# MAS-CommonKADS para el desarrollo de un Sistema Multiagente de Información de Recomendación de Rutas de Transporte: SINRUT.

## *MAS- CommonKADS in the Development of a Multi-Agent Information System for Transportation Routes Recommendation: SINRUT.*

María Isabel Marín Morales*
Luisa Fernanda Correa Ríos**
Luis Alejandro Aguilar Londoño***

Tipo de artículo: Resultado de Investigación.

Recibido: 10 de agosto, 2014
Aceptado: 15 de octubre, 2014

**Resumen**

Entre los principales inconvenientes para el uso del transporte público en ciudades como Medellín, está el desconocimiento de las rutas de transporte, así como de los recorridos que estas realizan. Esta necesidad motivó el desarrollo de este trabajo que presenta los aspectos clave de la implementación de un sistema multiagente (SMA) para la recomendación de rutas de transporte en el traslado de un lugar a otro en la ciudad de Medellín. Para el desarrollo de este sistema se utilizaron JADE y Protégé sobre la plataforma JAVA, que se integraron con un módulo de visualización desarrollado en el Framework.NET, extendiendo la funcionalidad del Sistema de Información Geográfica (SIG) de código abierto MapWindow. Las diferentes fases del desarrollo del SMA se llevaron a cabo utilizando la metodología MAS-CommonKADS que será evidenciada en este trabajo con el fin de que sirva de apoyo para el futuro uso de la misma en desarrollos similares.

**Palabras clave:** sistemas multiagentes, algoritmo A*, sistemas de recomendación, MAS-CommonKADS, MapWindow.

**Abstract**

The main drawbacks to the use of public transport in cities like Medellín, is the lack of transportation routes and the routes that they perform. This need motivated the development of this paper presents the key aspects of the implementation of a multi-agent system (MAS) for recommending transport routes in moving from one place to another in the city of Medellin. For the development of the JADE system and Protégé were used on the Java platform, which is integrated with a visualization module developed in the .NET Framework by extending the functionality of Geographic Information System (GIS) open source MapWindow . The different phases of the development of SMA were performed using MAS- CommonKADS methodology that will be evidenced in this work in order to provide back to the future use of the same in similar developments.

**Keywords:** multiagent systems, A* algorithm, recommendation systems, MAS-CommonKADS, MapWindow.

* Magíster en Ingeniería de Sistemas. Docente Asistente, Universidad de San Buenaventura. maria.marin@usbmed.edu.co

** Ingeniera de Sistemas e Informática. Analista de Soluciones, BTG Pactual Colombia. lfcorre0@unalmed.edu.co

*** Especialista en Sistemas, Ingeniero de Investigación. Universidad Nacional de Colombia. laaguilal@unalmed.edu.co

## Introducción

Como en muchas otras metrópolis, en la ciudad de Medellín diariamente las personas propias y foráneas se ven enfrentadas al dilema de decidir la mejor ruta para dirigirse a un lugar deseado, pensando en la optimización de recursos limitados como el dinero y el tiempo disponible para transportarse. La información empírica de cada ciudadano puede no ser suficiente para llevar a tomar la mejor decisión, teniendo en cuenta además que los datos conocidos previamente pueden tornarse obsoletos por el ingreso, eliminación o cambio en las rutas de transporte existentes.

Evaluando el abanico de posibilidades que los avances informáticos proporcionan, se consideró que este problema podría ser abordado por medio de un sistema de recomendación de rutas (Arthur et al., 2014). Estos sistemas normalmente realizan el filtrado con base en el perfil del usuario, que puede ser dado explícitamente por el usuario o inferido utilizando métodos adaptativos a partir de la interacción del usuario con el sistema (Gnjatović & Delić, 2014). Es así como se motiva el desarrollo de un Sistema Multiagente para la Recomendación de Rutas de Transporte para la ciudad de Medellín (SINRUT).

Se determinó la utilización del paradigma de la programación orientada a agentes por la necesidad de distribución y aprendizaje en el desarrollo del programa. Este sistema multiagente se desarrolla a la luz de la metodología MAS-CommonKADS (Iglesias & Garijo, 2005) y se implementa sobre la plataforma JAVA (Mohapatra et al., 2006) utilizando las interfaces de programación de aplicaciones de JADE (Vila et al., 2007) y Protege (Gennari et al., 2003), integrándose con un módulo de visualización desarrollado en el Framework.NET (Wenjie et al., 2012), a través del Sistema de Información Geográfica (SIG) de código abierto MapWindow (Aburizaiza & Ames, 2006) que permite extender su funcionalidad con el desarrollo de plugins.

En el campo de los sistemas de recomendación de rutas utilizando agentes se pueden identificar dos maneras de concluir recomendaciones como tales: a) recomendación basada en contenidos y b) recomendación colaborativa; la primera utiliza fuentes inertes para ello, mientras que la segunda acude a otros seres humanos para realizar las recomendaciones. Para complementar este procedimiento, existen algunos otros modelos de recomendación entre los cuales se pueden citar los modelos matemáticos, las redes neuronales y los sistemas difusos. Adicionalmente, un sistema de recomendación puede tener algún procedimiento de retroalimentación.

En el ámbito mundial existen diversos trabajos que pueden aproximarse a la idea y algunos de estos se abordaron desde la planificación, como es el caso de Castillo et al., (2008), el cual no se limita a la recomendación de rutas, sino que incluye también la recomendación de sitios y actividades utilizando un método de razonamiento basado en casos. Esta última característica es interesante, ya que se detecta la necesidad de un módulo de razonamiento dentro del sistema SINRUT.

En la ciudad de Medellín no se conoce de la creación de sistemas de recomendación relacionados, por lo que podría considerarse el presente trabajo como un caso piloto.

Las siguientes secciones se distribuyen como sigue: en la sección 2 se evidencia la metodología MAS-CommonKADS a través del diseño del sistema propuesto, en la sección 3 se expone la implementación del sistema, terminando con las secciones 4 y 5 en las cuales se presentan el trabajo futuro y las conclusiones.

## Metodología

Como se mencionó en la introducción, el sistema se desarrolló con base en la metodología MAS-CommonKADS, introducida por Iglesias & Garijo (2005). En esta sección se evidencia cada etapa de la metodología en el desarrollo del sistema de recomendación de rutas:

## Conceptualización

El propósito de la conceptualización es obtener el diccionario de actores y los casos de uso generales, además de los casos de uso específicos de cada actor. La descripción detallada de los actores se presenta en la Tabla 1 y los casos de uso generales en la Figura 1.

**Tabla 1.** Descripción de los actores

| Actores | Descripción |
|---|---|
| Usuario | Es el interesado que consulta el recorrido que debería llevar a cabo para trasladarse de su lugar de origen a un lugar de destino, teniendo en cuenta sus necesidades y sus limitantes en términos de tiempo y dinero. |
| Administrador | Es el encargado de mantener actualizada la base de rutas en caso de que se cree una nueva ruta, se elimine o cambie una ya existente. |
| Búsqueda | Este actor recibe la solicitud de Usuario para que sugiera la Ruta que mejor responda a sus necesidades de desplazamiento. |
| Filtro | A partir de la búsqueda obtenida por el actor Búsqueda, este actor filtra el resultado con base en las preferencias de Usuario. |
| Perfil | Administra las preferencias del Usuario, tanto cuando este las ingresa como cuando son requeridas por Filtro. |

Fuente: Elaboración propia (2014)

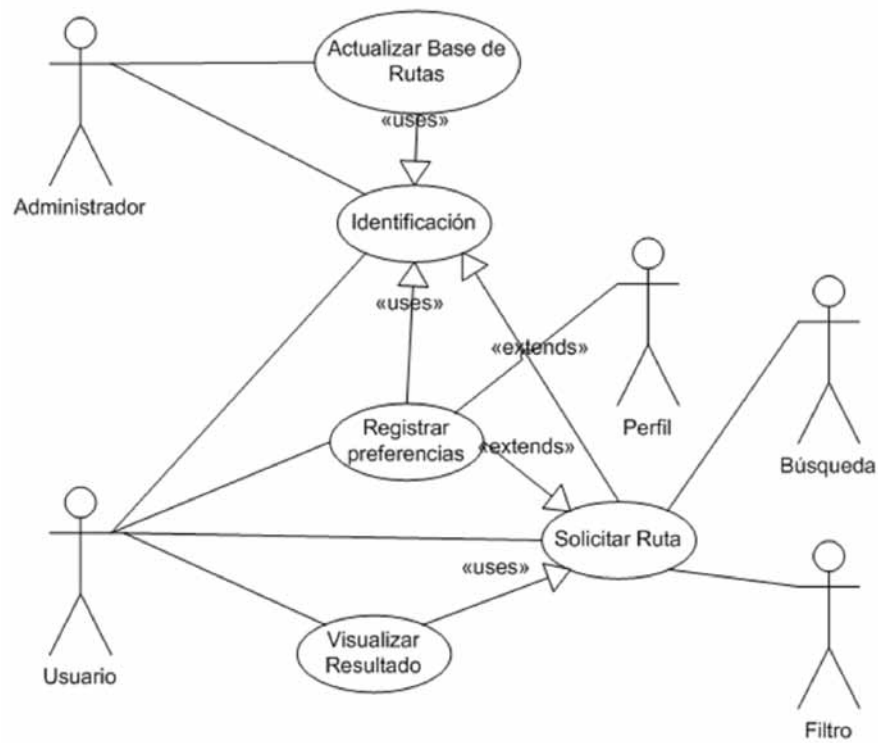



**Figura 1.** Casos de uso generales del sistema
Fuente: Elaboración propia (2014)

## Análisis

En la Fase de Análisis de MAS-CommonKADS se desarrollan 6 de los 7 modelos de la metodología, estos son: Modelo de Agentes, Modelo de Tareas, Modelo de Experiencia, Modelo de Coordinación, Modelo de Comunicación y Modelo de Organización.

El Modelo de Agentes tiene como fin identificar los Agentes del SMA. Para este proceso se utilizaron tres de las técnicas propuestas en la metodología MAS-CommonKADS: a) Análisis de los actores de la fase de conceptuación, b) Análisis del enunciado del problema, y c) Identificación de agentes empleando tarjetas CRC (Clases-Responsabilidades-Colaboraciones). Así, los Agentes Identificados en el problema son como se presentan en la Tabla 2. Adicionalmente, en la Tabla 3 se ilustra el artefacto utilizado para realizar la descripción de los cinco agentes, en este caso puntual se presenta el agente Perfil.

**Tabla 2.** Descripción de los agentes

| Agente | Descripción |
|---|---|
| Agente Administrador | Es un agente de interfaz que representa al Administrador dentro del sistema multi-agente. |
| Agente Usuario | Es un agente de interfaz que representa al Usuario dentro del sistema multi-agente. |
| Agente Búsqueda | Es un agente de búsqueda inteligente que representa al Buscador dentro del sistema multi-agente. |
| Agente Filtro | Es un agente inteligente que representa al Filtro dentro del sistema multi-agente. |
| Agente Perfil | Es un agente de perfil de usuario que representa al Perfil dentro del sistema multi-agente. |

Fuente: Elaboración propia (2014)

**Tabla 3.** Plantilla del Agente Perfil

| | |
|---|---|
| **Agente** *Perfil* | |
| **Tipo** | Agente software inteligente |
| **Papel** | Perfil |
| **Posición** | Contenido en la sociedad de agentes del sistema. |
| **Capacidades-razonamiento experiencia** | Capacidad de adaptarse al perfil del usuario.<br>Capacidad de comunicarse con otros agentes utilizando el lenguaje definido en la ontología. |
| **Descripción**<br>Este agente realiza varias actividades que son:<br>• Realiza búsqueda de perfiles en la base de datos.<br>• Maneja la base de datos de perfiles de usuarios (ingreso y modificación).<br>• Realiza verificaciones de perfiles sobre la base de datos y comparte esta información con otros agentes. | |
| **Objetivo:** Administrar los diferentes perfiles de los usuarios. | |
| **Servicios:**<br>• Validar inicio de sesión de Usuario.<br>• Actualizar perfil de Usuario.<br>• Buscar perfil de Usuario. | |
| **Comunicación:** No se comunica con ningún agente humano. | |
| **Coordinación:** Interactúa con el agente de interfaz Usuario y con el agente Filtro. | |
| **Parámetros de entrada:** Información de perfiles de usuario a ingresar y modificar (Nombre de usuario, preferencias como dinero disponible, tiempo disponible y distancia que el usuario está dispuesto a caminar, las coordenadas del punto de partida y llegada). | |
| **Parámetros de salida:** Muestra información de perfiles de usuario (Nombre de usuario, preferencias como dinero disponible, tiempo disponible y distancia que el usuario está dispuesto a caminar, las coordenadas del punto de partida y llegada). | |

Fuente: Elaboración propia (2014)

El Modelo de Tareas permite mostrar la descomposición funcional del sistema. Las tareas generales que se identificaron en este sistema propuesto fueron: Administrar Interfaces del Administrador, Administrar Interfaces del Usuario, Administrar el Perfil del Usuario, Buscar Rutas y Filtrar Rutas. En este artículo se mostrará la descripción de la tarea general Filtrar Rutas, cuya definición es análoga a la realizada con las demás tareas generales. La tarea Filtrar Rutas se lleva a cabo dentro del sistema multi-agente por el agente Filtro. En la Figura 2 (a) se muestra la descomposición de esta tarea en sub-tareas y en la misma figura, en la parte (b), se expone el diagrama de actividades que debe realizar el agente para llevar a cabo esta tarea.

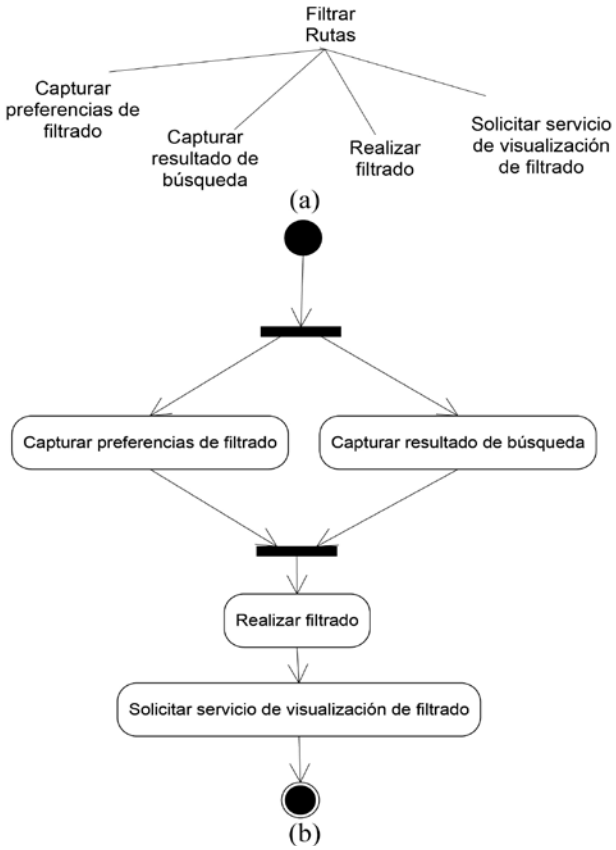



**Figura 2.** Descomposición (a) y Diagrama de actividades (b) de la tarea Filtrar Rutas
Fuente: Elaboración propia (2014)

En la Tabla 4 se muestra la plantilla textual propuesta por la metodología MAS-CommonKADS para describir en detalle las sub-tareas; en esta tabla se describe la sub-tarea Capturar preferencias de filtrado.

**Tabla 4.** Plantilla para la tarea Capturar preferencias de filtrado

| | |
|---|---|
| **Tarea:** *Capturar preferencias de filtrado* | |
| **Objetivo**: Obtener las preferencias que parametrizarán el filtrado. | |
| **Descripción**: El agente Filtro recibe del agente Usuario las preferencias del usuario para realizar la búsqueda de las rutas más útiles para el usuario. | |
| **Entrada**: Perfil del usuario. | **Salida**: Ninguna |
| **Precondición**: Ninguna | **Frecuencia:** Puede ser en cualquier momento. |

Fuente: Elaboración propia (2014)

El Modelo de la Experiencia involucra la identificación, descripción y estructuración del conocimiento que requieren los agentes para realizar sus tareas. El desarrollo del modelo incluye el desarrollo del dominio, el conocimiento de inferencias y el conocimiento de tareas.

Como parte del conocimiento del dominio se identifican todos los conceptos y predicados. Estos fueron generados por medio de la herramienta Protégé, creando una ontología cuyo nombre es: OntoRecomendacion Rutas Ontology y la cual incorpora los conceptos Concepto Nodo, Concepto Punto, Concepto Trayectoria, Concepto Ruta, Concepto Preferencia, Concepto Usuario, Concepto Agente, Concepto Lista Nodos, Concepto Lista Puntos, Concepto Lista Trayectorias y Concepto Lista Rutas; y predicados Predicado Actualizar Preferencia, Predicado Devolver Busqueda, Predicado Devolver Filtrado, Predicado Devolver Preferencia, Predicado Devolver Trayectoria, Predicado Realizar Busqueda, Predicado Recuperar Preferencia, Predicado Recuperar Trayectoria y Predicado Solicitar Filtrado.

Por otro lado, el conocimiento de inferencias en MAS-CommonKADS se especifica mediante la definición de las inferencias que se realizan en la resolución del problema y la estructura de inferencias que relacionan las inferencias y los papeles del conocimiento (metaclases) sobre los que operan las inferencias. En la Figura 3 se presenta un diagrama que ilustra el conocimiento de inferencias del sistema, en esta figura se puede apreciar que el proceso de filtrado se realiza utilizando métodos heurísticos, explícitamente se utilizó el algoritmo A* (Hart et al., 1968) utilizando el método Manhhatan (Leigh et al., 2007) y sin rodear las esquinas.

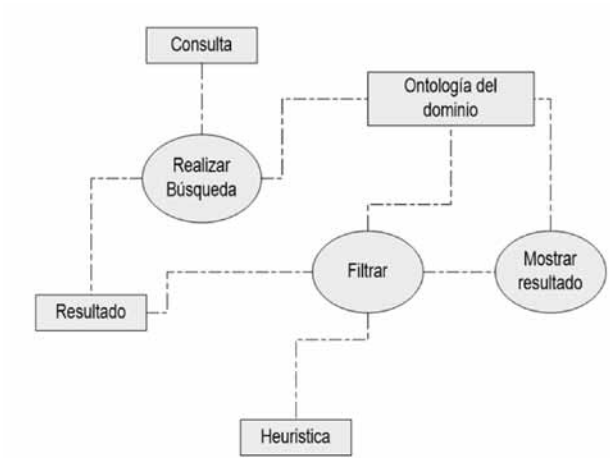



**Figura 3.** Conocimiento de inferencias
Fuente: Leigh et al. (2007)

En cuanto al conocimiento de las tareas, MAS-CommonKADS establece que se describan las tareas que requieren conocimiento para ser llevadas a cabo. Es así como se identificó que el conocimiento requerido por las tareas del sistema es acorde a la especificación resumida en la Tabla 5.

**Tabla 5.** Conocimiento de las tareas

| Tarea | Conocimiento de la tarea |
|---|---|
| Buscar perfil de usuario. | Conocer los usuarios del sistema y los perfiles de cada uno. |
| Capturar datos de inicio de sesión, capturar cambios en la BD, capturar preferencias de filtrado, capturar resultado de búsqueda, capturar consulta de búsqueda, capturar resultados del agente de filtrado. | Conocer la ontología del dominio para procesar la captura. |
| Solicitar servicio de validación de usuario, solicitar servicio de búsqueda, solicitar servicio de búsqueda de perfil, solicitar servicio de filtrado, solicitar servicio de visualización de filtrado. | Conocer la ontología del dominio para procesar la solicitud. |
| Realizar filtrado. | Conocimiento heurístico para obtener la mejor ruta. |

Fuente: Elaboración propia (2014)

Siguiendo con los modelos de la etapa de análisis de la metodología utilizada, se desarrolló el Modelo de Comunicación que involucra el modelado de las interacciones entre los agentes humanos y el sistema. Parte del modelo se presenta en la Tabla 6, en la cual se expone la conversación Solicitar acceso usuario registrado.

**Tabla 6.** Plantilla para la conversación Solicitar acceso usuario registrado

| **Conversación:** *Solicitar acceso usuario registrado* | |
|---|---|
| **Tipo:** | Reactiva |
| **Objetivo:** | Identificar el usuario en el sistema. |
| **Iniciador:** | Usuario |
| **Descripción:** | El usuario solicita el servicio de identificación ante el sistema. El Agente Interfaz de Usuario se encarga de capturar el nombre de usuario y contraseña para luego consultar en la base de datos e identificar al usuario en el sistema. |
| **Precondición:** | El usuario ha solicitado ingreso al sistema y ha ingresado su nombre de usuario y contraseña. |
| **Post-condición:** | El Agente Interfaz de Usuario ha capturado la consulta del usuario y busca en base de datos la información de identificación dada. |
| **Condición de terminación:** | El Agente Interfaz de Usuario responde a la petición de identificación en el sistema, siendo autorizado o denegado dicho acceso. Se pide al usuario que ingrese la petición de identificación nuevamente. |

Fuente: Elaboración propia (2014)

Por su parte, el Modelo de Coordinación se define con el fin de desarrollar y describir las interacciones entre los agentes involucrados en la resolución de un problema en un sistema multi-agente. Este modelo estructura las interacciones en conversaciones. Se define una conversación como un conjunto de interacciones iniciadas para alcanzar un objetivo. Cada interacción entre dos agentes se realiza mediante el envío de un mensaje, y tiene asociado un acto de habla.

En la Figura 4 se muestra el diagrama de coordinación general de los agentes. Adicionalmente, en la Tabla 7 se presenta la plantilla para la conversación Envío resultados de búsqueda y en la Figura 5 se muestra el Diagrama de secuencias para esta conversación; en este diagrama se observa que después de que el agente Búsqueda realiza la búsqueda de rutas de acuerdo a la ubicación (coordenadas) del usuario, envía la información de las rutas resultantes al agente Filtro, generando en la conversación las intervenciones Enviar Información Rutas por Ubicación, y Confirmación Entrega de Rutas por Ubicación.

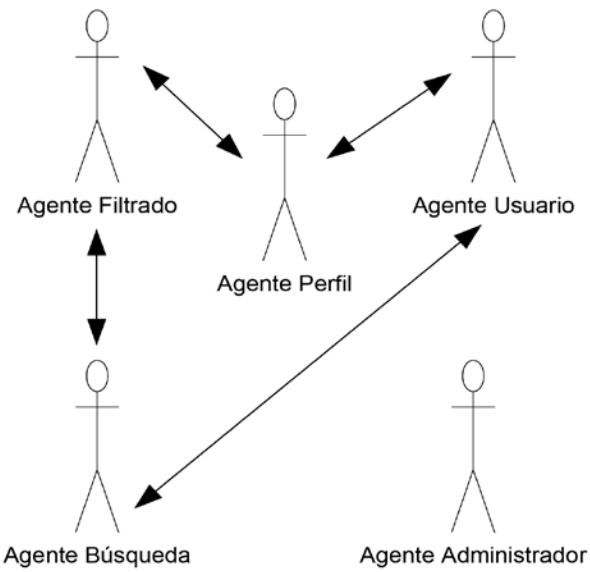



**Figura 4.** Modelo de coordinación
Fuente: Elaboración propia (2014)

**Tabla 7.** Plantilla para la conversación Envío resultados de búsqueda

| | |
|---|---|
| **Conversación:** ***Envío resultados de búsqueda*** | |
| **Tipo:** Paso de información. | |
| **Objetivo:** Agente Búsqueda envía las rutas encontradas en la cercanía de las ubicaciones al Agente Filtro. | |
| **Agentes:** Agente Filtro y Agente Búsqueda. | **Iniciador:** Agente Búsqueda. |
| **Descripción:**<br>Luego de que Agente Búsqueda obtiene las posibles rutas que pasan suficientemente cerca de los puntos acordados por el usuario, Agente Filtro adecua estas rutas a las preferencias de cada usuario.<br>• **Enviar rutas encontradas:** Se envían las rutas encontradas en la búsqueda hacia el Agente Filtro.<br>• **Aplicar filtro:** Aplica las preferencias del perfil encontradas en el Agente Filtro. | |
| **Servicio:** Proveer rutas al Agente Filtro. | |
| **Precondición:** Localización del Agente Filtro y Agente Búsqueda. Terminación del proceso de búsqueda del Agente Búsqueda. | **Post-condición:** Se entregaron las rutas de Agente Búsqueda. |
| **Condición terminación:** Ninguna. | |

Fuente: Elaboración propia (2014)

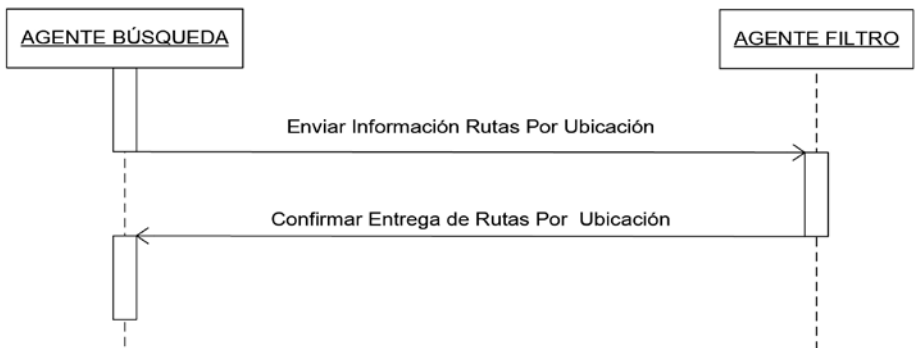



**Figura 5.** Diagrama de secuencias para la conversación Envío resultados de búsqueda
Fuente: Elaboración propia (2014)

El Modelo de Organización define las relaciones estáticas existentes entre los diferentes agentes del sistema. Para esto se llevan a cabo varias tareas principales como son: Estudio de las oportunidades de introducción del sistema, Identificación de la estructura organizacional e Identificación de relaciones de herencia. MAS-CommondKADS define constituyentes para identificar entre otros las funciones.

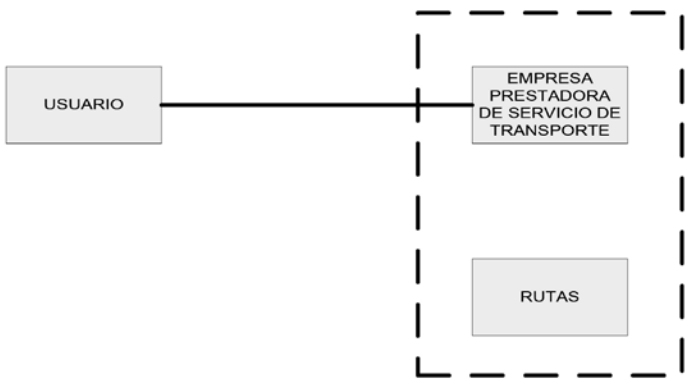



**Figura 6.** Diagrama de la organización
Fuente: Elaboración propia (2014)

La Empresa Prestadora de Servicio de Transporte y las Rutas están administradas por una sola entidad. El usuario y la empresa prestadora del servicio se localizan como cabezas principales, ya que el usuario es quien hace uso de los servicios prestados por la empresa. Las funciones identificadas se exponen en la Tabla 8 y la estructura organizativa del sistema se presenta en la Figura 7.

**Tabla 8.** Funciones en la organización

| Funciones | Descripción |
|---|---|
| Especificar ubicación y preferencias | Consiste en la solicitud de servicio de transporte (rutas) que hace el usuario a la empresa prestadora de servicios de transporte, de acuerdo a su ubicación y sus preferencias. |
| Buscar rutas aledañas | Consiste en la búsqueda de las rutas más cercanas, dependiendo de la ubicación del usuario. |
| Seleccionar rutas óptimas | Luego de realizar la búsqueda se realiza una selección de rutas, de acuerdo a las preferencias del usuario. |
| Entregar rutas | Se le entrega al usuario la información de las rutas seleccionadas. |
| Detectar cambios en las rutas | Consiste en detectar los cambios que se darán en las rutas actuales. |
| Actualizar rutas | Es la generación o modificación de rutas hecha por la empresa prestadora de servicio, después de ser autorizada por la entidad administradora. |

Fuente: Elaboración propia (2014)

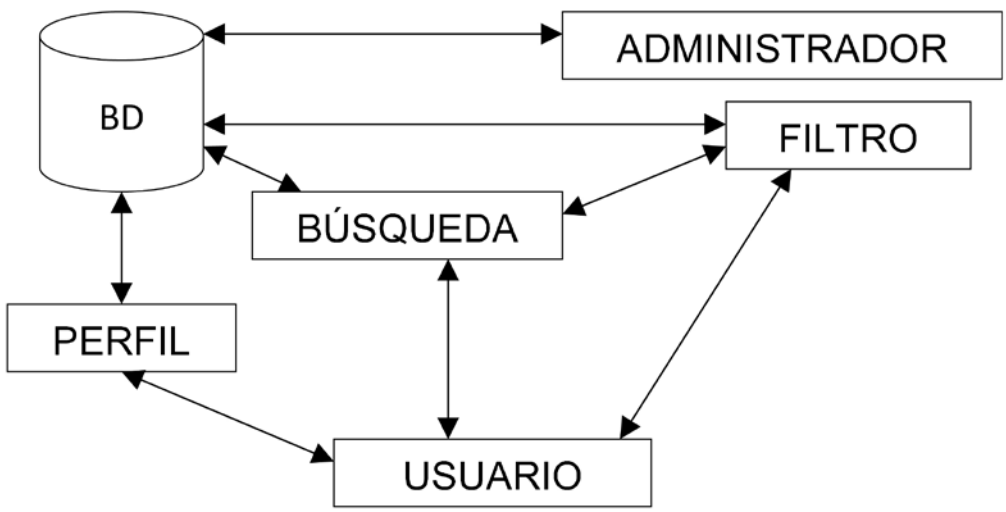



**Figura 7.** Estructura organizativa del sistema multi-agente
Fuente: Elaboración propia (2014)

## Diseño

La fase de diseño de la metodología MAS-CommonKADS se centra en el desarrollo del modelo de diseño que pretende transformar las especificaciones del resto de modelos para que se puedan describir con un lenguaje de programación.

Con este modelo se estructuran los componentes del sistema tomando como punto de entrada los resultados de la fase de análisis. El modelo de diseño de MAS-CommonKADS distingue tres clases de decisiones de diseño: diseño de la red, diseño de los agentes y diseño de la plataforma.
El diseño de la red consiste en desarrollar el "modelo de red" que proporciona a los agentes una visión uniforme de la red. Está definido como un modelo en capas. Para el diseño de la red se utiliza una plantilla propuesta por MAS–CommonKADS para definir los constituyentes del diseño de red (ver Tabla 9).

**Tabla 9.** Plantilla del diseño de red

| |
|---|
| **Red:** *RecomendacionRutas* |
| **servicios-de-red:** Los servicios de red que se prestan en el sistema multi-agente son:<br>• Registrarse (se realiza con la primitiva "register").<br>• Anunciar-servicio (se lleva a cabo con la primitiva "advertise").<br>• Abandonar-servicio (se realiza con "unadvertise").<br>• Consulta-páginas-amarillas (Cuando un agente desea consultar envía un mensaje con la primitiva "recommend-all"). |
| **agentes-de-red:** *Agente Facilitador de Directorios (DF)* |
| **Descripción:** Las funcionalidades de red (páginas amarillas) son proporcionadas por el agente facilitador. Este sistema multi-agente, por estar enfocado hacia consulta masiva de usuarios, puede orientarse hacia una arquitectura distribuida.<br><br>El Agente Facilitador estará encargado de realizar las siguientes funciones:<br>• *Registrar agentes:* Consiste en recibir el registro de los agentes que están interesados en promocionar un servicio al sistema multi-agente.<br>• *Proporcionar información:* El facilitador debe proporcionar la información que le solicite un agente; los agentes pedirán al facilitador que les recomiende agentes con los cuales puedan continuar el proceso de búsqueda hasta llevarlo a su etapa final.<br>En conclusión, el Agente Facilitador estará en la capacidad de recibir, manejar y proporcionar información de los servicios que registren los agentes *Usuario*, *Perfil*, *Búsqueda*, *Filtrado*, *Administrador*. |
| **Lenguaje de Comunicación:** El lenguaje de comunicación utilizado es FIPA-ACL. A través de este lenguaje se definen los diferentes protocolos de negociación e intercambio de mensajes entre los agentes del sistema.<br>**Ontología:** La ontología para el sistema es "OntoRecomendacionRutasOntology", y está definida en el modelo de la experiencia. |

Fuente: Elaboración propia (2014)

El diseño de los agentes se realiza siguiendo la plantilla propuesta por MAS-CommonKADS que se aprecia en la Tabla 10. En esta plantilla específica se describe el agente Administrador.

**Tabla 10.** Plantilla para el diseño de agentes: Agente Administrador

| **Sistema-Agente *Administrador*** |
|---|
| **Arquitectura**<br>Los agentes se desarrollaron en JAVA, usando la plataforma JADE para el desarrollo de sistemas multiagente. |
| **tiene-subsistema**<br>Proveer Rutas, Proveer Usuarios, Actualizar Usuarios, Actualizar Rutas. |
| **lenguaje-diseño**<br>JAVA |
| **Subsistema** *Proveer Rutas*<br>**Tipo** de ejecución de tareas<br>**Funcionalidades:** Hace las veces de Agente transducer para que el Administrador del sistema pueda solicitar las rutas existentes en la Base de Datos. Esta solicitud se hace por medio del Módulo de Visualización que se implementó en la herramienta .NET. A su vez este agente hace la consulta en la base de datos y recupera los datos necesitados por el usuario y los devuelve.<br>**Subsistema** *Proveer Usuarios*<br>**Tipo:** de ejecución de tareas.<br>**Funcionalidades:** Hace las veces de Agente Transducer para que el Administrador del sistema pueda solicitar la información de los usuarios existentes en la Base de Datos. Esta solicitud se hace por medio del Módulo de Visualización que se implementó en la herramienta .NET. A su vez este agente hace la consulta en la base de datos y recupera los datos necesitados por el usuario y los devuelve.<br>**Subsistema** *Actualizar Usuarios*<br>**Tipo** de ejecución de tareas.<br>**Funcionalidades:** Hace las veces de Agente Transducer para que el Administrador del sistema pueda actualizar la información de los usuarios existentes en la Base de Datos. Esta actualización se hace por medio del Módulo de Visualización que se implementó en la herramienta .NET. A su vez este agente hace la consulta en la base de datos y recupera los datos necesitados por el usuario y los devuelve.<br>**Subsistema** *Actualizar Rutas*<br>**Tipo** de ejecución de tareas.<br>**Funcionalidades** Hace las veces de Agente Transducer para que el Administrador del sistema pueda actualizar las rutas existentes en la Base de Datos. Esta actualización se hace por medio del Módulo de Visualización que se implementó en la herramienta .NET. A su vez este agente hace la consulta en la base de datos y recupera los datos necesitados por el usuario y los devuelve. |

Fuente: Elaboración propia (2014)

Finalmente, el último de los diseños es el de la plataforma, permite documentar las decisiones de bajo nivel sobre el lenguaje de implementación seleccionado, el software y hardware empleado, y los usuarios finales del sistema. En la Tabla 11 se muestra la plantilla textual para el diseño de la plataforma y en la Figura 8 se presenta el diagrama de despliegue.

**Tabla 11.** Plantilla para el diseño de la plataforma

| **Plataforma** *Sistema-Multiagente* |
|---|
| **Descripción**<br>El Sistema Multi-Agente está desarrollado en JAVA utilizando las herramientas JADE y PROTEGE, dadas las diferentes características ventajosas que tiene este lenguaje, con la optimización de estas herramientas de programación para la elaboración de software, como son:<br>**Ventajas de JAVA**<br>• El propósito general del lenguaje JAVA es totalmente orientado a objetos, este tipo de programación facilita la elaboración, el mantenimiento, la modificación y la reutilización de software.<br>• JAVA es independiente de la plataforma. Sus programas se compilan en *código de bytes (bytecodes)*, que es independiente de la arquitectura y del sistema operativo. Por tanto, los programas pueden escribirse y compilarse una vez y luego transmitirse para ejecutarse en cualquier lugar.<br>• JAVA proporciona múltiples flujos de control que se ejecutan de manera concurrente dentro de uno de sus programas. Los hilos permiten que un programa emprenda varias tareas de cómputo al mismo tiempo.<br>**Ventajas de JADE**<br>• Entorno de desarrollo para la creación de aplicaciones basadas en agentes.<br>• Entorno de ejecución para que los agentes vivan y se comuniquen.<br>• Arquitectura peer-to-peer (P2P).<br>• Interoperabilidad: especificaciones FIPA.<br>• Portabilidad: realizado en JAVA.<br>Además de lo anterior se implementará un módulo de visualización en el Framework Microsoft .NET, en el lenguaje C# que será manejado al interior del sistema multiagente por medio de dos agentes *Transducer* que serían el Agente Usuario y el Agente Administrador como se muestra en la Figura 9.<br>Finalmente, se implementará una base de datos en el motor SQLite por las grandes ventajas de portabilidad que proporciona. |
| **usa-lenguaje**<br>JAVA, FIPA-ACL, JADE, C# (.NET). |
| **hardware-requerido**<br>Un computador con 512MB RAM o superior y un procesador Pentium 4 o equivalente. |
| **software-requerido**<br>Máquina Virtual de Java (JVM) 1.5 o superior, Framework .NET 2.0 o superior y cualquier sistema operativo que soporte estas dos plataformas. |
| **Usuario**<br>Usuario<br>Administrador |

Fuente: Elaboración propia (2014)

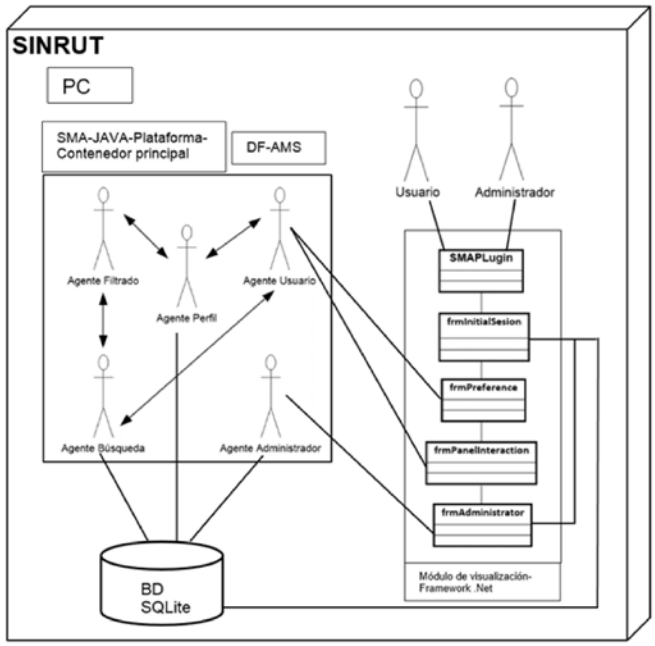
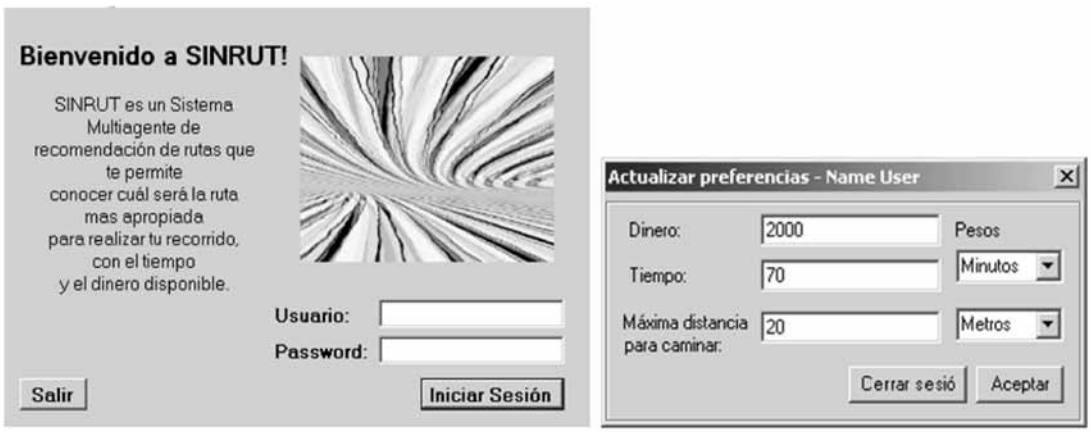



**Figura 8.** Diagrama de despliegue para entidades y relaciones de la plataforma de SINRUT
Fuente: Elaboración propia (2014)

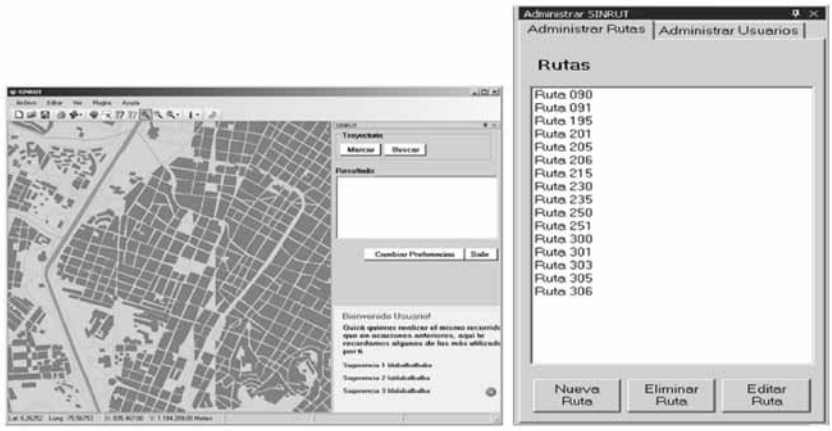
## Implementación

El módulo de visualización se desarrolló en el Framework .Net., se logró gracias a las posibilidades que brinda el SIG MapWindow para extender sus funcionalidades a través de plugins, dando así la facilidad de que el usuario utilice las herramientas de SIG para este sistema, como lo son el Zoom, entre otras. En las Figuras 10 a 14 se muestran las interfaces utilizadas en el SMA.




(a) (b)

**Figura 9.** Inicio de sesión (a) y definición de perfil (b)
Fuente: Elaboración propia (2014)




(a) (b)

**Figura 10.** Interfaz de interacción del usuario (a) y de administración (b)
Fuente: Elaboración propia (2014)


(a) (b)

**Figura 11.** Interfaz para el ingreso de nuevos usuarios (a) y rutas (b)
Fuente: Elaboración propia (2014)

## Resultados

El sistema adquiere el conocimiento de dos maneras: una cuando el usuario Administrador ingresa explícitamente los datos de las rutas del sistema, y la otra por medio de la interacción del usuario con el sistema, ya que este último va obteniendo una base de casos para en futuras ocasiones utilizarlos con el fin de resolver otros problemas, es decir, realizando un razonamiento basado en casos.

El sistema arroja como resultado una lista de trayectorias que el usuario puede utilizar para realizar su recorrido, cada trayectoria contiene a su vez una lista de rutas. La optimización de este resultado lo realiza inicialmente el Agente Búsqueda, que obtiene la lista de rutas por medio de la utilización del algoritmo heurístico A*, y seguidamente se refina la búsqueda por medio del Agente Filtrado basándose en las preferencias del usuario.

## Conclusiones

El sistema diseñado SINRUT es una alternativa a la recomendación de rutas de transporte público para la ciudad de Medellín, que brinda al usuario una solución a la hora de tomar decisiones de movilidad. Adicionalmente, la ejecución paso a paso de la metodología MAS-CommonKADS ofrece a los lectores una luz bajo la cual ejecutar proyectos similares. Por otro lado, la utilización del paradigma de agentes resultó apropiada, pues

permitió distribuir con éxito las tareas del sistema. La integración de las plataformas .NET y Java agregaron complejidad al desarrollo que se logró con el uso de archivos XML (eXtensible Markup Language). Específicamente en el tema del sistema de recomendación de rutas en la ciudad de Medellín, se plantea como trabajo futuro:

- Realizar un módulo de aprendizaje de manera que el sistema se adapte al perfil del usuario de una manera más inteligente.
- Indagar de qué manera segura y ágil se pueden comunicar las plataformas .NET y JAVA para realizar procesos como los propuestos en este trabajo.
- Ampliar el grafo de las rutas de la ciudad para que abarque más lugares y no se reduzca a una pequeña zona de estudio.

## Referencias

Aburizaiza, A. O. & Ames, D. P. (2006). GIS-Enabled Desktop Software Development Pardigms. *Advanced Geographic Information Systems & Web Services, 2009. GEOWS '09. International Conference on*, 1, (1), 75-79.

Arthur, L., Chien-Lung, H. & Eldon, L. (2014). Improving the Effectiveness of Experiential Decisions by Recommendation Systems. *Expert Systems with Applications*, 41(10), 4904-4914.

Castillo, L., Armengol, E., Onaindía, E., Sebastiá, S., González-Boticario, J., Rodríguez, A., Fernández, S., Arias, J.D. & Borrajo, D. (2008). SAMAP: An User-Oriented Adaptive System for Planning Tourist Visits. *Expert Systems with Applications*, 34(2), 1318-1332.

Gennari, J. H., Musen, M. A., Fergerson, R. W., Grosso, W. E., Crubézy, M., Eriksson, H., Noy, N. F. & Tu, S. W. (2003). The Evolution of Protégé: an Environment for Knowledge-based Systems Development, International. *Journal of Human-Computer Studies*, 58(1), 89-123.

Gnjatović, M. & Delić, V. (2014). Cognitively-inspired Representational Approach to Meaning in Machine Dialogue. *Knowledge-Based Systems*, 71, 25-33.

Hart, P., Nilsson, N. & Raphael, B. (1968). A Formal Basis for the Heuristic Determination of Minimum Cost Paths. *IEEE Trans. Syst. Sci. Cybernet*, 4(2), 100–107.

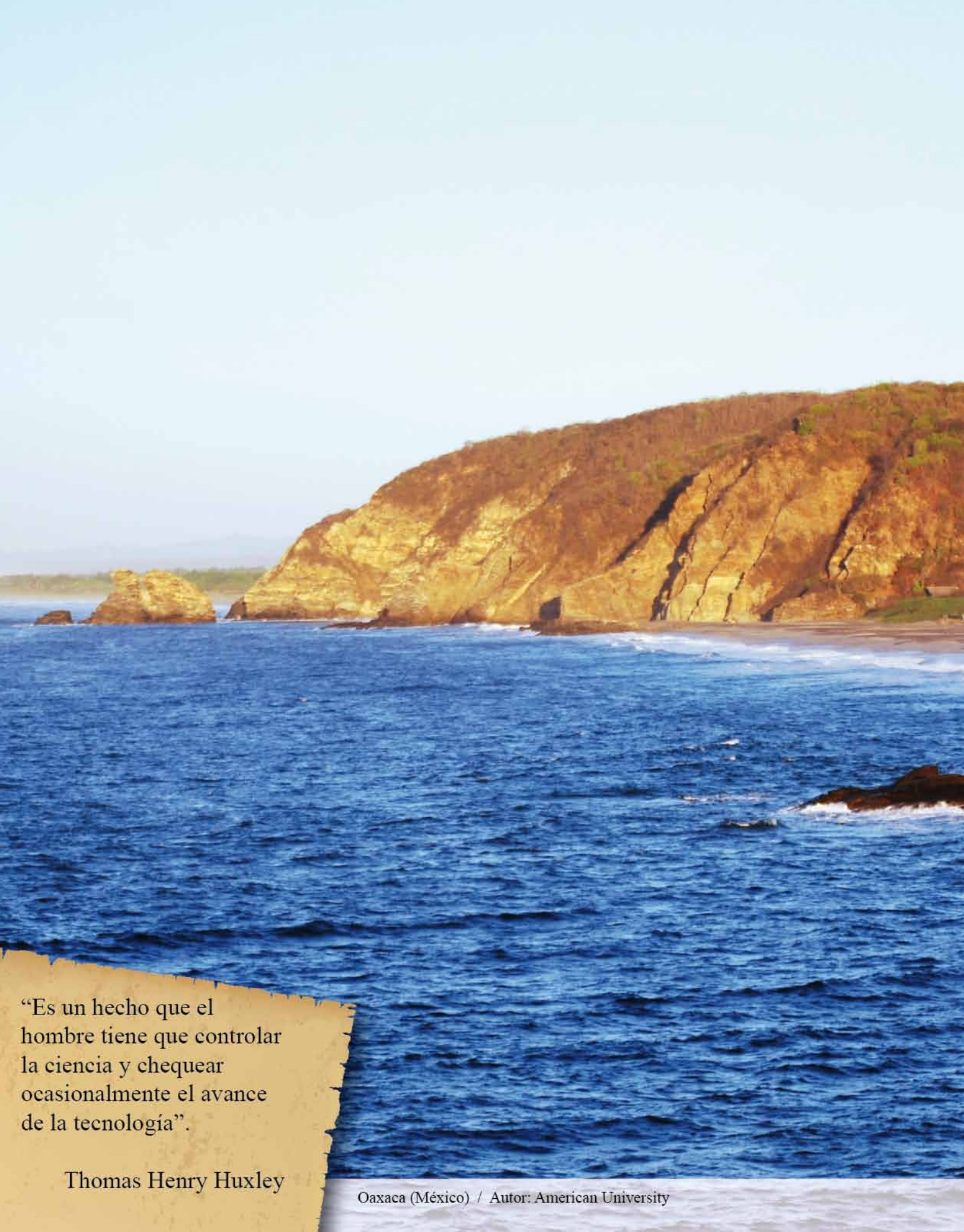
Iglesias, C. A. & Garijo, M. (2005). The Agent-Oriented Methodology MAS-CommonKADS. *In B. Henderson-Sellers, & P. Giorgini (Eds.) Agent-Oriented Methodologies*, 46-78.

Leigh, R., Louis, S.J. & Miles, C. (2007). Using a Genetic Algorithm to Explore A*-like Pathfinding Algorithms. *Computational Intelligence and Games, 2007. CIG 2007. IEEE Symposium on*, 1(1), 72-79.

Mohapatra, D.P., Kumar, R., Mall, R., Kumar, D.S. & Bhasin, M. (2006). Distributed Dynamic Slicing of Java Programs. *Journal of Systems and Software*, 79(12), 1661-1678.

Vila, X., Schuster A. & Riera, A. (2007). Security for a Multi-Agent System based on JADE. *Computers & Security*, 26(5), 391-400.

Wenjie, Z., Weiwei, L., Tao, X., Shengyi, X. & Jingxin, N. (2012). A Complete Development Process of Finite Element Software for Body-in-white Structure with Semi-rigid Beams in .NET Framework. *Advances in Engineering Software*, 45(1), 261-271.



"Es un hecho que el hombre tiene que controlar la ciencia y chequear ocasionalmente el avance de la tecnología".

Thomas Henry Huxley

Oaxaca (México) / Autor: American University