Universidad Autónoma de Sinaloa

Facultad de Informática y Facultad de Ciencias de la Tierra y el Espacio

Maestría en Ciencias de la Información

Sistemas Multi-Agentes para la solución del problema de programación de horarios escolares mediante negociación

Que como requisito parcial para obtener el grado de
Maestro en Ciencias de la Información

P R E S E N T A:
César Covantes Osuna

Director de Tesis:
Dr. Miguel Contreras Montoya

Culiacán, Sinaloa México, junio de 2014

Dedicatoria

A mis padres, por brindarme el apoyo a lo largo de mi vida para que pudiera lograr mis sueños.

A mi novia por su compañía, confianza, paciencia, comprensión y sacrificio de su tiempo para que pudiera cumplir con mis objetivos.

A todas las personas que creyeron en mí, me apoyaron y abrieron sus puertas.

A las instituciones y profesores que me han formado a lo largo de mi vida con su granito de arena del cual gracias a ellos soy parte de lo que soy ahora.

Aquellas personas que le dediquen su tiempo en leer este trabajo y les pueda servir de ayuda.

Agradecimientos

Quiero expresar mi gratitud a todas las personas e instituciones que de una u otra manera colaboraron en el desarrollo de esta Tesis de Maestría. En primer término mencionar a mi familia quienes han sido un pilar importante en mi desarrollo profesional, mi padre César Covantes Rodríguez, madre Elvia Y. Osuna Lizárraga, mis hermanos Edgar y Gerardo y sin olvidar a mi novia Irma G. Tirado Lerma por estar al pendiente durante la permanencia de mis estudios de maestría.

Agradezco también a mis compañeros de posgrado: Daniel, Jorge, Rosendo, Miguel, Marcia y a Edgar (hermano), al igual que a los estudiantes de doctorado Ángel y Oswaldo quienes con ellos conviví una buena experiencia, al igual de recibir apoyo y sugerencias en cada reunión de trabajo.

Mi reconocimiento y respeto a los asesores de los cursos del Programa de Maestría en Ciencias de la Información de la Facultad de Informática y Facultad de Ciencias de la Tierra y el Espacio de la Universidad Autónoma de Sinaloa por su apoyo y orientación brindados durante el desarrollo de la maestría: Dr. Jorge Adalberto Navarro Castillo, Dr. Eduardo René Fernández González, Dr. Inés Fernando Vega López que ayudaron en el desarrollo de esta tesis.

Así mismo reitero mi más profundo agradecimiento por las observaciones en el protocolo de investigación y avance de tesis a los investigadores: Dr. Ulises Zaldívar Colado, Dr. René Rodríguez Zamora, Dr. Guadalupe Esteban Vázquez Becerra y al Dr. Ramón Victorino García López quienes analizaron los puntos que surgían en la investigación y contribuyeron de manera importante con sus argumentos. Un reconocimiento a mi asesor Dr. Miguel Contreras Montoya por la formulación del Protocolo de Investigación de esta tesis.

También mi gratitud se hace extensiva: Dr. Santiago Inzunza Cázares, M. C. Roberto Bernal Guadiana, Dr. Juan Martín Aguilar Villegas, Dr. Wenseslao Plata Rocha y a la M.C. Thania Roxana Félix González, ex director de la Facultad de Informática, director actual de la Facultad de Informática, director de la Facultad de Ciencias de la Tierra y el Espacio, coordinador del posgrado y asistente de la coordinación del posgrado, respectivamente del Programa Educativo en Ciencias de la Información de la Universidad Autónoma de Sinaloa, institución que brindó en esta ocasión materializar el esfuerzo presente.

Así mismo, reconozco a la Dirección General de Investigación y Posgrado (DGIP) de la Universidad Autónoma de Sinaloa, quien a través de sus gestiones hicieron posible estar en el Programa Nacional de Posgrado de Calidad (PNPC) del Consejo Nacional de Ciencia y Tecnología (CONACYT), garantizando el desarrollo de la investigación. Igualmente reconozco el apoyo y financiamiento del CONACYT que me autorizó la beca no. 326030, sin la cual no hubiera sido posible la realización de esta investigación.

Resumen

Esta investigación está dirigida al público del área de las Tecnologías de la Información (IT – por sus siglas en inglés), específicamente al público del área de sistemas computacionales donde se realiza el análisis, diseño e implementación de un sistema basado en sistemas multi-agente (SMA) mediante negociación, por medio de un lenguaje de programación orientado a objetos para dar solución al problema de la programación de horarios escolares, además los resultados de esta investigación pueden ser de interés para aquellas personas ajenas de las IT que diseñan la programación de horarios.

Sin embargo, también en la investigación se encamina al público del área computacional (teoría de la computación), al considerarse la programación de horarios un problema de optimización que pertenece a la familia de los problemas NP (non-determenistic polynomial time).

En este trabajo, en el diseñó del sistema se consideraron tres tipos de agentes; un agente coordinador que se encarga de obtener las restricciones de actividad, además de instanciar, crea y administrar la interfaz gráfica de usuario de los agentes grupos y profesores donde el número de grupos y profesores dependen del caso de estudio considerado. La función de los agentes profesores es obtener sus restricciones, crear y envían su propuesta de horario al agente grupo donde el profesor imparte clase y los agentes grupos reciben las propuestas de los profesores para después realizar la negociación para resolver conflictos entre los agentes profesores. El sistema modela diferentes variables, formas de caracterizar y cuantificar las restricciones de tiempo, espacio, actividades y otros tipos de restricciones de manera generalizada en un formato XML ya propuesto, permitiendo la portabilidad de los casos de estudio para crear distintas soluciones y métodos para hacer posible en un futuro la comparación

y descartar métodos.

En el proceso de experimentación y resultados se contemplaron cuatro casos de estudio de instituciones de Belice, Brasil, España y Reino Unido de un total de 22 países disponibles en un repositorio de acceso libre en la web en el cual se comparan los conflictos resueltos, conflictos sin resolver, tiempo de solución, protocolos iniciados y número de mensajes.

Concluyéndose que la simulación con sistemas multi-agente permite resolver conflictos mediante negociación entre agentes para el problema de programación de horarios escolares. Asimismo, se propone continuar con la investigación al agregar una nueva implementación en la resolución de conflictos para los casos donde no se resolviendo en su totalidad.

Abstract

The present investigation is focused on the Information Technology area (IT), specially the people of computer science area where the analysis, design and develop of the system is based on multi-agent system with negotiation through an Object-oriented programming language to solve the school timetabling problem. In addition, in this research is also oriented to those people who design the school timetabling problem.

However, the research also aimed to the people of the theory of computation area for being the school timetabling problem a optimization problem belongs to a family of problems called NP (non-determenistic polynomial time).

In this work, the designed system considered three types of agents, a coordinator agent that is in charge of obtain the restriction activity, also to instantiate, create and manage the graphical user interface (GUI) of the agents groups and teachers where the number of the last agents depend on the case of study considered. The role of the agent's teachers is to obtain the restrictions, create and send a proposal to the respective agent group where the teacher gives the lectures; the agents groups receive the teacher proposals for later perform a negotiation to solve conflicts amongst agent. The system modelled different variables, ways to characterize and quantify the constraints of time, space, activities, and other type of restriction already widely proposed in XML format, allowing the portability of case studies to create different solutions and methods to make possible in the future the comparison and discard methods.

In the process of experimentation and results were consider four case studies of institutions of Belize, Brazil, Spain, and United Kingdom with 22 countries available in a repository in a web site, in the results shown the conflicts resolved, conflicts

unresolved, solution time, initiated protocols and number of messages between the case studies.

Deducing that the simulation with multi-agent systems it is possible to resolve conflicts with negotiation between agents for the problem of school timetabling problem. In future work will attempt to continue the research with a new implementation with the conflicts where they could not to solve.

Índice general

Índice de figuras

Índice de tablas

Introducción

El ser humano de manera diaria realiza actividades que se plantea hacer en un transcurso del día y en determinado periodo de tiempo. Este conjunto de actividades es necesario organizarlo para dar origen a un horario donde importa el orden y tiempo de realización. Teniendo los horarios un gran efecto en la vida cotidiana de las personas que lo realizan en el cual se consideran diversos factores como prioridades, tiempo de dedicación a las actividades, disponibilidad de los espacio, costo y/o valoración de las consecuencias satisfaciendo un conjunto de restricciones fuertes y débiles; las restricciones fuertes se tienen que cumplir en toda circunstancia, mientras que las restricciones débiles representan una mayor flexibilidad, pudiéndose cumplir o no en su totalidad.

Este problema de la programación de horarios llamado en inglés *timetabling problem* (TTP – iníciales en inglés) resulta en ocasiones difícil de modelar dado los recursos disponibles, en el área de la computación la programación de horarios representa un problema de optimización que pertenece a la familia de los problemas NP (non-determenistic polynomial time) que representan una complejidad computacional con un gran espacio de búsqueda en generar o combinar todas las soluciones, donde el objetivo es encontrar soluciones "buenas" por medio de una función de evaluación que describa la calidad de la soluciones en un tiempo "aceptable".

De hecho, éste tipo de problema sigue presente, aún cuando existen diferentes métodos que se han desarrollado y utilizado con éxito para resolver el problema de programación de horarios en departamentos e instituciones específicas, siendo métodos de carácter no universal por lo que una propuesta de solución no puede resolver "cualquier" problema TTP.

Estas propuestas de solución rara vez se comparan entre sí por la gran cantidad de variables diferentes y diversas formas de cuantificar las restricciones que se originan a partir de políticas y prácticas diferentes, por lo que cada una de ellas tiene sus características particulares de asignación de cursos. La comparación es necesaria para determinar cuáles son los mejores métodos computacionales dados los distintos tipos de datos de horarios, permitiendo descartar las técnicas utilizadas que son relativamente simples.

En la investigación que ahora se presenta, no se realiza comparación entre los métodos, los cuales siguen siendo necesarios para descartar técnicas computacionales. Además, esta investigación ésta enfocada desde una perspectiva sistémica dado que sigue sin haber una resolución y de acuerdo con las dificultades prevalecientes en la programación de horarios al ser un problema de optimización, actualmente se aplican técnicas de inteligencia artificial o con heurísticas para reducir el espacio de búsqueda, buscando resolver este problema de manera unificada de acuerdo a los requisitos y necesidades de las distintas instituciones educativas.

Al continuar las investigaciones, emerge la necesidad de apoyarse en el área de las TI, sustentada en la teoría de la computación y diseño de sistemas utilizando una técnica de la inteligencia artificial distribuida con sistemas multi-agente (SMA) por medio de un algoritmo de negociación ya que con SMA un problema puede ser modelado como humanos, adecuándose a distintos problemas haciendo difícil esta tarea para otros métodos, además de presentar características particulares, como; la comunicación, cooperación, coordinación e incluso negociación para resolver conflictos permitiendo una planificación distribuida y así lograr un objetivo en común.

Al estar profundizando en la investigación y determinar los mejores métodos computacionales y descartar técnicas utilizadas que eran relativamente simples, es necesario una forma en la cual se represente la información, se unifiquen las restricciones dadas las diferentes instituciones y que se pueda intercambiar información, se recurre en su análisis al lenguaje de marcado extensible (XML) al ser considerado

un estándar para el almacenamiento de los datos, siendo útil para varias aplicaciones que se comunican entre sí, además de intercambiar información entre diferentes plataformas.

Entre las principales contribuciones de este desarrollo se pueden citar las siguientes: (i) un sistema basado en SMA mediante negociación para dar solución al problema de la programación de horarios escolares con estándares de agentes reconocidos por la IEEE; (ii) la unificación de diferentes restricciones que permiten la modelación de diferentes casos de estudio; (iii) un sistema para en un futuro descartar o considerar técnicas que se han utilizado con éxito para el problema de la programación de horarios.

El cuerpo de la tesis se divide en cinco capítulos y el apartado de discusión y conclusiones. El primer capítulo, denominado *Planteamiento del problema*, describe tanto la problemática como la justificación de la investigación, el supuesto y objetivos relacionados con sus interrogantes planteadas que componen el núcleo de esta investigación que fueron desarrolladores en el aspecto metodológico enfocándose en el análisis cualitativo y cuantitativo.

Los aspectos del marco teórico por su extensión documental son analizados en dos capítulos para una mejor comprensión. En esta perspectiva, el segundo capítulo, titulado *Programación de horarios*, comprende una revisión bibliográfica de los diferentes enfoques epistemológicos y metodológicos que han estado aplicándose en los estudios acerca del desarrollo de un sistema para la solución del problema de programación de horarios, al igual que la implementación de un algoritmo basado en sistemas multi-agentes mediante negociación. Además, también dentro del marco teórico, está el tercer capítulo, titulado *Sistemas Multi-Agente (SMA)* se presenta una investigación documental sobre los agentes y sistemas multi-agentes, los modelos de referencia FIPA y la plataforma de desarrollo JADE al ser una organización reconocida por sus estándares de la IEEE y estar actualizada con la última versión desde el 12 de junio del 2013. Por otra parte, se expone la estructura de un agente JADE.

El *Diseño e implementación de la solución*, comprende el cuarto capítulo, en él se indica el diseño de solución y se exponen las representaciones de las restricciones de tiempo, espacio, actividades y otros tipos. Además, se plantean las tres etapas en el diseño de la estrategia de negociación para la solución al problema de la programación de horarios. Así mismo, en esta perspectiva se indica la implementación de la base de datos utilizada en un formato XML que dan lugar a la representación, comportamiento y características de las restricciones de las instituciones y por último el diagrama de clases del sistema propuesto con sus interfaces.

El quinto capítulo, titulado *Experimentación y análisis de Resultados*, se muestran las características de los cuatros casos de estudio elegidos de manera aleatoria de un total de 22 países, así como los resultados obtenidos en la resolución de conflictos, tiempo de solución, protocolos iniciados y numero de mensajes entre los agentes al implementar el algoritmo de negociación.

En el apartado de *Discusión y conclusiones*, está la aseveración de que en una plataforma de desarrollo con SMA para realizar la programación de horarios, es posible realizar el diseño e implementación del sistema mediante negociación para resolver el problema de programación de horarios escolares permitiendo la modelación de diferentes restricciones dados los diferentes casos de estudio. En otras palabras, permite que los agentes interacciones y resuelvan conflictos de manera dinámica y no determinista, así mismo, se puede ejecutar el algoritmo para el resto de los casos de estudio de FET (2013). Permitiendo con ello, dar respuesta a las seis interrogantes de la investigación.

En el último apartado de la investigación, llamado *Trabajo a futuro*, se describen los puntos a mejorar del sistema de los estatus de los agentes, indicando los pesos de las restricciones de tiempo y actividad que se están violando. Asimismo, implementar las restricciones faltantes y agregar en el sistema una opción en el cual el usuario pueda exportar los horarios por medio de un formato XSL o XSLT. Validar los resultados obtenidos con Sistemas Multi-Agentes contra las soluciones generadas por parte de

FET que implementa la técnica de enjambre de partículas.

Capítulo 1
PLANTEAMIENTO DEL PROBLEMA

1.1. Planteamiento del problema

Existen problemas para los cuales resulta conveniente combinar o buscar de manera exhaustiva todas las soluciones, por lo cual tratan de dar solución por medio de un problema de optimización que sigue sin resolverse de manera unificada por medio de un solo algoritmo. Resulta muy común en la vida real la aparición o presencia de problemas de planificación/programación de horarios que tienen un gran efecto diario en las vidas de las personas que los utilizan, donde la experiencia humana juega un papel fundamental en la construcción de sus soluciones. Ejemplos típicos de planificación: la programación de transporte público, programación de recursos limitados, programación de ligas deportivas, programación de horarios de empleados y la programación de horarios escolares.

La planificación/programación de horarios consiste en actividades que se plantean y se tienen que realizar en un transcurso del día y en determinado periodo de tiempo, considerando diversos factores como prioridades, tiempo de dedicación, disponibilidad de espacio, costo y/o valoración de las consecuencias de tal manera que se puedan satisfacer un conjunto de restricciones fuertes y débiles; las restricciones fuertes se tienen que cumplir en toda circunstancia, mientras que las restricciones débiles representan una mayor flexibilidad, pudiéndose cumplir o no en su totalidad.

En el caso particular de la programación de horarios escolares llamado en inglés timetabling problem, este problema aqueja desde hace varios años a los diseñadores de horarios y a los del área de la teoría de la computación, quienes señalan difícil de modelar dado los recursos disponibles en cada semestre/año y las diferentes formas

de realizarlo. A los diseñadores de horarios no les resulta conveniente utilizar una herramienta que resuelva TTP, ya que una herramienta no puede resolver cualquier TTP, debido a sus variables, prácticas y políticas diferentes donde la construcción del horario se termina realizando de manera manual. En el área de la teoría de la computación señalan al problema TTP como un problema computacional, pudiéndose tratar entonces como un problema de optimización utilizando diferentes propuestas paradigmáticas que evalúen la calidad de las soluciones por medio de una función objetivo, generándose distintas soluciones para los diferentes métodos que dependen de las distintas variables y formas de cuantificar las restricciones a partir de prácticas y políticas diferentes.

Existen diferentes métodos que se han desarrollado y utilizado con éxito para resolver el problema de programación de horarios en departamentos e instituciones específicas, siendo no métodos de carácter universal por lo que una propuesta de solución no puede resolver "cualquier" problema de timetabling. Estas propuestas de solución rara vez se comparan entre sí por gran cantidad de variables diferentes y formas de cuantificar las restricciones que se originan a partir de políticas y prácticas diferentes por lo que cada una de ellas tiene su característica particular de asignación de cursos. La comparación es necesaria para determinar cuáles son los mejores métodos computacionales dados los distintos tipos de datos de horarios, permitiendo descartar las técnicas utilizadas que son relativamente simples.

1.2. Motivación y justificación

En base a lo expuesto anteriormente se justifica la necesidad de una nueva propuesta que permita construir una solución para resolver el TTP de manera integral dada las distintas particularidades y variables de las instituciones por medio de un algoritmo de negociación con Sistemas Multi-Agentes para poder descartar métodos, obteniendo una aproximación importante para la solución de estos problemas al

utilizar las tecnologías de información como apoyo y complemento al desarrollo de problemas computacionales.

Además se contempla el lenguaje de marcado extensible (XML) para representar la información, consintiendo se unifiquen las restricciones dadas en las diferentes instituciones, presentando la ventaja de ser un estándar para el almacenamiento de datos, además de ser útil para las aplicaciones que se comunican entre sí, asimismo de integrar e intercambiar información bajo diferentes plataformas.

Otra razón para justificar, es que con técnicas de la inteligencia artificial distribuida con Sistemas Multi-Agentes se presentan características particulares como la comunicación, cooperación, coordinación e incluso negociación para resolver conflictos, permitiendo un problema modelarse como humanos donde existe un comportamiento global a partir de relaciones locales, el cual conduce aplicar SMA como paradigma de modelación de manera natural adecuándose a distintos problemas haciendo difícil esta tarea para otros métodos. Esta aplicación presenta casos de éxito en el que se ha demostrado científicamente su eficacia al resolver problemas de tiempo real, problemas de coordinación, de actividades, comercio electrónico, programación y planificación.

1.3. Hipótesis

Es posible diseñar e implementar un sistema multi-agente que utilice negociación entre agentes para resolver el problema de programación de horarios escolares de diferentes instituciones mediante la incorpotación de un conjunto generalizado de restricciones.

1.4. Objetivos

1.4.1. Objetivo general

- Diseñar e implementar un algoritmo y un sistema multi-agente para resolver el problema de programación de horarios escolares mediante negociación a través

de la incorporación de restricciones genéricas que permitan modelar diferentes casos de estudio.

1.4.2. Objetivos específicos

- Identificar los diferentes tipos de restricciones.
- Identificar un formato y estructura de datos para la especificación de problemas de programación de horarios.
- Seleccionar una plataforma para desarrollo de SMA.
- Diseñar un algoritmo de solución del problema de programación de horarios mediante SMA y negociación.
- Implementar el algoritmo diseñado en una plataforma para SMA.
- Validar el algoritmo con diferentes casos de estudio.

1.5. Preguntas de investigación

Con fundamento en lo expuesto y para explicar el sistema multi-agentes en la solución del problema de programación de horarios escolares mediante negociación, surgen las siguientes interrogantes de investigación:

- ¿Es posible modelar y resolver el problema de timetabling con sistemas multi-agentes?
- ¿La negociación entre agentes puede llevarse a cabo bajo un enfoque egoísta ó cooperativa?
- ¿Es posible considerar las propuestas de los horarios de los profesores desde un estado inicial y no generar soluciones iníciales por medio de una función de evaluación al validarlas con el profesor?

- ¿Es posible la re-planificación de propuestas cuando las actividades ya han sido asignadas o están a la espera de que el grupo evalué las propuestas?
- ¿Con sistemas multi-agentes es posible la resolución de conflictos?
- ¿El algoritmo funcionara de manera uniforme para todos los casos de estudio?

1.6. Metodología

La investigación que se presenta, se utilizó la metodología cualitativa y cuantitativa. En relación con la metodología cualitativa, se recurrió primeramente a la revisión de diferentes fuentes bibliográficas para documentar el problema de la programación de horarios, así mismo de sus tres diferentes vertientes; programación de horarios escolares, programación de cursos y programación de exámenes, donde en cada uno presenta sus formas sus variantes y los diferentes métodos paradigmáticos que se han implementado.

Así mismo parte de la revisión consistió en conocer las diferentes formas de representar el problema, igualmente analizar los software existentes. En la representación del problema se seleccionó el formato propuesto por FET que representó el mayor conjunto de restricciones y se revisaron los software comerciales y libres que tratan de dar solución a este problema, dada las diferentes restricciones y políticas o prácticas de las diferentes instituciones.

De los diferentes métodos se seleccionó el enfoque distribuido de agentes y se hizo una revisión bibliográfica dando como resultado la existencia de una organización reconocida por la IEEE que promueve estándares de software para agentes y sistemas multi-agentes, teniendo diferentes plataformas que promueven esta tecnología, seleccionándose la de JADE por sus ventajas, seguridad, usabilidad, gran documentación y aceptación por la comunidad científica y empresas.

Una vez efectuada la revisión bibliográfica se realizó el diseño e implementación de un software para sistema multi-agente, consistentes en tres etapas con 3 tipos de agentes; el agente coordinador, grupo y profesor. Se utilizó el protocolo iterativo contract-net por permitir realizar nuevas rondas de negociación en espacios ocupados que anteriormente estaban disponibles, mientras que en el diseño de clases se consideró el patrón de diseño Factory Method debido a que los agentes permiten crear las restricciones ya que de manera inicial no pueden anticipar los tipos de restricciones que estos deben de crear.

En el trabajo de gabinete y experimental se llevaron a cabo los análisis cuantitativos con la selección de dos casos de estudio de Latinoamérica y dos europeos de un total de 22 países para probar el algoritmo, organizando la información en tablas de concentrados con su análisis estadístico enfocado en la variación de conflictos, protocolos iniciados, mensajes y tiempo de ejecución sobre los casos de estudio considerado en el que también se muestran graficas sobre el número de conflictos de manera inicial, los resueltos por el algoritmo y los que no se pudieron resolver.

Capítulo 2
PROGRAMACIÓN DE HORARIOS

Cotidianamente existen actividades que se tienen que realizar en el transcurso del día y en un determinado periodo de tiempo, pero el problema consiste en seleccionar, asignar recursos y tiempo a un conjunto de actividades de una planificación, teniendo en cuenta cumplir con un conjunto de restricciones que reflejan la relación temporal entre las actividades, dada la capacidad limitada de los recursos compartidos.

Los problemas en común detrás de estas asignaciones, se encuentra el TTP, donde en algunos casos consiste formularlo como un problema de búsqueda, tratando de encontrar algún horario que satisfaga todas las restricciones, mientras que en otros casos, el problema se formula de optimización. Esto es, un horario que satisfaga todas los restricciones duras y minimice (o maximice según sea el caso) una función objetivo con las restricciones suaves, aplicando técnicas de optimización a un problema de búsqueda.

Entre los problemas de programación de horarios se encuentra el problema de programación del transporte público, la programación de recursos limitados, la programación de ligas deportivas, la programación de horarios de empleados y la programación de horarios escolares.

En todos los casos, siempre existirá la búsqueda y optimización; en el caso del problema de búsqueda, el inconveniente principal es el mismo, el cual consiste en decidir si existe una solución y en el caso del problema de optimización, decidir si existen una solución con un valor dado en la función objetivo.

Por lo anterior, se dice que el problema principal en la programación de horarios es *NP-completo* en casi todas las variantes. Por lo tanto, una solución exacta es alcanzable solo para un pequeño grupos de casos (*e.g.*, tamaño de entrada de 10 cur-

sos)(Schaerf, 1999), mientras en instancias reales que normalmente pueden involucrar alrededor de cientos de datos de entrada, es necesario el uso de métodos heurísticos, los cuales no garantizan alcanzar la solución óptima.

2.1. Caracterización general del problema

En el caso particular del objeto de estudio en la programación de horarios escolares, en el trabajo de Schmidt and Ströhlein (1980), describen a los "participantes" en el sentido general del término, como maestros, clases, aulas, laboratorios, piezas de equipo, etc. Además, existe un conjunto de "horas" llamadas en ocasiones ranuras o periodos. El término "disponibilidades" se describe para cualquier participante un subconjunto de horas en las cuales una persona o cosa esta libre/disponible para participar ya sea en una clase, materia, conferencia o examen en el que puede estar involucrado y por ultimo describe la noción de un "compromiso" a una colección de participantes de los cuales tienen que reunirse por el número de horas requeridas para ello. Además, se establece el clásico problema de programación de horarios clase-profesor y este se logra si cada compromiso mantiene ocupado exactamente un profesor y en una clase como participante, aún cuando pueden existir demandas para el cumplimiento de algún otro compromiso.

Estas responsabilidades, se logran cuando en la programación de horario se asigna el número preciso de horas requeridas, para que todas estas horas estén disponibles a todos los participantes y tal que, como requerimiento fundamental, ninguno de los participantes sea programado dos veces a la misma hora.

Sin embargo, para la realización de una programación de horarios, según Schmidt and Ströhlein (1980) señalan que existe una enorme diversidad de requerimientos especiales para que la programación de horarios cumpla dependiendo principalmente del tipo de institución y peculiaridades administrativas del país. Porque si no es así, se pueden presentar problemas, por ello se deben considerar las siguientes recomen-

daciones: contemplar el conjunto de horas en días si el ciclo de horas es semanal; en algunos participantes (principalmente las clases), es necesario evitar horas libres entre lecciones; pueden existir horas libres al inicio o al final del día; algunas materias requieren horas consecutivas no separadas por interrupciones; pueden existir limitaciones en la carga diaria del profesor, y puede ser necesario proveer al profesor de un día libre; asignaturas impartidas varias veces a la semana; dos asignaturas con tareas no deben asignarse los sábados y lunes; los profesores pueden indicar una preferencia en intervalo de horas libres entre sus asignaturas. Es claro que, no todas estas restricciones son igualmente importantes y algunas son meramente restricciones estéticas.

Schmidt and Ströhlein (1980), también mencionan que los requerimientos especiales son ligeramente distintos. Mientras en escuelas, el tamaño de una clase de secundaria es de menor interés comparado con las universidades, debido al número de estudiantes en una sesión que varía de uno a tres mil. Normalmente, los salones pueden ser seleccionados de un conjunto de salones de tamaño comparable. Mientras en el problema de la escuela, cada clase debe ser ocupada todo el tiempo, esta condición no es requerida en la universidad. Por otro lado, los requerimientos para la distribución de horas libres a la semana tanto de estudiantes como de profesores es por mucho menos restrictiva.

Las aportaciones de las investigaciones realizadas por Schmidt and Ströhlein (1980), son retomadas por Schaerf (1999), al señalar la existencia de muchos aportes en la literatura relacionada con el problema de programación de horarios, entre ellos se encuentran: Schmidt and Ströhlein (1980) quienes proveen más de 200 publicaciones, listando virtualmente todos los trabajos en el campo que aparecieron hasta 1980. Mientras Werra (1985), provee la definición formal y diferentes formas de formular los problemas de programación de horarios, describiendo también los enfoques más importantes del problema (a la fecha) y destacando los relacionados a la teoría de grafos. Un año después, Carter (1986) provee un estudio acerca de los enfoques en

el análisis del problema de programación de horarios, centra su estudio en los enfoques basados en el problema de reducción de coloreo de grafos. En ese mismo año, Junginger (1986) realiza una investigación en una escuela en Alemania sobre el problema de programación de horarios, en el que describe varios productos de software implementados y su utilización por las instituciones. El trabajo también describe los enfoques subyacentes, los cuales están basados en heurísticas directas. Después, Corne et al. (1994) provee un estudio de la aplicación de los algoritmos genéticos para la programación de horarios y discute las perspectivas de tal enfoque, comparando los resultados obtenidos hasta el momento con respecto a otros enfoques.

Igual de relevante, están las contribuciones de Schaerf (1999) al indicar la existencia de una cantidad de problemas para la programación de horarios en la literatura, los cuales difieren unos de otros basados en el tipo de institución involucrada y tipo de restricciones. Por ello, Schaerf (1999) clasifica los problemas de programación de horarios en tres clases principales a considerar:

- Horarios escolares: La calendarización semanal de todas las clases en una escuela, evitando que profesores tengan dos clases al mismo tiempo, y viceversa.

- Cursos: La calendarización semanal de todas las conferencias de un conjunto de cursos universitarios, minimizando las conferencias al mismo tiempo con estudiantes en común.

- Evaluaciones: La calendarización de evaluaciones de un conjunto de cursos, evitando exámenes al mismo tiempo teniendo estudiantes en común, distribuyendo los exámenes para los estudiantes lo más posible.

Las aportaciones realizadas por Schaerf (1999), al describir su tipología relacionada con las tres clases, hacen una denotación formal y destaca que la presente clasificación no es estricta, en el sentido en el que un mismo problema puede formar parte de una o más clases y es muy difícil precisar una sola clase para el problema.

2.1.1. Programación de horarios escolares

En esta sección se describe con detalle la programación de horarios escolares, conocido como el modelo clase/profesor, seguido con una descripción de una versión simple cada vez que es resuelto en tiempo polinómico por medio de una formulación básica. Posteriormente, se introduce el problema básico de búsqueda y al problema de optimización, describiendo sus variantes consideradas en la literatura.

2.1.1.1. Problema Polinomial Simplificado

Sea $c_1, \ldots, c_m$ m clases, $t_1, \ldots, t_n$ sea n profesores y $1, \ldots, p$ sea p periodos. Dada una matriz de números enteros no negativos $R_{m \times n}$ llamada matriz de requerimientos, donde r_{ij} es el número de materias impartidas por el profesor t_j a la clase c_i.

Schaerf (1999) señala que el problema consiste en asignar actividades(materias) a periodos de tal suerte que ningún profesor o clase se encuentre involucrada en más de una actividad a la vez. Apoyándose en la formulación matemática de acuerdo con Werra (1985), se establece lo siguiente:

$$
\begin{aligned}
\textit{encontrar} \quad & x_{ijk} \quad (i = 1, \ldots, m; j = 1, \ldots, n; k = 1, \ldots, p) \\
\sum_{k=1}^{p} x_{ijk} \quad &= \quad r_{ij} \ (i = 1, \ldots, m; j = 1, \ldots, n) && (1) \\
\sum_{j=1}^{n} x_{ijk} \quad &\leq \quad 1 \ (i = 1, \ldots, m; k = 1, \ldots, p) && (2) \\
\sum_{i=1}^{m} x_{ijk} \quad &\leq \quad 1 \ (j = 1, \ldots, n; k = 1, \ldots, p) && (3) \\
x_{ijk} \quad &= \quad 0 \text{ ó } 1 \ (i = 1, \ldots, m; j = 1, \ldots, n; k = 1, \ldots, p) && (4)
\end{aligned}
$$

donde $x_{ijk} = 1$ si la clase c_i y el profesor t_j es agendado en el periodo k, en caso contrario $x_{ijk} = 0$.

La restricción de la ecuación 1 asegura que el profesor imparte el número correcto de materias a cada clase. La restricción 2 y 3 asegura que cada profesor (clases) esté involucrado en al menos una materia para cada periodo.

Basándose Schaerf (1999) en Even et al. (1975), señala existir siempre una solución a este problema, a menos que el profesor o una clase sea requerido estar involucrado en más de un p materias. Más precisamente, existe una solución si y solo si.

$$\sum_{i=1}^{m} r_{ij} \leq p \ (j = 1, \ldots, n) \tag{5}$$

$$\sum_{j=1}^{n} r_{ij} \leq p \ (i = 1, \ldots, m) \tag{6}$$

Para poder resolver el problema de la programación de horarios, se asocia a una instancia del problema del multígrafo bipartito: Las clases y profesores están asociados a los vértices y cada clase c_i es vinculada a cada profesor t_j por los bordes paralelos r_{ij}. La técnica de solución empleada en Even et al. (1975) se basa en la búsqueda de una secuencia de coincidencia máximas en el multígrafo bipartito, donde una coincidencia es un conjunto de aristas sin nodos comunes.

En cuanto a la complejidad del método demostraron que la coincidencia requerida puede encontrarse en tiempo polinómico con respecto al tamaño de un multígrafo dado que el método de Even et al. (1975) requiere p coincidencias y el tamaño del multígrafo implicado es polinomial n, m, p, donde todo el método se ejecuta en tiempo polinómico.

Alternativamente, según Schaerf (1999), el inconveniente puede ser reducido a un problema de coloreo de grafos conforme a Werra (1985): Dados p colores (cada periodo corresponde a un color), el problema consiste en encontrar una asignación de un color a cada borde de tal manera que no hay dos bordes adyacentes con el mismo color. A partir de entonces, la variable x_{ijk} obtiene el valor de 1 si uno de los bordes entre c_i y t_j tiene el color de k.

Schaerf (1999) apoyándose aún en Werra (1985), considera también algunas variantes del problema de programación de horarios que todavía se pueden resolver en tiempo polinomial. El considera la posibilidad de que un profesor (y clase) puede participar en más de una actividad para cada período. Agrega, en tal variante de un periodo no representa una ranura de tiempo atómico sino a un conjunto de ellos (por ejemplo, un día). También considera el caso en el que las materias están limitadas por lo que deben ser repartidos tanto como sea posible a lo largo de todos los períodos.

2.1.1.2. Problema básico de búsqueda

Schaerf (1999), considera que el TTP no incluye ninguna restricción sobre la programación de las materias. Pero en algunos casos se debe de tomar en la posibilidad de que un profesor (o clase) no esté disponible en un momento dado. Ahora se vuelve un problema de programación de horarios sin disponibilidad de los profesores y clases. La siguiente formulación se debe a Junginger (1986); las alternativas se pueden encontrar por ejemplo en Even et al. (1975):Werra (1985); Garey and Johnson (1990). Según Junginger (1986) apoyándose en los investigadores anteriores, introduce dos matrices binarias $T_{m\times p}$ y $C_{n\times p}$ tal que $t_{ik} = 1$ (resp. $c_{jk} = 1$) si el profesor t_i está disponible en el periodo k y $t_{ik} = 0$ (resp. c_{jk}). A partir de entonces, se sustituye la restricción 2 y 3 en TTP por la restricción 7 y 8 de la siguiente manera:

$$
\begin{aligned}
&\textit{encontrar} \quad x_{ijk} \quad (i = 1, \ldots, m; j = 1, \ldots, n; k = 1, \ldots, p) \\
&\sum_{k=1}^{p} x_{ijk} \quad = \quad r_{ij} \ (i = 1, \ldots, m; j = 1, \ldots, n) \\
&\sum_{j=1}^{n} x_{ijk} \quad \leq \quad t_{ik} \ (i = 1, \ldots, m; k = 1, \ldots, p) \qquad (7) \\
&\sum_{i=1}^{m} x_{ijk} \quad \leq \quad c_{jk} \ (j = 1, \ldots, n; k = 1, \ldots, p) \qquad (8) \\
&x_{ijk} \quad = \quad 0 \text{ ó } 1 \ (i = 1, \ldots, m; j = 1, \ldots, n; k = 1, \ldots, p)
\end{aligned}
$$

Igualmente leyendo a Werra (1985) considera también las restricciones por su pre-asignamiento: Una materia puede ser impuesta para ser programada en un momento dado. Las pre-asignaciones se pueden expresar agregando de un conjunto de restricciones de la siguiente forma:

$$x_{ijk} \geq p_{ijk}(i = 1 \ldots m; j = 1 \ldots n; k = 1 \ldots p) \tag{9}$$

Donde $p_{ijk} = 0$ si no hay asignaciones, y $p_{ijk} = 1$ cuando una materia del profesor t_j a la clase c_i es pre-asignada al período k. Werra (1985) también muestra que la indisponibilidad se puede expresar como pre-asignaciones con clases o profesores vacíos.

Even et al. (1975) al emplear la restricción 7 y 8 demuestran que el problema es *NP-completo* a través de una reducción 3SAT (Garey and Johnson (1990)) para aquellas clases que están siempre disponible y cada profesor disponible para exactamente dos períodos.

2.1.1.3. Problema de optimización

La siguiente función objetivo.

$$\text{mín} \sum_{i=1}^{m} \sum_{j=1}^{n} \sum_{k=1}^{p} d_{ijk} x_{ijk} \tag{10}$$

Donde d_{ijk} es asignado al periodo k en donde la materia del profesor t_j a la clase c_j es menos deseada.

Otro ejemplo para la definición de la función objetivo son los propuesto por Colorni et al. (1993), donde una función objetivo mucho más compleja es definida, la cual incluye muchos aspectos de la programación de horarios escolares. Esta función objetivo está basada en las siguientes cantidades (con peso decreciente):

- El costo didáctico, *e.g.*, el esparcimiento de las clases sobre la semana;

- El costo organizacional, *e.g.*, tener un profesor disponible para posibles puestos temporales;
- El costo personal, *e.g.*, un día libre específico para cada profesor.

Complementa Yoshikawa et al. (1996) al introducir un lenguaje restrictivo y asocia una penalización por cada restricción violada. De acuerdo a los investigadores, el objetivo es minimizar las penalizaciones globales, *e.g.*, considerando la posibilidad de que un profesor sea forzado a impartir su materia en un periodo donde él/ella no se encuentre disponible.

2.1.2. Programación de Cursos

En la programación de cursos Schaerf (1999), por su parte menciona agendar un conjunto de materias para cada curso dentro de un número de salones y periodo de tiempo. Tomando en cuenta que la principal diferencia de la programación de horarios escolares es que en la programación de cursos se pueden tener estudiantes en común, mientras que en horarios escolares son conjuntos de estudiantes disjuntos. El investigador advierte, si dos cursos tienen estudiantes en común entonces existe conflicto, y no pueden ser agendados en el mismo periodo. Agrega, por otra parte, los profesores en escuelas siempre enseñan en más de una clase, mientras que en el problema de cursos, un profesor normalmente imparte solo un curso. Además, la disponibilidad de aulas (y su tamaño) juega un rol importante, mientras que en el problema de horarios escolares esta característica es obviada debido (a la mayoría de los casos) que se asume que cada clase tiene su propia aula.

2.1.2.1. Problema básico de búsqueda

Continuando Schaerf (1999) con Werra (1985), indica que en el problema básico de búsqueda existen q cursos $K_1, \ldots, K_q$, y por cada i, curso K_i consiste de k_i materias. Explica la existencia de r curriculas $S_1, \ldots, S_r$, los cuales son grupos de cursos que

tienen estudiantes en común. Esto significa que los cursos en S_l deben de ser agendados en diferentes horas. Además, el número de periodos p y l_k es el máximo número de materias que pueden ser agendadas por periodo k (*i.e.*, el número de aulas disponibles en el periodo k). Por ello, expone la formulación siguiente:

$$
\begin{aligned}
encontrar \quad & y_{ik} \quad (i = 1, \ldots, q; k = 1, \ldots, p) \\
\sum_{k=1}^{p} y_{ik} &= k_i \ (i = 1, \ldots, q) && (11) \\
\sum_{i=1}^{q} y_{ik} &\leq l_k \ (k = 1, \ldots, p) && (12) \\
\sum_{i \in S_l} y_{ik} &\leq 1 \ (l = 1, \ldots, r; k = 1, \ldots, p) && (13) \\
y_{ik} &= 0 \text{ ó } 1 \ (i = 1, \ldots, q; k = 1, \ldots, p) && (14)
\end{aligned}
$$

donde $y_{ik} = 1$ si una materia de un curso K_i es agendada en el periodo k, de lo contrario $y_{ik} = 0$.

La restricción número 11 impone que cada curso este compuesto por el correcto número de materias, la restricción 12 hace cumplir que cada periodo no existan más materias que aulas, la restricción 13 previene conflictos entre materias a ser agendadas en el mismo periodo. Dicho planteamiento se puede probar que es *NP-completo* mediante una reducción del problema de coloreo de grafosWerra (1985).

Otra forma de plantear el problema es basarse en una matriz de conflicto en lugar de una curricula. La matriz de conflicto $C_{q \times q}$ es una matriz binaria tal que $c_{ij} = 1$ si los cursos K_i y K_j tienen estudiantes en común, de otra forma $c_{ij} = 0$.

2.1.2.2. Problema de Optimización

Así mismo Werra (1985), al problema de búsqueda que hizo mención anteriormente se le puede definir la siguiente función objetivo:

$$\text{máx} \sum_{i=1}^{q} \sum_{k=1}^{p} d_{ik} y_{ik}$$

donde d_{ik} es el grado de deseo de tener la materia del curso K_i en el periodo k.

Una forma de representar este problema es la mencionada por Tripathy (1992), al considera una matriz de conflictos $C_{q\times q}$ con valores enteros, tal que c_{ij} representa el número de estudiantes tomando ambos cursos K_i y K_j. De esta forma, c_{ij} representa también una medida de insatisfacción en caso de que una conferencia de K_i y K_j se encuentren agendadas al mismo tiempo. Completa el investigador, busca minimizar la insatisfacción global obtenida de la suma de todas estas insatisfacciones.

De acuerdo con Schaerf (1999), diversos autores optan por separar los requerimientos en duros (fuertes) y suaves (débiles). Los requerimientos duros son aquellos que tienen que ver con las restricciones y definen el espacio de búsqueda, mientras que los suaves incluyen la función objetivo, generalmente incluyen las restricciones de capacidad del aula y el esparcimiento de las conferencias en el periodo de tiempo, etc.

2.1.3. Programación de exámenes

El presente problema requiere la creación de un horario dado un número de exámenes (uno para cada curso) dentro de un determinado periodo de tiempo, según Schaerf (1999). Este problema es similar al anterior, sin embargo, es posible establecer algunas diferencias ampliamente aceptadas para estos dos problemas. Este problema en particular tiene las siguientes características que lo diferencian de la programación de cursos:

- Existe solo un examen por asignatura.
- Los conflictos son generalmente de carácter estricto, de hecho, es aceptable que un participante sea obligado a saltarse una materia para evitar materias al mismo tiempo, pero no es permitido saltarse un examen.
- Existen diferentes tipos de restricciones, *e.g.*, un solo examen por estudiante, pero no exámenes consecutivos para cada estudiante.

- El número p de periodos puede variar, en contraste con la programación de cursos donde los periodos son fijos.
- Puede haber más de un examen en una sola aula.

2.1.3.1. Problema básico de búsqueda

Este puede ser formulado de forma similar al problema de programación de cursos donde existen q cursos $K_1, \ldots, K_q$, y un examen para cada curso K_i. Existe un grupo r de exámenes $S_1, \ldots, S_r$ tal que en cada S_l existen participantes que deben tomar todos los examenes en S_l. El número de periodos es p y l_k es el número máximo de exámenes que pueden ser agendados en el periodo k (el cual no necesariamente es el número de aulas, debido a que más de un examen puede realizarse en la misma aula).

$$
\begin{aligned}
\textit{encontrar} \quad & y_{ik} \quad (i = 1, \ldots, q; k = 1, \ldots, p) \\
\sum_{k=1}^{p} y_{ik} \quad &= \quad 1 \ (i = 1, \ldots, q) && (15) \\
\sum_{i=1}^{q} y_{ik} \quad &\leq \quad l_k \ (k = 1, \ldots, p) && (16) \\
\sum_{i \in S_l} y_{ik} \quad &\leq \quad 1 \ (l = 1, \ldots, r; k = 1, \ldots, p) && (17) \\
y_{ik} \quad &= \quad 0 \text{ ó } 1 \ (i = 1, \ldots, q; k = 1, \ldots, p) && (18)
\end{aligned}
$$

donde $y_{ik} = 1$ si el examen del curso K_i es agendado en el periodo k, de otra forma $y_{ik} = 0$.

De la misma forma que el problema de programación de cursos, este pertenece a la familia de problemas *NP-completo* mediante su reducción al problema de coloreo de grafos *cf.* Werra (1985).

2.1.3.2. Problema de Optimización

Los tipos más comunes de restricciones suaves en este tipo de problemáticas son aquellos relacionados con las restricciones de segundo orden, es decir, el sistema debe

de evitar que un participante tome dos exámenes en periodos consecutivos. Bajo este objetivo, la siguiente función objetivo para el problema básico de búsqueda definido anteriormente sería:

$$\sum_{k=1}^{p-1} \sum_{l=1}^{r} \sum_{j \in S_l} y_{ik} y_{jk+1}$$

La función anterior cuenta los pares de 1's correspondientes a los exámenes pertenecientes al mismo grupo S_l agendados al periodo siguiente. De hecho, el producto $y_{ik}y_{jk+1}$ tiene como resultado 1 solo si ambos y_{ik} y y_{ik+1} son 1.

Algunos autores como Mehta (1981) consideran la función objetivo el número de estudiantes involucrados en cada conflicto. En cambio Carter et al. (1994) hace una generalización de las restricciones anteriormente descritas y considera una penalización por el hecho que un participante sea forzado a tomar 'x' examen en 'y' periodos consecutivos. Su sistema considera como consecutivo el último periodo del día y el primer periodo del siguiente día y los periodos anteriores y posteriores de un descanso. Por lo tanto, las insatisfacciones de un participante que toma dos exámenes en periodos consecutivos no es la misma. En el mismo periodo, Corne et al. (1994) modifica la función objetivo penalizando (de forma descendente en peso) un participante que toma; a) más de dos exámenes en el mismo día, b) dos exámenes en periodos consecutivos de tiempo, y c) dos exámenes antes y después de un descanso.

Considerando tales premisas con sus restricciones para la problemática en programación de horarios, anteriormente debieron haberse confrontado diferentes posturas que adquieren relevancia temporal, como es el caso de los siguientes planteamientos paradigmáticos.

2.2. Propuestas paradigmáticas

2.2.1. Enfoques paradigmáticos

En relación con la literatura existen un gran conjunto de técnicas de optimización que se han utilizado para resolver el TTP. Sin embargo, los problemas de TTP se han estudiado hace poco tiempo desde la aparición de las meta heurísticas alrededor de 1983. Actualmente es una de las prioridades por el que las ciencias se abren a la temática propia de una ciencia del conocimiento y esto representa un revolucionario "cambio del objeto". A partir de lo anterior, existen varias interpretaciones con respecto al TTP. El tema entra en conflicto entre dos enfoques metodológicos y epistemológicos básicos: el ***reduccionismo*** y el ***heurístico***, expresados particularmente alrededor del estudio de la TTP.

Dentro del ***reduccionismo*** se encuentran tres paradigmas: programación biológica, psicológica y e informática. El reduccionismo biológico, se fundamenta en dos enfoques: el **algoritmo evolutivo** y el **enjambre de abejas**. El **algoritmo evolutivo**, se basa en la optimización de funciones y es una abstracción de la evolución al nivel de las especies, por lo que no se requiere el uso de un operador de recombinación (diferentes especies no se pueden cruzar entre sí). Los algoritmos evolutivos presentan una estructura que puede aplicarse en distintos problemas, favoreciendo así grandemente las tareas de diseño e implementación. Como único requisito del usuario que desee aplicar esta técnica para resolver un problema concreto es saber programar en cualquier lenguaje de propósito general en el que codificaría el algoritmo evolutivo. Dentro de su exigencia está la de obtener buenos resultados con estos algoritmos, el cual es necesario conocerlos en el más mínimo detalle, ya que dentro del esquema general de un algoritmo evolutivo hay que elegir múltiples componentes y parámetros como son la diversificación e intensificación, de los que va a depender la calidad del resultado y la eficiencia del algoritmo. Asimismo, utiliza selección probabilística. También se encuentran las estrategias evolutivas, cuyos métodos computacionales

trabajan con una población de individuos que pertenecen al dominio de los números reales, que mediante los procesos de mutación y de recombinación evolucionan para alcanzar el óptimo de la función objetivo (Colorni et al., 1992; Abramson and Abela, 1992; Fernandes et al., 1999; Bufé et al., 2001; Stefano and Tettamanzi, 2001; Filho and Lorena, 2001; Wilke et al., 2002; Bedoya and Santos, 2003; Yigit, 2007; Beligiannis et al., 2008, 2009; Mohammadi and Lucas, 2008; Nurmi and Kyngas, 2008; Raghavjee and Pillay, 2009, 2010; Srndic et al., 2009).

En cambio, **enjambre de abejas** se funda en el comportamiento colectivo de los sistemas descentralizados auto-organizados, naturales o artificiales. El concepto se emplea en los trabajos sobre inteligencia artificial. Los sistemas de Inteligencia de Enjambres están típicamente formados por una población de agentes simples que interactúan localmente entre ellos y con su medio ambiente. La inspiración proviene a menudo de la naturaleza, especialmente de los sistemas biológicos. Los agentes siguen reglas muy simples, y aunque no existe una estructura de control centralizado que dicta cómo deben comportarse los agentes individuales, locales, y en un grado determinado al azar, las interacciones entre tales agentes conducen a la aparición de "inteligente" comportamiento global. Como ejemplos naturales se incluyen las colonias de hormigas, y enjambre de partículas (Lara et al., 2008).

La posición reduccionista psicológica sostiene la **programación lógica** y las redes neuronales. En el primero, el objeto legítimo de estudio es un tipo de paradigma de programación dentro de la programación declarativa. La mayoría de los lenguajes de programación lógica se basan en la teoría lógica de primer orden, aunque también incorporan algunos comportamientos de orden superior como la lógica difusa. En este sentido, destacan los lenguajes funcionales, ya que se basan en el cálculo lambda, que es la única teoría lógica de orden superior que es demostradamente computable (Kang and White, 1992). Por lo que concierne a las **redes neuronales**, son un paradigma de aprendizaje y procesamiento automático inspirado en la forma en que funciona el sistema nervioso de los animales. Se trata de un sistema de interconexión de neuronas

que colaboran entre sí para producir un estímulo de salida. En inteligencia artificial es frecuente referirse a ellas como redes neuronales. Con un paradigma convencional de programación en ingeniería del software, el objetivo del programador es modelar matemáticamente (con distintos grados de formalismo) el problema en cuestión y posteriormente formular una solución (programa) mediante un algoritmo que permita resolver dicho problema. El algoritmo parte de un conjunto de datos de entrada suficientemente significativo y el objetivo es conseguir que la red aprenda automáticamente las propiedades deseadas. En este sentido, el diseño de la red tiene menos que ver con cuestiones como los flujos de datos y la detección de condiciones, y más que ver con cuestiones tales como la selección del modelo de red, la de las variables a incorporar y el pre-procesamiento de la información que formará el conjunto de entrenamiento. Asimismo, el proceso por el que los parámetros de la red se adecuan a la resolución de cada problema no se denomina genéricamente programación sino que se suele denominar entrenamiento neuronal (Carrasco and Pato, 2004).

Referente a la programación en informática, se consideran dos, entre ellos: **enfoque basado en restricciones** y la **programación entera**. El **enfoque basado en restricciones**, se trata de un paradigma de establecido en la especificación de un conjunto de restricciones, las cuales deben ser satisfechas por cualquier solución del problema planteado, en lugar de especificar los pasos para obtener dicha soluciones un paradigma de la programación en informática, donde las relaciones entre las variables son expresadas en términos de restricciones (ecuaciones). Actualmente es usada como una tecnología de software para la descripción y resolución de problemas combinatorios particularmente difíciles, especialmente en las áreas de planificación y programación de tareas (calendarización). Este paradigma representa uno de los lenguajes de programación desde 1990 y recientemente haya sido identificada por la ACM (Asociación de Maquinaria Computacional) como una dirección estratégica en la investigación en computación (Minton et al., 1993; Meisels et al., 1994; Yoshikawa et al., 1996; Abbas and Tsang, 2001; Marte, 2002; Valouxis and Housos, 2003).

La **programación entera**, esta requiere que la solución óptima se componga de valores enteros para algunas de las variables. La resolución de un problema se obtiene analizando las posibles alternativas de valores enteros de esas variables en un entorno alrededor de la solución obtenida considerando las variables reales. Muchas veces la solución del programa lineal truncado está lejos de ser el óptimo entero, por lo que se hace necesario usar algún algoritmo para hallar esta solución de forma exacta. El más famoso es el método de 'Ramificar y Acotar' o Branch and Bound por su nombre en inglés. El método de Ramificar y Acotar parte de la adición de nuevas restricciones para cada variable de decisión (acotar) que al ser evaluado independientemente (ramificar) lleva al óptimo entero (Lawrie, 1969; Birbas T. and E., 1997; Birbas et al., 2009; Papoutsis K. and E., 2003; Schutt et al., 2010; Boland et al., 2008; Ribić and Konjicija, 2010).

Por su parte, dentro de la posición ***heurística*** se distingue cuatro variantes filosóficas acerca de los problemas perceptuales; los diversos paradigmas representan aproximaciones diferenciadas en la manera de entender la programación de cursos con sistemas multi-agentes, aunque una teoría con respecto a la otra, no están desconectadas entre sí, ya que los autores que las desarrollan parten de elementos o de ideas comunes. En esta perspectiva, se establece la teoría del **Tabú** que hace énfasis en la su resolución un metaheurístico: **recocido simulado** de Abramson (1991); Abramson et al. (1997); Melicio F and A. (2006); Liu et al. (2009); **Grasp** de Vieira et al. (2010) y la del **enfoque distribuido**.

Los aportes de Costa (1994); Alvarez-Valdes R. and M. (1996); Schaerf (1996); Hertz (1991, 1996); Alvarez-Valdés et al. (2002); Santos et al. (2005); Frank Jacobsen and Gehring (2006), constituyen la esencia **Tabú** para un problema de distribución de espacios que habitualmente se presenta en situaciones reales cuando se deben asignar simultáneamente diferentes conjuntos de espacios (despacho, habitaciones, salas, etc.) distribuidos entre edificios y/o plantas entre varios grupos de personas de tal forma que se minimicen las distancias entre los espacios signados a cada grupo y la sede

de dicho grupo: esta situación da lugar a un problema combinatorio con una función objetiva cuadrática, lo cual complica enormemente su resolución mediante un método exacto.

En la segunda variante, está la posición del **recocido simulado** que pone énfasis en el estudio un algoritmo de aproximación a la solución óptima, y se basa en una analogía del comportamiento de sistemas termodinámicos simples. Esta técnica se utiliza en la búsqueda de soluciones a problemas de optimización de gran tamaño, desde el punto de vista práctico se ha usado para resolver el problema del viajero. También, se ha empleado en problemas prácticos como el diseño automático de circuitos integrados, análisis sintáctico del lenguaje, simulación del funcionamiento neuronal, comunicaciones, teoría de grafos, predicción de estructuras de cromosomas, clasificación de imágenes y diseño de circuitos electrónicos. La simulación del proceso puede usarse para describir la generación de soluciones de un problema de optimización combinatoria, en donde conforme el proceso cambia (azar) y teniendo como resultado la mejor solución. Las soluciones de un problema de optimización combinatoria son equivalentes a los estados de un sistema físico. El costo de una solución es equivalente a la energía de un estado (Abramson, 1991; Melicio F and A., 2006; Liu et al., 2009).

La teoría **Grasp** (Greede Randomized Adaptive Search Procedures), de uno de sus exponentes como son Vieira et al. (2010), tiene un antecedente en el principio básico de asignación de responsabilidades. Indica, por ejemplo, que la responsabilidad de la creación de un objeto o la implementación de un método, debe recaer sobre la clase que conoce toda la información necesaria para crearlo. De este modo se obtiene un diseño con mayor cohesión y así la información se mantiene encapsulada (disminución del acoplamiento).El hecho de mantener un encapsulado, los objetos se benefician por utilizar su propia información para llevar a cabo sus tareas. Se distribuye el comportamiento entre las clases que contienen la información requerida. Son más fáciles de entender y mantener. Siempre que se tenga que llevar a cabo una responsabilidad que dependa del tipo, se tiene que hacer uso del polimorfismo, cuando

las alternativas o comportamientos relacionados varían según el tipo (clase), asigne la responsabilidad para el comportamiento, utilizando operaciones polimórficas a los tipos para los que varía el comportamiento. Asigna el mismo nombre a servicios en diferentes objetos. La fabricación pura se da en las clases que no representan un ente u objeto real del dominio del problema, sino que se ha creado intencionadamente para disminuir el acoplamiento, aumentar la cohesión y/o potenciar la reutilización del código. Es la solución cuando el diseñador se encuentre con una clase poco cohesiva y no tenga otra clase en la que implementar algunos métodos. Es decir que es una clase "inventada" o que no existe en el problema como tal, pero que añadiéndola se logra mejorar estructuralmente el sistema. Como contraindicación deberemos mencionar que al abusar de este patrón suelen aparecer clases función o algoritmo (que tienen un solo método).

Contrario a las anteriores posiciones heurística, está el **enfoque distribuido**. En esta variante, según Šlechta (2005) y Obit et al. (2011), encuentran más elementos que permiten demostrar que con técnicas distribuidas el TTP presentan una buena forma de dividir el problema en sub-tareas. Así mismo, Obit et al. (2011) mencionan que la implementación con tecnología de agentes, un problema puede ser modelado como humanos, adecuándose a tantas soluciones como sea posible permitiendo modelar casos particulares de la programación de horarios donde existan profesores impartiendo clase en diferentes escuelas, haciendo difícil esta tarea para otros métodos.

Además, los agentes también han presentado casos de éxito en el que se ha demostrado científicamente su eficacia en resolver problemas tiempo real, problema de coordinación, actividades, comercio electrónico, programación y planificación.

2.2.2. Enfoques distribuidos

Gaspero et al. (2004), propone una resolución de manera distribuida al problema de la programación de cursos con sistemas multi-agentes una vez que está construido el horario, presentando los problemas de asignar los grupos cuando cambian de aula,

las restricciones del caso de estudio no se especifican, además de que el mecanismo de negociación no se describe. Un año después por parte de Šlechta (2005) de igual manera plantea un modelo para descomponer el problema de horarios de secundaria en sub-problemas y resolver cada sub-problema en paralelo por medio de un algoritmo de descomposición usado para dividir un grafo, en sub-grafos que presentan sub-problemas ejecutándose en una maquina diferente, no explicando las restricciones y el modo de resolver los conflictos.

Por parte de Yang et al. (2004, 2006) presentan dos trabajos con sistemas multi-agentes. En el primero utilizan dos agentes móviles con cuyo comportamiento es verificar diariamente los conflictos, no especificando las restricciones y la obtención de una mejor solución, mientras que en el trabajo dos cambia solo en que los agentes representan las restricciones fuertes, no presentando una abstracción completa al problema.

Para Kingston (2007, 2010), presenta un enfoque jerárquico que combina de forma recursiva horarios más pequeños en un calendario más grande. Así mismo, tres años más tarde Kingston (2010) extiende el problema de horarios escolares empleando el modelo de procesamiento paralelo 'coarseg rained' y facilitando el intercambio de casos y la creación independiente de soluciones múltiples en paralelo, no especificando en ambos las restricciones y el mejoramiento de una solución.

Finalmente Obit et al. (2011) aborda el problema de programación de cursos con sistemas multi-agentes desde un enfoque distribuido, sin especificar las restricciones y la forma de resolver conflictos. Sin duda, en todos los trabajos, la técnica distribuida con SMA ha quedado limitada en cuanto a las especificaciones del problema que se está resolviendo, además de que el mecanismo de negociación, resolución de conflictos y las restricciones no se especifica.

2.3. Herramientas y estándares

2.3.1. Análisis de las herramientas existentes

Existen diferentes métodos que se han desarrollado y utilizado para resolver el TTP como se describe en la sección de propuestas paradigmáticas, sin embargo, sólo se utilizan en un departamento o institución en particular debido a sus variables, reglas y restricciones específicas. Por lo tanto, un programa escrito para programar el horario de una escuela difícilmente se puede utilizar para uno diferente dado a las características particulares de cada problema que se originan a partir de políticas y prácticas diferentes, teniendo cada método características particulares de asignación por lo que rara vez se comparan entre sí.

La comparación es necesaria para determinar cuáles son los mejores métodos computacionales dados los distintos tipos de datos de horarios y por ello decir que técnicas utilizadas son simples y fáciles de reproducir. Otro problema son los requisitos específicos en donde solamente son conocidos por los expertos del área que realizan la programación de horarios, donde rara vez se realizan por escrito de manera universal para que otros programadores puedan leer y entender. Por ello, el intercambio de información entre los investigadores es insipiente.

Ante esto, se hace conveniente en la búsqueda de soluciones tener un problema estándar que incluya un súper conjunto de las restricciones para un conjunto de escuelas con el fin de escribir programas universales. Esto facilita la comparación de diferentes metodologías para la solución del problema de horarios escolares y promover aún más el desarrollo del dominio del horario de la escuela.

Aprovechando a la vez la existencia de repositorios con variedad de datos de programación de horarios escolares al público para probar y comparar el desempeño de diferentes metodologías. Además, la comunidad de investigación también ha investigado el uso de un lenguaje o formato de datos uniforme para expresar problemas de horarios escolares y por lo tanto facilitar el intercambio de problemas entre los

investigadores.

2.3.2. Estructura de representación de los datos del problema

La investigación de la programación de horarios escolares carece de la disponibilidad de puntos de referencia intercambiables en un formato uniforme. Por esta razón un grupo de investigadores acordó utilizar el estándar XML para el intercambio de sus bases de datos. El objetivo principal de esta iniciativa es proporcionar un formato estándar para expresar los conjuntos de datos a los problemas de horarios escolares, facilitando así el uso público de estos problemas. Reis and Oliveira (2001) presentan el lenguaje UniLang para definir la escuela, cursos universitarios y los problemas de horarios de exámenes. UniLang ofrece un evaluador para probar si un calendario determinado cumple todos los requisitos y las limitaciones del problema definido. Más recientemente, Wilke and Ostler (2010) presentan un formato XML para los dos horarios y problemas de programación.

Kingston (2001) creó un lenguaje llamado STTL para especificar los problemas de programación de horarios para escuelas secundarias y evaluar soluciones a estos problemas. Post et al. (2012), después adopta un enfoque similar con un formato de datos XML con su propio tipo de especificaciones y estructura llamada XHSTT para facilitar el intercambio de datos y así promover la investigación en este ámbito para las escuelas secundarias. Estos casos de estudio de 16 países se encuentran localizados en un repositorio para el acceso al público con un evaluador de los casos de estudio se encuentra sin funcionamiento, además que las restricciones están limitadas y generalizadas.

FET (2013) presenta un formato también en XML con su propia estructura y extensión llamada .FET (2013) con diferentes casos de estudio de 22 países localizados en un repositorio de manera gratuita. FET (2013) presenta de manera específica los diferentes tipos de restricciones con su representación además de servir como herramienta para captura del problema.

2.4. Software existente

En la actualidad existe software que trata de resolver el TTP, este software se encuentra de manera comercial y de manera libre donde cada uno utiliza sus propias técnicas de optimización y estructura de restricciones, mientras que otros simplemente como herramienta de apoyo para la asignación de los horarios de manera manual (Véase Tabla 2.1).

Dentro del software comercial se encuentra a aSc Applied Software Consultants (2013), este permite realizar la programación de horarios de manera automática en el cual no especifica el método de búsqueda y sus restricciones. El software Mimosa Software Ltd. (2013) permite la programación de horarios para escuelas, reuniones, cursos y planificación de conferencias de manera automática sin especificar el método de solución y restricciones.

Tabla 2.1: Comparativa de Software programación de programación de horarios

Nombre	**Licencia**	**Tipo de Programación**	**Tipo de Método**	**Estructura de BD**	**Accesible**	**Especificación de las restricciones**
aSCTimeTables	Comercial	Automático	Sin especificar	Propio	No	No
Mimosa	Comercial	Automático	Sin especificar	Propio	No	No
FET Free Timetabling Software	Libre	Automático	Enjambre de particulas	Propio	Si	Si
Fedena 2.2 Released	Libre	Automático	Sin especificar	Propio	No	No
Imagic Timetable Software	Libre	Automático	Sin especificar	Propio	No	No
Mimosa 12 Scheduling Software	Libre	Automático	Sin especificar	Propio	No	No
openSIS Community Edition	Libre	Automático	Sin especificar	Propio	No	No
TimeFinder Software	Libre	Automático	Sin especificar	Propio	No	No
Lantiv Scheduling Studio 7	Libre	Manual	No aplica	Propio	No	No
University Timetabling Comprehensive Academic Timetabling Solution	Libre	Automático	Busqueda Local	Propio	No	No

Por otra parte en el software libre se encuentra FET (2013) con licencia GNU GPL, que permite realizar la programación de horarios de manera automática con enjambre de partículas especificando su estructura del problema y restricciones en formato XML. iMagic Software (2013) permite realizar la programación de horarios de manera automática sin especificar el método de búsqueda para su solución y sin especificar sus restricciones. Mimosa Software Ltd. (2013), presenta también una versión libre con el tipo de solución de manera automática sin especificar el método de solución y el formato de sus restricciones. OpenSIS (2013) permite la programación de horarios, además de herramienta para la información de calificaciones e inscripciones de los estudiantes sin especificar el método de solución y formato de restricciones. Time Finder (2013) emplea un algoritmo de optimización el cual no se especifica al igual que tampoco se especifican las restricciones. Lantiv (2013) es un software que sirve como herramienta para realizar la programación de horarios de manera manual sin especificar la estructura de las restricciones. En cambio, UniTime (2013) es de código abierto con licencia GNU, permite la programación de horarios de exámenes y utiliza la minimización de los conflictos utilizando búsqueda local con la programación de restricciones (variables, valores y limitaciones) sin especificar su formato de restricciones.

Capítulo 3
SISTEMAS MULTI-AGENTE

3.1. Origen de los agentes

Aun cuando se apoyan en las investigaciones de Reis and Oliveira (2001); Wilke and Ostler (2010); Kingston (2001) y FET (2013) en los formatos de datos y software existente mencionados anteriormente, se sigue planteando resolver problemas que son difíciles o imposibles de resolver para un agente individual. Por ello, se establece el dominio del sistema multi-agente o de inteligencia artificial (IA) quien ha tenido como objetivo el desarrollo de software para simular las llamadas capacidades inteligentes de los seres humanos, tales como el razonamiento, comunicación del lenguaje natural y el aprendizaje: con este tipo de programas, el rol de la computadora se ha convertido en un instrumento para convertirse progresivamente en una especie de asistente para los seres humanos.

En cierto modo, Tweedale et al. (2007) menciona que a principios y finales de 1990 la IA presentaba poca interacción entre los seres humanos para la resolución de problemas dando a luz un nuevo campo denominado Inteligencia Artificial Distribuida (Distributed Artificial Intelligence –DAI-) que incluye la tecnología de agentes con el fin de formar inteligencia a partir de interacción. De acuerdo a esta posición, Durfee et al. (1989) menciona que la DAI se encarga de investigar los modelos de conocimiento, así como la comunicación y las técnicas de razonamiento para que los agentes puedan interactuar y coordinar en sociedad con el fin de resolver un problema o un grupo de tareas de manera eficiente.

El notable interés en la tecnología de agentes desde mediados de los años 1980 ha renovado el interés y aún más la consolidación de la DAI y es que desde el momento en que apareció la idea de agente, un tema de estudio tan importante en la inteligencia artificial como lo ha sido desde siempre la representación del conocimiento y el aprendizaje (Moulin and Chaib-draa, 1996), poniendo énfasis en tres características: En primer lugar la necesidad de trabajar con conocimiento distribuido con aplicaciones que están distribuidas geográficamente; el segundo en tratar de extender la cooperación hombre-máquina con un método basado en la resolución distribuida entre el hombre y la máquina; por último, el DAI aporta una nueva perspectiva en la representación del conocimiento y resolución de problemas, proporcionando formulaciones científicas más ricas y una representación que en la práctica es más realista.

3.2. Agente

Uno de los primeros términos de agente corresponde al trabajo realizado por Minsky (1986), titulado "Society of Mind", en su trabajo relaciona a una colección de agentes o sistemas multi-agentes (SMA), que agentes individuales. Asimismo, menciona que la inteligencia humana puede ser construida a partir entre partes simples llamado agentes dando además cierta credibilidad a la teoría de que en los humanos, la toma de decisiones puede ser modelada eficientemente, usando técnicas de simulación con SMA.

Un hecho relevante a nivel mundial en sus investigaciones, es que el termino agente, según Wooldridge and Jennings (1995) por ser polisémico presenta diferentes conceptos básicos para una variedad de sub-disciplinas como la IA en computación, en ingeniería de software, comunicaciones de datos, investigaciones de sistemas concurrentes, así como la robótica e incluso en la programación orientada por lo que definir el término de agente no resulta del todo posible.

Siguen exponiendo Wooldridge and Jennings (1995) hechos precisos al definir agente, como aquella entidad actuante y con un papel activo que conlleva a acciones que afectan su entorno teniendo autonomía y racionalidad. La autonomía, generalmente sin la intervención humana o guía directa y la racionalidad en el sentido de pseudo-teoría de juegos en el que un agente maximiza su rendimiento con respecto a una "función de evaluación".

En cambio en 1996, Russell et al. (1996) dentro de sus aportaciones, establecen cuatro tipos de agentes que encarnan los principios que subyacen en casi todos los sistemas inteligentes:

- Reactivo Simple: Es el tipo de agente más sencillo. Este agente selecciona las acciones sobre la base de las percepciones actuales, ignorando el resto de las percepciones históricas. Como dice su nombre tienen la propiedad de ser simples con una inteligencia muy limitada.

- Basado en Modelos: Este tipo de agente para manejar una efectiva visibilidad parcial, es almacenar información de las partes del mundo que no pueden ver. Es decir mantener algún tipo de estado interno que depende de la historia percibida y que de ese modo depende por lo menos algunos de los aspectos no observables del estado actual.

- Basado en Objetivo: Debido que el conocimiento sobre el estado actual del mundo no es siempre suficiente para decidir qué hacer. Además de descripción del estado actual, el agente necesita algún tipo de información sobre su meta que describa las situaciones que son deseables.

- Basado en Utilidad: Las metas por sí solas no son realmente suficientes para generar comportamiento de gran calidad en la mayoría de los entornos. Una función de utilidad proyecta un estado (o una secuencia de estados) en un número real, que representa un nivel de felicidad. La definición completa de

> una función de utilidad permite tomar decisiones racionales en dos tipos de casos en los que las metas son inadecuadas. Primero cuando haya objetivos conflictivos, y sólo se puedan alcanzar algunos de ellos (por ejemplo, costo-beneficio), la función de utilidad determina el equilibrio adecuado. Segundo, cuando haya varios objetivos por los que se pueda guiar el agente, y ninguno de ellos se pueda alcanzar con certeza, la utilidad proporciona un mecanismo para ponderar la probabilidad de éxito de función de la importancia de los objetivos.

Nwana (1996) en sus indagaciones sobre lo polisémico del término, lo define como una entidad computacional que actúa en comportamiento de otras entidades de manera autónoma, realizando acciones con un cierto nivel de pro-actividad y/o reactividad y además de presentar algún nivel de los atributos clave del aprendizaje, cooperación y movilidad. Pero, uno de los términos de agentes más utilizado es por parte de Russell et al. (1996), que establecen: "Un agente es cualquier cosa que sea capaz de percibir su ambiente a través de sensores y actuar en él a través de efectores".

Entre las características de los agentes es que estos presentan actitudes de información (creencia y conocimiento y pro-actividad, deseo, intención, obligación, compromiso y elección). Además de autonomía, otro aspecto importante de la conducta de un agente es el grado de pro-actividad y reactividad y finalmente los agentes también exhiben algún nivel de atributos principales como aprender, cooperar y movilidad.

Posteriormente, Jennings and Wooldridge (2000) define a un agente como un sistema de computación encapsulado que se sitúa en algún ambiente y que es capaz de actuar en forma flexible y autónoma en ese ambiente para alcanzar los objetivos de su diseño.

Por otro lado, existen enfoques que se le dan mayor peso al aspecto social de los agentes, por ejemplo, según Woolridge and Wooldridge (2001) "un agente es un hardware o, más comúnmente, un sistema de cómputo basado en software, que posee las siguientes propiedades:

- Autonomía: Los agentes operan sin la intervención directa de humanos u otros, y tienen algún tipo de control sobre sus acciones y su estado interno;
- Aptitud social: Los agentes interactúan con otros agentes (y posiblemente con humanos) vía algún tipo de lenguaje de comunicación;
- Reactividad: Los agentes perciben su ambiente (el cual puede ser el mundo físico, un usuario a través de una interfaz gráfica, una colección de otros agentes, el internet o una combinación que ocurren en él);
- Pro actividad: Los agentes no actúan simplemente en respuesta a su ambiente, sino que tienen la capacidad de exhibir comportamiento dirigido a objetivos, llevando la iniciativa".

3.3. Descripción de sistemas multi-agentes

La resolución de problema realizada por agentes en un sistema multi-agente (SMA) se denomina Solución de Problemas Distribuidos e implica la investigación en las áreas de coordinación, negociación y comunicación. Para que un SMA resuelva problemas comunes con coherencia, los agentes deben coordinarse entre sí, coordinar sus actividades y negociar una vez que se encuentra en conflicto. Los conflictos pueden resultar del simple contenido limitado de recursos a más complejos cálculos basados en computación donde los agentes están en desacuerdo debido a las discrepancias entre sus ámbitos de competencias. Es necesaria la coordinación para determinar la estructura de la organización entre un grupo de agentes, la asignación de recursos y negociación para detección y resolución de conflictos.

Al igual que con el término agente, es difícil encontrar también una definición formal del término SMA. En el trabajo de Huhns and Stephens (1999) establecen las siguientes características para ambientes multi-agentes:

- Deben contar con comunicación y protocolos de interacción.
- Son capaces de organizarse de manera autónoma.
- Contienen agentes distribuidos y autónomos, que pueden ser cooperativos o egoístas.

La definición de SMA que da Ferber (1999) indica que comprende los siguientes elementos: ambiente, objetos, agentes, relaciones, operaciones y leyes. Mientras que el término de SMA, Weiss (1999) lo define como un sistema en el cual varios agentes inteligentes interactúan y persiguen algún conjunto de metas o realizan algún conjunto de tareas.

Ante esto, para Weiss (1999) señala que los SMA han demostrado ser útiles para:

- Resolver problemas que son demasiado grandes para un solo agente, por los recursos que consumen.
- Resolver problemas en los que resulta arriesgado tener centralizado el sistema en un solo agente.
- Interconectar sistemas que ya existen, tales como sistemas expertos y sistemas para toma de decisiones.
- Ofrecer soluciones a problemas inherentes distribuidos, como el control de tráfico aéreo.
- Ofrecer soluciones que emanan de fuentes de información distribuida.
- Ofrecer soluciones donde el conocimiento está distribuido.
- Incrementar la velocidad de ejecución, confianza y extensibilidad de las soluciones.
- Ofrecer claridad y sencillez en el diseño.

3.3.1. Características de los sistemas multi-agentes

A todos los agentes racionales se les puede caracterizar como poseedores de una función de utilidad. Así pues, un agente que tiene una función de utilidad explícita, está en posibilidad de tomar decisiones racionales, aunque quizás tenga que comparar las utilidades obtenidas mediante diversas acciones. Las metas, no obstante permiten al agente optar de inmediato por una acción, cuando esta permite ser alcanzada.

Además, en algunos casos, se puede traducir la función de utilidad en un conjunto de metas, de manera que las decisiones adoptadas por el agente basado en metas, tome como base tal conjunto de metas y resulte idéntica a las que haría el agente basado en la utilidad. Algunas técnicas a las que haría el agente basado en la utilidad en incrementar su utilidad, constan de seis fases a considerar: coordinación, cooperación, negociación, conducta coherente, planificación y comunicación.

La **coordinación**, es una cuestión importante dentro de los SMA, es como se deben organizar los agentes para promover la colaboración entre ellos, de acuerdo a Genesereth and Ketchpel (1994) estas son posibles por medio de comunicación directa y coordinación asistida. En la coordinación directa los agentes controlan su propia coordinación, la arquitectura emplea este tipo de enfoque por medio de una red de contratos (contract-net) y la repartición de especificaciones (specification sharing). En la primera, los agentes necesitan servicios, distribuyen peticiones de propuestas a otros agentes. Los agentes receptores evalúan las propuestas y mandan sus ofertas a los agentes originadores. Los originadores deciden que ofertas aceptar, y establecen contratos con los elegidos. En la repartición de especificaciones, los agentes informan a otros agentes sus capacidades y necesidades, por lo cual pueden coordinar sus actividades. Las desventajas de las comunicaciones directas son su alto costo (sobre todo cuando el número de agentes en el sistema es muy grande) y las dificultades de implantación, pues cada agente debe incluir código para habilitar la coordinación.

En la coordinación asistida, los agentes se apoyan en otros sistemas para lograr su coordinación, como son los sistemas de federación. En estos sistemas, los agentes no se comunican directamente entre sí, sino lo hacen solo con sistemas especiales, llamados *facilitadores* que se comunican entre sí. En un sistema de federación los agentes utilizan un ACL (Lenguaje de Comunicación entre Agentes por sus siglas en ingles) para comunicar sus capacidades y necesidades a sus facilitadores, así como información a nivel aplicación. También reciben en ACL información a nivel aplicación proveniente de los facilitadores. Los facilitadores utilizan la información proporcionada por sus agentes, para transformar los mensajes de aplicación y encaminarlos a los destinos adecuados. La desventaja de la coordinación asistida, es que los agentes de aplicación dependen de otros sistemas para lograr su interacción; además, están sujetos a las capacidades de estos.

Por otra parte Mintzberg (1979), considera tres tipos de mecanismos fundamentales de coordinación: Ajuste mutuo, supervisión directa y estandarización. El ajuste mutuo logra una coordinación de los trabajos por el simple proceso de la comunicación informal. Esta ocurre donde quiera que haya dos o más agentes acordando compartir recursos para alcanzar una meta común. El segundo tipo de coordinación llamado supervisión directa logra la coordinación al tener una persona encargada que asuma la responsabilidad por el trabajo de los demás, dando instrucciones a los mismos y el control de sus acciones. Por último, la coordinación estandarizada se logra desde un principio en el que el coordinador impone sus reglas bajo ciertas circunstancias de las cuales los demás saben exactamente que esperar de ellos y actuar en consecuencia.

Por supuesto, la coordinación es importante para los SMA y para la resolución de problemas distribuidos. Sin coordinación, un grupo de agentes puede rápidamente colapsar, en una caótica colección de individuos ya que, un agente solo tiene una vista parcial e imprecisa del sistema completo, y sus acciones pueden interferir en vez de ayudar a las acciones de otros agentes.

En el caso de la **cooperación**, punto central para el desarrollo de SMA es la cooperación. Durfee et al. (1989) propone cuatro metas genéricas para la cooperación entre un grupo de agentes: implementar la razón de terminación de tareas a través del paralelismo; incrementar el conjunto o ámbitos de tareas alcanzables por la compartición de recursos (información, dispositivos físicos, etc.) incrementar la probabilidad de realizar tareas completas encargándose de las tareas duplicadas, posiblemente con diferentes métodos de realización de esas tareas; decrementando la interferencia entre tareas, evitando interacciones dañinas.

La cooperación total es aplicada en el estudio de resolución de problemas distribuidos (CDPS). Durfee et al. (1989), mencionan el espacio donde los agentes pueden trabajar juntos en una red para resolver problemas que están más allá de sus capacidades individuales. En esta red, cada agente es capaz de resolver problemas complejos y trabajar independientemente; el problema es que los agentes no pueden estar completamente sin cooperación total. La cooperación total es necesaria porque un agente sencillo no tiene el suficiente conocimiento y recursos para resolver un problema dado. Generalmente, los agentes CDPS construyen cooperativamente una solución al problema, mediante el uso de su conocimiento local y de sus recursos para resolver dichos problemas; luego, integran estas soluciones de sub-problemas dentro de una solución total.

La tercera fase conocida como negociación, juega también un papel importante en las actividades de los SMA, además de las actividades cooperativas de los agentes, permitiendo resolver conflictos que podrían interferir con la construcción de soluciones. Durfee et al. (1989) definen la negociación como un proceso de acuerdo mejorado (reduciendo la inconsistencia e incertidumbre) sobre puntos de vista comunes o planes a través de la estructura de intercambio de información relevante.

Uno de los protocolos de negociación más estudiados para la negociación es el protocolo de red de contratos (contract net protocol) es una de las aproximaciones que más ha influido en la negociación propuesta por SMA. Esta fue inspirada por un

proceso de contratos en la negociación humana. Los agentes coordinan sus actividades a través de contratos, para completar metas específicas.

En el caso de la **conducta coherente** global significa que las acciones de los agentes, acercan al sistema a alcanzar las metas comunes del grupo entero. La coherencia global puede ser implementada por el mejoramiento solo en las habilidades sociales del agente, mejorando solo las políticas de comunicación o por una combinación de ambas. En cada uno de estos casos, la decisión más coherente resulta si un agente tiene una mejor comprensión de lo que ya se ha hecho y de las actividades e intenciones de otros agentes.

Acerca de la **planificación**, Bond and Gasser (1988) indican que se puede alcanzar la meta global, mediante la alineación de las conductas de los agentes con divisiones explicitas de labores. Técnicas tales como planeación centralizada para múltiples agentes, plan de reconciliación, planeación distribuida y análisis organizacional; son formas de ayudar a ordenar las actividades de los agentes y asignar tareas, teniendo en cuenta las consecuencias de hacer estas tareas en un orden particular.

En la planeación centralizada multi-agente, un agente es responsable de construir un plan multi-agente que especifique todas las acciones de los agentes. Debido a ello, Georgeff and Lansky (1987) mencionan que implementar una planeación centralizada multi-agente se plantea formar primero agentes individuales; luego plantear un agente central que una los planes locales y los analice para identificar conflictos potenciales. Este agente trata de resolver el conflicto, modificando el plan local e introduciendo comandos de comunicación dentro de ellos. Así los agentes son sincronizados apropiadamente.

En la sexta fase implicada a la **comunicación**, los agentes pueden interactuar a través de acciones lingüísticas explicitas (tal como la comunicación) o a través de acciones no lingüísticas, modificando el mundo en el que actúan: la comunicación habilita a los agentes para intercambiar información y coordinar sus actividades.

3.4. Plataforma de agentes

Existe una organización llamada Fundación de Agentes Físicos Inteligentes (Foundation for Intelligent Physical Agents -FIPA-) para producir estándares de software para agentes heterogéneos y sistemas basados en agentes. En la elaboración de estas normas, la construcción de las especificaciones se utiliza para lograr la interoperabilidad entre los sistemas basados en agentes desarrollados por diferentes empresas y organizaciones. La organización FIPA pertenece a la IEEE Computer Society Standards por sus estándares, interoperabilidad y desarrollo de software que promueve sus normas con la tecnología de agentes con otras tecnologías (FIPA, 2012).

La plataforma de agentes proporciona la infraestructura física en la cual los agentes se pueden desarrollar. La plataforma de agentes (PA) consiste en máquina(s), sistema operativo, software de soporte a agentes, componentes de administración de agentes FIPA (DF, AMS, MTS) y agentes.

3.5. Modelo de referencia FIPA

El modelo de referencia FIPA provee el marco normativo de trabajo dentro del cual, los agentes existen y operan, también se establece los alcances de referencia lógicos para la creación, registro, ubicación, migración y retiro de agentes FIPA AMS (2001).

El modelo de referencia considera una Plataforma de Agentes (PA) como un conjunto de cuatro componentes cuyos nombres aparecen abreviados por sus siglas en inglés: Agente, Sistema de Administración de Agentes (AMS), Facilitador de Directorios (DF) y Servicio de Transporte de Mensajes (MTS). Los primeros tres componentes son tipos especiales de agentes que dan soporte a la administración de agentes, mientras que el MTS provee un servicio de entrega de mensajes. La funcionalidad de estos elementos está descrita en la especificación en FIPA (FIPA AMS, 2001).

El diseño interno de un PA es un problema para los desarrolladores de sistemas y no un tema de estándares dentro de FIPA y en donde se muestran las entidades contenidas en el modelo de referencia (Véase Figura 3.1), entre ellas: software externo, agente, sistema de administración de agentes, facilitadores de directorio y el sistema de transporte de mensajes.

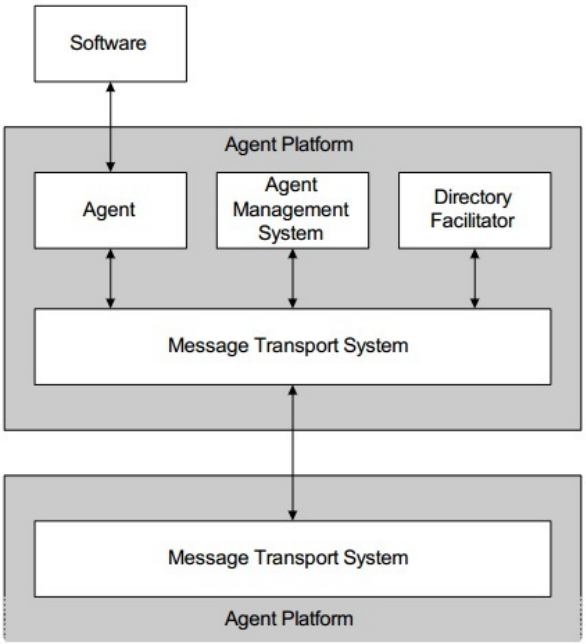


Figura 3.1: Modelo de Referencia de FIPA.

En este entorno, el **software** son todos aquellos sistemas que no tienen las características de agentes, pero que son utilizados por los agentes para llevar a cabo sus tareas accesibles desde un dominio a través de un agente. Los agentes pueden acceder al software, por ejemplo:

- Añadir nuevos servicios.
- Adquirir nuevos protocolos de comunicaciones.
- Adquirir nuevos protocolos o algoritmos de seguridad.

- Adquirir nuevos protocolos de negociación, etc.

Los **agentes** son la parte principal de una plataforma del modelo de referencia. Un agente encapsula uno o más servicios, dentro de un modelo unificado e integrado de ejecución. FIPA mantiene un concepto abierto de lo que es un agente, para poder incluir un gran número de aplicaciones de agentes, y no limitar la forma en la que son implementados. Los agentes se comunican utilizando un lenguaje de comunicación de agente (ACL) de modo que se puede distinguir sin ambigüedades en el universo agente por medio de un identificador (AID), además puede ser registrado en un número de direcciones de transporte a la que puede ser contactado. Un agente combina uno o más servicios, dentro de un modelo de ejecución integrado y unificado. Este modelo puede incluir acceso a software externo, usuarios humanos y facilidades de comunicación.

Con respecto al **sistema de administración de agentes** (AMS), es un componente obligatorio de la PA entorno en el que puede existir solo un AMS en la PA. Al Sistema de Administración de Agentes corresponde la responsabilidad de administrar las actividades de un PA. Estas responsabilidades incluyen borrar y crear agentes con el fin de tener un AID, decidir si un agente puede registrarse dinámicamente a la plataforma y supervisar la migración de agentes hacia y desde otras plataformas. Así, su función compleja deriva en que el AMS, mantiene una lista de todos los agentes que residen actualmente sobre la plataforma. La lista incluye un identificador único global (GUID) y la dirección de transporte de cada agente.

Los **facilitadores de directorio** (DF) proporcionan servicios de sección amarilla con otros agentes. El DF es el encargado de mantener un directorio de agentes confiable, en el sentido de que mantiene actualizada una lista precisa y completa de los agentes; proporciona la información acerca de los agentes de su directorio a todos los agentes autorizados. El DF no controla el ciclo de vida interno de los agentes.

Los agentes pueden registrar sus servicios con el DF o consultar cuales servicios son ofrecidos por otros agentes. Debe existir por lo menos un DF en cada PA, sin embargo, una PA puede incluir cualquier número de ellos. Los AMS, ACC y DF; pueden registrar sus servicios, en múltiples DFs. La pertenencia a un directorio DF define un dominio de agentes. Un dominio es un espacio lógico dentro del cual, los agentes se pueden organizar y localizar unos a otros. Una PA puede soportar varios dominios y a su vez, un dominio puede extenderse a varias PA.

En relación al **sistema de transporte de mensajes** (MTS), este proporciona un mecanismo para la transferencia ACL de mensajes entre los agentes. Los agentes implicados pueden ser locales en una PA o en diferentes PAs. En cualquier PA dado, el MTS es proporcionado por un canal de comunicación de agentes (ACC). Cualquier MTP puede utilizar una representación interna diferente para describir un mensaje, pero debe expresarse bajo los mismos términos, representando la misma semántica.

Las siguientes son declaraciones generales sobre la forma de un mensaje:

- Un mensaje consiste en un conjunto de parámetros.
- Un parámetro es un par nombre/valor.
- Un mensaje puede contener parámetros opcionales.

Cada vez que de lanza un mensaje ACC puede añadir nueva información al mensaje, pero nunca se puede sobrescribir la información existente. ACC puede añadir nuevos parámetros a un mensaje que se sobrescribe con el mismo nombre del parámetro, el mecanismo para mensajes sin ambigüedad de entradas es especifica por cada sintaxis de mensaje completo.

3.5.1. Ciclo de vida de la plataforma

Considerando el modelo de referencia, se contempla que los agentes existen físicamente en una plataforma y utilizan las facilidades ofrecidas por esta para llevar a

cabo sus propias funciones. En particular, un agente como un proceso de software, tiene un ciclo de vida físico que tiene que ser administrado por la plataforma. Para cada agente, este ciclo de vida físico y los estados asociados pueden ser diferentes del ciclo de vida lógico externo y los estados en el dominio, los cuales son administrados por el DF.

En una plataforma dada, el ciclo de vida de un agente tiene las siguientes propiedades:

1. Limitado a la plataforma: Un agente es administrado físicamente dentro de una plataforma. Esta circunstancia determina que el ciclo de vida de un agente, esté, por tanto, siempre limitado a una plataforma específica.

2. Independiente de la aplicación: Puesto que define solamente los estados y transiciones del agente, en su ciclo de vida.

3. Orientado a Instancias: Se asume que el agente descrito en el modelo del ciclo de vida, es una instancia (un agente que tiene un nombre único y es ejecutado en forma independiente).

4. Unicidad: Diferente al ciclo de vida en un dominio, donde un agente puede tener estados diferentes en diferentes dominios al mismo tiempo; cada agente tiene solamente un ciclo de vida en la plataforma y dentro de una sola plataforma.

Para eso, el ciclo de vida de un agente en la plataforma, se representa por estados (círculos) y transiciones (flechas) (Véase Figura 3.2). En las Tablas 3.1 y 3.2, se muestran las descripciones de las transiciones y estados respectivamente.

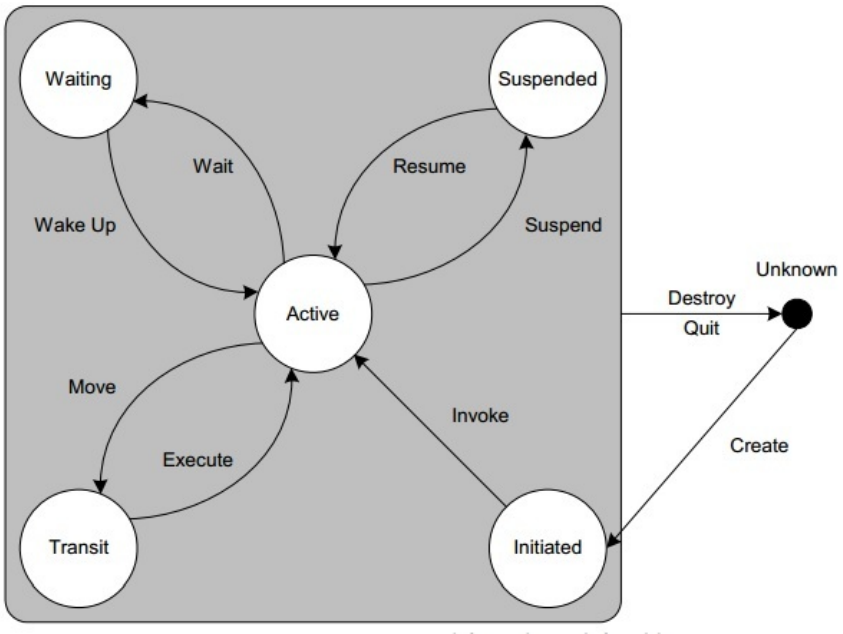


Figura 3.2: Ciclo de vida de los agentes, AMS.

Tabla 3.1: Transiciones del ciclo de Vida en la Plataforma

Estado	**Descripción**
Create	La creación (instalación) de un nuevo agente
Invoke	La invocación de un agente.
Start	El inicio o reinicio de la operación de un agente.
Suspend	Suspensión de la operación de un agente, ya sea por el AMS o por solicitud del agente mismo
Resume	Activación de un agente suspendido. Soló puede ejecutarla el AMS.
Wait	El agente está en un estado de espera para ciertos eventos. Es diferente a la acción *suspend* ya que sólo puede ser iniciado por el agente mismo.
Wake Up	Despertar del agente de un estado de espera. Este sólo puede ser iniciada por el AMS
Destroy	Terminación forzosa de la ejecución de un agente. Sólo pede ser iniciado por el agente
Quit	Indica al agente que debe terminar su ejecución apropiadamente, y puede ser ignorada por él
Move	Pone al agente en su modo “transitorio". Sólo puede ser iniciado por el agente
	Continúa en la página siguiente.

Tabla 3.1 – Continuación de la página anterior

Estado	Descripción
Execute	Regresa al agente del modo “transitorio", y sólo puede ser ejecutado por el AMS.

Tabla 3.2: Estados del ciclo de vida en la plataforma, FIPA AMS (2001)

Estado	Descripción
Unknown	El agente no existe en realidad dentro de la plataforma, el mecanismo de entrega de mensajes puede rechazar los mensajes para un agente en este estado, o almacenarlo para cuando el agente esté en estado Active. Esto dependerá de las políticas de la plataforma.
Initiated	El agente ha sido creado o ha llegado a la plataforma. El MTS puede almacenar sus mensajes hasta que vuelva al estado active, o puede enviárselos a otra dirección (sólo si se ha registrado otra).
Active	El agente está trabajando sobre la PA, y recibe mensajes del MTS de forma normal.
Suspended	La ejecución del agente ha sido suspendida, ya sea por el AMS, o por el agente mismo. El manejo de los mensajes de idéntico a Initiated.
Waiting	El agente está esperando cierto evento, *e.g.* la llegada de mensajes ACL, o de otros eventos de administración. La entrega de mensajes se comporta igual que en Initiated.
Transit	Sólo existe en plataformas que soportan movilidad de agentes. El agente se encuentra en camino a una plataforma distinta. En esa circunstancia, el mecanismo de entrega de mensajes puede almacenar sus mensajes hasta que el agente llegue a la otra plataforma y registre su nueva dirección, o hasta que vuelva a la plataforma; puede también enviárselos a otra dirección conocida.

3.5.2. Comunicación y lenguajes de comunicación

Como se señala anteriormente, la comunicación juega un papel importante dentro de las características de los SMA. En este sentido, Finin et al. (1994) mencionan

que el problema de comunicación entre SMA se puede descomponer en tres aspectos: protocolos de interacción, protocolos de transporte y lenguajes de comunicación. El protocolo de interacción se refiere a la estrategia de alto nivel efectuada por los agentes que controla su interacción con otros agentes. Los protocolos de interacción se identifican como patrones preestablecidos de conversación en el cual se describen más adelante.

El protocolo de transporte es realmente el mecanismo de transporte que se usa para comunicarse, tal como TCP, SMTP o HTTP. Green et al. (1997) indican que los Lenguajes de Comunicación entre Agentes (ACL por sus siglas en inglés) se pueden dividir en *ad hoc* y estándares. Los primeros son hechos a la medida para aplicaciones específicas. Algunos de ellos ni siquiera tienen realmente el grado de lenguaje, sino que utilizan estructuras de datos para pasar información. Este tipo de lenguajes son empleados por muchas aplicaciones, pero su desventaja es que dificultan mucho la comunicación entre agentes construidos por diferentes desarrolladores.

Aunque actualmente existen varios estándares ACL, dos dominan la escena: El ACL que forma parte de la especificación de FIPA (FIPA ACL) y el Knowledge Query and Manipulation Language (KQML)- El FIPA-ACL es el lenguaje que se basa la implementación.

Para el caso del KQML, este se desarrolló como parte de un proyecto de la Defense Adanced Research Projects agency (DARPA) y compuesto de tres capas: la capa de contenido, la capa de mensajes y la capa de comunicación. La primera encierra el contenido real del mensaje, la segunda la constituye el conjunto de acciones que los agentes pueden ejecutar al comunicarse (performatives) y la última contiene el protocolo para la entrega del contenido.

3.5.2.1. FIPA Agent Comunication Languages (ACL)

En el año 1995, la fundación de agentes físicos e inteligentes (FIPA) comenzó su trabajo en el desarrollo de normas para los sistemas de agente. La clave central de

esta iniciativa fue la elaboración de una lista ACL (FIPA ACL, 2002). Esta ACL es similar al KQML; por definir un lenguaje 'exterior' para los mensajes. Además, la sintaxis concreta para los mensajes FIPA ACL se asemeja mucho al de KQML.

El rango de dominios al que puede pertenecer el contenido de un mensaje no está restringido; de hecho, no se obliga a ajustarse a ningún lenguaje de contenido en particular. Sin embargo, se asume que los agentes que conversan entre sí son capaces de darle una interpretación correcta a los mensajes que intercambian (no se contempla el problema de cómo compartir una ontología). Por otro lado, sí se define un conjunto de tipos de mensaje y su significado, independientemente del contenido.

La idea fundamental de la especificación, es: los mensajes representan actos comunicativos. La especificación FIPA, tanto una definición formal (basada en las actitudes mentales de los agentes, y utilizando lógica modal de primer orden con identidad) como una descripción informal de los actos comunicativos.

En lo que se refiere a la utilización del lenguaje ACL, se imponen ciertas reglas a los agentes:

1. Los agentes están obligados a implementar todo el conjunto de actos comunicativos.

2. Los agentes deben enviar un acto *not-understood*, como respuesta a un mensaje que no reconozcan o que no sean capaces de procesar. De manera analógica, deben estar preparados para recibir un *not-understood* como respuesta a un mensaje que hayan enviado.

3. Si un agente decide implementar un acto que aparece en la especificación, entonces debe hacerlo de acuerdo a la especificación.

4. Los agentes pueden implementar actos que no están incluidos en la especificación, siempre que no tengan el mismo significado que uno que ya exista. En el

caso de definir nuevos actos, deben responsabilizarse de que los agentes que los reciban, le den la interpretación correcta.

5. Los agentes deben ser capaces de generar mensajes sintácticamente bien formados, que correspondan a los actos que desean mandar. De la misma manera, deben ser capaces de interpretar correctamente un mensaje sintácticamente bien formado.

En la ejecución de dichas normas, FIPA ACL (2002) recomienda tener presente los nombre de parámetros de los mensajes y su función (Véase Tabla 3.3).

Tabla 3.3: Parámetros de los mensajes FIPA ACL(FIPA ACL (2002)).

Parámetro	**Descripción**
performative	Indica el acto comunicativo.
sender	Identifica al agente remitente del mensaje.
receiver	Denota al agente al que se le pretende entregar el mensaje.
reply-to	Indica a quien contestar el mensaje.
content	Describe el contenido del mensaje.
language	Denota el lenguaje en el que está escrito el contenido del mensaje.
encoding	Descripción del campo Contect.
ontology	Especifica la ontología utilizada, para darle sentido al contenido del mensaje.
protocol	Especifica el protocolo empleado por el remitente. Los protocolos se describen más adelante en esta sección del trabajo.
converstion-id	Denota una expresión que se usa, para identificar la secuencia de mensajes que forman una conversación.
reply-with	Establece un identificador para que el receptor lo utilice con el parámetro in-reply-to al contestar. Útil para seguir una conversación cuando existen varios diálogos.
in-reply-to	Denota un identificador, que hace referencia a una petición de la que el mensaje es respuesta (ver reply-with).
reply-by	Indica el tiempo que se esperará la respuesta

3.5.2.2. Catálogos de actos comunicativos

Para el catálogo de actos comunicativos, existen los mensajes FIPA-ACL que contienen los siguientes campos que se muestran los actos comunicativos con su respectiva descripción (Véase Tabla 3.4).

Tabla 3.4: Catálogo de Actos Comunicativos (CALS (2002)).

Acto Comunicativo	**Contenido del mensaje**	**Descripción**
accept-proposal	Una acción por efectuarse y una propuesta a las condiciones del acuerdo.	Aceptación de una propuesta antes recibida (generalmente vía un acto *propose*). El agente que manda la aceptación, informa al receptor que el espera que el receptor ejecute la acción, una vez cumplidas las condiciones.
agree	La acción a efectuar y las condiciones del acuerdo	Se realiza acuerdo sobre una petición antes recibida para ejecutar una acción. El remitente informa al receptor que acepta ejecutar acción, pero hasta que se cumplan las condiciones.
cancel	La acción a cancelar	El remitente cancela una acción previamente requerida, que está ejecutando o va a ejecutar el receptor. Si la acción ya se ejecutó, el remitente del cancel recibirá un mensaje refuse.
cfp	La acción a ejecutar y las precondiciones de la acción	CFP (Call For Proposals) indica negociación al solicitar propuestas para ejecutar una acción. Normalmente, el agente que contesta un CFP manda sus condiciones, las cuales deben ser compatibles con las especificadas en el CFP.
confirm	Una proposición lógica	El remitente informa al destinatario que la proposición es verdadera. Se asume: el remitente cree que así es, intenta que el destinatario también lo crea, y que el destinatario tiene incertidumbre sobre esa verdad.
disconfirm	Una proposición lógica	El remitente informa al destinatario que la proposición es falsa. Se asume que remitente cree que así es, intenta que el destinatario también lo crea, y que el destinatario tiene duda o cree la verdad de la proposición.
failure	La acción que falló y la razón de la falla	Informa que la acción era factible de realizarse, pero que no se completó por alguna razón.
inform	Una proporsición lógica	El remitente informa al destinatario que la proposición es verdadera. Se asume que el remitente cree que así es, he intenta que destinatario también lo crea, y que el destinatario no tiene conocimiento sobre la verdad de la proposición.
inform-if	Una proposición lógica	Informa la proposición si el remitente cree que es verdadera, en otro caso informa la negociación de la proposición.

Continúa en la página siguiente.

Tabla 3.4 – Continuación de la página anterior

Acto Comunicativo	**Contenido del mensaje**	**Descripción**
inform-ref	Una descripción de un objeto	El remitente informa al destinatario el objeto que corresponde a un descriptor.
not-understood	Una acción y una razón	El remitente informa al destinatario que aun que percibió que realizó una acción, no entiende lo que hizo. El caso más común es que remitente avisa al destinatario que no entiende el mensaje que previamente le envió.
propagate		El remitente tiene la intención de que el receptor trate el mensaje incrustado como enviado directamente al receptor y quiere que el receptor identifique los agentes denotados por el descriptor dado y envié el mensaje recibido propagado por ellos.
Propose	La acción y las pre- condiciones ejecutadas	El remitente propone ejecutar cierta acción cuando sean ciertas las precondiciones.
proxy		El remitente quiere que el remitente seleccione los agentes destinatarios indicados por la descripción dada y para enviar un mensaje incorporado a ellos.
query-if	Una proposición lógica	El remitente pide al destinatario que le informe si la proposición es verdadera. Se asume que el remitente no lo sabe, y cree que el destinatario sí.
query-ref	Un descriptor de objeto	Remitente solicita al destinatario que le informe qué objeto corresponde al descriptor.
refuse	Acción rechazada y la razón	Negación del remitente a ejecutar la acción, pues no se cumplieron las condiciones pre-establecida.
reject-proposal	Propuesta de acción, sus pre-condiciones, y razón de rechazo	El remitente informa al destinatario que no tiene ninguna intención de que el destinatario ejecute la acción dada, bajo las pre-condiciones dadas.
request	Una acción	El remitente pide al destinatario que ejecute la acción dada. Uso importante de esto, es cuando la acción es a su vez un acto comunicativo.
request-when	Una acción y sus pre-condiciones	El remitente pide al destinatario que ejecute la acción dada cuando se cumplan las pre-condiciones dadas.
request-whenever	Una acción y sus pre-condiciones	El remitente pide al destinatario que ejecute la acción dada, cuando y cada vez que se cumplan las precondiciones dadas.
subscribe	Un descriptor de objeto	El destinatario pide ser informado constantemente de los cambios que le sucedan al objeto cancel. Identificado por el descriptor. La suscripción se da por terminada con un acto.

3.5.2.3. Protocolos de interacción

La mayoría de las conversaciones que sostienen los agentes siguen ciertos patrones. Inician de modo similar y en determinado momento se puede inferir qué tipo de actos sigue. Por ejemplo, después de un request, es común que siga un *not-understood*, un *agree* o un *refuse*, pero no es usual que siga un subscribe. A estos patrones de conversación se les llama protocolos FIPA IPLS (2000).

Un diseñador de aplicaciones de agentes, puede decidir hacer a los agentes lo suficientemente conscientes del significado de los mensajes, sus objetivos, creencias y otros estados mentales que el agente posea, de tal forma que el proceso de planeación cause que los protocolos de interacción emerjan espontáneamente de las decisiones del agente. Esto, sin embargo, implica una gran carga de capacidades y complejidad en la implementación del agente. Una forma alternativa y muy pragmática, es predefinir los protocolos de interacción para que los agentes implementados en forma sencilla puedan involucrarse en conversaciones que tengan sentido con otros agentes, simplemente apegándose al protocolo conocido (Véase Tabla 3.5).

Tabla 3.5: Protocolos de interacción

Protocolo	Descripción
Blokering	Está diseñado para apoyar en las iteraciones de intermediaciones en los sistemas de mediación y sistemas multi-agente.
Contract Net	Permite el rechazar o aceptar actos comunicativos de negociación. Este permite negociar con los participantes que proponen sus propuestas.
Dutch Auction	Subasta holandesa donde el subastador trata de encontrar un buen precio de mercado, empezando con una licitación con un precio mucho más alto que el valor de mercado esperado, entonces se reduce de manera progresiva el precio hasta que uno de los compradores acepta el precio.
English Auction	Subasta inglesa donde el subastador trata de encontrar un buen precio de mercado proponiendo un precio inferior al del valor del mercado y luego subir gradualmente el precio.

Continúa en la página siguiente.

Tabla 3.5 – Continuación de la página anterior

Protocolo	Descripción
Iterated Contract Net	Es una extensión del protocolo contract net, pero la diferencia es que permite múltiples ronda de licitaciones.
Propose	Permite a un agente proponer a los agentes de recepción que el iniciador hará las acciones descritas en el acto propose.
Query	Permite a un agente solicitar la realización de una acción de otro agente.
Recruiting	Está diseñado para apoyar en las interacciones de reclutamiento en los sistemas de mediación y en los sistemas multi-agente.
Request When	Permite a un agente solicitar la realización de una acción en un momento dado.
Subscribe	Permite a un agente solicitar otro agente (receptor) realizar una acción en la suscripción cuando se cambia el objeto de referencia.

El apegarse al uso de protocolos trae ventajas a los desarrolladores de agentes, pues pueden enfocarse a implementar, sólo aquellos actos involucrados en los protocolos, en lugar de considerar todo el espectro de actos. Se cuenta con varios protocolos predefinidos, los desarrolladores pueden crear sus propios protocolos, y no se obliga a los agentes a implementar ninguno de los protocolos predefinidos, tampoco se les prohíbe usar otros protocolos. Sin embargo, si un agente usa un protocolo predefinido, debe hacerlo de manera consistente a como está especificado.

3.6. Plataforma para el desarrollo de SMA

Una plataforma de agentes proporciona la estructura física o ambiente donde los agentes se implantan para alcanzar sus metas. FIPA presenta una lista de las principales implementaciones de desarrollo SMA de acceso al público (FIPA, 2012).

Para el caso de las características, se menciona primeramente a **Agent Development Kit (ADK)** que es Tryllian's Agent Development Kit (ADK), además es una plataforma de desarrollo en Java con una arquitectura JXTA-based P2P basado en la comunicación de mensajes XML y SOAP, servicio de directorio JNDI (FIPA ACL,

2002), utilizando un entorno de ejecución Java fundada en componentes móviles que permite la construcción, desarrollo, administración de aplicaciones distribuidas a gran escala que operan independientemente de la ubicación, el ambiente o protocolo. Esto permite a las empresas responder de forma dinámica a los retos complejos, construcción confiable y escalable. El ADK permite a los desarrolladores comunicarse con los componentes de software de agente que no necesariamente están controladas en el mundo del agente, tales como bases de datos, motores de búsqueda, sistemas de correo, servidores Web y aplicaciones J2EE (Tryllian, 2013).

April Agent Platform (AAP), es una plataforma de agentes FIPA-Compliant que está se encuentra desactualizada y fue diseñada para ser una solución ligera y de gran alcance para el desarrollo de sistemas basados en agentes. El lenguaje de programación april y el sistema de comunicación InterAgent (IMC), proporciona características para acelerar el desarrollo y el despliegue de agentes y plataformas de agentes (FIPA Agent Platform, 2012).

La plataforma **CAPNET** (Component Agent Platform base on .NET) no propia de FIPA pero cumple completamente con las especificaciones, implementado como 100 % código .NET en el marco de Microsoft .NET Framework y Compact Framework escrito en el lenguaje C# para la plataforma Windows, tales como Servicios Web (WS) y comunicación remota. CAPNET contiene una infraestructura integrada que cubre la programación, implementación, administración y la integración con las aplicaciones de escritorio, dispositivos móviles, servicios web, entre otros que consiste en un entorno de tiempo de ejecución que admite la implementación del SMA, entorno de desarrollo en forma de plantillas de agentes, herramientas de programación, galería de componentes y algunos conectores para permitir la integración con las aplicaciones empresariales (Contreras et al., 2004).

En el caso de **Comtec Agent Platform**, plataforma de agentes realizada por la agencia de promoción de la información y tecnología, Japón y la tecnología de comunicación es de código abierto basada en agentes de comunicación FIPA, agentes

administrativo, agente de transporte de mensajes y algunas de las aplicaciones que requieren SL2 como el idioma de los contenidos de código abierto escrito en Java JDK 1,2 o mayor con licencia GPL de código abierto (FIPA Agent Platform, 2012).

Para **FIPA-OS**, esta plataforma se encuentra desactualizada y fue la primera implementación Open Source de FIPA y con miles de descargas. Para la construcción de FIPA-OS contribuyeron diferentes desarrolladores de todo el mundo con numerosas correcciones de errores y mejoras con un total de 10 lanzamientos. FIPA-OS 2 es un kit de herramientas basado en componentes implementados en 100 % Java, uno de los aportes más significativos recibidos es una versión a pequeña de la FIPA-OS destinada a PDAs y teléfonos móviles inteligentes, que ha sido desarrollado por la Universidad de Helsinki como parte del proyecto IST Crumpet (FIPA Agent Platform, 2012).

Las particularidades de **Grasshopper**, indican una plataforma desactualizada de código abierto 100 % en Java de agentes inteligentes móviles compatible con los estándares OMG MASIF y especificaciones FIPA. Grasshopper es la primera plataforma de agentes móviles que cumplen con las especificaciones de MASIF al ser una extensión de código abierto proporcionado el OMG y las interfaces para agente/plataforma FIPA. Se distribuye comercialmente por IKV++, una empresa de Berlín. Grasshopper soporta varios protocolos de transporte mediante el uso de un ORB interno: un protocolo propietario basado en TCP/IP, Java RMI, CORBA IIOP, MAF IIOP, TCP/IP con SSL y RMI con SSL. La plataforma de soporte integral para la seguridad, la comunicación y persistencia del agente (FIPA Agent Platform, 2012).

JACK Intelligent Agent, desarrollada por Agente Orientado a Software Pty. Ltd (AOS) en Melbourne Australia presenta un software comercial versión 5 (Intelligent Agent Development Toolkit) que permite el desarrollo de agentes en java, compatible con la arquitectura Belief-Desire-Intention (BDI) o bien seguido el modelo PRS. Jack comenzó con una aplicación de dMARS en Java, pero se ha diseñado con un enfoque en la extensibilidad y con el objetivo de apoyar la no-BDI, así como la arquitectura BDI, el framework de agentes inteligentes JACK por Software Orientado

Agente trae el concepto de agentes inteligentes de la ingeniería de software comercial y Java. Los objetivos principales de Jack son ofrecer a los desarrolladores un robusto producto estable para satisfacer una amplia variedad de necesidades prácticas, facilitar la transferencia de tecnología desde la investigación a la industria, y para permitir la investigación aplicada. El entorno de ejecución JACK proporciona un modelo de ejecución en el que un agente (o capacidad) responde a los eventos de los cuales tiene un plan. JACK se caracteriza por un conjunto de herramientas de desarrollo gráfico que ayudan en la visualización y desarrollo de los diseños y de las interacciones de los componentes. JACK está equipado con un entorno gráfico JACK Development Environment (JDE) (FIPA Agent Platform, 2012).

Java Agent Services API, los Servicios de Agente de Java (JAS), es una plataforma de agentes desactualizada cuyo proyecto se define de una industria estándar y una API que proporciona interfaces para la creación de mensajes, codificación de mensajes, transporte de mensajes, directorios y nombres para el despliegue de agentes de servicios de la plataforma de infraestructura. Esta plataforma se implementó de la Arquitectura de FIPA en el Java Community Process donde la iniciativa dio lugar a la intención de formar la base para crear aplicaciones comerciales basadas en las especificaciones de FIPA. JAS es una plataforma abierta que apoya el plug-in de la plataforma tecnológica de servicios a terceros. Este diseño tiene por objetivo garantizar que un sistema basado en el despliegue JAS siga transparente a los cambios en la tecnología sin causar interrupciones en la prestación de servicios y por lo tanto el proceso de negocio (FIPA Agent Platform, 2012).

En relación al distintivo de **LEAP**, que viene siendo una Plataforma Ligera de Agente Extensible (LEAD), es un entorno de desarrollo ya desactualizada que permitía ejecutar entornos de ejecución derivados de JADE, implementados en dispositivos (ordenadores, PDA y teléfonos móviles) con mecanismo de comunicación TCP/IP y WAP (FIPA Agent Platform, 2012).

En relación a **ZEUS**, quien tiene una plataforma de agentes desactualizada de código abierto implementado en Java, desarrollado por los Laboratorios BT y considerado como un conjunto de herramientas de colaboración para la construcción de aplicaciones multi-agente. Por otra parte, Zeus proporciona facilidades para el apoyo a las comunicaciones que utilizan los agentes FIPA ACL como el transporte de mensajes y sockets TCP/IP como el mecanismo de entrega. Zeus también proporciona facilidades para la construcción de agentes (FIPA Agent Platform, 2012). ZEUS provee un conjunto de componentes de software y herramientas que son usadas para el diseño, desarrollo y organización de los agentes, además provee una herramienta de ejecución que permite a las aplicaciones ser observadas y otro asistente de herramientas como: reportes, estadística y visualizador de agentes.

Por último, **JADE** (Java Agent Development Framework), desarrollado por el laboratorio Telecom en Italia en julio de 1998, en el 2000 fue liberado como código abierto, además de cumplir las especiaciones FIPA. La plataforma JADE puede ser distribuido en varios equipos, además los agentes pueden tener diferentes comportamientos concurrentes y migrar o clonarse así mismo a otros hosts de la plataforma, sin importar el sistema operativo. El ciclo de vida de los agentes puede ser controlado remotamente a través de una interfaz gráfica de usuario, que también permite herramientas de depuración. JADE se implementa en la versión 1,2 de Java y no tiene la dependencia sobre el software de terceros. La última versión de JADE es JADE 4,3,1 liberada el 12 de junio del 2013 (JADE, 2013).

Capítulo 4
DISEÑO E IMPLEMENTACIÓN DE LA SOLUCIÓN

4.1. Representación de las restricciones

Debido a la tabla comparativa del software existente que se mostró en el capítulo II, se consideró la estructura y diseño de FET debido a que se encuentra disponible, tiene una estructura definida, es de fácil acceso al público con una gran cantidad de casos de estudio de diferentes países, además de ser portable y contar con una gran cantidad de restricciones dividida en cuatro grupos llamados:

- Restricciones de tiempo
- Restricciones de espacio
- Restricciones de actividades
- Otros tipos de restricciones

En las restricciones de tiempo se tiene las horas no disponibles, máximo de días por semana, máximo de espacios libres sin asignación entre actividades por semana, máximo de espacios libres sin asignación entre actividades por día, máxima de horas diarias, mínimo de horas diarias y máximo de horas continuas (Véase Tabla 4.1). Las horas no disponibles corresponde al día con hora donde no debe de existir alguna actividad tanto para el profesor o el grupo, la restricción máximo de días por semana son los días permitidos por semana que puede impartir clase un profesor, a varios profesores, a un grupo o a varios grupos para tener actividades, el máximo de espacios sin asignación por día o semana son el total de espacios no asignados permitidos entre

actividades asignadas, el máximo y mínimo de horas diarias permitidas que pueden tener los profesores o grupos, y por último el máximo de horas continuas permitidas que pueden ser asignadas para profesores, grupos en específico o de manera general a todos los profesores o grupos que pertenecen en la institución.

Tabla 4.1: Restricciones de tiempo

Restricción de tiempo	**Profesor**	**Estudiante**	**Profesores**	**Estudiantes**
No disponible	X	X		
Máximo de días por semana	X	X		
Máximo de espacios libres sin asignación entre actividades por semana	X	X	X	X
Máximo de espacios libres sin asignación entre actividades por día	X	X	X	X
Máximo de horas diarias	X	X	X	X
Mínimo de horas diarias	X	X	X	X
Máximo de horas continuas	X	X	X	X

Las restricciones de espacio son donde las clases tienen su sede, también se tiene; el máximo de cambio de edificios por día, máximo de cambio de edificio por semana y mínimo de espacios libres sin asignación entre cambios de edificios para un profesor o grupo. El aula base es donde un profesor, profesores, grupo o varios grupos tienen su sede para las clases, mientras que la restricción de máximo cambio de edificio por día o semana son las veces que tiene permitida los profesores o grupos de manera individual o de manera general de moverse entre los edificios y por último, el mínimo de espacios sin asignar entre cambio de edificio (Véase Tabla 4.2).

Tabla 4.2: Restricciones de espacio

Restricciones de espacio	**Profesor**	**Estudiante**	**Profesores**	**Estudiantes**
Aula base	X	X	X	X
Máximo cambio de edificios por día	X	X	X	X
Máximo cambio de edificios por semana	X	X	X	X
Mínimo de espacios libres sin asignación entre cambio de edificios	X	X	X	X

Para las restricciones de actividades es necesario conocer como está compuesta una actividad. Una actividad se compone por un profesor, materia, grupo, duración, total de duración, un identificador, un identificador de actividad de grupo y un comentario, el cual un profesor tiene conjunto de actividades que están vinculadas con el grupo correspondiente, de tal manera que un conjunto de actividades componen un horario (Véase Tabla 4.3).

Tabla 4.3: Atributos de una actividad

Profesor
Materia
Grupo
Duración
Total de duración
Id
Id de actividad de grupo
Comentarios

Dados los atributos de una actividad, es posible asignar restricciones a las mismas ya sea de manera individual o grupal, dentro de las restricciones de actividades que se muestran en la Tabla 4.4, se tiene la restricción de tiempo de inicio preferido, actividad con intervalos de tiempo preferido, actividades con mínimo de días, actividad con la misma hora de inicio, actividades con el mismo día de inicio, dos actividades se

encuentran ordenadas, actividades sobre puestas, mínimo de espacios libres entre actividades y dos actividades consecutivas.

Tabla 4.4: Restricciones de actividades

Restricciones de tiempo para actividades	
Actividad de tiempo inicio preferido	Actividades con el mismo día de inicio
2 actividades consecutivas	2 actividades ordenadas
Actividades con mínimo de días	Conjunto de actividades no sobre puestas
Actividades con la misma hora de inicio	Mínimo de espacios libres entre actividades

La restricción de tiempo de inicio preferido se refiere a que una actividad tiene preferencia de impartirse temprano, las actividades con el mismo día de inicio como se indica se tiene que impartir actividades en el mismo día que con frecuencia depende de la asignatura, las actividades consecutivas son actividades que están una seguida con otra a diferencia de las ordenadas que puede estar en la mañana y la otra en la tarde sin importan que existan actividades de por medio. Actividades con el número de días se refiera a que un conjunto de actividades distribuidas a lo largo de la semana, las actividades que no se encuentren sobre puestas indica que una actividad no se encuentre en el mismo día con la misma hora al mismo grupo, actividades con la misma hora de inicio tal como su nombre lo indica y la de mínimo de espacios libres entre actividades con frecuencia los espacios libres sin asignación para permitir a los grupos movilidad.

Los otros tipos de restricciones corresponden a las restricciones que no entran en una clasificación de las anteriores pero de igual manera son importantes, dentro de otros tipos de restricciones se tienen las restricciones de tiempo que corresponde a que un profesor no puede dar clase en dos o más actividades al mismo tiempo, al igual que un grupo debe tener una actividad por periodo. En las restricciones de espacio

pertenece a que un aula no puede tener dos o más actividades al mismo tiempo. En las restricciones de tiempos libres son los tiempos que no debe de existir alguna asignación, regularmente para indicar un receso y por último las aulas de preferencia para una materia, grupo, profesor o actividad (Véase Tabla 4.5).

Tabla 4.5: Otros tipos de restricciones

Otro tipo de restricción	**Profesores**	**Estudiantes**	**Materia**	**Materias**	**Actividad**
Limitaciones de tiempos					
Limitaciones de espacios					
Tiempos libres	X	X			
Aula no disponible					
Aula(s) de preferencia			X	X	X

4.2. Diseño del sistema multi-agente

En el diseño del sistema de la plataforma multi-agente para solución al problema de la programación de horarios los actores/agentes involucrados se encuentran descritos a continuación (Véase Tabla 4.6):

Tabla 4.6: Diseño de plataforma multi-agente

Agente	**Objetivo**	**Representación**
Coordinador	Agente que se crea, se registra en el DF al iniciar y se encarga de leer el archivo XML para crear a los agentes profesores, grupos y cargar las restricciones de actividad.	
Profesor	Agente que se registra en el DF, carga sus restricciones de tiempo y espacio. Además de generar su horario, enviar sus propuestas.	
Grupo	Agente que se registra en el DF, carga sus restricciones solicita el horario de los profesores para generar el suyo y genera las negociaciones en caso de conflictos con profesores involucrados en el mismo horario.	

Continúa en la página siguiente.

Tabla 4.6 – Continuación de la página anterior

Agente	Objetivo	Representación
Usuario Final	Usuario que inicia e interactúa con el sistema y genera el horario una vez terminado la generación de horario de los profesores y grupos.	
AMS	El AMS ejerce control de supervisión sobre el acceso y uso de la AP. Es el que mantiene un directorio de los identificadores de los agentes que contienen la dirección de transporte para los agentes registrados en el AP.	
DF	El DF ofrece servicios de páginas amarillas a otros agentes. Los agentes pueden registrar sus servicios con el DF o consultar el DF para saber qué servicios son ofrecidos por otros agentes.	

Estos agentes se crean en 3 diferentes etapas a lo largo del sistema (Véase Figura 4.1); en la primer etapa se crea un agente coordinador que se encarga de administrar la interfaz gráfica principal, así como leer las restricciones de actividades y crear los agentes profesores. Posteriormente al estarse creando los agentes profesores, estos se registran con el DF y van creando su horario de manera egoísta de acuerdo a sus actividades y restricciones asignadas para luego notificar al coordinador que han terminado de programar su horarios. En la segunda etapa una vez que todos los profesores están listos, el coordinador crea a los agentes grupos y estos a su vez se registran con el DF, obtiene sus restricciones, actividades y solicita el horario de cada profesor que imparte en ese grupo para luego el grupo revisar los conflictos en la tercer etapa donde se realiza la negociación con los profesores involucrados y después terminadas las negociaciones notificar al coordinador.

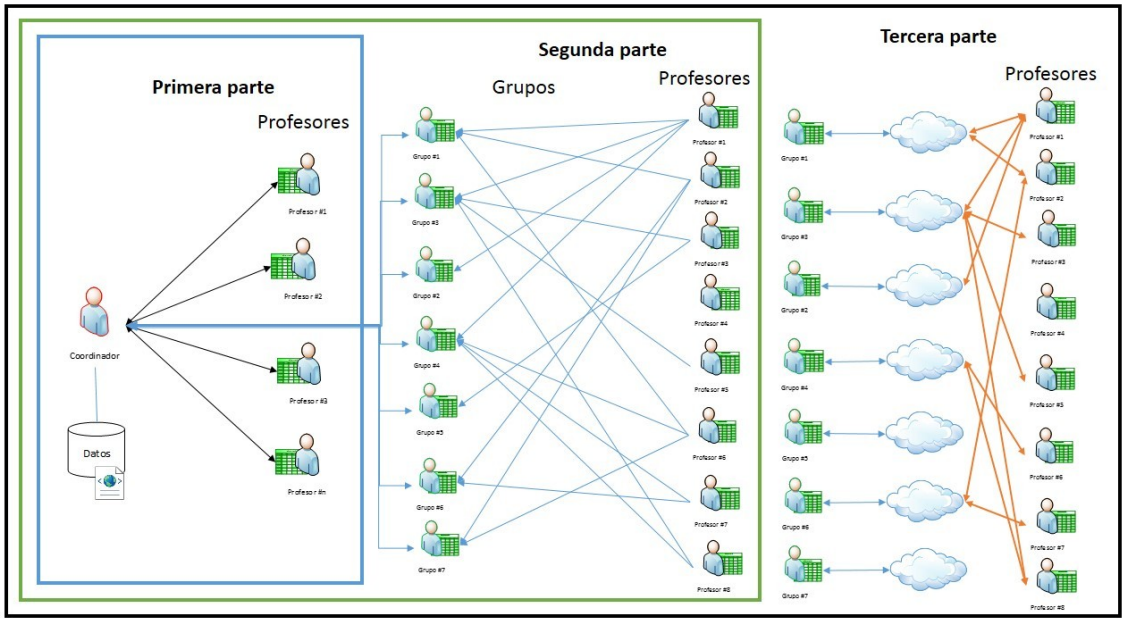


Figura 4.1: Diseño de la estructura multi-agente propuesta

Las etapas del sistema multi-agente se explican con mayor detalle en los siguientes sub-incisos.

4.2.1. Diseño de la estrategia de Negociación - Primera etapa

En esta primer etapa se crea un agente coordinador, se registra en el DF y se encarga de administrar la interfaz principal de los cambios efectuados por los agentes profesores y grupos. El coordinador captura las restricciones de actividad por medio de un archivo XML que contiene la información de la institución del caso de estudio, lee el archivo y crea los agentes profesores quienes se registran con el DF, obtienen sus restricciones y sus actividades, posteriormente en base a sus actividades realizan su propuesta de manera individual y egoísta de acuerdo a sus preferencias, siempre y cuando se respeten las restricciones generales de la institución, en caso de existir conflictos, el horario del profesor se modifica acorde a los lineamientos de la institución

pero respetando parte del horario sugerido por el profesor. Los profesores implementan el protocolo FIPA-Request (Véase Figura 4.2), donde el Iniciador es el profesor y el Participante el Coordinador.

El profesor cuando termina de realizar su programación de horario, le hace la petición al Coordinador para que lo muestre en la interfaz gráfica, así mismo el coordinador puede aceptar o rechazar la petición y de ser favorable lo muestra en la interfaz e informa al profesor.

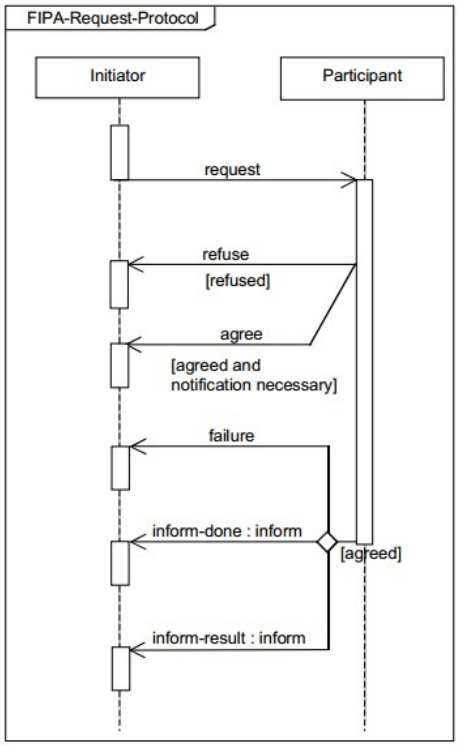


Figura 4.2: Protocolo FIPA-Request-Primera fase

La representación de una programación de horario de un profesor, está compuesta por los días, horas y en cada posición un identificador, el grupo y la materia del profesor (Véase Tabla 4.7).

Tabla 4.7: Estructura de horario profesor

Horario	Lunes	Martes	Miércoles	Jueves	Viernes
7:00 - 7:50	Id/Grupo/Materia		Id/Grupo/Materia		Id/Grupo/Materia
7:50 - 8:40	Id/Grupo/Materia		Id/Grupo/Materia		Id/Grupo/Materia
8:40 - 9:30	R E C E S O				
9:30 - 9:50				Id/Grupo/Materia	
9:50 - 10:40				Id/Grupo/Materia	
10:40 - 11:30				Id/Grupo/Materia	
11:30 - 12:20				Id/Grupo/Materia	
12:20 - 1:10				Id/Grupo/Materia	

4.2.2. Diseño de la estrategia de Negociación - Segunda Etapa

En esta segunda etapa cuando todos los profesores se encuentran listos, el coordinador crea a los agentes Grupos y cada grupo se registra en el DF, obtiene sus restricciones, actividades y solicita el horario de los profesores por medio del protocolo FIPA-request (Véase Figura 4.3), siendo el Iniciador el grupo y los Participantes los profesores. El Grupo solicita cada uno de los profesores su horario y el profesor envía como respuesta su horario.

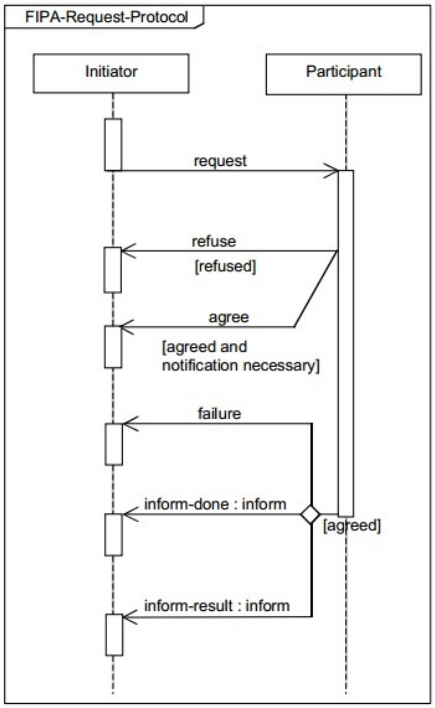


Figura 4.3: Protocolo FIPA-Request-Segunda fase

4.2.3. Diseño de la estrategia de Negociación - Tercera Etapa

Una vez que todos los profesores hayan mandado su horario al grupo, este verifica los conflictos y en caso de existir conflictos en un determinado día y hora se implementa el protocolo de negociación iterativo contract-net (Véase Figura 4.4).

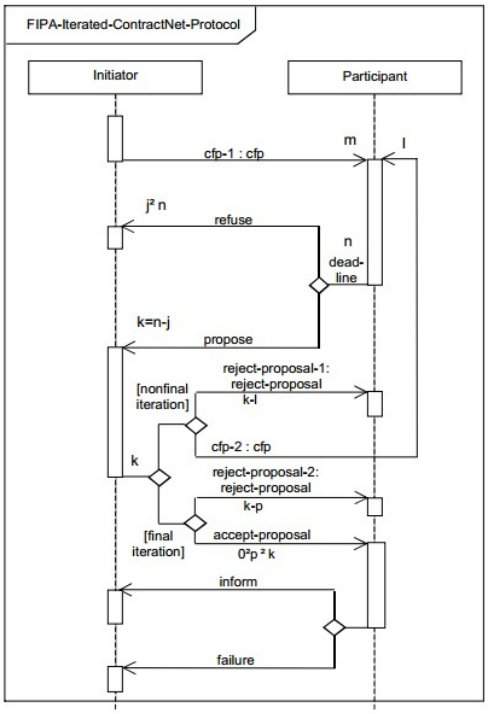


Figura 4.4: Protocolo Iterativo Contract-Net

En el protocolo iterativo contract net el Iniciador es el Grupo y los Participantes los profesores. El Grupo envía la solicitudes a los profesores de que propongan una nueva posición donde no presente ningún problema o donde se respete la mayor cantidad de restricciones por medio de una función de evaluación que determine la calidad de la solución, los profesores envían las propuestas que le solicito el grupo y este evalúa todas las propuestas recibidas para aceptarlas, rechazarlas o de ser necesario solicitarle a los profesores una nueva posición.

A diferencia del protocolo contract-net el protocolo iterativo contract-net permite nuevas rondas de negociación, el cual es útil cuando existe conflicto con el mismo profesor en diferentes grupos donde el profesor envía la misma propuesta (día-hora) a ambos grupos del cual ambos tienen que evaluar la propuesta y gana el que la asigne primero, mientras que el otro agente (grupo) si también gana la misma propuesta del profesor entonces el grupo verifica que no esté asignado y de ser así el grupo solicita que le envíen otra propuesta de negociación. Los escenarios posibles de ejemplificación con tres profesores se muestran a continuación (Véase Tabla 4.8).

Tabla 4.8: Posibles escenarios entre tres profesores

Escenario 1	**Escenario** 2	**Escenario** 3	**Escenario** 4	**Escenario** 5
Profesor 1:0	Profesor 1:100	Profesor 1:100	Profesor 1:100	Profesor 1:MAX
Profesor 2:0	Profesor 2:0	Profesor 2:100	Profesor 2:200	Profesor 2:MAX
Profesor 3:0	Profesor 3:0	Profesor 3:100	Profesor 3:300	Profesor 3:300

En una evaluación de conflictos por parte de un Grupo dado el escenario 1 con tres profesores es que estos no presentan un ningún inconveniente en moverse a otro espacio por lo que el agente grupo selecciona un agente al azar cancelándole su petición, mientras que a los demás se las acepta y se cambian. En un escenario 2 el profesor 1 muestra que puede cambiarse de posición pero violando una o varias restricciones con un peso de 100, mientras que los demás profesores no tienen problemas en moverse por lo que el grupo cancela la petición del profesor 1 y acepta el cambio de los profesores 2 y 3. En un escenario 3 el grupo selecciona un profesor al azar entre el 1 y el 2 cancelándole su petición debido a que ambos violan una o varias restricciones, mientras los demás aceptan su nueva posición. En el escenario 4, cancela la petición del profesor 3 debido a que presenta mayor problema al cambiarse mientras que los demás les acepta su nueva posición y por último en el escenario 5, los profesores 1 y 2 presentan un valor máximo que indica que no encontraron una posición disponible el cual el grupo acepta al profesor 3 y cancela la petición de los profesores 1 y 2 (Véase

Tabla 4.8).

De lo anterior por ser un enfoque distribuido donde todos los grupos están tratando de resolver conflictos de manera simultánea con todos los profesores, existen casos en que un profesor tenga un conflicto con otros profesores en varias horas con diferentes grupos y que el mismo profesor mande la misma posición a los diferentes grupos, por lo que el grupo valida y si ya fue asignado se solicita una nueva negociación para que el profesor proporcione una nueva posición, tal como se explicó anteriormente en el protocolo iterativo contract-net.

La estructura representativa al de un grupo se muestra en la Tabla 4.9 donde se indica los días con el horario y para cada posición un identificador, el nombre del profesor y la materia.

Tabla 4.9: Representación grupo

Horario	**Lunes**	**Martes**	**Miércoles**	**Jueves**	**Viernes**
7:00 - 7:50	Id / Profesor / Materia		Id / Profesor / Materia		Id / Profesor / Materia
7:50 - 8:40	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia		Id / Profesor / Materia
8:40 - 9:30	R E C E S O				
9:30 - 9:50	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia
9:50 - 10:40	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia
10:40 - 11:30	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia
11:30 - 12:20	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	Id / Profesor / Materia	
12:20 - 1:10			Id / Profesor / Materia	Id / Profesor / Materia	

4.3. Implementación

En este apartado se describe la implementación de la base de datos utilizada en un formato XML por medio de etiquetas, el cual de manera ilustrativa se muestra en tablas y atributos similar a un enfoque relacional, así mismo se ilustra los dia-

gramas de clases implementados que dan lugar al comportamiento y características de las restricciones y por último el diagrama de clases del sistema propuesto con sus interfaces.

4.3.1. Base de datos

La estructura de la base de datos se encuentra organizada por medio de un formato XML. Cada fragmento de dato se encuentra rodeado por una etiqueta donde cada una de esas etiquetas describe el significado de ese fragmento de datos donde la combinación de etiqueta y datos se denomina nodo. En la Figura 4.5 se muestra de manera general la estructura por parte de FET en un formato XML en donde se desprenden todas las ramificaciones con su información. En esta investigación para ilustrar de manera práctica la base de datos, se representa de manera similar al modelo relacional por medio de tablas y atributos para poder identificar incluso los tipos de datos (numéricos, cadenas).

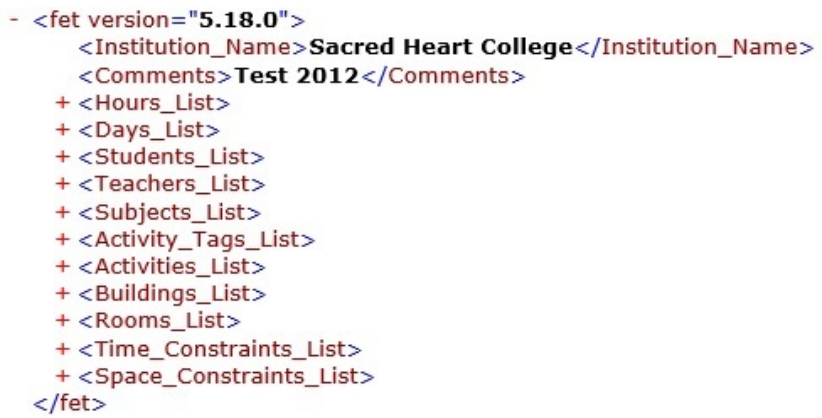


Figura 4.5: Estructura general de la base de datos FET

En la Figura 4.6 se muestran diferentes tablas que describen la situación de una escuela. En primer lugar se tiene la tabla *Fet* con sus campos comentarios y nombre de la institución, la tabla *Day* los días permitidos para dar clases, *Hour* las horas permitidas para dar clase a lo largo de un día. En la tabla *Subject* se enlistan todas

las materias que se imparten en la institución, en *Students* los grupos ya sea de manera individual o agrupados por años, la tabla *Building* para algunas instituciones identificar los edificios con los que cuenta la escuela, mientras que en la tabla *Room* el nombre de las aulas, la capacidad y el edificio donde se localiza. La tabla *Activity* compuesta por una duración, identificador, materia, grupo, maestro y el total de duraciones por semana, en estas actividades se tiene un listado donde cada profesor tiene varias actividades para el mismo o diferentes grupos.

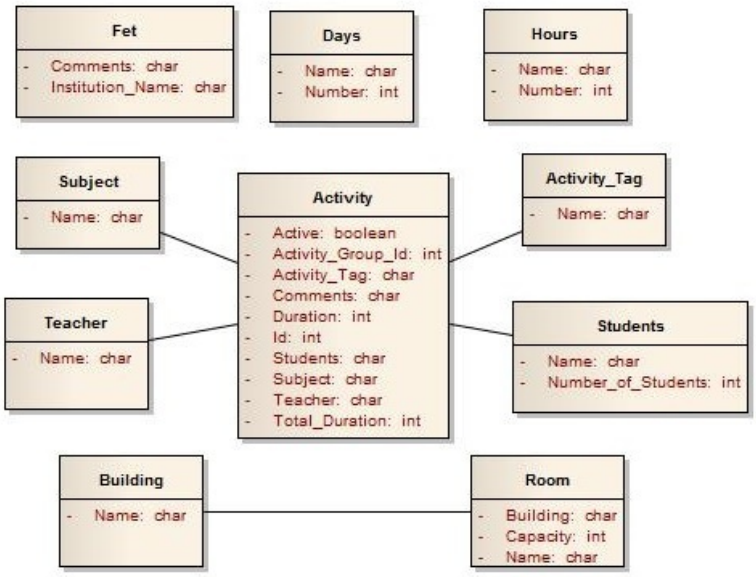


Figura 4.6: Base de datos de actividades

Siguiendo con la estructura de la base de datos, se tienen las tablas de las restricciones de los profesores de la Figura 4.7, donde cada una de las tablas se encuentra localizada en las restricciones de tiempo de la raíz principal en caso de que se aplique la restricción, de no ser necesarias ciertas restricciones entonces no se muestra en el archivo XML. En algunas tablas se muestra el atributo *Teacher_Name* que indica que la restricción es asignada para un profesor en específico, mientras que los que carecen con este atributo la restricción es considerada de manera general para todos los profesores. Todas las tablas presentan el atributo Active con el tipo de valor ver-

dadero o falso para indicar si la restricción esta activa o no, el otro atributo que cada uno presenta es el *Weight_ Percentage* que es un tipo de dato numérico que indica el peso de la restricción, donde a mayor cantidad la restricción se considera difícil y por lo tanto no debe violarse.

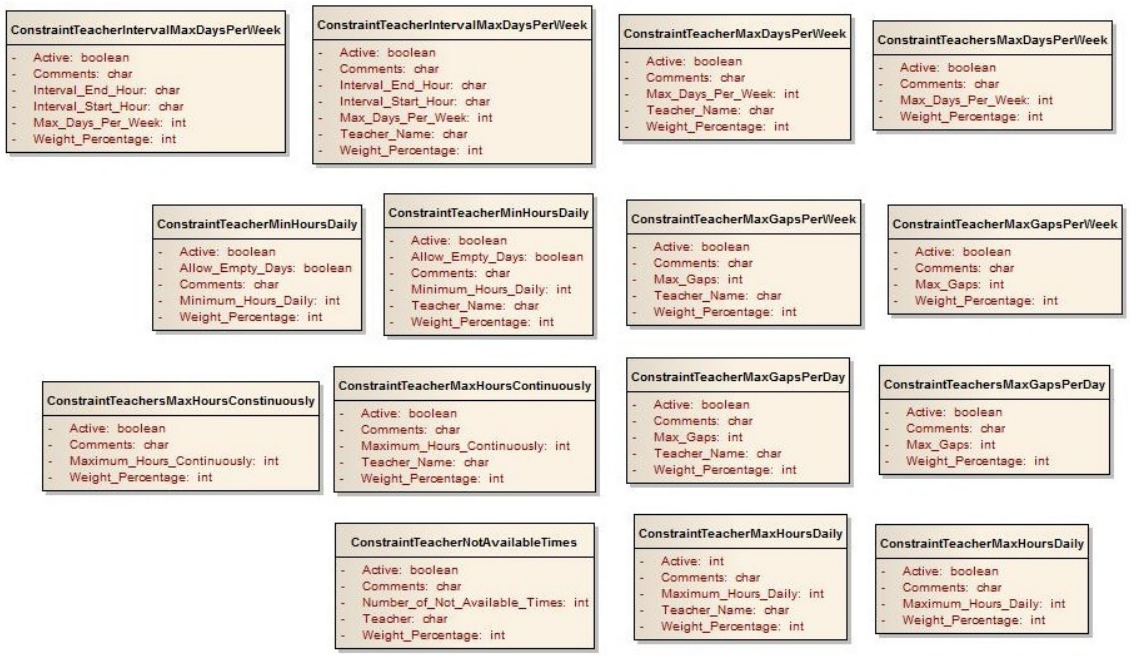


Figura 4.7: Restricciones de tiempo de profesores

Al igual que los profesores, también los grupos presentan las restricciones de tiempo con los mismos atributos *Active*, *Weight_ Percentage* que se describieron anteriormente solo cambiando el atributo *Teacher_ Name* por el *Student* que indica que la restricción es para un grupo en específico y los que no tienen este campo es para todos los grupos. Las restricciones de tiempo de los grupos se muestran en la Figura 4.8.

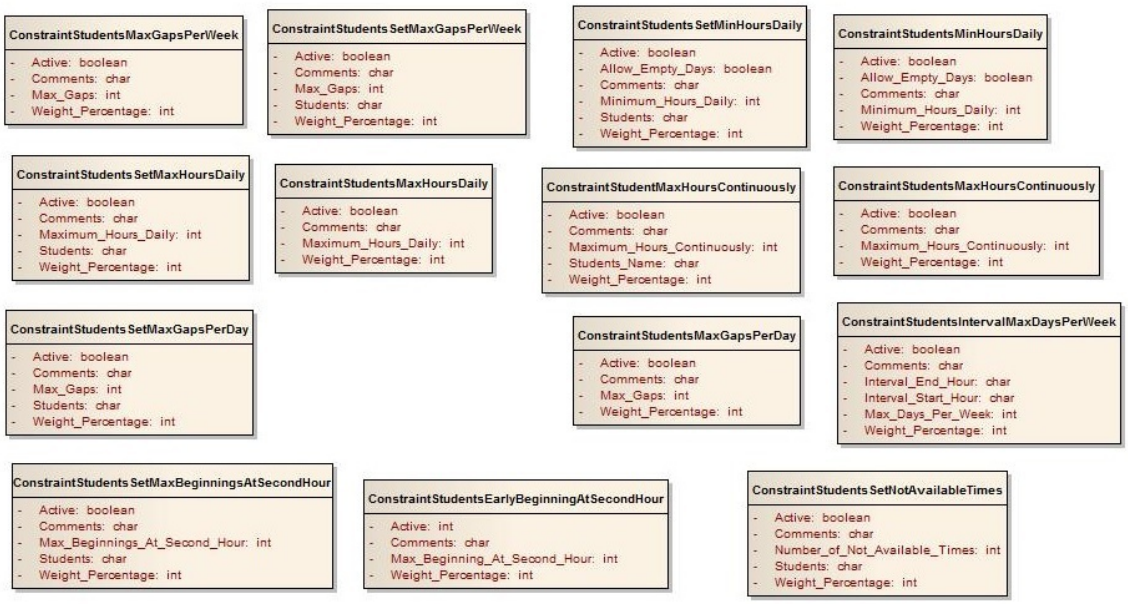


Figura 4.8: Restricciones de tiempo de estudiantes

Así mismo, siguiendo con la estructura de la base de datos se tienen las restricciones de actividades (Véase Figura 4.9) que están en la parte de restricciones de tiempo del archivo XML. Algunas actividades presentan los atributos *Subject_ Name* y *Teacher_ Name* y *Teacher_ Name* que pueden ser asignadas para el grupo, materia y profesor, mientras que otras se tiene el campo *Activity_ Id* que indica un identificado a la cual está ligada la actividad a la que se le esta asignada la restricción.

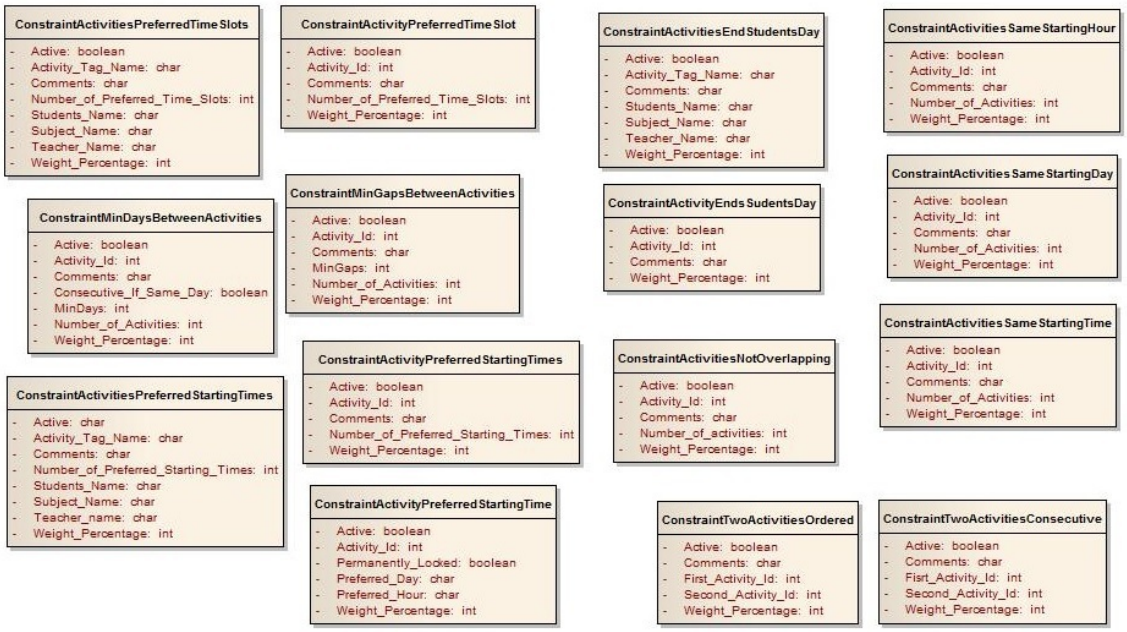


Figura 4.9: Restricciones de actividades

También se encuentran las restricciones de espacio, estas restricciones que se muestran en la Figura 4.10 se hallan localizadas en la parte de restricciones de espacio del archivo XML. Estas restricciones dan lugar a una actividad o materia donde un aula tiene preferencia. La tabla *ConstraintBasicConsultoryTime* se refiere a que una actividad no puede asignarse al mismo tiempo, mientras que la tabla *ConstraintBasicConsultorySpace* las aulas no pueden tener dos o más actividades.

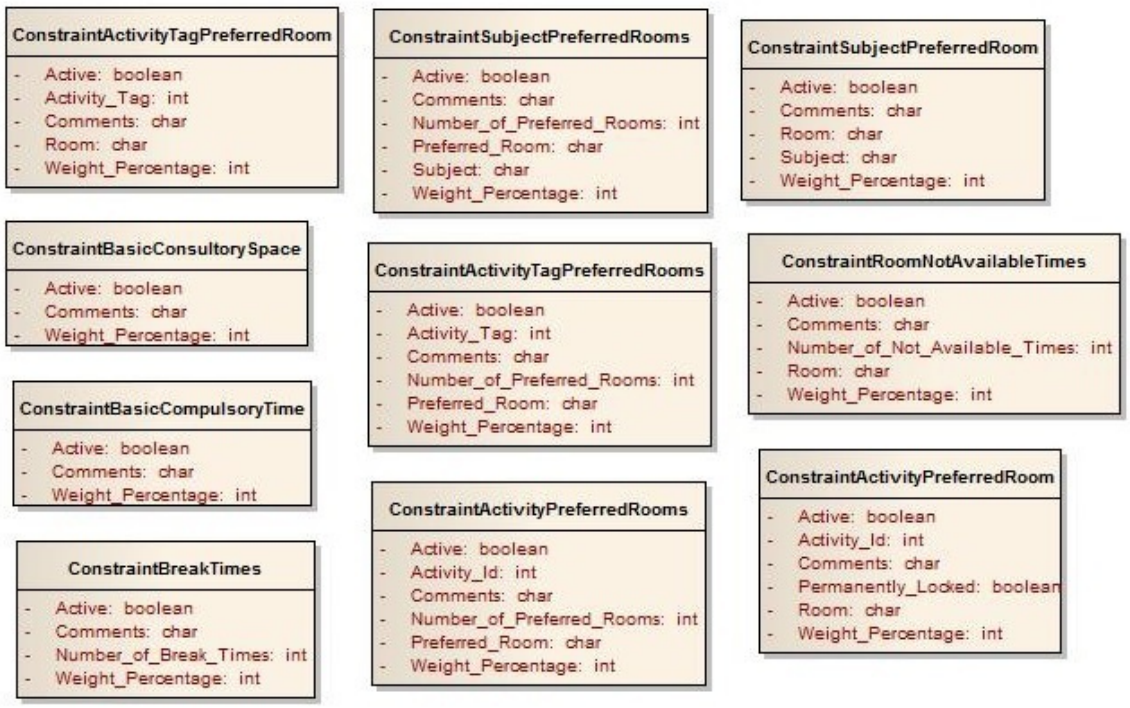


Figura 4.10: Restricciones de espacio

Por último, siendo parte también de las restricciones de espacio, están las restricciones de espacio para profesores y grupos representadas en la Figura 4.11, en estas tablas se describen los atributos donde un profesor o grupo tiene un aula de preferencia, también el máximo de cambios de edificios permitidos por día o por semana, al igual de espacios disponibles entre cambio de edificios.

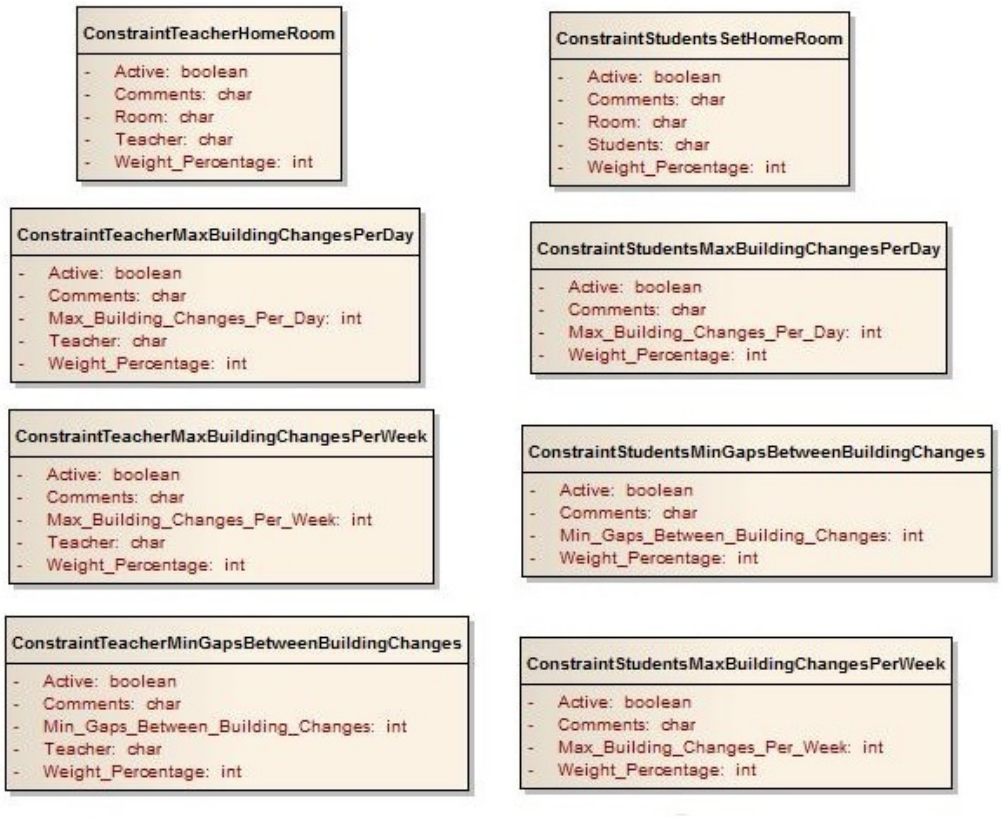


Figura 4.11: Restricciones de espacio de profesores y grupos

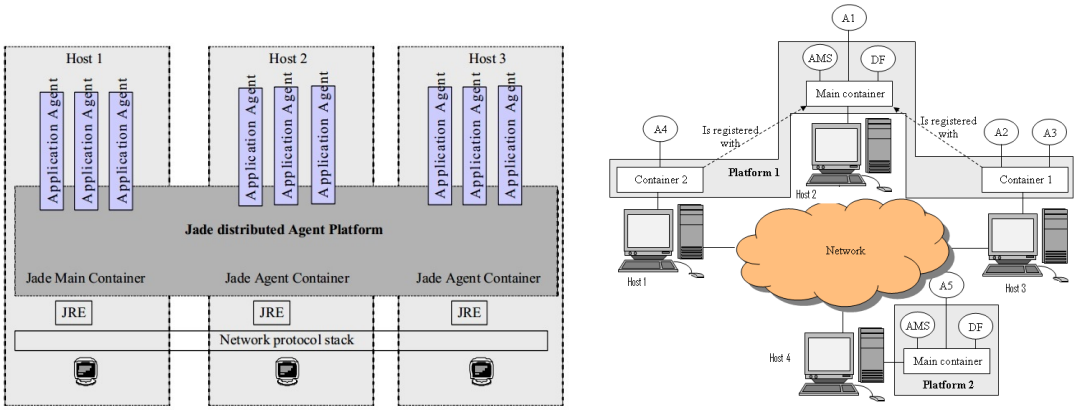
En el diseño del sistema como primera etapa no se consideraron las tablas *Room* y *Building* para aulas y los edificios, también en las restricciones de actividad no se consideraron las tablas; *ConstraintActivitiesSameStartingHour*, *ConstraintActivities-SameStartingDay*, *ConstraintActivitiesSameStartingTime*, *ConstraintTwoActivities-Consecutive*, *ConstraintTwoActivitiesOrdered* y *ConstraintAcitivitiesEndStudentsDay* debido a que su validación tiene que ser de manera general cuando el horario está construido con actividades de diferentes grupos o profesores y no de manera individual de un grupo o profesor al validar sus propias actividades. En la segunda iteración se contempla un agente de tipo general que valide los horarios cuando ya estén asig-

nadas las actividades con diferentes profesores y grupos, al igual que considerar las restricciones de espacio de los profesores y grupos.

4.3.2. Análisis de la plataforma seleccionada

De las 11 plataformas que cumplen con los estándares de FIPA se consideró JADE debido a que presenta la ventaja de estar actualizada a diferencia de April, FIPA-OS, Grasshopper, Java Agent Services API, LEAD y ZEUS. Así mismo, JADE es libre y de código abierto a diferencia de CAPNET y JACK que ocupan de licencia. JADE presenta características de seguridad en la autenticación de las conexiones, validación de usuario y cifrado de mensajes. Además de presentar una completa interfaz gráfica, gran documentación y alta aceptación en empresas, comunidad científica y proyectos de desarrollo.

Además, JADE (2013) presenta las ventajas de poderse distribuir en diferentes contenedores o equipos de manera remota con el fin de reducir el número de hilos por huésped en diferentes computadoras que no tengan incluso el mismo sistema operativo. Además otra característica de JADE es que puede ser controlado a través de una interfaz gráfica para la comunicación entre plataformas. Un sistema en JADE es hecho por uno o más contenedores de agentes, cada uno con una máquina virtual de Java separada con una arquitectura de agentes en JADE (Véase Figura 4.12).



A) Distribución JADE 1 B) Distribución JADE 2

Figura 4.12: Distribución de la plataforma JADE

El contenedor principal que se muestra en la Figura 4.12 mantiene una tabla de todos los contenedores con su referencia hacia el objeto RMI de cada uno de estos. Además, una tabla descriptora global de agentes es mantenida para relacionar cada nombre del agente con sus datos AMS y con su referencia RMI del contenedor. Cuando el contenedor principal se comienza a ejecutar, crea un registro RMI interno en el host actual escuchando usuarios por el puerto TCP/IP, luego se ejecutan el ACC, AMS y DF.

El entorno de JADE se presentan cuatro agentes auxiliares; el RMA que se encarga de gestionar la plataforma en iniciar, suspender, reiniciar agentes, así como matarlos, mandar mensajes y clonar agentes (Véase Figura 4.13).

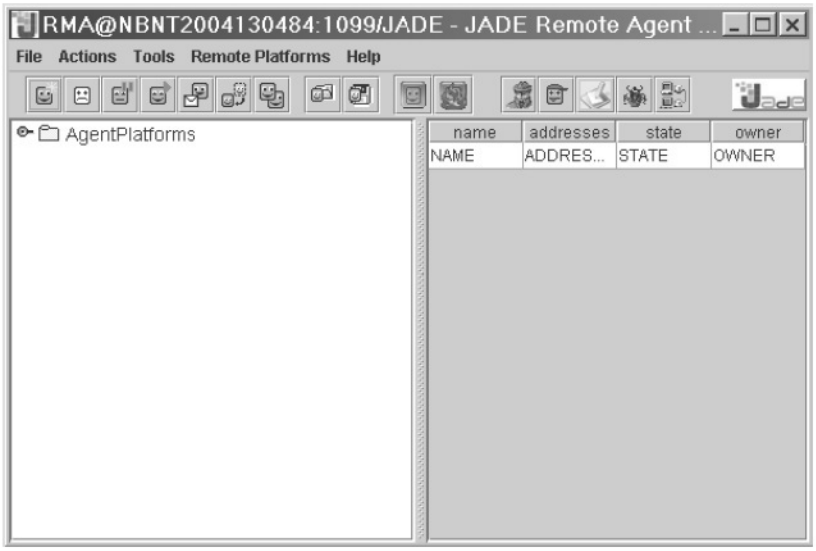


Figura 4.13: Interfaz RMA

El agente Dummy puede ser iniciado desde el RMA, permite de forma sencilla interactuar con agentes en la composición y envió de mensajes ACL, también permite el almacenamiento de los mensajes para emplearlos posteriormente (Véase Figura 4.14).

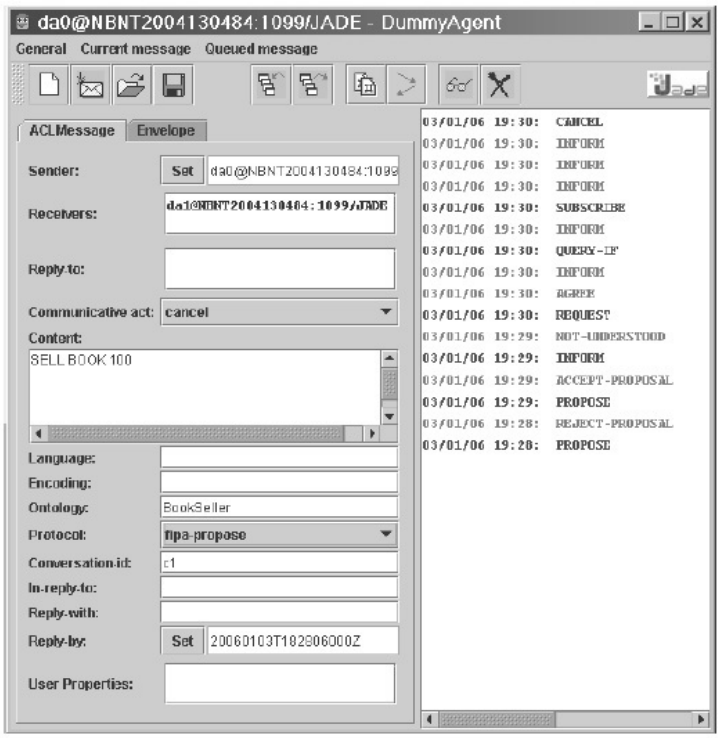


Figura 4.14: Interfaz Dummy

El agente Sniffer muestra las iteraciones que se producen en un agente donde el usuario selecciona que agentes quiere monitorizar para ver el contenido de cada mensaje (Véase Figura 4.15).

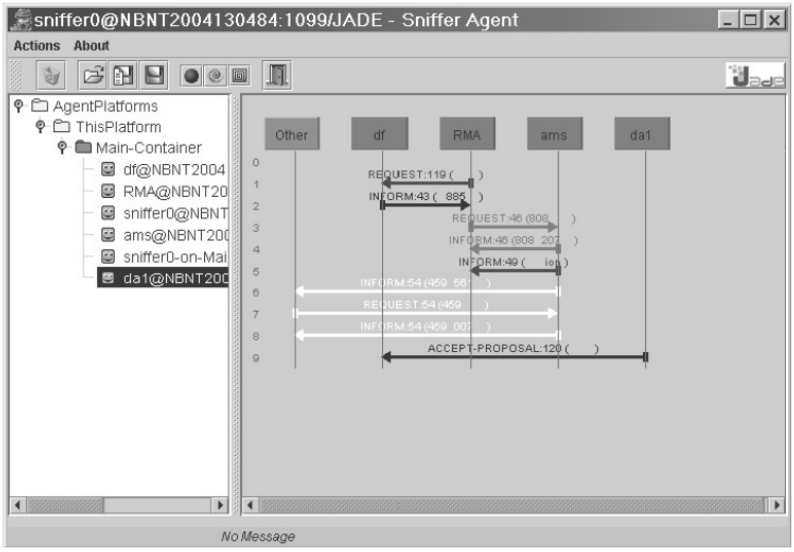


Figura 4.15: Interfaz Sniffer

Mientras que el agente Introspector permite monitorizar la ejecución de un agente mostrando las colas de entrada y salida de mensajes intercambiados por este, así como la de sus comportamientos y su ejecución paso a paso (Véase Figura 4.16).

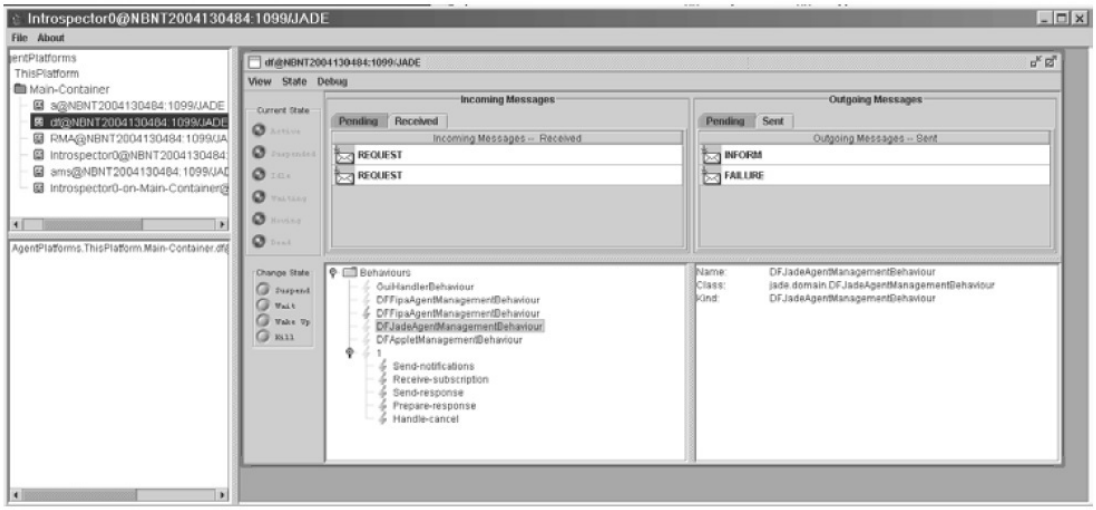


Figura 4.16: Interfaz Introspector

4.3.3. Estructura de un agente en JADE

Lo anteriormente es posible siempre y cuando se consideren la estructura de un agente de JADE, en donde se implementa instancias de clases que extienden de *jade.core.behaviours.Behaviours* e implementa los métodos *action()* y *done()* para dar un enfoque más simple y limpio para la ejecución de tareas complejas mediante la composición de comportamientos simples. La base para esta función es proporcionada por la clase *CompositeBehaviour* incluido en el paquete *jade.core.behaviours*. Un comportamiento compuesto (una instancia de la clase *CompositeBehaviour*) es en sí un comportamiento que incorpora una serie se sub-comportamientos hijos. La clase *CompositeBehaviour* implementa el método *action()* de tal manera que cada vez que se llama se invoca el método action() de uno de los hijos. La política se utiliza para seleccionar cuál de los hijos lanzar en cada ronda que delega en el *ScheduleFirst()* (primera ronda) y *ScheduleNext()* (rondas sucesivas). Estos métodos se declaran abstractos y deben ser definidos en las subclases *CompositeBehaviour* que implementan los tipos reales de composiciones.

Los sub-comportamientos incrustados en un comportamiento compuesto también pueden ser comportamientos compuestos, por lo que es posible la creación de tareas complejas mediante la combinación jerárquica de comportamientos simples. La jerarquía completa del paquete jade.core.behaviour (Véase Figura 4.17) con tres tipos de comportamientos compuestos que se proporcionan en la distribución JADE llamados *SequentialBehaviour*, *FSMBehaviour* y *ParallelBehaviour*; *cf.* Bellifemine et al. (2007).

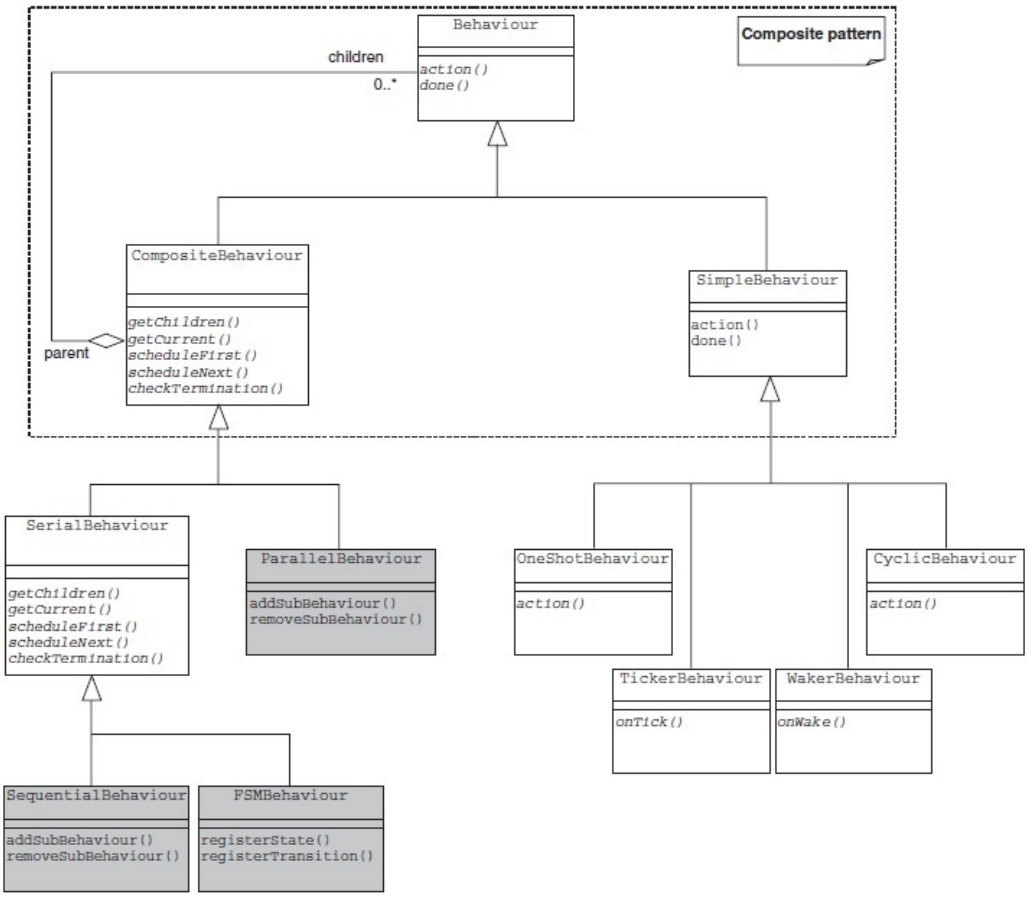


Figura 4.17: Jerarquía de clases JADE

4.3.4. Diseño de clases

Es común trabajar con agentes en un entorno grafico GUI (interfaz gráfica de usuario). La cuestión es que se tienen que utilizar los patrones de comunicación entre hilos, primeramente que el hilo AWT (componentes GUI) envié un mensaje ACL por medio de sus eventos de diferentes tipos (por ejemplo, presionar un botón) permitiendo despertar al hilo del agente que recibe el mensaje ACL para realiza alguna acción.

Para el entorno grafico (GUI), JADE incluye la clase *jade.gui.GuiAgent* para este propósito específico. Esta clase es una extensión de la clase *jade.core.Agent* desde el principio, es decir cuando se ejecuta el método *setup()* se inicializa un comportamiento ad hoc que maneja una cola de eventos (objetos de tipo *jade.gui.GuiEvent*)) que pueden ser recibidos por otros hilos.

Este comportamiento se oculta al programador, que solo necesita implementar el código específico para gestionar cada evento. Un hijo (particularmente el GUI) que desea notificar un evento a un agente de tipo *jade.gui.GuiAgent* debe crear un nuevo objeto de tipo jade.gui.GuiEvent y pasárselo como parámetro a la llamada del método *postGuiEvent()* de dicho agente. Después de que el método *postGuiEvent()* sea llamado, el agente reacciona despertando todos sus comportamientos activos, en particular, el comportamiento del agente que activa dicho evento en el método *onGuiEvent()*.

De acuerdo a lo anterior, la implementación del sistema con los agentes *Profesor*, *Principal* (coordinador) y *Grupo*, cada uno con su composición de eventos de tipo GUI y que a su vez estos agentes extienden de la clase *GUIAgent* para que cada uno de ellos implemente su propia interfaz, que a su vez esta extiende de la clase *Agent* para obtener todos los comportamientos de un agente (Véase Figura 4.18).

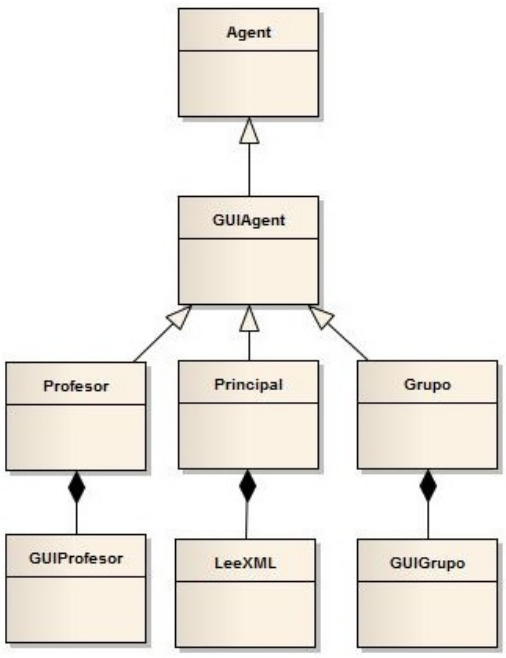


Figura 4.18: Diseño de agentes

Para conocer como un agente crea e implementa sus restricciones, es necesario explicar el patrón de diseño *Factory Method*, este patrón permite crear objetos definiendo una interfaz de creación de un cierto tipo de objeto, permitiendo que las subclases decidan que clase concreta necesita instanciar. Este patrón es de importancia debido a que a veces ocurre que una clase no puede anticipar el tipo de objetos que debe crear, ya que la jerarquía de clases que tiene requiere delegar la responsabilidad a una subclase.

En la Figura 4.19 se muestra la clase *Factory* que declara el método de fabricación y devuelve un objeto de tipo Product, en la clase *ConcreteFactory* se redefine el método de fabricación para devolver un producto. La clase *ConcreteProduct* es el resultado final donde el Factory se apoya en sus subclases para definir el método de fabricación que devuelve el objeto apropiado.

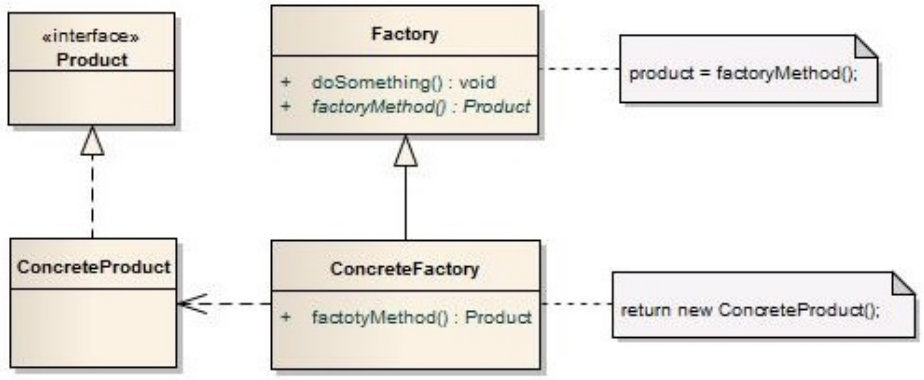


Figura 4.19: Patrón de diseño Factory Method

Siguiendo con esta estructura, la implementación del patrón de diseño Factory method se muestra en la Figura 4.20 para poder instanciar a los profesores y grupos sus tipos de restricciones. En la implementación existe una clase abstracta llamada PersonTimeConstraints de la cual heredan ocho clases del tipo de restricciones de tiempo (Véase Figura 4.21).

Pero quien se encarga de crear los tipos de restricciones concretas no debería tener que conocer cómo se compone internamente. Para ello se crea la clase *ConstraintFactoryMethod* y desde el punto de vista del agente se crean las restricciones.

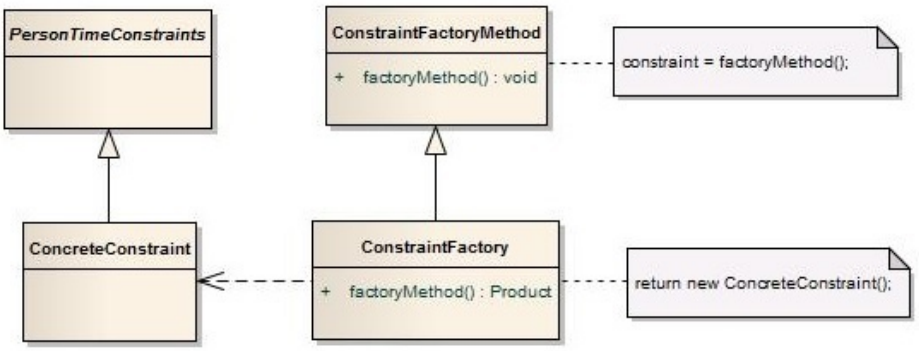


Figura 4.20: Implementación del pátron Factory Method en restricciones de tiempo

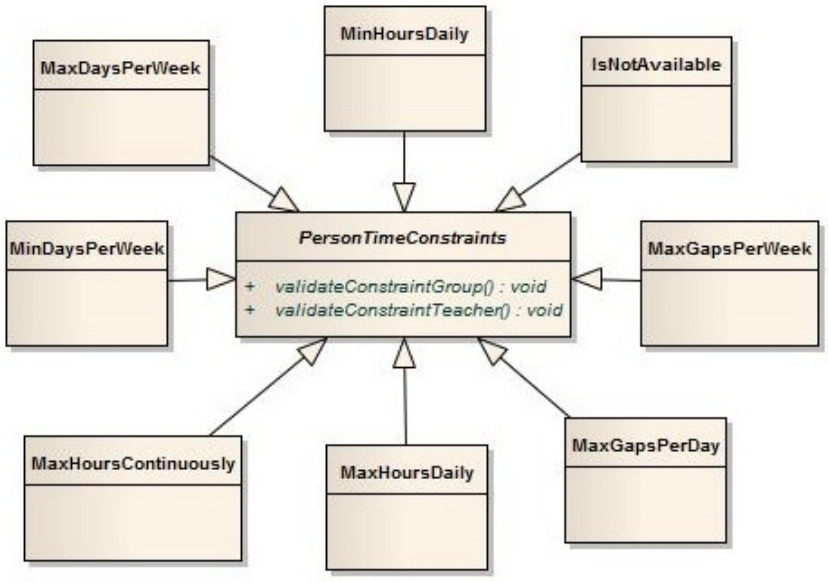


Figura 4.21: Restricciones de tiempo

También se implementó el patrón Factory Method con las restricciones de actividades (Véase Figura 4.22), estas restricciones se instancian al agente coordinador. En la implementación se tiene una clase abstracta llamada ActivitiesTimeConstraints de la cual heredan 12 clases de restricciones de actividad que se muestra en la Figura 4.23.

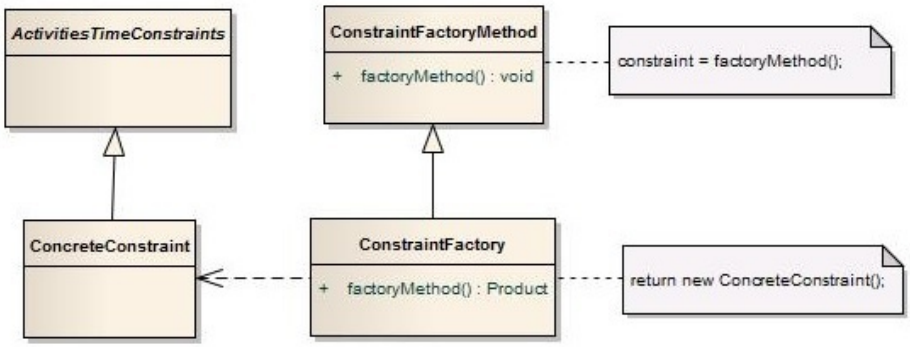


Figura 4.22: Implementación del patrón Factory Method en restricciones de actividades

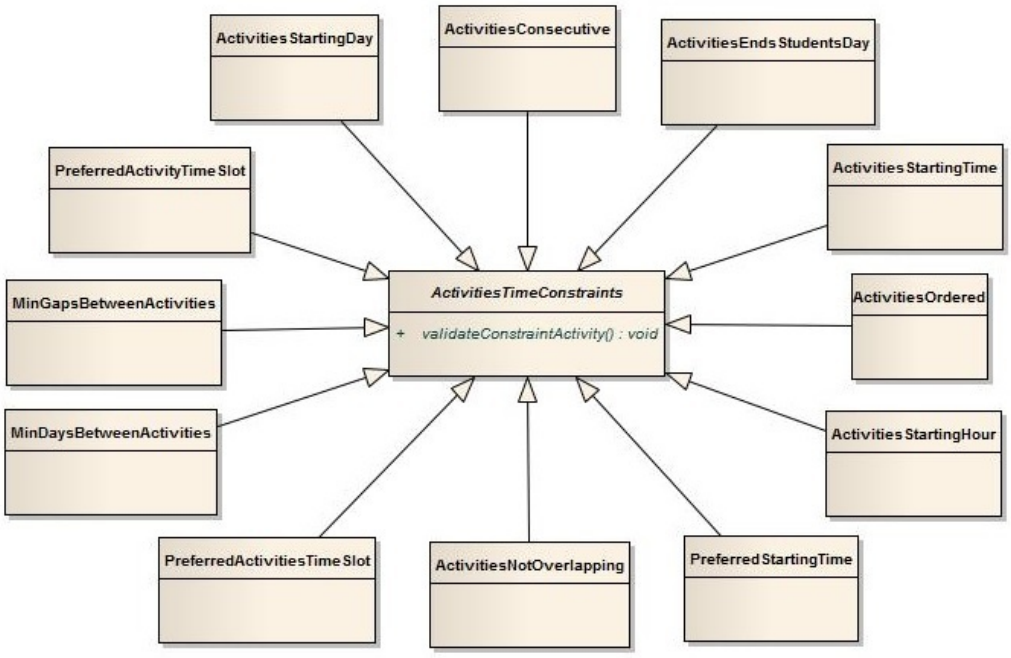


Figura 4.23: Restricciones de actividades

4.3.5. Interfaces del sistema

En la implementación del sistema en el entorno grafico GUI, como se ha mencionado anteriormente, al iniciar el sistema se crea el agente coordinador que se encarga de obtener las restricciones de actividad y crear a los agentes profesores. Al crearse los agentes profesores estos obtienen sus restricciones, actividades y crean su propuestas de horario para luego notificar al agente coordinador que ha finalizado de crear su horario. El coordinador al recibir la notificación del profesor le asigna al agente el estatus de listo (Ready).

En la Figura 4.24 se muestra la interfaz del agente coordinador indicando la primer columna el nombre del agente (Agent), en la segunda el tipo de agente (Type of Agent) en caso de ser profesor o grupo y por último el estatus del agente (Status).

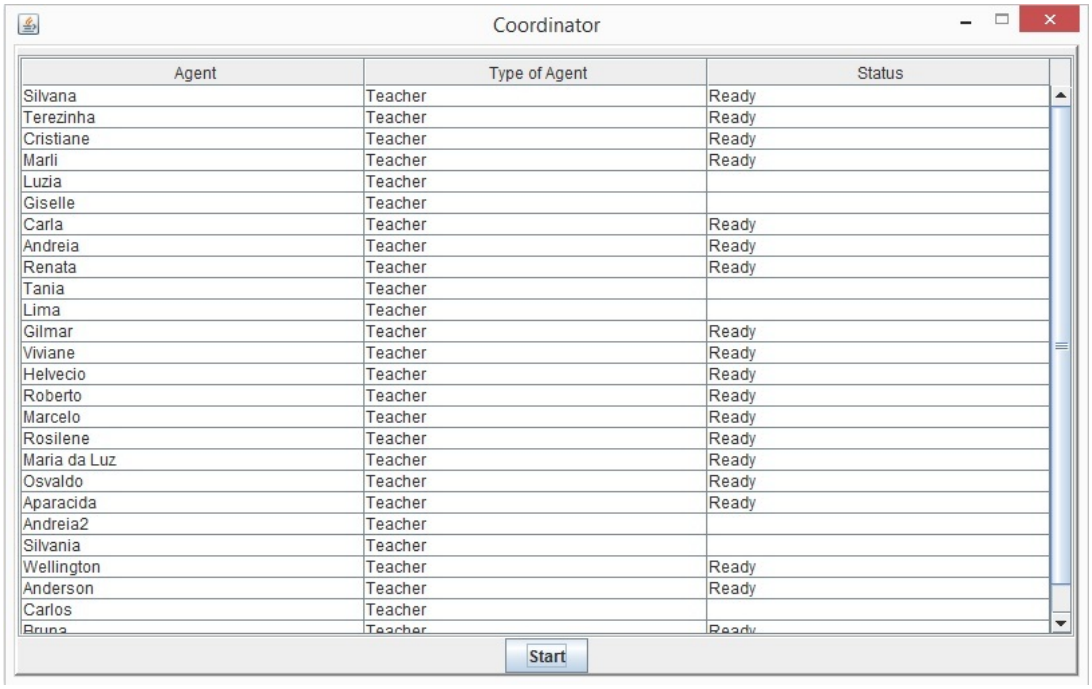


Figura 4.24: Pantalla coordinador con profesores

Una vez que todos los profesores se encuentran listos en un estatus “Ready”, el usuario puede presionar el botón *Start* para que los profesores envíen las propuestas a los grupos y estos revisen los espacios donde existen conflictos para iniciar la negociación entre los profesores involucrados. En la Figura 4.25 se muestran los agentes grupos que se han creado y están en negociación.

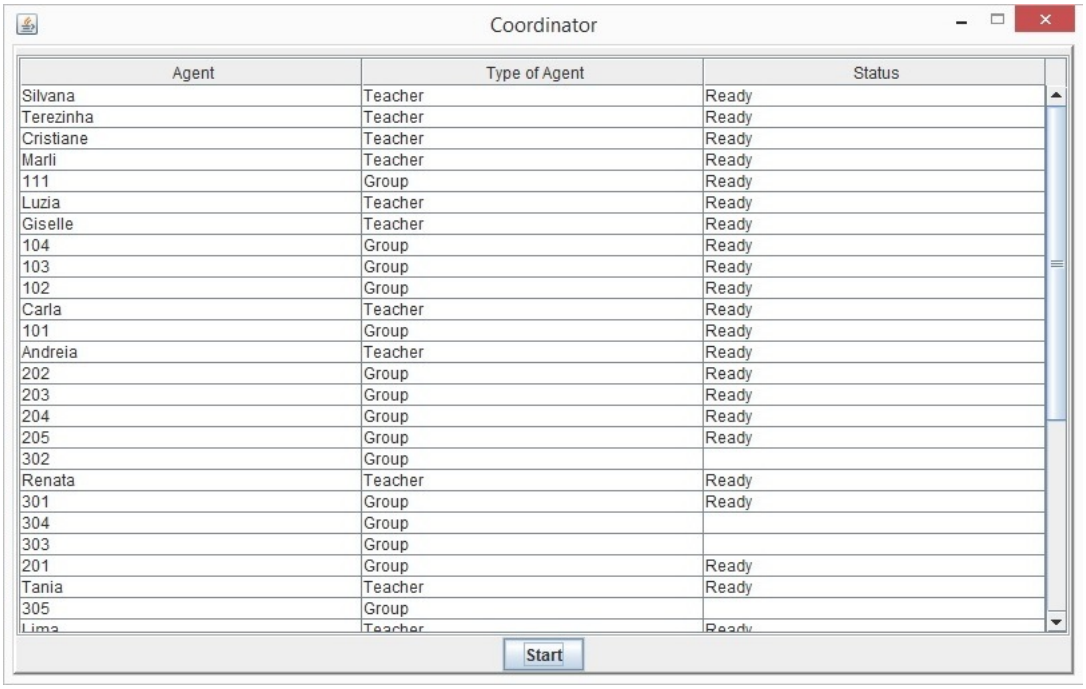
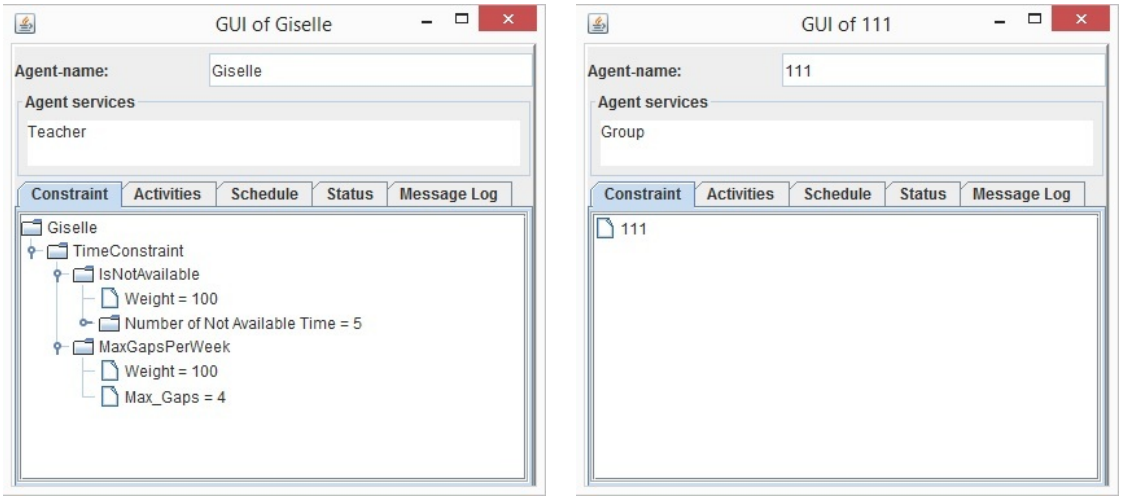


Figura 4.25: Pantalla coordinador con profesores y grupos

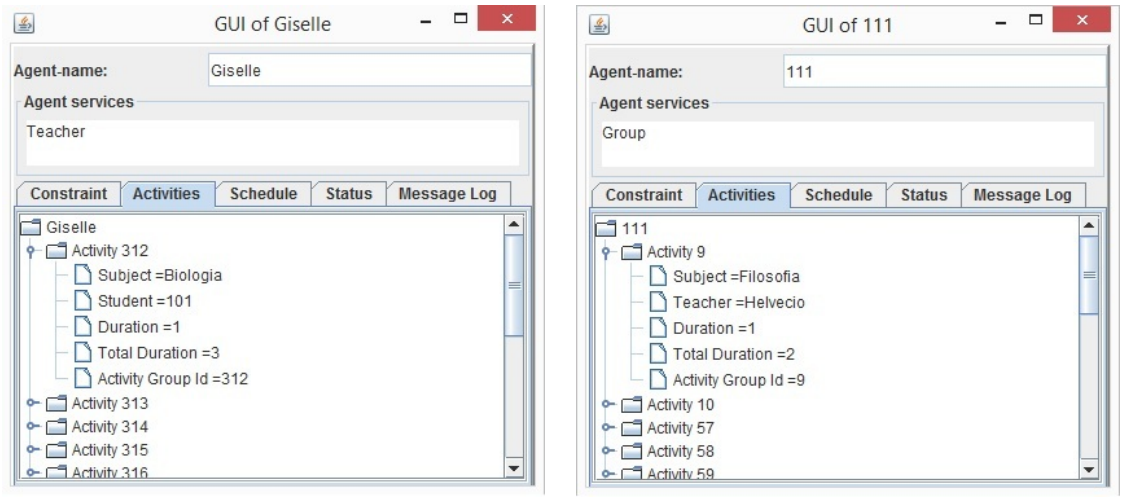
En cualquier momento que un profesor o un grupo se encuentren listo, es posible acceder a sus propiedades dando doble clic sobre la interfaz del coordinador. En la Figura 4.26 se muestran las propiedades de los agentes profesor (izquierda) y grupo (derecha); en la parte superior se muestra el nombre del agente, después el tipo de agente y posteriormente una pestaña de restricciones.



A) Estatus profesor B) Estatus grupo

Figura 4.26: Propiedades de las restricciones

En Figura 4.27 se muestra la segunda pestaña llamada Activities donde se presentan las actividades que tiene un profesor o un grupo.



A) Actividades profesor B) Actividades grupo

Figura 4.27: Propiedades de las actividades

Por último, se tiene la pestaña Schedule que indica el horario de un profesor en el caso de la Figura 4.28 y el de un grupo de la Figura 4.29.

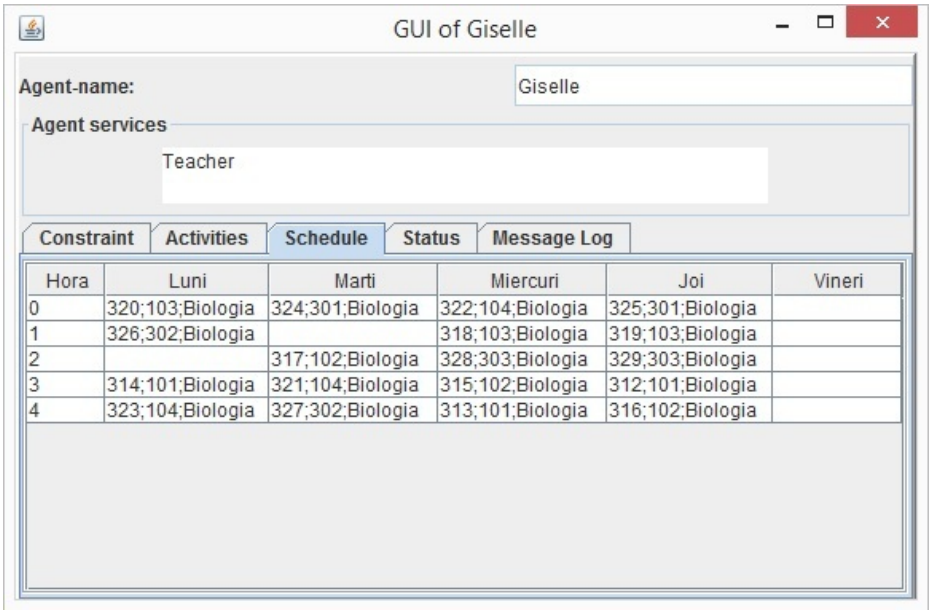


Figura 4.28: Propiedades del profesor – horario

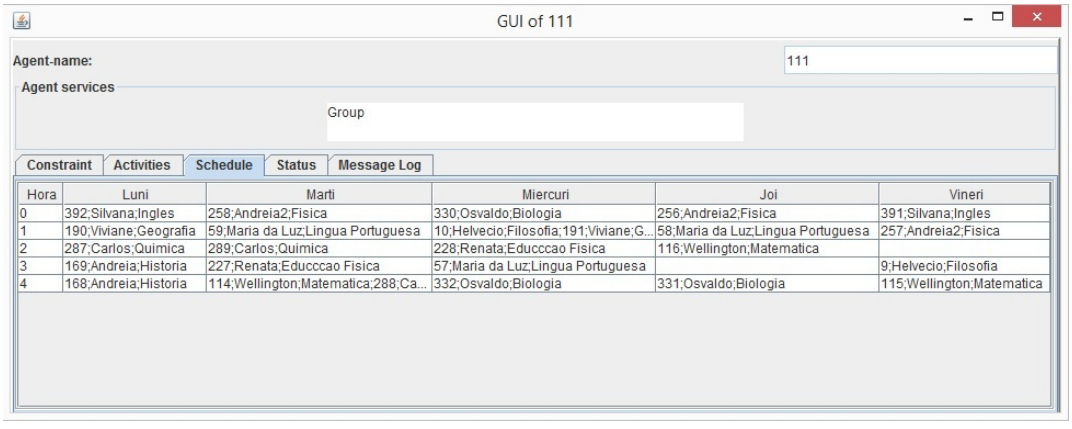


Figura 4.29: Propiedades del grupo – horario

Capítulo 5
EXPERIMENTACIÓN Y ANÁLISIS DE RESULTADOS

5.1. Selección de casos de estudio

En esta sección se muestran los resultados obtenidos al implementar el algoritmo basado en sistemas multi-agentes mediante negociación para la resolución la resolución de conflictos entre profesores. En este proceso de experimentación se tomaron en cuenta cuatro casos de estudio de manera aleatoria de un total de 22 países. Estos casos de estudio se encuentran disponibles de manera libre en un repositorio por parte de FET (2013).

Los países considerados para la experimentación son; Brasil, Belice, España y Reino Unido, donde cada uno de ellos presentan características particulares en días, horas, grupos, profesores, materias y actividades. Las actividades de Belice y España fueron modificadas debido a que existen datos nulos, Belice de un total de 952 se tuvieron 249 datos nulos, mientras que España de un total de 1086 actividades se encontraron 269, dando como resultados los datos que se muestran en la Tabla 5.1.

Tabla 5.1: Características de los casos de estudio

	Belice	**Brasil**	**España**	**Reino Unido**
Días	5	5	5	6
Horas	6	5	7	5
Grupos	23	16	185	46
Profesores	44	27	56	26
Materias	24	12	78	25
Actividades	703	400	817	163

En la Tabla 5.2, se indica la cantidad de restricciones de tiempo y espacio que tiene cada uno de los casos de estudio de los cuales en la primera etapa no están consideradas las restricciones de espacio.

Tabla 5.2: Restricciones de los casos de estudio.

Restricciones	Tipo de restricción	Brasil	Belice	España	Reino Unido
ConstraintBasicCompulsoryTime	Tiempo	1	1	1	1
ConstraintMinDaysBetweenActivities	Tiempo	160	433	273	
ConstraintTeacherNotAvailableTimes	Tiempo	23		2	
ConstraintTeacherMaxDaysPerWeek	Tiempo	13			
ConstraintTeachersMaxGapsPerWeek	Tiempo	1			
ConstraintStudentsSetNotAvailableTimes	Tiempo		12		
ConstraintActivitiesSameStartingDay	Tiempo		5		
ConstraintBreakTimes	Tiempo			12	
ConstraintActivitiesNotOverlapping	Tiempo			1	
ConstraintActivitiesSameStartingTime	Tiempo			21	
ConstraintTwoActivitiesConsecutive	Tiempo			1	
ConstraintActivityPreferredTimeSlots	Tiempo			2	
ConstraintActivityPreferredStartingTime	Tiempo				167
ConstraintBasicCompulsorySpace	Espacio	1	1	1	1
ConstraintSubjectPreferredRooms	Espacio		1		
ConstraintSubjectPreferredRoom	Espacio		4	46	167
ConstraintSubjectActivityTagPreferredRoom	Espacio		7		
ConstraintRoomNotAvailableTimes	Espacio		1		
ConstraintActivitiesOccupyMaxDifferentRooms	Espacio		2		
TOTAL		199	476	360	336

5.1.1. Belice

En la experimentación, para cada caso de estudio se consideraron 30 ejecuciones (iteraciones) con el fin de identificar en promedio los conflictos que presenta cada grupo, así como los conflictos resueltos, los protocolos iniciados, el número de mensajes y el tiempo de duración de la ejecución. En la Tabla 5.3 se muestran los resultados del caso de estudio de Belice que fue construida por las Tablas de los Anexos (Véase Anexo, Tablas A.1 y A.2).

Tabla 5.3: Resultados Belice

Iteraciones	**Conflictos**	**Resueltos**	**Diferencia**	**Protocolos Iniciados**	**Mensajes**	**Tiempo de ejecución (minutos)**
1	136	125	11	383	1748	11.37
2	135	125	10	384	1755	10.50
3	118	112	6	356	1583	11.28
4	132	120	12	376	1726	11.69
5	119	111	8	375	1662	11.01
6	127	117	10	381	1716	10.84
7	129	117	12	387	1743	10.69
8	133	128	5	385	1750	10.35
9	125	117	8	397	1735	10.65
10	126	115	11	374	1691	10.58
11	116	111	5	372	1641	10.25
12	129	117	12	368	1681	11.17
13	125	114	11	381	1707	10.48
14	128	121	7	389	1756	11.11
15	118	111	7	375	1653	11.17
16	124	112	12	372	1669	11.09
17	124	118	6	372	1684	10.55
18	131	124	7	378	1724	11.22
19	124	111	13	374	1635	11.16
20	130	117	13	383	1724	10.58
21	118	107	11	369	1635	10.62
22	128	123	5	381	1742	10.68
23	121	115	6	388	1702	10.75
24	125	117	8	382	1699	10.47
25	111	111	0	367	1591	11.55
26	126	121	5	378	1687	11.71
27	124	115	9	380	1709	11.51
28	123	117	6	388	1713	10.99
29	131	124	7	369	1685	11.35
30	116	112	4	376	1662	10.36
Total	3752	3505	247	11340	50845	327.73
Media	125.07	116.83	8.23	378	1694.83	10.92
Moda	124	117	11	381	1662	
Desviación estándar	6.02	5.25	3.18	8.26	44.74	0.42

Los resultados por parte de Belice indican que en promedio se tienen 125.07 (± 6.02) conflictos, se resuelven 116.83 (± 5.25) y quedan sin resolverse 8.23 (± 3.18), mientras que en los protocolos iniciados se tienen en promedio 378 (± 8.26) y mensajes entre agentes 1694.83 (± 44.74).

5.1.2. Brasil

En los resultados obtenidos también del otro país latinoamericano, es el caso de estudio de Brasil, estos se muestran en la Tabla 5.4 que fue construida por las Tablas de los Anexos (Véase Anexo, Tablas A.3 y A.4).

Tabla 5.4: Resultados Brasil

Iteraciones	**Conflictos**	**Resueltos**	**Diferencia**	**Protocolos Iniciados**	**Mensajes**	**Tiempo de ejecución (minutos)**
1	135	103	32	364	1713	2.50
2	131	104	27	366	1694	2.57
3	137	95	42	367	1694	2.64
4	125	99	26	368	1674	2.58
5	148	109	39	364	1733	2.38
6	132	100	32	376	1699	2.34
7	140	107	33	365	1712	2.62
8	148	104	44	383	1778	2.54
9	150	106	44	377	1774	2.63
10	144	111	33	382	1784	2.52
11	147	107	40	372	1756	2.66
12	142	103	39	374	1696	2.47
13	151	99	52	388	1797	2.49
14	132	102	30	360	1645	2.78
15	141	113	28	366	1715	3.01
16	145	99	46	365	1718	2.83
17	134	103	31	360	1688	3.06
18	139	103	36	372	1701	2.88
19	131	92	39	369	1683	3.12
20	146	117	29	390	1829	2.84
21	136	102	34	370	1708	2.79
22	145	103	42	370	1728	3.09

Continúa en la página siguiente.

Tabla 5.4 – Continuación de la página anterior

Iteraciones	**Conflictos**	**Resueltos**	**Diferencia**	**Protocolos Iniciados**	**Mensajes**	**Tiempo de ejecución (minutos)**
23	141	94	47	369	1698	2.73
24	127	104	23	368	1682	2.57
25	133	101	32	353	1621	2.38
26	131	96	35	367	1652	2.42
27	138	97	41	372	1716	2.38
28	135	100	35	365	1680	2.64
29	126	93	33	359	1623	2.55
30	139	101	38	384	1778	2.51
Total	4149	3067	1082	11095	51369	79.51
Media	138.30	102.23	36.07	369.83	1712.30	2.65
Moda	131	103	32	364	1694	
Desviación estándar	7.30	5.72	6.83	8.67	49.77	0.22

Los resultados muestran que en promedio se tienen 138.3 (± 7.3) conflictos encontrados de manera inicial, se resuelven 102.23 (± 5.72) con el algoritmo y no se resuelven por el algoritmo 36.07 (± 6.83), mientras que en los protocolos iniciados en promedio se tienen 369.83 (± 8.67) y el intercambio de mensajes entre agentes con 1712.3 (± 49.77).

5.1.3. España

En los resultados de los casos de estudio de las instituciones Europeas se tienen también un total de 30 iteraciones, a diferencia de las instituciones de Latinoamérica, las europeas presentan un tiempo de ejecución en segundos debido en sus características al contar España con una hora extra y Reino Unido con un día extra y ambas con más grupos, permiten una mayor asignación y distribución más uniforme de los maestros, teniendo como respuesta una resolución más rápida.

Los resultados de España se muestran la Tabla 5.5, mientras que en la Tabla 5.6 la de Reino Unido que fue complementada por las Tablas de los Anexos (Véase Anexo, Tablas A.5 y A.6 para España; Tablas A.7 y A.8 a Reino Unido).

Tabla 5.5: Resultados España

Iteraciones	**Conflictos**	**Resueltos**	**Diferencia**	**Protocolos Iniciados**	**Mensajes**	**Tiempo de ejecución (segundos)**
1	14	14	0	463	1446	53.75
2	19	19	0	469	1486	48.88
3	10	10	0	458	1414	49.37
4	11	11	0	459	1421	50.33
5	12	12	0	462	1436	24.49
6	16	16	0	464	1456	57.66
7	11	11	0	459	1421	55.74
8	18	18	0	466	1471	61.66
9	11	11	0	456	1412	36.61
10	12	12	0	459	1425	36.46
11	10	10	0	457	1411	42.45
12	11	11	0	458	1418	43.23
13	8	8	0	456	1400	37.61
14	12	12	0	461	1432	29.39
15	17	17	0	464	1460	37.02
16	9	9	0	457	1407	40.79
17	13	13	0	461	1435	42.25
18	7	7	0	455	1393	41.19
19	11	11	0	458	1418	41.14
20	12	12	0	460	1427	35.25
21	15	15	0	463	1450	48.29
22	13	13	0	461	1436	45.03
23	12	12	0	459	1425	33.85
24	12	12	0	455	1393	43.19
25	12	12	0	460	1428	52.64
26	9	9	0	457	1407	41.48
27	11	11	0	459	1421	40.78
28	16	16	0	463	1453	38.85
29	14	14	0	462	1442	38.46
30	11	11	0	459	1421	52.81
Total	369	369	0	13800	42865	1300.69
Media	12.3	12.3	0	460	1428.83	43.36

Continúa en la página siguiente.

Tabla 5.5 – Continuación de la página anterior

Iteraciones	Conflictos	Resueltos	Diferencia	Protocolos Iniciados	Mensajes	Tiempo de ejecución (segundos)
Moda	11	11	0	459	1421	
Desviación estándar	2.83	2.83	0	3.29	22.05	8.44

Los resultados de España muestran los conflictos encontrados y resueltos con el mismo promedio de 12.3 ($\pm$ 2.83), teniéndose poca variabilidad entre los conflictos pudiéndose resolver en su totalidad con el algoritmo.

5.1.4. Reino Unido

Tabla 5.6: Resultados Reino Unido

Iteraciones	Conflictos	Resueltos	Diferencia	Protocolos Iniciados	Mensajes	Tiempo de ejecución (segundos)
1	0	0	0	236	708	5.50
2	1	1	0	237	715	5.032
3	0	0	0	236	708	5.26
4	0	0	0	236	708	4.92
5	0	0	0	236	708	5.02
6	0	0	0	236	708	5.09
7	0	0	0	236	708	4.97
8	0	0	0	236	708	4.94
9	0	0	0	236	708	4.95
10	0	0	0	236	708	4.86
11	1	1	0	237	714	4.85
12	0	0	0	236	708	5.44
13	0	0	0	236	708	5.07
14	0	0	0	236	708	5.08
15	0	0	0	236	708	4.96
16	0	0	0	236	708	5.05
17	0	0	0	236	708	4.92

Continúa en la página siguiente.

Tabla 5.6 – Continuación de la página anterior

Iteraciones	**Conflictos**	**Resueltos**	**Diferencia**	**Protocolos Iniciados**	**Mensajes**	**Tiempo de ejecución (segundos)**
18	0	0	0	236	708	5.16
19	0	0	0	236	708	4.99
20	0	0	0	236	708	4.95
21	0	0	0	236	708	5.17
22	0	0	0	236	708	5.09
23	1	1	0	237	715	5.11
24	0	0	0	236	708	5.09
25	0	0	0	236	708	5.12
26	0	0	0	236	708	5.11
27	0	0	0	236	708	4.50
28	0	0	0	236	708	5.41
29	0	0	0	236	708	4.86
30	0	0	0	236	708	5.08
Total	3	3	0	7083	21260	151.69
Media	0.1	0.1	0	236.1	708.66	5.05
Moda	0	0	0	236	708	5.08
Desviación estándar	0.30	0.30	0	0.30	2.03	0.19

Los resultados muestran en Reino Unido solo en 3 iteraciones con conflictos de los cuales se pudieron resolver teniendo un promedio de 0.1 ($\pm$ 0.31).

5.2. Resolución de conflictos

En las siguientes figuras se presentan de manera ilustrativa los resultados de las tablas mostradas anteriormente. La Figura 5.1 representa el resultado de Belice de acuerdo a los conflictos encontrados, resueltos y los no que se pudieron resolver.

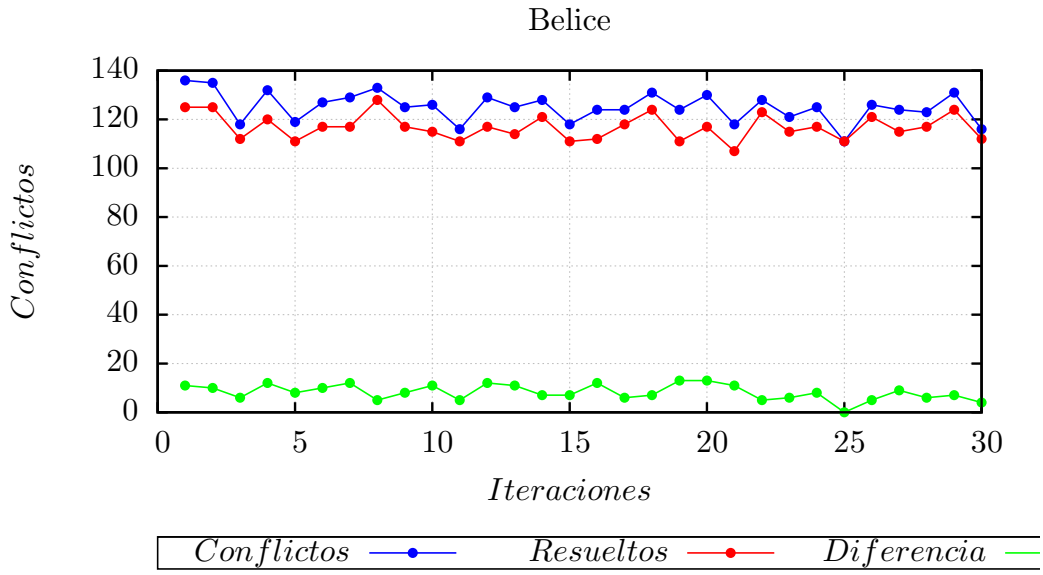


Figura 5.1: Conflictos resueltos Belice

La Figura 5.2 se evidencia los resultados obtenidos del caso de estudio de Brasil de acuerdo a los conflictos encontrados, resueltos y sin resolver.

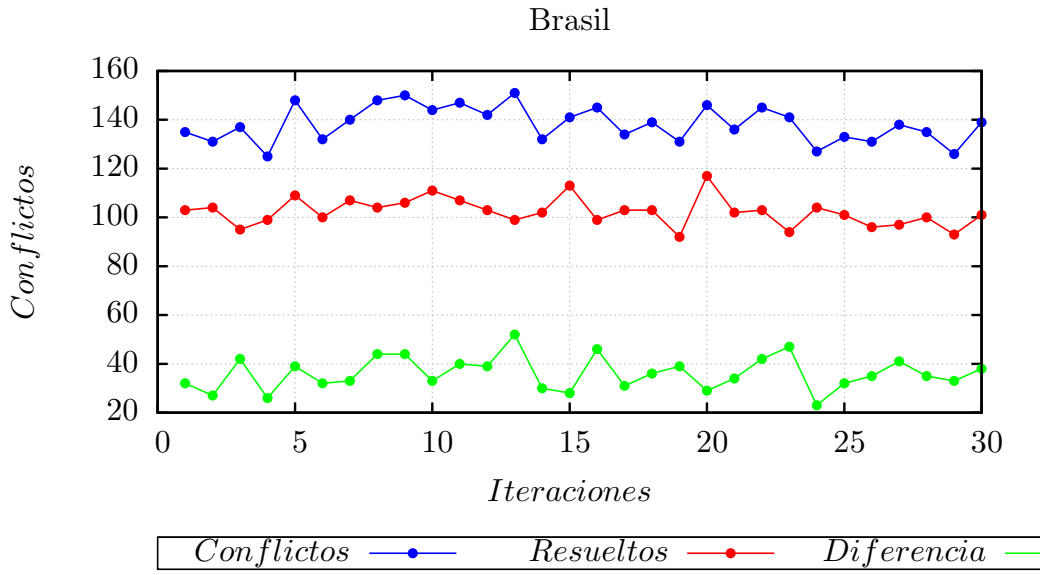


Figura 5.2: Conflictos resueltos Brasil

En la Figura 5.3 se ilustran los resultados del caso de estudio España, el cual muestra los resueltos sobrepuestos de los conflictos debido a que el algoritmo pudo resolver en su totalidad los conflictos.

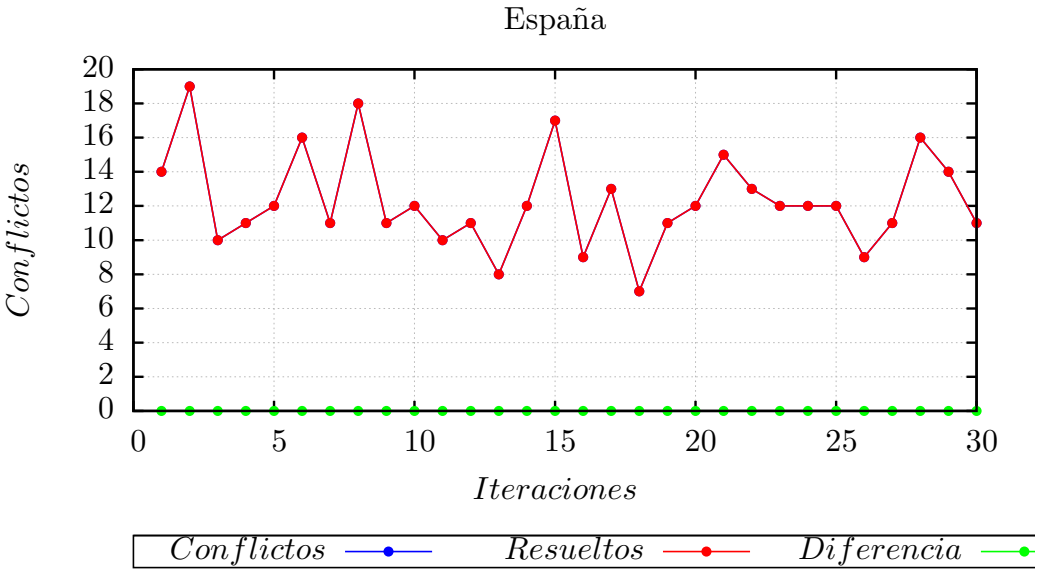


Figura 5.3: Conflictos resueltos España

En la Figura 5.4 se muestra los resultados del caso de estudio de Reino Unido con solo 3 conflictos que se pudieron resolver con una diferencia de cero de no haber conflictos en las demás iteraciones.

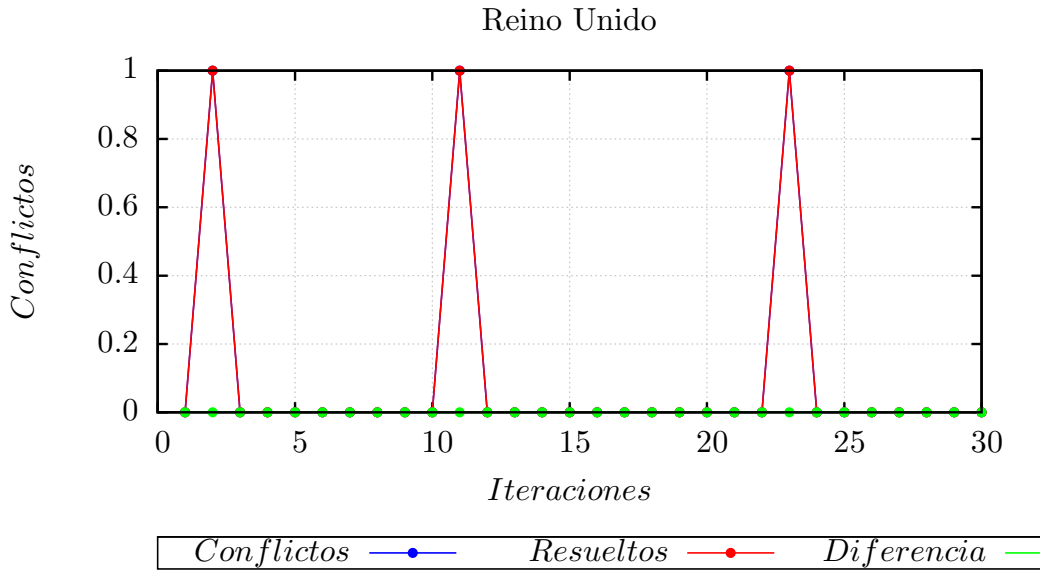


Figura 5.4: Conflictos resueltos Reino Unido

Las gráficas muestran las variabilidades de los conflictos, para los casos de Belice y Brasil las gráficas se comportan de manera similar debido a que en sus características presenta menos espacio para la asignación de las actividades de los profesores comparado con España y Reino Unido, además en los conflictos encontrados de Belice y Brasil los profesores desde un estado inicial presentan un horario que mayor se adapta a su conveniencia, imponiendo un mayor número de restricciones, mientras que en España y Reino Unido los profesores se imponen menos restricciones dando como resultado mayor variabilidad en la asignación de las actividades.

DISCUSIÓN Y CONCLUSIONES

En base a los objetivos generales y especificos planteados al inicio pudieron ser alcanzados, pues en los específicos fue posible identificar las restricciones ó problemas que presenta la mayoría de las escuelas en la programación de horarios, también fue posible optar por un formato estándar ya propuesto para estructurar y ser portable la información, así como identificar una plataforma para el desarrollo del SMA, mientras que el objetivo general fue posible mediante la metodología propuesta en la investigación con el diseño e implementación de un algoritmo y un sistema multiagente para resolver el problema de la programación de horarios escolares a través de la incorporación de restricciones genéricas para modelar diferentes casos de estudio.

Entre las contribuciones en este trabajo corresponde a un sistema que implementa un algoritmo basado en sistemas multi-agentes mediante negociación para resolver el problema de la programación de horarios escolares permitiendo la modelación de diferentes restricciones para diferentes casos de estudio, independientemente de la institución o país que se quiera resolver, siempre y cuando cumpla con las restricciones y estándares de FET.

Una de las ventajas de la propuesta es que es compatible con las especificaciones de FIPA al implementar con estándares reconocidos, al igual que presentar independencia del sistema operativo en el cual se quiera implantar. En la parte del sistema como ventaja al implementar sistemas multi-agentes fue posible considerar las propuestas iníciales de los profesores, siendo una tarea difícil para otras técnicas en las que tienen que generar soluciones, validando cada asignación.

Otra de las aportaciones de esta tesis es la de permitir que los agentes interaccionen y resuelvan conflictos de manera dinámica y no determinista, así mismo, se puede ejecutar el algoritmo para el resto de los casos de estudio de FET. Con base en los

resultados se ha logrado responder con éxito a las seis interrogantes planteadas en la investigación y las cuales se muestran a continuación:

¿Es posible modelar y resolver el problema de timetabling con sistemas multi-agentes? En base a los resultados obtenidos es evidente la modelación y resolución del problema de programación de horarios mediante negociación en el cual los agentes interaccionan y resuelven conflictos. Ante esto, surge la segunda pregunta en relación a que sí; ¿La negociación entre agentes puede llevarse a cabo bajo un enfoque egoísta ó cooperativa? Ciertamente los agentes profesores de manera inicial tienen un comportamiento egoísta al proponer un horario que mejor se adapte a su conveniencia, siempre y cuando no violen las restricciones de manera general por parte de la institución, posteriormente los agentes grupos trabajan de manera cooperativa con los agentes profesores en las negociaciones para resolver conflictos entre ellos.

Sin embargo, en este contexto, se cuestiona; ¿Es posible considerar las propuestas de los horarios de los profesores desde un estado inicial y no generar soluciones iníciales por medio de una función de evaluación al validarlas con el profesor? Con el deseo de evitar conflictos desde el inicio es recomendable considerar las propuestas de los profesores desde un estado inicial ya que permite reducir el espacio de búsqueda, es decir, darse los casos en que las propuestas de los profesores no existan conflictos, quedando resuelta la programación de horarios desde un estado inicial sin existir negociación, permitiendo con esto la reducción de un esfuerzo computacional extra al estar asignando actividades para luego evaluarlas por medio de una función de evaluación que describe la calidad de solución.

En respuesta a la cuarta pregunta acerca; ¿Es posible la re-planificación de propuestas cuantos las actividades ya han sido asignadas o están a la espera de que el grupo evalué las propuestas? Se recomienda a los agentes grupos al encontrar espacios donde existen conflicto con 2 o más profesores, este solicita a los profesores que le envíen una nueva propuesta, es decir un día y hora donde un profesor puede cambiar una actividad, pero puede darse el caso que dos grupos exista conflicto con el mismo

maestro, y este envié la misma propuesta a ambos grupos del cual ambos tiene que evaluar la propuesta y gana el que la asigne primero, mientras que el otro agente (grupo) si también gana la misma propuesta del profesor entonces el grupo verifica que no esté asignado y de ser así el grupo solicita mediante una nueva negociación a todos los profesores una nueva posición.

En relación a; ¿Con sistemas multi-agentes es posible la resolución de conflictos? En base a los resultados que se muestran en las tablas y gráficas, es evidente que con sistemas multi-agentes permite en gran medida la resolución de conflictos; resolviendo en el caso de Belice un 93.41 %, así mismo con Brasil un 73.92 %, mientras que con España un 100 % y con Reino Unido un 100 %.

Por último, e igual de importante que las anteriores: ¿El algoritmo funcionara de manera uniforme para todos los casos de estudio? La respuesta es; no, debido a las características y particulares de cada institución ya que cada una tiene sus limitantes en profesores, grupos, espacios, días, horas, etc. En base a los resultados de las Tablas 5.3, 5.4, 5.5 y 5.6 de de cada caso de estudio, Brasil es donde menos conflictos se resolvieron debido a que cuenta con un menor número de grupos y la matriz de horas por días es menor a diferencia de los demás casos, de tal manera que las asignaciones se ven más limitadas, el siguiente caso de estudio donde se resolvieron un poco más de conflictos es Belice el cual cuenta con más profesores, más grupos y una hora extra para cada día teniendo entonces un mayor espacio para la asignación.

En los casos de estudio de España y Reino Unido se pudieron resolver en gran medida debido a que tiene una mayor cantidad de grupos, días y horas para la asignación de actividades.

TRABAJO A FUTURO

Como parte del trabajo futuro se pretende mejorar el estatus de los agentes profesores y grupos, en las pestañas "Constraint" y "Activities" en identificar con un color actividades y restricciones que no presentan problema, mientras que con otro color las que presentan inconveniente, también en los estatus de los agentes implementar la pestaña "Status" indicando los pesos de las restricciones de tiempo y actividad que se están violando. Además, agregar la pestaña "Message Log" para monitorear los registros de los agentes sin la necesidad de ir al agente Sniffer.

También, implementar las restricciones faltantes y agregar en el sistema una opción en el cual el usuario pueda exportar los horarios por medio de un formato XSLT (eXtensible Stylesheet Language Transformations).Validar los resultados obtenidos con Sistemas Multi-Agentes contra las soluciones generadas por parte de FET que implementa la técnica de enjambre de partículas.

Otro punto a tratar en futuras investigaciones, será el implementar una nueva técnica para resolver los conflictos que no se pudieron resolver por medio de una técnica similar a la propuesta por los investigadores Drogoul and Dubreuil (1993) por medio de Eco-Problem-Solving, el cual se utiliza para resolver el problema N-PUZZLE por medio de agentes que interactúan. Donde la idea, es que un agente ataca a otro para solicitar el espacio, mientras que el agente que fue atacado tiende a huir y a la vez atracar a otro agente hasta cumplir con el objetivo.

Bibliografía

Abbas, A. M. and Tsang, E. P. K. (2001). Constraint-based timetabling-a case study. In *2001 ACS / IEEE International Conference on Computer Systems and Applications (AICCSA 2001), 26-29 June 2001, Beirut, Lebanon*, pages 67–72. IEEE Computer Society.

Abramson, D. (1991). Constructing school timetables using simulated annealing: Sequential and parallel algorithms.

Abramson, D. and Abela, J. (1992). A parallel genetic algorithm for solving the school timetabling problem. In *Division of Information Technology, C.S.I.R.O*, pages 1–11.

Abramson, D., Krishnamoorthy, M., and Dang, H. (1997). Simulated annealing cooling schedules for the school timetabling problem.

Alvarez-Valdes R., M. G. and M., T. J. (1996). Constructing good solutions for the spanish school timetabling problem. *Journal of the Operational Research Society*.

Alvarez-Valdés, R., Parreño, F., and Tamarit, J. (2002). A tabu search algorithm for assigning teachers to courses. *Top*, 10(2):239–259.

aSc Applied Software Consultants (2013). asc timetables. [visitado Junio 2013].

Bedoya, C. F. and Santos, M. (2003). A non-standard genetic algorithm approach to solve constrained school timetabling problems.

Beligiannis, G. N., Moschopoulos, C. N., Kaperonis, G. P., and Likothanassis, S. D. (2008). Applying evolutionary computation to the school timetabling problem: The greek case. *Computers & Operations Research*, 35(4):1265 – 1280.

Beligiannis, G. N., Moschopoulos, C. N., and Likothanassis, S. D. (2009). A genetic algorithm approach to school timetabling. *JORS*, pages 23–42.

Bellifemine, F. L., Caire, G., and Greenwood, D. (2007). *Developing Multi-Agent Systems with JADE (Wiley Series in Agent Technology)*. John Wiley & Sons.

Birbas, T., Daskalaki, S., and Housos, E. (2009). School timetabling for quality student and teacher schedules. *J. of Scheduling*, 12(2):177–197.

Birbas T., D. S. and E., H. (1997). Timetabling for greek high schools. *Journal of the Operational Research Society*, 48(2):1191–1200.

Boland, N., Hughes, B. D., Merlot, L. T. G., and Stuckey, P. J. (2008). New integer linear programming approaches for course timetabling. *Comput. Oper. Res.*, 35(7):2209–2233.

Bond, A. and Gasser, L. (1988). *Readings in Distributed Artificial Intelligence*. M. Kaufmann.

Bufé, M., Fischer, T., Gubbels, H., Häcker, C., Hasprich, O., Scheibel, C., Weicker, K., Weicker, N., Wenig, M., and Wolfangel, C. (2001). Automated solution of a highly constrained school timetabling problem - preliminary results. In Boers, E., editor, *Applications of Evolutionary Computing*, volume 2037 of *Lecture Notes in Computer Science*, pages 431–440. Springer Berlin Heidelberg.

CALS, F. (2002). Fipa communicative act library specification.

Carrasco, M. P. and Pato, M. V. (2004). A comparison of discrete and continuous neural network approaches to solve the class/teacher timetabling problem. *European Journal of Operational Research*, 153:65–79.

Carter, M. W. (1986). A survey of practical applications of examination timetabling algorithms. *Operations Research*, 34(2):193 – 202.

Carter, M. W., Laporte, G., and Chinneck, J. W. (1994). A general examination scheduling system. *Interfaces*, 24(3):109 – 120.

Colorni, A., Dorigo, M., and Maniezzo, V. (1992). Genetic algorithms: A new approach to the timetable problem. In Akgül, M., Hamacher, H., and Tüfekçi, S., editors, *Combinatorial Optimization*, volume 82 of *NATO ASI Series*, pages 235–239. Springer Berlin Heidelberg.

Colorni, A., Dorigo, M., and Maniezzo, V. (1993). A genetic algorithm to solve the timetable problem. Technical report, Politecnico di Milano, Italy.

Contreras, M., Germán, E., Chi, M., and Sheremetov, L. (2004). Design and implementation of a fipa compliant agent platform in .net. *JOURNAL OF OBJECT TECHNOLOGY*, 3(9):5–28.

Corne, D., Ross, P., and lan Fang, H. (1994). Evolutionary timetabling: Practice, prospects and work in progress. In *In Proceedings of the UK Planning and Scheduling SIG Workshop, Strathclyde.*

Costa, D. (1994). A tabu search algorithm for computing an operational timetable. *European Journal of Operational Research*, 76(1):98 – 110.

Drogoul, A. and Dubreuil, C. (1993). A distributed approach to n-puzzle solving.

Durfee, E. H., Lesser, V. R., and Corkill, D. D. (1989). Trends in cooperative distributed problem solving. *IEEE Trans. on Knowl. and Data Eng.*, 1(1):63–83.

Even, S., Itai, A., and Shamir, A. (1975). On the complexity of time table and multi-commodity flow problems. In *Foundations of Computer Science, 1975., 16th Annual Symposium on*, pages 184–193.

Ferber, J. (1999). *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1st edition.

Fernandes, C., Caldeira, J. a. P., Melicio, F., and Rosa, A. (1999). High school weekly timetabling by evolutionary algorithms. In *Proceedings of the 1999 ACM symposium on Applied computing*, SAC '99, pages 344–350, New York, NY, USA. ACM.

FET (2013). Fet free timetabling software. [visitado Junio 2013].

Filho, G. R. and Lorena, L. A. N. (2001). A constructive evolutionary approach to school timetabling. In *Proceedings of the EvoWorkshops on Applications of Evolutionary Computing*, pages 130–139, London, UK, UK. Springer-Verlag.

Finin, T., Fritzson, R., McKay, D., and McEntire, R. (1994). Kqml as an agent communication language. In *Proceedings of the Third International Conference on Information and Knowledge Management*, CIKM '94, pages 456–463, New York, NY, USA. ACM.

FIPA (2012). The foundation for intelligent physical agents. [visitado Junio 2013].

FIPA ACL (2002). Fipa acl message structure specification.

FIPA Agent Platform (2012). Publicly available agent platform implementations. [visitado Junio 2013].

FIPA AMS (2001). Fipa agent management specification.

FIPA IPLS (2000). Fipa interaction protocol library specification.

Frank Jacobsen, A. B. and Gehring, H. (2006). Timetabling at german secondary schools: tabu search versus constraint programming. In (Eds.), R., editor, *Pro-*

ceedings of the international conference on the practice and theory of automated timetabling (PATAT 2006), pages 439–442.

Garey, M. R. and Johnson, D. S. (1990). *Computers and Intractability; A Guide to the Theory of NP-Completeness*. W. H. Freeman & Co., New York, NY, USA.

Gaspero, L. D., Mizzaro, S., and Schaerf, A. (2004). A multiagent architecture for distributed course timetabling. In *Proceedings of the 5th International Conference on the Practice and Theory of Automated Timetabling*, pages 471–474.

Genesereth, M. R. and Ketchpel, S. P. (1994). Software agents. *Commun. ACM*, 37(7):48–ff.

Georgeff, M. P. and Lansky, A. L. (1987). Reactive reasoning and planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence - Volume 2*, AAAI'87, pages 677–682. AAAI Press.

Green, S., Hurst, L., Nangle, B., Cunningham, D. P., and Somers, F. (1997). Software agents: A review. Technical report.

Hertz, A. (1991). Tabu search for large scale timetabling problems. *European Journal of Operational Research*, 54(1):39 – 47.

Hertz, A. (1996). School timetabling using heuristic search. *Journal of the Operational Research Society*, 47(3):347 – 357.

Huhns, M. N. and Stephens, L. M. (1999). Multiagent systems. In Weiss, G., editor, *none*, chapter Multiagent systems and societies of agents, pages 79–120. MIT Press, Cambridge, MA, USA.

iMagic Software (2013). imagic timetable master. [visitado Junio 2013].

JADE (2013). Java agent development framework. [visitado Junio 2013].

Jennings, N. R. and Wooldridge, M. (2000). On agent-based software engineering. *Artificial Intelligence*, 117:277–296.

Junginger, W. (1986). Timetabling in germany – a survey. *Interfaces*, 16(4):66 – 74.

Kang, L. and White, G. M. (1992). A logic approach to the resolution of constraints in timetabling. *European Journal of Operational Research*, 61(3):306 – 317.

Kingston, J. H. (2001). Modelling timetabling problems with sttl. In *Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III*, PATAT '00, pages 309–321, London, UK, UK. Springer-Verlag.

Kingston, J. H. (2007). Hierarchical timetable construction. In Burke, E. and Rudová, H., editors, *Practice and Theory of Automated Timetabling VI*, volume 3867 of *Lecture Notes in Computer Science*, pages 294–307. Springer Berlin Heidelberg.

Kingston, J. H. (2010). Solving the general high school timetabling problem. In *In Proceedings of the 8th international conference on the practice and theory of automated timetabling (PATAT 2010)*, pages 517–518.

Lantiv (2013). Lantiv scheduling studio. [visitado Junio 2013].

Lara, C., Flores, J., and Calderón, F. (2008). Solving a school timetabling problem using a bee algorithm. In Gelbukh, A. and Morales, E., editors, *MICAI 2008: Advances in Artificial Intelligence*, volume 5317 of *Lecture Notes in Computer Science*, pages 664–674. Springer Berlin Heidelberg.

Lawrie, N. L. (1969). An integer linear programming model of a school timetabling problem. *The Computer Journal*, 12(4):307–316.

Šlechta, P. (2005). Decomposition and parallelization of multi-resource timetabling problems. In Burke, E. and Trick, M., editors, *Practice and Theory of Automated*

Timetabling V, volume 3616 of *Lecture Notes in Computer Science*, pages 177–189. Springer Berlin Heidelberg.

Liu, Y., Zhang, D., and Leung, S. C. (2009). A simulated annealing algorithm with a new neighborhood structure for the timetabling problem. In *Proceedings of the first ACM/SIGEVO Summit on Genetic and Evolutionary Computation*, GEC '09, pages 381–386, New York, NY, USA. ACM.

Marte, M. (2002). *Models and Algorithms for School Timetabling*. PhD thesis, none.

Mehta, N. K. (1981). The application of a graph coloring method to an examination scheduling problem. *Interfaces*, 11(5):57–65.

Meisels, A., Ell-sana', J., and Gudes, E. (1994). Decomposing and solving timetabling constraint networks. *Computational Intelligence*, 1997:486–505.

Melicio F, C. J. and A., R. (2006). Thor: a tool for school timetabling. In *Proceedings of the 6th international conference on the practice and teaching of automated timetabling (PATAT 2006)*, pages 532–535. Rudova (Eds.).

Mimosa Software Ltd. (2013). Mimosa softwre. [visitado Junio 2013].

Minsky, M. (1986). *The society of mind*. Simon & Schuster, Inc., New York, NY, USA.

Minton, S., Philips, A., Johnston, M. D., and Laird, P. (1993). Minimizing conflicts: A heuristic repair method for constraint-satisfaction and scheduling problems. *J. ARTIFICIAL INTELLIGENCE RESEARCH*, 58:161–205.

Mintzberg, H. (1979). *The structuring of organizations: A synthesis of the research*. Englewood Cliffs, NJ: Prentice-Hall.

Mohammadi, M. and Lucas, C. (2008). Cooperative co-evolution for school timetabling problem. In *Cybernetic Intelligent Systems, 2008. CIS 2008. 7th IEEE International Conference on*, pages 1–7.

Moulin, B. and Chaib-draa, B. (1996). Foundations of distributed artificial intelligence. In O'Hare, G. M. P. and Jennings, N. R., editors, *Foundations of distributed artificial intelligence*, chapter An overview of distributed artificial intelligence, pages 3–55. John Wiley & Sons, Inc., New York, NY, USA.

Nurmi, K. and Kyngas, J. (2008). A conversion scheme for turning a curriculum-based timetabling problem into a school timetabling problem. In *In Proceedings of the 7th international conference on the practice and theory of automated timetabling (PATAT 2008)*, pages 1–7.

Nwana, H. S. (1996). Software agents: An overview. *Knowledge Engineering Review*, 11:205–244.

Obit, J., Landa-Silva, D., Ouelhadj, D., Vun, T. K., and Alfred, R. (2011). Designing a multi-agent approach system for distributed course timetabling. In *Hybrid Intelligent Systems (HIS), 2011 11th International Conference on*, pages 103–108.

OpenSIS (2013). Opensis. [visitado Junio 2013].

Papoutsis K., V. C. and E., H. (2003). A column generation approach for the timetabling problem of greek high schools. *Journal of Operational Research Society*, 54(3):230–238.

Post, G., Ahmadi, S., Daskalaki, S., Kingston, J., Kyngas, J., Nurmi, C., and Ranson, D. (2012). An xml format for benchmarks in high school timetabling. *Annals of Operations Research*, 194(1):385–397.

Raghavjee, R. and Pillay, N. (2009). Evolving solutions to the school timetabling problem. In *Nature Biologically Inspired Computing, 2009. NaBIC 2009. World Congress on*, pages 1524–1527.

Raghavjee, R. and Pillay, N. (2010). Using genetic algorithms to solve the south african school timetabling problem. In *Nature and Biologically Inspired Computing (NaBIC), 2010 Second World Congress on*, pages 286–292.

Reis, L. P. and Oliveira, E. (2001). A language for specifying complete timetabling problems. In *Selected papers from the Third International Conference on Practice and Theory of Automated Timetabling III*, PATAT '00, pages 322–341, London, UK, UK. Springer-Verlag.

Ribić, S. and Konjicija, S. (2010). A two phase integer linear programming approach to solving the school timetable problem. In *Information Technology Interfaces (ITI), 2010 32nd International Conference on*, pages 651–656.

Russell, S. J., Norvig, P., Candy, J. F., Malik, J. M., and Edwards, D. D. (1996). *Artificial intelligence: a modern approach*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

Santos, H. G., Ochi, L. S., and Souza, M. J. (2005). A tabu search heuristic with efficient diversification strategies for the class/teacher timetabling problem. *J. Exp. Algorithmics*, 10.

Schaerf, A. (1996). Tabu search techniques for large high-school timetabling problems. In *IEEE TRANSACTIONS ON SYSTEMS, MAN, AND CYBERNETICS*, pages 363–368.

Schaerf, A. (1999). A survey of automated timetabling. *Artificial Intelligence Review*, 13(2):87–127.

Schmidt, G. and Ströhlein, T. (1980). Timetable construction – an annotated bibliography. *The Computer Journal*, 23(4):307–316.

Schutt, A., Feydy, T., Stuckey, P. J., and Wallace, M. G. (2010). Solving the resource constrained project scheduling problem with generalized precedences by lazy clause generation. *CoRR*, abs/1009.0347.

Srndic, N., Pandzo, E., Dervisevic, M., and Konjicija, S. (2009). The application of a parallel genetic algorithm to timetabling of elementary school classes: A coarse grained approach. In *Information, Communication and Automation Technologies, 2009. ICAT 2009. XXII International Symposium on*, pages 1–5.

Stefano, C. D. and Tettamanzi, A. G. B. (2001). An evolutionary algorithm for solving the school time-tabling problem. In *Proceedings of the EvoWorkshops 2001*, pages 452–462. Springer.

Time Finder (2013). Timefinder. [visitado Junio 2013].

Tripathy, A. (1992). Computerised decision aid for timetabling — a case analysis. *Discrete Applied Mathematics*, 35(3):313 – 323.

Tryllian (2013). Tryllian's agent development kit. [visitado Junio 2013].

Tweedale, J., Ichalkaranje, N., Sioutis, C., Jarvis, B., Consoli, A., and Phillips-Wren, G. (2007). Innovations in multi-agent systems. *Journal of Network and Computer Applications*, 30(3):1089 – 1115.

UniTime (2013). University timetabling. [visitado Junio 2013].

Valouxis, C. and Housos, E. (2003). Constraint programming approach for school timetabling. *Computers & Operations Research*, 30(10):1555 – 1572. <ce:title>Part Special Issue: Analytic Hierarchy Process</ce:title>.

Vieira, A., Rafael, M., and Scaraficci, A. (2010). A grasp strategy for a more constrained school timetabling problem.

Weiss, G., editor (1999). *Multiagent systems: a modern approach to distributed artificial intelligence*. MIT Press, Cambridge, MA, USA.

Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research*, 19(2):151 – 162.

Wilke, P., Gröbner, M., and Oster, N. (2002). A hybrid genetic algorithm for school timetabling. In *Proceedings of the 15th Australian Joint Conference on Artificial Intelligence: Advances in Artificial Intelligence*, AI '02, pages 455–464, London, UK, UK. Springer-Verlag.

Wilke, P. and Ostler, J. (2010). The erlangen advanced time tabling system (eatts) unified xml file format for the specification of time tabling systems. In Burke, editor, *Proc. PATAT 2010*, page 1, Berlin. Springer.

Wooldridge, M. and Jennings, N. R. (1995). Agent theories, architectures, and languages: a survey. In *Proceedings of the workshop on agent theories, architectures, and languages on Intelligent agents*, ECAI-94, pages 1–39, New York, NY, USA. Springer-Verlag New York, Inc.

Woolridge, M. and Wooldridge, M. J. (2001). *Introduction to Multiagent Systems*. John Wiley & Sons, Inc., New York, NY, USA.

Yang, Y., Paranjape, R., and Benedicenti, L. (2004). An examination of mobile agents system evolution in the course scheduling problem. In *Electrical and Computer Engineering, 2004. Canadian Conference on*, volume 2, pages 657–660 Vol.2.

Yang, Y., Paranjape, R., and Benedicenti, L. (2006). An agent based general solution model for the course timetabling problem. In *Proceedings of the fifth international*

joint conference on Autonomous agents and multiagent systems, AAMAS '06, pages 1430–1432, New York, NY, USA. ACM.

Yigit, T. (2007). Constraint-based school timetabling using hybrid genetic algorithms. In Basili, R. and Pazienza, M., editors, *AI*IA 2007: Artificial Intelligence and Human-Oriented Computing*, volume 4733 of *Lecture Notes in Computer Science*, pages 848–855. Springer Berlin Heidelberg.

Yoshikawa, M., Kaneko, K., Yamanouchi, T., and Watanabe, M. (1996). A constraint-based high school scheduling system. *IEEE Expert: Intelligent Systems and Their Applications*, 11(1):63–72.

Anexo A
Iteraciones

En esta sección se muestran dos Tablas para cada caso de estudio; en la primera se muestra la composición de las primeras dos iteraciones de los conflictos encontrados, resueltos y sin resolver; mientras que en la segunda Tabla se muestra la composición de las primeras dos iteraciones sobre los protocolos iniciados e intercambio de mensajes para cada agente. En las Tablas de los conflictos y protocolos iniciados e intercambio de mensajes de cada caso de estudio, los totales concuerdan con los resultados de las Tablas 5.3, 5.4, 5.5 y 5.6 respectivamente.

En la Tabla A.1 se muestra la representación de la iteración 1 y 2 para el caso de estudio de Belice donde se enlistan los grupos, se tienen los conflictos de manera inicial, los conflictos resueltos y los que no se pudieron resolver.

Tabla A.1: Número de conflictos en la Iteraciones 1 y 2 Belice

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
C1A	8	8	0	6	6	0
C1B	9	9	0	7	5	2
C1C	7	4	3	11	11	0
C1D	11	10	1	10	8	2
C1E	9	9	0	8	7	1
C1F	8	8	0	11	10	1
C1G	11	10	1	6	6	0
C2A	6	6	0	8	6	2
C2B	9	9	0	9	9	0
C2C	7	7	0	9	9	0
C2D	7	6	1	10	9	1
C2E	10	10	0	9	8	1
C2F	9	8	1	9	9	0
C3A	0	0	0	0	0	0
C3B	0	0	0	0	0	0
C3C	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.1 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
C3D	0	0	0	0	0	0
C3E	0	0	0	0	0	0
C3F	0	0	0	0	0	0
C4A	0	0	0	0	0	0
C4B	0	0	0	0	0	0
C4C Aca I	12	11	1	11	11	0
C4D	0	0	0	0	0	0
C4E Aca II	13	10	3	11	11	0
Total	**136**	**125**	**11**	**135**	**125**	**10**

En la Tabla A.2 del caso de estudio de Belice se muestra los protocolos iniciados y los intercambios de mensajes de todos los agentes involucrados para las dos primeras iteraciones.

Tabla A.2: Número de protocolos iniciados y mensajes en la iteración 1 y 2 Belice

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Coordinador	Coordinador	24	160	24	160
C1A	Grupo	20	47	18	40
C1B	Grupo	22	54	18	41
C1C	Grupo	17	39	23	60
C1D	Grupo	23	60	22	57
C1E	Grupo	22	54	21	49
C1F	Grupo	21	50	25	64
C1G	Grupo	25	64	17	37
C2A	Grupo	17	37	19	46
C2B	Grupo	20	51	20	50
C2C	Grupo	14	34	19	49
C2D	Grupo	18	41	20	51
C2E	Grupo	24	59	18	46
C2F	Grupo	20	50	19	49
C3A	Grupo	1	3	1	3
C3B	Grupo	1	3	1	3
C3C	Grupo	1	3	1	3
C3D	Grupo	1	3	1	3
C3E	Grupo	1	3	1	3

Continúa en la página siguiente.

Tabla A.2 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
C3F	Grupo	1	3	1	3
C4A	Grupo	1	3	1	3
C4B	Grupo	1	3	1	3
C4C Aca I	Grupo	22	62	29	74
C4D	Grupo	1	3	1	3
C4E Aca II	Grupo	21	65	19	57
ALMA	Profesor	1	20	1	26
AYEL	Profesor	1	32	1	37
CARI	Profesor	1	11	1	5
CHRA	Profesor	1	23	1	20
CRAN	Profesor	1	38	1	35
DIYV	Profesor	1	53	1	52
ECME	Profesor	1	19	1	21
FRKE	Profesor	1	15	1	19
GAOS	Profesor	1	20	1	9
GRLU	Profesor	1	35	1	36
GRST	Profesor	1	16	1	22
HAMA	Profesor	1	25	1	29
HAST	Profesor	1	12	1	12
HUTA	Profesor	1	31	1	29
HUTR	Profesor	1	20	1	25
JOKA	Profesor	1	28	1	28
JUTA	Profesor	1	27	1	22
LEDE	Profesor	1	33	1	24
LIBE	Profesor	1	1	1	1
LLJU	Profesor	1	26	1	25
LOLD	Profesor	1	6	1	5
MADE	Profesor	1	1	1	1
MALE	Profesor	1	8	1	8
MEAL	Profesor	1	18	1	11
MEMY	Profesor	1	1	1	1
MOBE	Profesor	1	7	1	13
O-BR	Profesor	1	10	1	11
O-MA	Profesor	1	24	1	22
PAOD	Profesor	1	15	1	20
PEBE	Profesor	1	9	1	9
PERO	Profesor	1	19	1	21
PIME	Profesor	1	5	1	5
PUCA	Profesor	1	9	1	12
PULA	Profesor	1	7	1	5
SALU	Profesor	1	9	1	21
SAMA	Profesor	1	14	1	19
TESA	Profesor	1	1	1	1

Continúa en la página siguiente.

Tabla A.2 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
TRTH	Profesor	1	10	1	11
UNEN	Profesor	1	22	1	23
UNSC	Profesor	1	30	1	25
VAAB	Profesor	1	44	1	49
WACA	Profesor	1	15	1	7
YAKI	Profesor	1	8	1	3
ZANO	Profesor	1	17	1	18
Total		**383**	**1748**	**384**	**1755**

En el caso de estudio de Brasil, en la Tabla A.3 se muestran los resultados de los conflictos encontrados, resueltos y sin resolver por los agentes grupos de las primeras dos iteraciones.

Tabla A.3: Número de conflictos en la Iteraciones 1 y 2 Brasil

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
101	10	9	1	9	7	2
102	9	6	3	6	6	0
103	9	7	2	8	8	0
104	9	6	3	7	6	1
111	8	8	0	10	9	1
201	6	5	1	8	6	2
202	9	7	2	10	7	3
203	8	4	4	10	7	3
204	9	7	2	9	7	2
205	8	5	3	6	4	2
206	6	4	2	8	7	1
301	10	9	1	8	5	3
302	6	4	2	4	2	2
303	9	7	2	9	8	1
304	11	9	2	11	9	2
305	8	6	2	8	6	2
Total	**135**	**103**	**32**	**131**	**104**	**27**

En la tabla A.4 se muestran los protocolos iniciados y mensajes iniciados de las dos primeras iteraciones del caso de estudio de Brasil.

Tabla A.4: Número de protocolos iniciados y mensajes en la iteración 1 y 2 Brasil

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Coordinador	Coordinador	16	102	16	102
101	Grupo	19	52	26	68
102	Grupo	16	36	20	50
103	Grupo	23	53	23	56
104	Grupo	20	47	20	55
111	Grupo	24	59	19	50
201	Grupo	22	53	16	36
202	Grupo	21	56	23	57
203	Grupo	20	51	16	39
204	Grupo	18	45	18	47
205	Grupo	16	37	17	42
206	Grupo	20	51	17	38
301	Grupo	22	50	21	54
302	Grupo	17	32	20	42
303	Grupo	21	54	22	54
304	Grupo	25	66	24	64
305	Grupo	19	45	19	46
Anderson	Profesor	1	40	1	38
Andreia	Profesor	1	5	1	5
Andreia2	Profesor	1	36	1	29
Aparacida	Profesor	1	34	1	37
Bruna	Profesor	1	30	1	33
Carla	Profesor	1	21	1	22
Carlos	Profesor	1	36	1	36
Cristiane	Profesor	1	32	1	29
Dulcimar	Profesor	1	29	1	34
Gilmar	Profesor	1	19	1	13
Giselle	Profesor	1	33	1	31
Helvecio	Profesor	1	31	1	38
Lima	Profesor	1	38	1	34
Luzia	Profesor	1	30	1	29
Marcelo	Profesor	1	38	1	36
Maria da Luz	Profesor	1	33	1	31
Marli	Profesor	1	33	1	35
Osvaldo	Profesor	1	19	1	14
Renata	Profesor	1	38	1	34

Continúa en la página siguiente.

Tabla A.4 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Roberto	Profesor	1	25	1	25
Rosilene	Profesor	1	35	1	36
Silvana	Profesor	1	38	1	43
Silvania	Profesor	1	31	1	30
Tania	Profesor	1	24	1	30
Terezinha	Profesor	1	21	1	20
Viviane	Profesor	1	34	1	30
Wellington	Profesor	1	30	1	33
Total		**366**	**1702**	**364**	**1705**

En la Tabla A.5, se expresan los resultados de los conflictos encontrados, resueltos y sin resolver para la iteración 1 y 2 de España de un total de 185 grupos, mientras que en la Tabla A.6 se muestran los protocolos iniciados e intercambio de mensajes de todos los agentes involucrados.

Tabla A.5: Número de conflictos en la Iteraciones 1 y 2 España

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
1º Bach A Est Asist	0	0	0	0	0	0
1º Bach A GH	0	0	0	1	1	0
1º Bach A HU	2	2	0	4	4	0
1º Bach A MU	0	0	0	0	0	0
1º Bach A Rel	0	0	0	0	0	0
1º Bach B CI	0	0	0	0	0	0
1º Bach B CS	0	0	0	0	0	0
1º Bach B Est Asist	0	0	0	0	0	0
1º Bach B Rel	0	0	0	0	0	0
1º ESO A BIL EST	0	0	0	0	0	0
1º ESO A BIL Fr	0	0	0	0	0	0
1º ESO A BIL Ref	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.5 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
1º ESO A BIL REL	0	0	0	0	0	0
1º ESO A EST	0	0	0	0	0	0
1º ESO A Fr	0	0	0	0	0	0
1º ESO A Ref	0	0	0	0	0	0
1º ESO A REL	0	0	0	0	0	0
1º ESO B BIL EST	0	0	0	0	0	0
1º ESO B BIL Fr	0	0	0	0	0	0
1º ESO B BIL Ref	0	0	0	0	0	0
1º ESO B BIL REL	0	0	0	0	0	0
1º ESO B EST	0	0	0	0	0	0
1º ESO B Fr	0	0	0	0	0	0
1º ESO B Ref	0	0	0	0	0	0
1º ESO B REL	0	0	0	0	0	0
1º ESO C BIL EST	0	0	0	0	0	0
1º ESO C BIL Fr	0	0	0	0	0	0
1º ESO C BIL Ref	0	0	0	0	0	0
1º ESO C BIL REL	0	0	0	0	0	0
1º ESO C EST	0	0	0	0	0	0
1º ESO C Fr	0	0	0	0	0	0
1º ESO C Ref	0	0	0	0	0	0
1º ESO C REL	0	0	0	0	0	0
1º ESO D BIL EST	0	0	0	0	0	0
1º ESO D BIL Fr	0	0	0	0	0	0
1º ESO D BIL Ref	0	0	0	0	0	0
1º ESO D BIL REL	0	0	0	0	0	0
1º ESO D EST	0	0	0	0	0	0
1º ESO D Fr	0	0	0	0	0	0
1º ESO D Ref	0	0	0	0	0	0
1º ESO D REL	0	0	0	0	0	0
1º ESO E Apoyo EST	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.5 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
1º ESO E Apoyo Fr	0	0	0	0	0	0
1º ESO E Apoyo Ref	0	0	0	0	0	0
1º ESO E Apoyo REL	0	0	0	0	0	0
1º ESO E EST	0	0	0	0	0	0
1º ESO E Fr	0	0	0	0	0	0
1º ESO E Ref	0	0	0	0	0	0
1º ESO E REL	0	0	0	0	0	0
1º ESO EAE EST	0	0	0	0	0	0
1º ESO EAE Fr	0	0	0	0	0	0
1º ESO EAE Ref	0	0	0	0	0	0
1º ESO EAE REL	0	0	0	0	0	0
1º ESO F1 EST	0	0	0	0	0	0
1º ESO F1 Fr	0	0	0	0	0	0
1º ESO F1 Ref	0	0	0	0	0	0
1º ESO F1 REL	0	0	0	0	0	0
1º ESO F2 EST	0	0	0	0	0	0
1º ESO F2 Fr	0	0	0	0	0	0
1º ESO F2 Ref	0	0	0	0	0	0
1º ESO F2 REL	0	0	0	0	0	0
2º Bach A ECOA	0	0	0	0	0	0
2º Bach A ECOM	0	0	0	0	0	0
2º Bach A Est	0	0	0	0	0	0
2º Bach A GHA	0	0	0	0	0	0
2º Bach A GHM	2	2	0	0	0	0
2º Bach A HU	1	1	0	1	1	0
2º Bach A HUG	0	0	0	0	0	0
2º Bach A HUI	0	0	0	0	0	0
2º Bach A MU	0	0	0	0	0	0
2º Bach A Rel	0	0	0	0	0	0
2º Bach B CSCT	1	1	0	2	2	0
2º Bach B CSF	0	0	0	0	0	0
2º Bach B DT	1	1	0	0	0	0
2º Bach B Est	0	0	0	0	0	0
2º Bach B Rel	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.5 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
2º ESO A BIL EST	0	0	0	0	0	0
2º ESO A BIL Fr	0	0	0	0	0	0
2º ESO A BIL Ref	0	0	0	0	0	0
2º ESO A BIL REL	0	0	0	0	0	0
2º ESO A EST	0	0	0	0	0	0
2º ESO A Fr	0	0	0	0	0	0
2º ESO A Ref	0	0	0	0	0	0
2º ESO A REL	0	0	0	0	0	0
2º ESO B BIL EST	0	0	0	0	0	0
2º ESO B BIL Fr	0	0	0	0	0	0
2º ESO B BIL Ref	0	0	0	0	0	0
2º ESO B BIL REL	0	0	0	0	0	0
2º ESO B EST	0	0	0	0	0	0
2º ESO B Fr	0	0	0	0	0	0
2º ESO B Ref	0	0	0	0	0	0
2º ESO B REL	0	0	0	0	0	0
2º ESO C Apoyo EST	0	0	0	0	0	0
2º ESO C Apoyo Fr	0	0	0	0	0	0
2º ESO C Apoyo Ref	0	0	0	0	0	0
2º ESO C Apoyo REL	0	0	0	0	0	0
2º ESO C EST	0	0	0	0	0	0
2º ESO C Fr	0	0	0	0	0	0
2º ESO C Ref	0	0	0	0	0	0
2º ESO C REL	0	0	0	0	0	0
2º ESO D EST	0	0	0	0	0	0
2º ESO D Fr	0	0	0	0	0	0
2º ESO D Ref	0	0	0	0	0	0
2º ESO D REL	0	0	0	0	0	0
3º ESO A BIL EST	0	0	0	0	0	0
3º ESO A BIL Fr	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.5 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
3º ESO A BIL Ref	0	0	0	0	0	0
3º ESO A BIL REL	0	0	0	0	0	0
3º ESO A EST	0	0	0	0	0	0
3º ESO A Fr	0	0	0	0	0	0
3º ESO A Ref	0	0	0	0	0	0
3º ESO A REL	0	0	0	0	0	0
3º ESO B BIL EST	0	0	0	0	0	0
3º ESO B BIL Fr	0	0	0	0	0	0
3º ESO B BIL Ref	0	0	0	0	0	0
3º ESO B BIL REL	0	0	0	0	0	0
3º ESO B EST	0	0	0	0	0	0
3º ESO B Fr	0	0	0	0	0	0
3º ESO B Ref	0	0	0	0	0	0
3º ESO B REL	0	0	0	0	0	0
3º ESO C DIV EST	0	0	0	0	0	0
3º ESO C DIV Fr	0	0	0	0	0	0
3º ESO C DIV Ref	0	0	0	0	0	0
3º ESO C DIV REL	0	0	0	0	0	0
3º ESO C EST	0	0	0	0	0	0
3º ESO C Fr	0	0	0	0	0	0
3º ESO C Ref	0	0	0	0	0	0
3º ESO C REL	0	0	0	0	0	0
3º ESO D DIV EST	0	0	0	0	0	0
3º ESO D DIV Fr	0	0	0	0	0	0
3º ESO D DIV Ref	0	0	0	0	0	0
3º ESO D DIV REL	0	0	0	0	0	0
3º ESO D EST	0	0	0	0	0	0
3º ESO D Fr	0	0	0	0	0	0
3º ESO D Ref	0	0	0	0	0	0
3º ESO D REL	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.5 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
4º ESO A BIO	2	2	0	4	4	0
4º ESO A DIB	0	0	0	0	0	0
4º ESO A DIV	0	0	0	0	0	0
4º ESO A Est Asist	0	0	0	0	0	0
4º ESO A LAT	1	1	0	0	0	0
4º ESO A Rel	0	0	0	0	0	0
4º ESO A TEC	0	0	0	1	1	0
4º ESO B BIO BIL	2	2	0	5	5	0
4º ESO B DIB BIL	0	0	0	0	0	0
4º ESO B DIV	1	1	0	0	0	0
4º ESO B Est Asist	0	0	0	0	0	0
4º ESO B LAT BIL	0	0	0	0	0	0
4º ESO B Rel	0	0	0	0	0	0
4º ESO B TEC	1	1	0	1	1	0
Total	**14**	**14**	**0**	**19**	**19**	**0**

Tabla A.6: Número de protocolos iniciados y mensajes en la iteración 1 y 2 España

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Coordinador	Coordinador	150	562	150	562
1º Bach A Est Asist	Grupo	2	4	2	4
1º Bach A GH	Grupo	4	6	5	10
1º Bach A HU	Grupo	6	14	9	24
1º Bach A MU	Grupo	1	3	1	3
1º Bach A Rel	Grupo	2	4	2	4
1º Bach B CI	Grupo	2	4	2	4
1º Bach B CS	Grupo	2	4	2	4
1º Bach B Est Asist	Grupo	2	4	2	4
1º Bach B Rel	Grupo	1	3	1	3
1º ESO A BIL EST	Grupo	2	4	2	4

Continúa en la página siguiente.

Tabla A.6 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
1º ESO A BIL Fr	Grupo	2	4	2	4
1º ESO A BIL Ref	Grupo	1	3	1	3
1º ESO A BIL REL	Grupo	1	3	1	3
1º ESO A EST	Grupo	2	4	2	4
1º ESO A Fr	Grupo	2	4	2	4
1º ESO A Ref	Grupo	3	5	3	5
1º ESO A REL	Grupo	2	4	2	4
1º ESO B BIL EST	Grupo	1	3	1	3
1º ESO B BIL Fr	Grupo	1	3	1	3
1º ESO B BIL Ref	Grupo	1	3	1	3
1º ESO B BIL REL	Grupo	1	3	1	3
1º ESO B EST	Grupo	1	3	1	3
1º ESO B Fr	Grupo	1	3	1	3
1º ESO B Ref	Grupo	1	3	1	3
1º ESO B REL	Grupo	1	3	1	3
1º ESO C BIL EST	Grupo	2	4	2	4
1º ESO C BIL Fr	Grupo	2	4	2	4
1º ESO C BIL Ref	Grupo	1	3	1	3
1º ESO C BIL REL	Grupo	2	4	2	4
1º ESO C EST	Grupo	2	4	2	4
1º ESO C Fr	Grupo	1	3	1	3
1º ESO C Ref	Grupo	3	5	3	5
1º ESO C REL	Grupo	2	4	2	4
1º ESO D BIL EST	Grupo	1	3	1	3
1º ESO D BIL Fr	Grupo	1	3	1	3
1º ESO D BIL Ref	Grupo	1	3	1	3
1º ESO D BIL REL	Grupo	1	3	1	3
1º ESO D EST	Grupo	1	3	1	3

Continúa en la página siguiente.

Tabla A.6 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
1º ESO D Fr	Grupo	1	3	1	3
1º ESO D Ref	Grupo	1	3	1	3
1º ESO D REL	Grupo	1	3	1	3
1º ESO E Apoyo EST	Grupo	1	3	1	3
1º ESO E Apoyo Fr	Grupo	1	3	1	3
1º ESO E Apoyo Ref	Grupo	1	3	1	3
1º ESO E Apoyo REL	Grupo	1	3	1	3
1º ESO E EST	Grupo	2	4	2	4
1º ESO E Fr	Grupo	1	3	1	3
1º ESO E Ref	Grupo	3	5	3	5
1º ESO E REL	Grupo	1	3	1	3
1º ESO EAE EST	Grupo	1	3	1	3
1º ESO EAE Fr	Grupo	1	3	1	3
1º ESO EAE Ref	Grupo	1	3	1	3
1º ESO EAE REL	Grupo	1	3	1	3
1º ESO F1 EST	Grupo	2	4	2	4
1º ESO F1 Fr	Grupo	1	3	1	3
1º ESO F1 Ref	Grupo	1	3	1	3
1º ESO F1 REL	Grupo	1	3	1	3
1º ESO F2 EST	Grupo	2	4	2	4
1º ESO F2 Fr	Grupo	1	3	1	3
1º ESO F2 Ref	Grupo	1	3	1	3
1º ESO F2 REL	Grupo	2	4	2	4
2º Bach A ECOA	Grupo	1	3	1	3
2º Bach A ECOM	Grupo	1	3	1	3
2º Bach A Est	Grupo	1	3	1	3
2º Bach A GHA	Grupo	1	3	1	3
2º Bach A GHM	Grupo	6	15	3	5
2º Bach A HU	Grupo	6	11	6	11
2º Bach A HUG	Grupo	2	4	2	4
2º Bach A HUI	Grupo	2	4	2	4
2º Bach A MU	Grupo	1	3	1	3
2º Bach A Rel	Grupo	1	3	1	3

Continúa en la página siguiente.

Tabla A.6 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
2º Bach B CSCT	Grupo	5	10	6	14
2º Bach B CSF	Grupo	1	3	1	3
2º Bach B DT	Grupo	4	9	3	5
2º Bach B Est	Grupo	1	3	1	3
2º Bach B Rel	Grupo	1	3	1	3
2º ESO A BIL EST	Grupo	1	3	1	3
2º ESO A BIL Fr	Grupo	2	4	2	4
2º ESO A BIL Ref	Grupo	1	3	1	3
2º ESO A BIL REL	Grupo	2	4	2	4
2º ESO A EST	Grupo	2	4	2	4
2º ESO A Fr	Grupo	2	4	2	4
2º ESO A Ref	Grupo	2	4	2	4
2º ESO A REL	Grupo	1	3	1	3
2º ESO B BIL EST	Grupo	1	3	1	3
2º ESO B BIL Fr	Grupo	1	3	1	3
2º ESO B BIL Ref	Grupo	1	3	1	3
2º ESO B BIL REL	Grupo	1	3	1	3
2º ESO B EST	Grupo	2	4	2	4
2º ESO B Fr	Grupo	1	3	1	3
2º ESO B Ref	Grupo	2	4	2	4
2º ESO B REL	Grupo	1	3	1	3
2º ESO C Apoyo EST	Grupo	1	3	1	3
2º ESO C Apoyo Fr	Grupo	1	3	1	3
2º ESO C Apoyo Ref	Grupo	1	3	1	3
2º ESO C Apoyo REL	Grupo	1	3	1	3
2º ESO C EST	Grupo	2	4	2	4
2º ESO C Fr	Grupo	2	4	2	4
2º ESO C Ref	Grupo	2	4	2	4
2º ESO C REL	Grupo	2	4	2	4
2º ESO D EST	Grupo	2	4	2	4

Continúa en la página siguiente.

Tabla A.6 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
2º ESO D Fr	Grupo	2	4	2	4
2º ESO D Ref	Grupo	2	4	2	4
2º ESO D REL	Grupo	2	4	2	4
3º ESO A BIL EST	Grupo	1	3	1	3
3º ESO A BIL Fr	Grupo	2	4	2	4
3º ESO A BIL Ref	Grupo	1	3	1	3
3º ESO A BIL REL	Grupo	1	3	1	3
3º ESO A EST	Grupo	2	4	2	4
3º ESO A Fr	Grupo	2	4	2	4
3º ESO A Ref	Grupo	3	5	3	5
3º ESO A REL	Grupo	2	4	2	4
3º ESO B BIL EST	Grupo	1	3	1	3
3º ESO B BIL Fr	Grupo	1	3	1	3
3º ESO B BIL Ref	Grupo	1	3	1	3
3º ESO B BIL REL	Grupo	1	3	1	3
3º ESO B EST	Grupo	1	3	1	3
3º ESO B Fr	Grupo	1	3	1	3
3º ESO B Ref	Grupo	1	3	1	3
3º ESO B REL	Grupo	1	3	1	3
3º ESO C DIV EST	Grupo	2	4	2	4
3º ESO C DIV Fr	Grupo	1	3	1	3
3º ESO C DIV Ref	Grupo	1	3	1	3
3º ESO C DIV REL	Grupo	1	3	1	3
3º ESO C EST	Grupo	1	3	1	3
3º ESO C Fr	Grupo	1	3	1	3
3º ESO C Ref	Grupo	2	4	2	4
3º ESO C REL	Grupo	2	4	2	4
3º ESO D DIV EST	Grupo	1	3	1	3
3º ESO D DIV Fr	Grupo	1	3	1	3

Continúa en la página siguiente.

Tabla A.6 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
3º ESO D DIV Ref	Grupo	1	3	1	3
3º ESO D DIV REL	Grupo	1	3	1	3
3º ESO D EST	Grupo	2	4	2	4
3º ESO D Fr	Grupo	1	3	1	3
3º ESO D Ref	Grupo	1	3	1	3
3º ESO D REL	Grupo	1	3	1	3
4º ESO A BIO	Grupo	7	15	10	25
4º ESO A DIB	Grupo	1	3	1	3
4º ESO A DIV	Grupo	2	4	2	4
4º ESO A Est Asist	Grupo	2	4	2	4
4º ESO A LAT	Grupo	4	9	3	5
4º ESO A Rel	Grupo	2	4	2	4
4º ESO A TEC	Grupo	3	5	4	9
4º ESO B BIO BIL	Grupo	7	15	10	27
4º ESO B DIB BIL	Grupo	2	4	2	4
4º ESO B DIV	Grupo	4	9	3	5
4º ESO B Est Asist	Grupo	2	4	2	4
4º ESO B LAT BIL	Grupo	3	5	3	5
4º ESO B Rel	Grupo	1	3	1	3
4º ESO B TEC	Grupo	4	9	4	9
Prof 1	Profesor	1	3	1	3
Prof 2	Profesor	1	1	1	1
Prof 3	Profesor	1	1	1	1
Prof 4	Profesor	1	1	1	1
Prof 5	Profesor	1	1	1	1
Prof 6	Profesor	1	1	1	1
Prof 7	Profesor	1	6	1	6
Prof 8	Profesor	1	8	1	14
Prof 9	Profesor	1	7	1	7
Prof 10	Profesor	1	1	1	1
Prof 11	Profesor	1	9	1	10
Prof 12	Profesor	1	16	1	10
Prof 13	Profesor	1	5	1	5
Prof 14	Profesor	1	4	1	7
Prof 15	Profesor	1	6	1	6
Prof 16	Profesor	1	1	1	1

Continúa en la página siguiente.

Tabla A.6 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Prof 17	Profesor	1	1	1	1
Prof 18	Profesor	1	5	1	5
Prof 19	Profesor	1	1	1	1
Prof 20	Profesor	1	1	1	1
Prof 21	Profesor	1	5	1	5
Prof 22	Profesor	1	13	1	9
Prof 23	Profesor	1	5	1	5
Prof 24	Profesor	1	5	1	7
Prof 25	Profesor	1	1	1	1
Prof 26	Profesor	1	1	1	1
Prof 27	Profesor	1	3	1	3
Prof 28	Profesor	1	9	1	9
Prof 29	Profesor	1	7	1	12
Prof 30	Profesor	1	8	1	7
Prof 31	Profesor	1	1	1	1
Prof 32	Profesor	1	3	1	3
Prof 33	Profesor	1	6	1	9
Prof 34	Profesor	1	9	1	13
Prof 35	Profesor	1	3	1	3
Prof 36	Profesor	1	13	1	13
Prof 37	Profesor	1	15	1	14
Prof 38	Profesor	1	10	1	16
Prof 39	Profesor	1	1	1	1
Prof 40	Profesor	1	4	1	7
Prof 41	Profesor	1	5	1	3
Prof 42	Profesor	1	3	1	3
Prof 43	Profesor	1	1	1	1
Prof 44	Profesor	1	3	1	3
Prof 45	Profesor	1	1	1	1
Prof 46	Profesor	1	8	1	12
Prof 47	Profesor	1	5	1	5
Prof 48	Profesor	1	23	1	23
Prof 49	Profesor	1	8	1	9
Prof 50	Profesor	1	1	1	1
Prof 51	Profesor	1	1	1	1
Prof 52	Profesor	1	8	1	5
Prof 53	Profesor	1	1	1	1
Prof 54	Profesor	1	9	1	6
Prof 55	Profesor	1	1	1	1
Prof 56	Profesor	1	5	1	5
Total		**463**	**1446**	**469**	**1486**

Por último para el caso de estudio de Reino Unido se tiene la Tabla A.7 de los conflictos y la Tabla A.8 se muestra el número de protocolos iniciados y mensajes intercambiados por agente para las 2 primeras iteraciones.

Tabla A.7: Número de conflictos en la Iteraciones 1 y 2 Reino Unido

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
Basic ??	0	0	0	0	0	0
Basic Eve A	0	0	0	0	0	0
Basic Eve B	0	0	0	0	0	0
Basic F/T A	0	0	0	0	0	0
Basic F/T B	0	0	0	0	0	0
Basic P/T A	0	0	0	0	0	0
Basic P/T B	0	0	0	1	1	0
Basic plus	0	0	0	0	0	0
Basic Skills	0	0	0	0	0	0
Begin	0	0	0	0	0	0
Begin Eve A	0	0	0	0	0	0
Begin Eve B	0	0	0	0	0	0
Begin F/T	0	0	0	0	0	0
Begin P/T A	0	0	0	0	0	0
Begin P/T B	0	0	0	0	0	0
CAE	0	0	0	0	0	0
CAE Eve	0	0	0	0	0	0
EAL group	0	0	0	0	0	0
EFL	0	0	0	0	0	0
Elem Eve A	0	0	0	0	0	0
Elem Eve B	0	0	0	0	0	0
Elem F/T	0	0	0	0	0	0
Elem P/T A	0	0	0	0	0	0
Elem P/T B	0	0	0	0	0	0
ESOL	0	0	0	0	0	0
ESOL Craft (FACE)	0	0	0	0	0	0
ESOL IT	0	0	0	0	0	0
ESOL new	0	0	0	0	0	0
FCE	0	0	0	0	0	0
FCE Eve	0	0	0	0	0	0
IELTS	0	0	0	0	0	0
IELTS Found F/T	0	0	0	0	0	0
IELTS Inter P/T	0	0	0	0	0	0
IELTS Up Int P/T	0	0	0	0	0	0
Inter Eve A	0	0	0	0	0	0
Inter Eve B	0	0	0	0	0	0
Inter F/T	0	0	0	0	0	0

Continúa en la página siguiente.

Tabla A.7 – Continuación de la página anterior

	Iteración 1			Iteración 2		
Grupo	Conflictos	Resueltos	Diferencia	Conflictos	Resueltos	Diferencia
Inter P/T	0	0	0	0	0	0
Pre-Int Eve A	0	0	0	0	0	0
Pre-Int Eve B	0	0	0	0	0	0
Pre-Int F/T	0	0	0	0	0	0
Pre-Int P/T A	0	0	0	0	0	0
Pre-Int P/T B	0	0	0	0	0	0
Staff	0	0	0	0	0	0
Upper Int P/T	0	0	0	0	0	0
Women Group	0	0	0	0	0	0
Total	**0**	**0**	**0**	**1**	**1**	**0**

Tabla A.8: Número de protocolos iniciados y mensajes en la iteración 1 y 2 Reino Unido

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Coordinador	Coordinador	46	190	46	190
Basic ??	Grupo	2	4	2	4
Basic Eve A	Grupo	3	5	3	5
Basic Eve B	Grupo	3	5	3	5
Basic F/T A	Grupo	6	8	6	8
Basic F/T B	Grupo	6	8	6	8
Basic P/T A	Grupo	5	7	5	7
Basic P/T B	Grupo	4	6	5	10
Basic plus	Grupo	2	4	2	4
Basic Skills	Grupo	4	6	4	6
Begin	Grupo	4	6	4	6
Begin Eve A	Grupo	3	5	3	5
Begin Eve B	Grupo	3	5	3	5
Begin F/T	Grupo	5	7	5	7
Begin P/T A	Grupo	4	6	4	6
Begin P/T B	Grupo	5	7	5	7
CAE	Grupo	2	4	2	4
CAE Eve	Grupo	2	4	2	4
EAL group	Grupo	3	5	3	5
EFL	Grupo	3	5	3	5
Elem Eve A	Grupo	3	5	3	5
Elem Eve B	Grupo	3	5	3	5
Elem F/T	Grupo	5	7	5	7
Elem P/T A	Grupo	4	6	4	6

Continúa en la página siguiente.

Tabla A.8 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Elem P/T B	Grupo	4	6	4	6
ESOL	Grupo	2	4	2	4
ESOL Craft (FACE)	Grupo	2	4	2	4
ESOL IT	Grupo	3	5	3	5
ESOL new	Grupo	5	7	5	7
FCE	Grupo	5	7	5	7
FCE Eve	Grupo	3	5	3	5
IELTS	Grupo	2	4	2	4
IELTS Found F/T	Grupo	2	4	2	4
IELTS Inter P/T	Grupo	2	4	2	4
IELTS Up Int P/T	Grupo	2	4	2	4
Inter Eve A	Grupo	3	5	3	5
Inter Eve B	Grupo	3	5	3	5
Inter F/T	Grupo	8	10	8	10
Inter P/T	Grupo	4	6	4	6
Pre-Int Eve A	Grupo	3	5	3	5
Pre-Int Eve B	Grupo	3	5	3	5
Pre-Int F/T	Grupo	7	9	7	9
Pre-Int P/T A	Grupo	4	6	4	6
Pre-Int P/T B	Grupo	4	6	4	6
Staff	Grupo	4	6	4	6
Upper Int P/T	Grupo	2	4	2	4
Women Group	Grupo	3	5	3	5
Adrian	Profesor	1	7	1	7
Angela	Profesor	1	19	1	19
Angie	Profesor	1	13	1	13
Ann	Profesor	1	5	1	5
Anne Rush	Profesor	1	1	1	1
Ayesha	Profesor	1	3	1	3
David	Profesor	1	19	1	19
Eugenia	Profesor	1	11	1	11
Fozia	Profesor	1	3	1	3
Inka	Profesor	1	17	1	17
John	Profesor	1	15	1	17
Julie	Profesor	1	15	1	15
Kausar	Profesor	1	15	1	15
Leonora	Profesor	1	21	1	22
Lynn	Profesor	1	5	1	5
Nargis	Profesor	1	5	1	5

Continúa en la página siguiente.

Tabla A.8 – Continuación de la página anterior

Agente	Tipo de Agente	Iteración 1		Iteración 2	
		Protocolos Iniciados	Mensajes	Protocolos Iniciados	Mensajes
Patricia	Profesor	1	13	1	13
Paul	Profesor	1	3	1	3
Pauline	Profesor	1	9	1	9
Philip	Profesor	1	23	1	23
Robert	Profesor	1	13	1	13
Rona	Profesor	1	3	1	3
Saima	Profesor	1	17	1	17
Stephen	Profesor	1	3	1	3
Vanessa	Profesor	1	3	1	3
Wilf	Profesor	1	1	1	1
Total		**236**	**708**	**237**	**715**