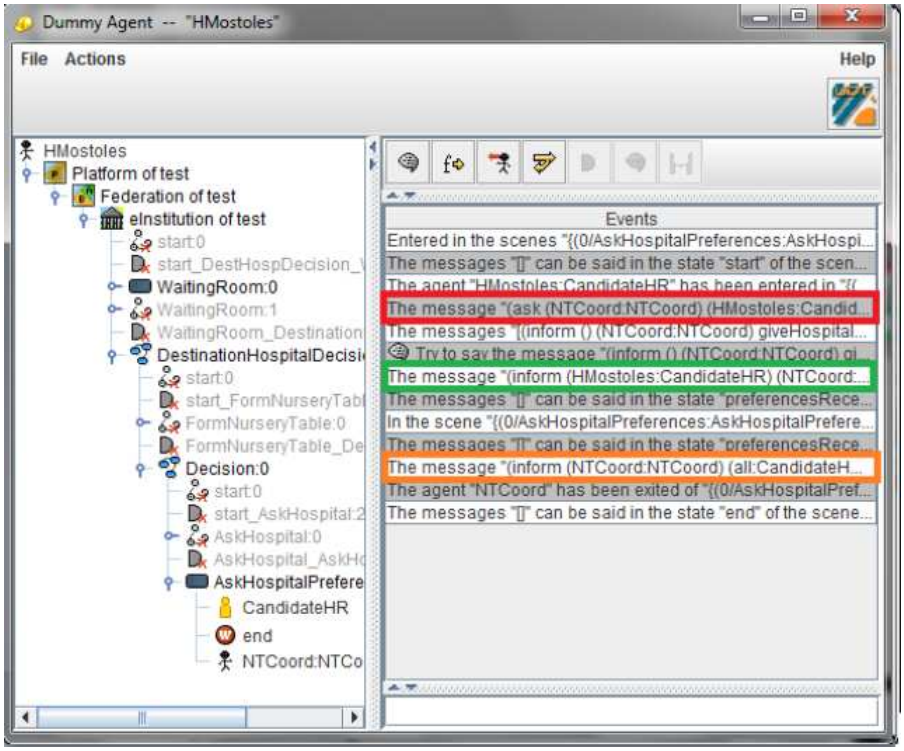

Figura 6.16: El estado del CandidateHR después de ser preguntado por sus preferencias

6.2.4. Disolución de la mesa de enfermería

Aquí, podemos ver cómo la mesa de enfermería es disuelta (escena **DissolveNursery-Table**). En las figuras 6.17 y 6.18, se observa cómo el NTCood informa tanto a HMostoles como a HLaPaz de la decisión tomada, mediante el envío de dos mensajes *inform*, uno de ellos con la función *notSelected()* como contenido, y el otro con la función *notSelected()* como contenido. Estos mensajes están marcados en rojo, y las respuestas de los hospitales (mensajes *inform* con la función *acceptDecision()* como contenido) están marcadas en verde.

Tras esto, el NTCoord notifica a todos los hospitales que ha terminado de informar de la selección, estando preparados por tanto para dejar la escena, nuevamente mediante un mensaje *inform* con la función *finished()*. Este mensaje está marcado en naranja. Tras esto, finaliza la estructura performativa **DestinationHospitalDecision**.

En este caso, HMostoles es el elegido por la mesa de enfermería, por lo que avanza a la siguiente escena. HLaPaz, por el contrario, puede salir de la institución electrónica y de la aplicación (al menos ese *alteroide*), dado que su papel ya ha finalizado en este proceso de traslado concreto. Sin embargo, recordemos, el otro *alteroide* está aún en disposición de atender peticiones en la escena **WaitingRoom**.

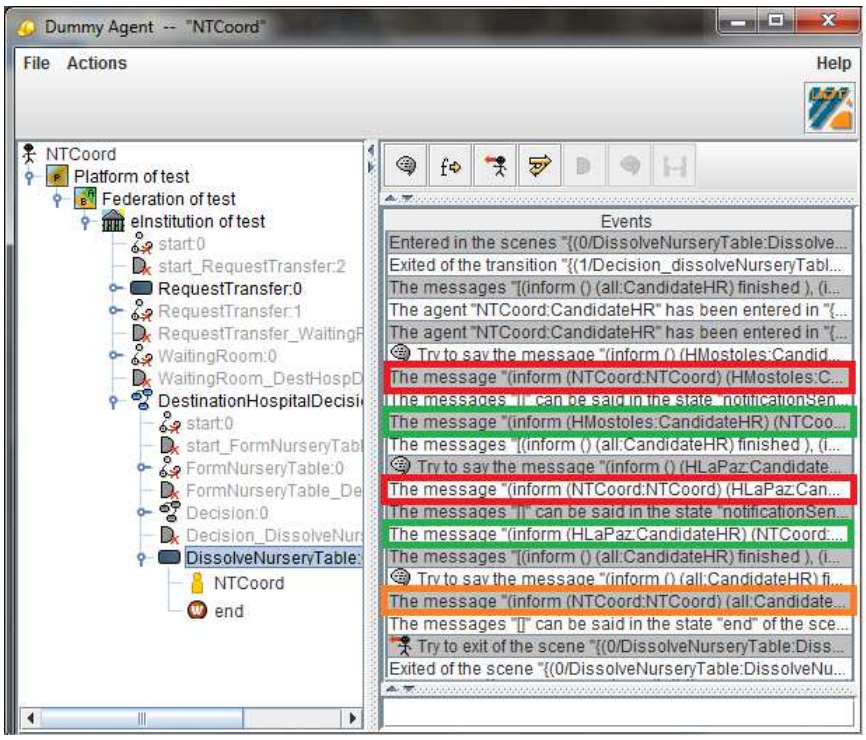

Figura 6.17: El estado de NTCoord después de disolver la mesa de enfermería

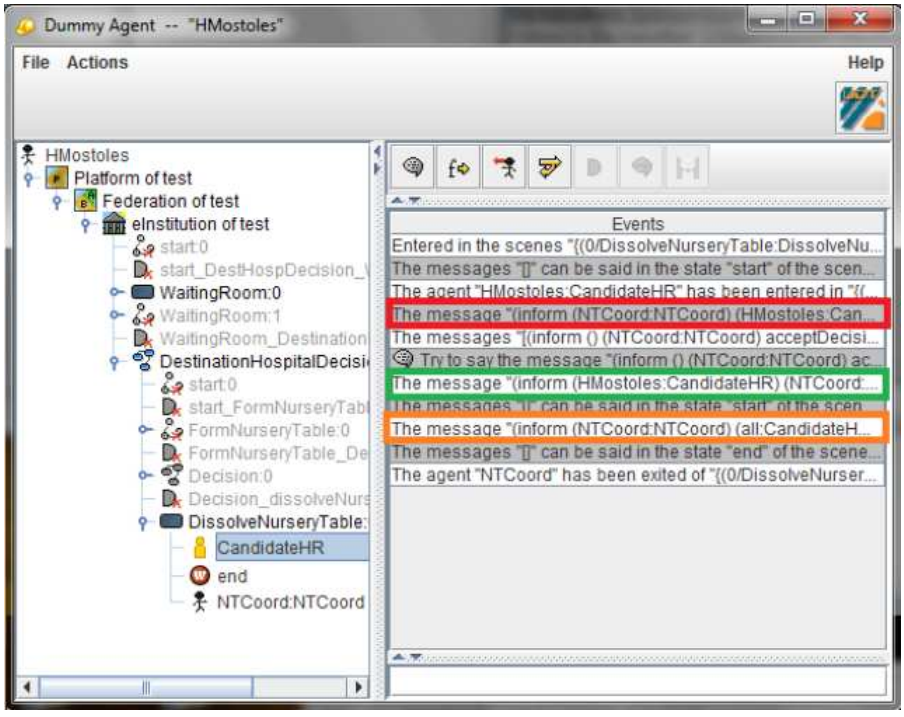

Figura 6.18: El estado del CandidateHR después de que la mesa de enfermería sea disuelta

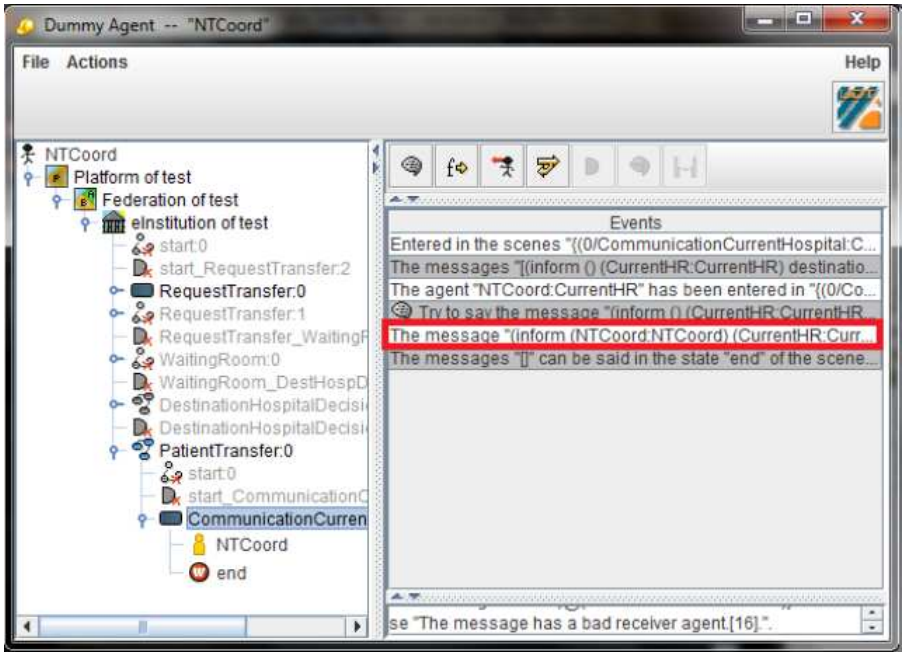

Figura 6.19: El estado de NTCoord después de informar al CurrentHR de la decisión

6.2.5. Traslado del paciente

Aquí podemos ver cómo, una vez tomada la decisión, el traslado al destino del paciente es planificado. Esto ocurre en la estructura performativa **PatientTransfer**.

En las figuras 6.19 y 6.20, vemos cómo el NTCoord, en la escena **CommunicationCurrentHospital**, informa al hospital del paciente de la decisión tomada por la mesa de enfermería (mediante un mensaje *inform* con la función *destinationDecision()*, conteniendo el nombre del hospital de destino). Este mensaje está marcado en rojo.

Tras ello, en las figuras 6.21 y 6.24, correspondientes a la escena **CommunicationEmCentre**, vemos al NTCood informando al EmCentre de que tiene un nuevo traslado que debe realizar, mediante un mensaje *inform* consistente en la función *transfer()*. Esta función contendrá la información del paciente (”*Pepe Perez, Edad 20, Perforación del Apéndice, Prioridad 1*”), la del hospital de origen (”*Hospital Universitario de Alcorcon, C/ Budapest, 1, 200*”), y la del hospital de destino (”*Hospital Unversitario de Móstoles, C/ Rio Jucar S/N, 300*”), para que el EmCentre pueda hacer los preparativos que sean necesarios. Tras eso, el NTCoord (o al menos, ese *alteroide*) sale de la Institución Electrónica, dado que ya no tiene más tareas que realizar. Esta información está marcada en rojo.

Mientras, las figuras 6.22 y 6.23, correspondientes a la escena **InformationTransfer** muestran como el hospital del paciente transfiere su información al hospital de destino mediante un mensaje *inform*, consistente en una función *patientInfoTransfer()* con la información del paciente (este mensaje aparece marcado en rojo).

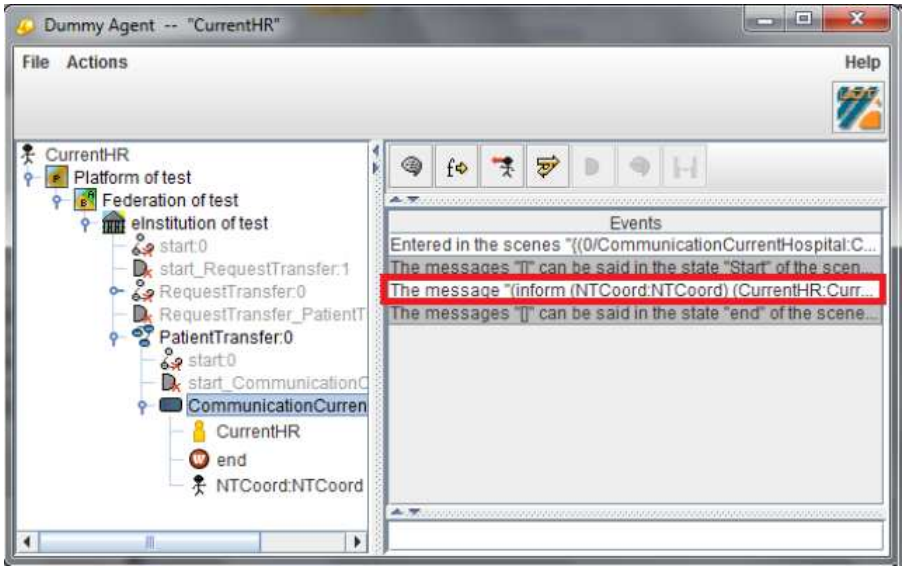

Figura 6.20: El estado del CurrentHR después de ser informado de la decisión tomada

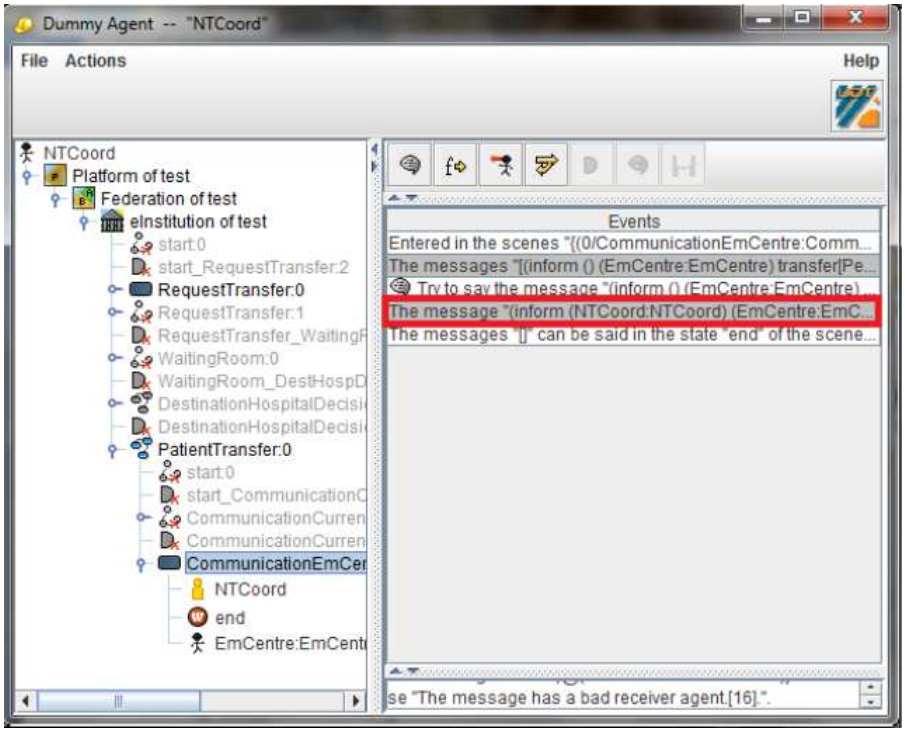

Figura 6.21: El estado de NTCoord después de informar al EmCentre del nuevo traslado

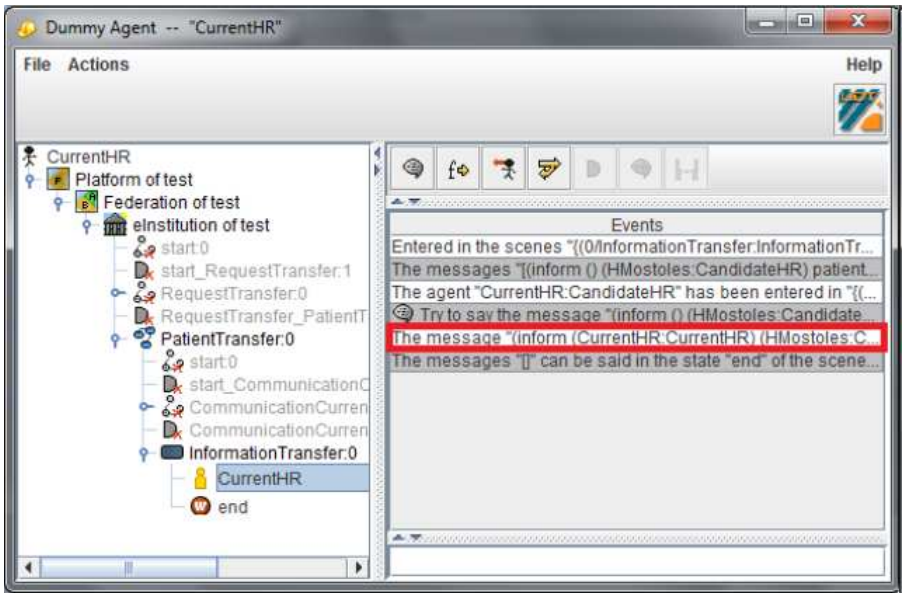

Figura 6.22: El estado del CurrentHR después de enviar la información del paciente

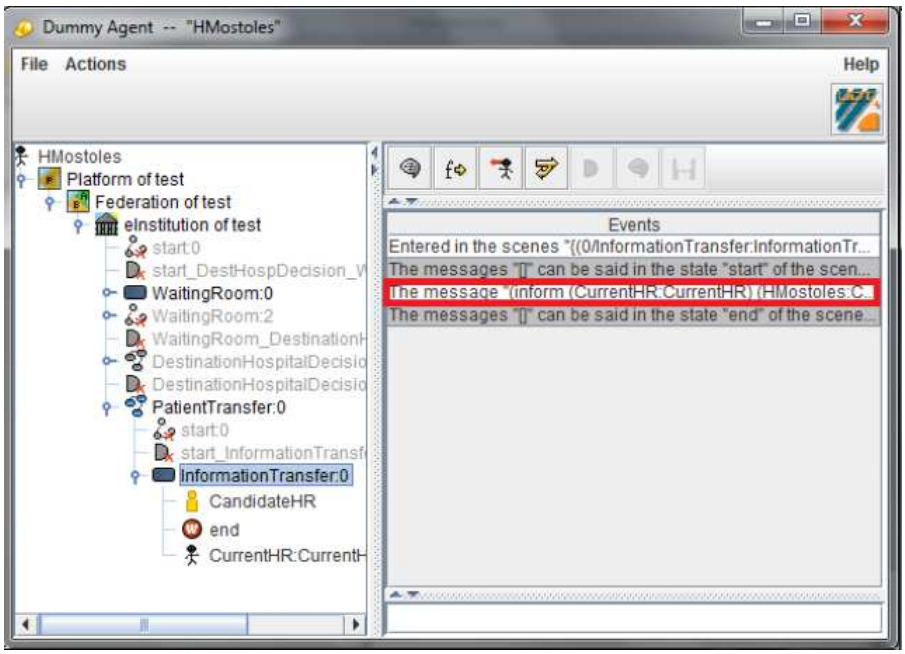

Figura 6.23: El estado del CandidateHR después de recibir la información del paciente

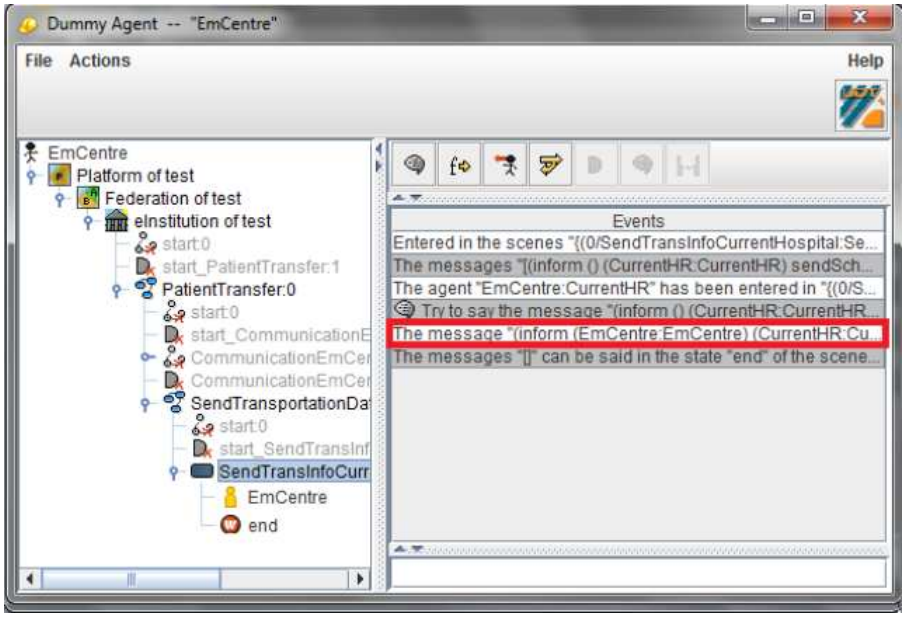

Figura 6.24: El estado del EmCentre tras informar de la planificación del traslado al CurrentHR

Por último, el EmCentre informa a los dos hospitales involucrados de la planificación horaria del traslado en la estructura performativa **SendTransportationData**. Así, en las figuras 6.24 y 6.25 (escena **SendTransInfoCurrentHospital**) vemos cómo el CurrentHR recibe la hora de llegada de la ambulancia que recogerá al paciente (un mensaje *inform* consistente en una función *sendSchedule()* conteniendo la hora *14:30*), y en las figuras 6.26 y 6.27, correspondientes a la escena **SendTransInfoDestinationHospital** vemos al hospital de destino (HMostoles en nuestro caso) recibiendo la hora de llegada de la ambulancia transportando al paciente (un mensaje *inform* consistente en una función *sendSchedule()* conteniendo la hora 15:30). Ambos mensajes están marcados en rojo.

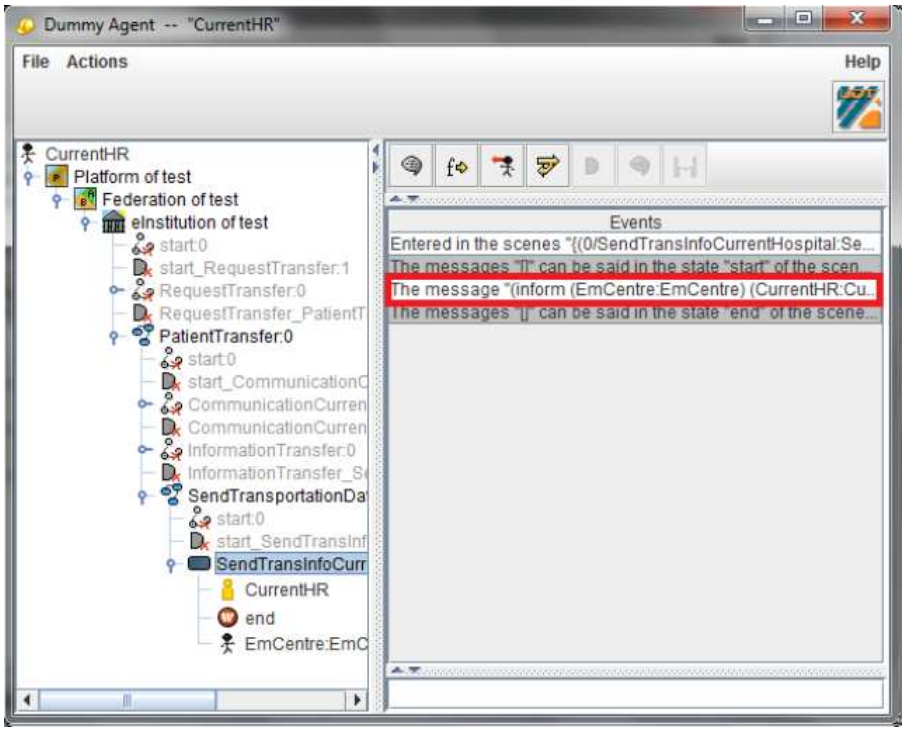

Figura 6.25: El estado del CurrentHR tras ser informado de la hora de llegada de la ambulancia

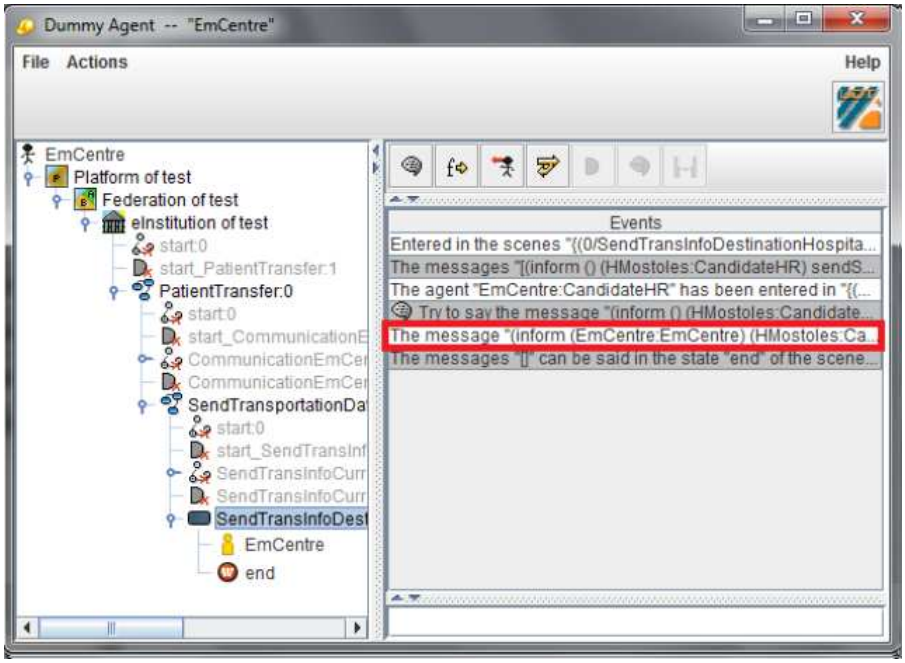

Figura 6.26: El estado del EmCentre tras informar de la hora de llegada del paciente al CandidateHR

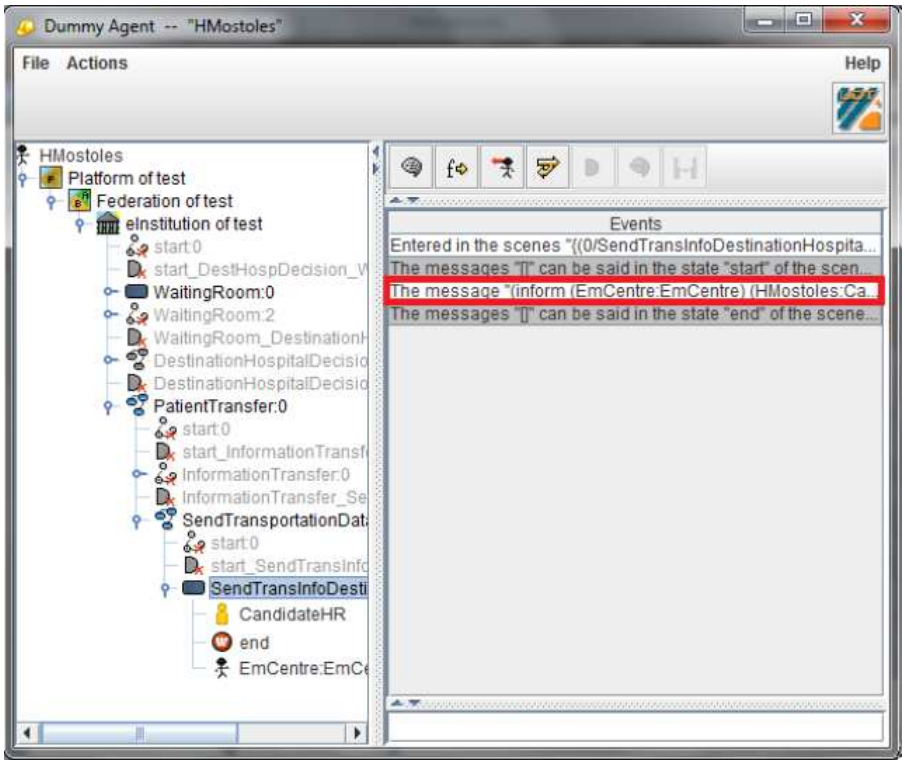

Figura 6.27: El estado del CandidateHR tras ser informado de la hora de llegada del paciente

Capítulo 7

Conclusiones y líneas futuras de trabajo

7.1. Logros principales

En este trabajo se han establecido los siguientes logros:

- Haber determinado con precisión los requisitos, roles que intervienen, y el comportamiento general de una estructura organizativa utilizada como base para una aplicación que solucione el problema de la coordinación de traslados inter-hospitalarios, diseñada mediante el paradigma de las Instituciones Electrónicas.
- Haber descrito y diseñado la especificación de la Institución Electrónica correspondiente a dicha estructura, usando para ello la herramienta gráfica ISLANDER perteneciente al entorno de desarrollo EIDE.
- Haber probado con éxito la validez funcional de dicho diseño, gracias al AMELI y al Dummy Agent (pertenecientes también al EIDE), demostrando su correcto funcionamiento.

7.2. Líneas futuras de investigación relacionadas

Existen dos líneas de investigación para continuar el trabajo aquí presentado:

- Concentrarse en el problema de encontrar un acuerdo entre los hospitales involucrados en un traslado para decidir a cuál de ellos es trasladado el paciente en cuestión. Se estudiará la aplicación de diferentes patrones de acuerdo (votación, subasta, consenso, etc.) para poder optimizar el proceso de decisión en base a medidas de calidad como:

    1. *Distribución de la ocupación de camas:* Para poder asegurar un servicio de calidad, es deseable tener una ocupación similar en todos los hospitales.

2. *Tiempo de traslado:* Aqui se consideraría el intervalo de tiempo desde que se realiza una petición hasta que el paciente llega al hospital de destino. Dicho intervalo a su vez podría dividirse en dos medidas que describen, por un lado, la eficiencia de la mesa de enfermería y, por otra, la calidad del proceso de traslado en sí.

   La primera de ellas sería el intervalo entre una petición de traslado y la notificación de la decisión de destino alcanzada en la mesa de enfermería, y la segunda, el intervalo desde que se toma la decisión hasta que el paciente llega a su destino. Obviamente, en ambos casos, es deseable que dichos tiempos sean lo menores posible.

3. *Ratio de asignaciones no colaborativas:* Esta medida describe el porcentaje de asignaciones en las cuales ninguno de los hospitales que ofrecían el tratamiento adecuado quiere aceptar al paciente, bien por no tener capacidad suficiente, o por no estar interesados en el caso (debido a su dificultad, a la edad del paciente, etc.). Dado que todos los casos deben ser resueltos, se transferiría el paciente a uno de dichos hospitales.

   También se podrían considerar casos en los que los hospitales esconden su capacidad real para evitar asignaciones de pacietnes no deseados, asi como aquellos casos en los que la decisión es que no haya traslado (es decir, el hospital del paciente puede prestar el tratamiento requerido, pero pide un traslado por otra razón).

   Esta medida describe la calidad del sistema desde la perspectiva de cómo son satisfechas las preferencias individuales de los hospitales. También presenta cómo el sistema es capaz de tratar con posibles casos de hospitales que proporcionen datos incompletos o erróneos sobre sus capacidades para impedir ciertas asignaciones. Se prevee el análisis e implementación de diferentes mecanismos de decisión para prevenir dichos problemas.

- La implementación de agentes reales con la ayuda del programa ABuilder del entorno EIDE, proporcionando así una forma de desarrollar por completo el sistema entero, pudiéndolo probar con agentes reales.

  Esta implementación permitirá comprobar el comportamiento del sistema en condiciones reales y realizar experimentos con cantidades mayores de hospitales y pacientes que las expuestas en las pruebas (solamente dos y uno, respectivamente), así como profundizar en los siguientes puntos de interés, los cuales ha sido imposible desarrollar en la especificación ISLANDER del presente trabajo, al ser cuestiones puramente de implementación:

  – El modo de implementar las decisiones de los agentes sin sacrificar la modularidad del código, a fin de que éste sea lo más accesible posible. Una de las opciones, por ejemplo, sería crear una clase "*Decisiones*" por cada tipo de agente, y que sea llamada desde las distintas escenas que requieran una deci-

sión. De este modo, se podrán probar diferentes métodos de decisión en cada una de ellas.

- El mecanismo de selección de una instancia de escena u otra, a fin de que un hospital solamente participe en los procesos de decisión y traslado que le corresponden.
- La selección de los hospitales que van a tomar parte en la mesa de enfermería.
- La forma de relacionar la información de los hospitales registrados con los agentes que se encuentran activos en el sistema.
- El desarrollo del mecanismo que utilizará el NTCoord para llevar a cabo varios traslados simultáneos, y llevar una traza de cada uno (en qué etapa se encuentra, los hospitales involucrados, etc.).
- Relacionado con el punto anterior, el comportamiento del sistema completo en situaciones de alto tráfico de pacientes.
- El desarrollo de una interfaz gráfica que permita a un agente humano monitorizar el comportamiento del agente.

Dichas líneas de investigación podrían permitir, en un futuro, que esta aplicación se implantase en el sistema sanitario, redundando en una mejor calidad del servicio, y una mayor calidad de vida para los pacientes.

Bibliografía

[BFD+] Holger Billhardt, Moser Fagundes, José Javier Durán, Roberto Centeno, and Diego Fradejas Muñoz. *AT Project Deliverable 8.3.1 – mHealth Requirements Analysis and Design*.

[CFB+09] Roberto Centeno, Moser Fagundes, Holger Billhardt, Sascha Ossowski, Juan Manuel Corchado, Vicente Julian, and Alberto Fernández. An organisation-based multiagent system for medical emergency assistance. In *International Work-Conference on Artificial Neural Networks*, page to appear. Springer-Verlag, 2009.

[CFBO09] Roberto Centeno, Moser Fagundes, Holger Billhardt, and Sascha Ossowski. Supporting medical emergencies by mas. In *Distributed Systems and Artificial Intelligence Applications (KES AMSTA 2009)*, page to appear. Springer-Verlag, 2009.

[CFO05] César Cáceres, Alberto Fernández, and Sascha Ossowski. Cascom - context-aware health-care service coordination in mobile computing environments. *ERCIM News*, (60):77–78, 2005.

[Dev] EIDE Developers. *Documentación de ayuda del entorno EIDE* (disponible con la descarga del mismo).

[dlpwdpA] Sección ”Project” de la página web del proyecto AT. http://www.agreement-technologies.org/project.

[Fer] Jacques Ferber. *Multi-Agent Systems. An Introduction to Distributed Artificial Intelligence*. Addison Wesley.

[JAS05] Pablo Noriega Juan Rodriguez-Aguilar Josep Arcos, Marc Esteva and Carles Sierra. Engineering open environments with electronic institutions. *Engineering Applications of Artificial Intelligence*, (18):191–204, 2005.

[LIB08] Beatriz López, Bianca Innocenti, and Dídac Busquets. A multiagent system to support ambulance coordination of urgent medical transportation. *IEEE Intelligent Systems*, 23(5):50 – 57, 2008.

[Mas05] Ana Mas. *Agentes software y sistemas multiagente. Conceptos, arquitecturas y aplicaciones*. Pearson Educacion, S.A., 2005.

[mdedlpwdS] Sección ”medios/mesa de enfermería”de la página web del SUMMA112.

[wdeE] Página web del entorno EIDE. http://e-institutions.iiia.csic.es/.

[wdpI] Página web del programa INGENIO. ”http://www.ingenio2010.es/contenido.asp”.

[WJ95] Michael Wooldridge and Nicholas R. Jennings. Intelligent agents: theory and practice. *The Knowledge Engineering Review*, 10(2):115–152, 1995.