MODELO DE REFERENCIA PARA SIMULACIÓN CON AGENTES Y BASES DE DATOS

VIRGINIA PADILLA SIFONTES

Integrando el simulador GALATEA con un sistema de información geográfica

tutor – Dr. Jacinto Dávila Q.

Centro de Simulación y Modelos
Facultad de Ingeniería
Universidad de los Andes
Mérida, Venezuela

Departamento de Ciencia y Tecnología
Área de Informática
Universidad Nacional Experimental de Guayana
Ciudad Guayana

junio 2013 –

Virginia Padilla Sifontes: *Modelo de Referencia para Simulación con Agentes y Bases de Datos,* Integrando el simulador GALATEA con un sistema de información geográfica, Universidad Nacional Experimental de Guayana, Departamento de Ciencia y Tecnología, Ciudad Guayana; Universidad de los Andes, Centro de Simulación y Modelos ©, Mérida, Venezuela. junio 2013

A mis padres, Virgilio y Thais.

A mis hijas, Ana Virginia y Andrea Isabel.

RESUMEN

Este trabajo de investigación tiene como objetivo proponer un modelo de referencia de un geosimulador multiAgentes que describa qué es un agente, en qué consiste un sistema constituído por agentes, y como se integran los agentes, y las bases de datos en un sistema de simulación. El propósito es que este modelo de referencia de un geosimulador multiAgentes sirva para, ambos aspectos, especificar y como guía de desarrollo de aquellos sistemas que usan la tecnología de agentes para prestar servicios inéditos como los que requieren grandes almacenes de conocimiento, los simuladores de sistemas complejos y los sistemas basados en ontologías.

Los sistemas de geosimulación multiAgentes son sistemas informáticos para modelado de los fenómenos que tienen lugar en zonas geográficas a través del uso del enfoque basado en agente. Estos sistemas se han utilizado para solucionar problemas de planificación territorial, fenomenos ambientales y ecológicos, fenomenos sociales, entre otros(Murgante et al. [121]).

Nuestra hipótesis fundamental de trabajo es que la mejor manera de construir tales herramientas es a partir de una conceptualización de nociones como bases de datos, sistema de información geográfica y agentes adaptable a este dominio de aplicación y servicio. Básicamente las investigaciones en estos sistemas han estado enfocadas hacia su arquitectura, sobre cómo lograr su construcción.

Para probar los conceptos expresados en el modelo formal propuesto se presenta el diseño de un sistema en el dominio de la gestión de desastres y reducción del riesgo, orientado al servicio público y que simulará la ocurrencia de cambios en la dinámica de lluvias sobre una geografía determinada que podrían significar períodos de sequía. La construcción del sistema es un proceso planificado de reuso de otros sistemas. Por ello se han utilizado sistema de simulación funcionales como el software de simulación de sistemas hidraulicos, EpaNet, y el software para sistemas multiAgentes, GALATEA, sistema de información geográfica con una interfaz versátil como la propuesta por gvSIG y arquitecturas para integración de sistemas de simulación distribuidos como la arquitectura de alto nivel, HLA.

ABSTRACT

This research aims to propose a reference model of a multiagent geosimulador that allow describing what an agent is, what a system composed of agents, and how to integrate the agents, and databases in a simulation system. The purpose is that this reference model of a multiagent geosimulador serves to both aspects, specify and guide development of those systems using agent technology to provide unprecedented services as requiring large stores of knowledge, systems simulators complex systems based on ontologies.

Geosimulación multiagent systems are systems for modeling phenomena that occur in geographic areas through the use of agent based approach. These systems have been used to solve problems of territorial planning, environmental and ecological phenomena, social phenomena, among others (Murgante et al. [121]).

Our working hypothesis is essential that the best way to build such tools is based on a conceptualization of notions such as databases, GIS and agents suitable for this application domain and service. Basically, investigations in these systems have been focused towards its architecture, how to achieve their construction.

To show the application of the proposed formal model we presents the design of a system in the field of Disaster Management and Risk Reduction, public service oriented and will simulate the occurrence of changes in the dynamics of rain on a particular geography that could mean periods of drought. The construction of the system is a planned process of reuse of complete and entire architectures. For this purpose have been used for the construction of the experimental framework, functional simulation system as EpaNet and GALATEA, GIS such as gvSIG, and system integration architectures for distributed simulation such as HLA.

We have seen that computer programming is an art,
because it applies accumulated knowledge to the world,
because it requires skill and ingenuity, and especially
because it produces objects of beauty.

— Donald E. Knuth Knuth [97]

RECONOCIMIENTOS

Quisiera agradecer el apoyo de mi tutor, el prof. Jacinto Dávila, un maestro y guía excepcional, generoso y gentil.

A la prof. Magdiel Ablan por las horas dedicadas al estudio de los sistemas hidrológicos y por sus consejos que ayudaron a darle forma a este trabajo.

A los miembros del CESIMO por el apoyo que me prestaron durante mi estadía en ese centro de investigación.

Y a todas aquellas personas que de una u otra manera me han ayudado en el logro de este objetivo.

A todos, muchas gracias.

ÍNDICE GENERAL

LISTA DE FIGURAS

LISTA DE CUADROS

LISTA DE PROGRAMAS

ABREVIATURAS

AC Autómata Celular

API Application Programming Interface

AUML Agent Unified Modeling Language

AAII Australian AI Institute

BD Base de Datos

BDA Base de Datos Activas

BDD Base de Datos Deductiva

BDI Belief, Desire e Intention

BDOOA Base de Datos Orientada a Objeto Activa

CommonKADS Common Knowledge Acquisition Design System

CORBA Common Object Request Broker Architecture

CDDL Common Development and Distribution License

CQL Contextual Query Language

DIAS Dynamic Information Architecture System

DIF Data Interchange Format

DEVS Discrete Event System Specification

DM Declaration Management

DDM Data Distribution Management

D-W Darcy-Weisbach

ECQL Extend Contextual Query Language

EDLC Evolved Dynamic Link Compatible

E/S Entrada/Salida

ECA Evento, Condición, Acción

EM Empirical Modelling

EpaNet Environmental Protection Agency NETwork

EPSG European Petroleum Survey Group

FED Federation Execution Data

FEDEP Federation Development and Execution Process

FOM Federation Object Model

GALATEA GLIDER with Autonomous Logic-based Agents, TEmporal reasoning and Abduction

GDAL Geospatial Data Abstraction Library

GLIDER Gate, Line, Input, Decision, Exit, Resource

GDRR Gestión de Desastres y Reducción de Riesgo

GNU GNU is Not Unix

GOPRR Graph, Object, Property, Relationship, and Role

GPL General Public License

GRASS Geographic Resources Analysis Support System

gvSIG generalitat valenciana SIG

HLA High Level Architecture

IA Inteligencia Artificial

IDE Integrated Development Environment

IEEE Institute of Electrical and Electronics Engineers

I/O Input/Output

IGVSB Instituto Geográfico Venezolano Simón Bolívar

INIA Instituto Nacional de Investigaciones Agrícolas

JAI Java Advanced Imaging

JDE The JACK Development Enviroment

JVM Java Virtual Machine

LBTS Lower Bound Time Stamp

LUCITA Land-Use Change in the Amazon

MaSE Multiagent System Engineering

MAGI Multi-Agent Geosimulation Infrastructure

MAS-CommonKADS Multi Agent System- CommonKADS

MESSAGE/UML MESSAGE extends UML

MOM Management Object Model

MRA Marco de Referencia de Agentes

MSC Message Sequence Charts

MSNM Metros Sobre el Nivel del Mar

mseg mili segundos

OO Orientado a Objetos

OGC Open Geospatial Consortium

O-MaSE Open-Multiagent System Engineering

OMT Object Model Template

OPF OPEN Process Framework

PD Procesamiento Distribuido

PDA Personal Digital Assistant

PDT The Prometheus Design Tool

PSAP Predictor de Servicio de Agua Potable

QGIS Quantum Geographic Information System

RBSIM Recreation Behavior Simulation

RI Restricciones de Integridad

REST Representational State Transfer

RO Receive Order

RTI Run-Time Infrastructure

SDL Specification and Description Language

SMA Sistema Multi Agentes

SIG Sistema de Información Geográfica

SISO Simulation Interoperability Standards Organization

SOA Service Oriented Architecture

SOC Service Oriented Computing

SOM Simulation Object Model

TSO Time Stamp Ordered

UML Unified Modeling Language

USDP Unified Software Development Process

XML eXtensible Markup Language

WMS Web Map Service

WFS Web Feature Service

WCS Web Coverage Service

WSDL Web Service Description Language

Parte I

INTRODUCCIÓN Y OBJETIVOS

1

INTRODUCCIÓN Y OBJETIVOS

1.1 INTRODUCCIÓN

Un modelo de referencia es la base epistémica de una ontología, se hace referencia a la ontología en su sentido histórico, más que el sentido técnico moderno. Es la descripción de cosas y conceptos que *habitan* cierto dominio del conocimiento. El modelo de referencia de un geosimulador multiAgente tiene que describir qué es un agente, en qué consiste un sistema constituído por agentes, y como se integran los agentes, y las Base de Datos (BD) en un sistema de simulación.

La noción de agentes ha comenzado a ser extremadamente popular en el mundo tecnológico en tiempos recientes. La Inteligencia Artificial (IA) gira alrededor de este concepto Russell and Norvig [147] y existen proyectos para desarrollar un nuevo paradigma orientado a agentes con el fin de reforzar el paradigma de la OO Rumbaugh et al. [146]. En ese contexto, un agente es concebido como un objeto con una interface a su ambiente a través del cual llegan entradas al objeto y hay salidas desde el objeto. Es una capsula en el sentido usual de la OO. Lo que hace a los agentes un objeto activo y especial es su dinámica interna, la cual opera sobre sus estados internos y conecta significativamente sus entradas con su salida. Cuando esta conexión produce cierto tipo de comportamiento, se habla acerca de agentes inteligentes. Se puede decir que el objetivo del proyecto total de la IA es definir ciertos tipos de comportamientos y encontrar la manera de generarlos.

Los investigadores han buscado la ayuda de avanzadas tecnologías informáticas en el desarrollo de sistemas integrados de modelado y simulación basado en agentes y SIG, para adquirir una comprensión más profunda de la complejidad de los sistemas naturales, sistemas sociales y de planificación urbana. Evidencia de esta afirmación descansa en una área de investigación muy dinámica expresada en los trabajos de Gimblett [73], Gilbert and Troitzsch [71], Batty et al. [7], Murgante et al. [121], Torrens [158]. La integración de estas tres tecnologías, modelado y simulación, agentes de software y SIG, ha sido llamada por Benenson and M.Torrens [10]: *geosimulación*, reconociendola además como un nuevo campo de investigación

y una oportunidad para contribuir con el desarrollo de nuevas herramientas.

Este trabajo de investigación tiene como objetivo proponer un modelo de referencia de un geosimulador multiAgente que sirva para, ambos aspectos, especificar y como guía de desarrollo de sistemas de geosimulación multiagente. El modelo de referencia de un sistema multiagente es un intento de generalizar las características de GALATEA[1] (GLIDER with Autonomous Logic-based Agents, TEmporal reasoning and Abduction (GALATEA))[38, 42, 43, 44, 36], un sistema de simulación basado en multiagentes, dirigido a lograr una plataforma más integral, uniforme y bien fundamentada, para servicios de gestión del conocimiento como una manera de superar la "... debilidad relativa de la ingeniería de software en el lado de los sistemas multiagentes: existen muchas metodologías o disponibilidad de lenguajes orientados a objetos, pero no un fuerte compromiso con una semántica operacional determinada" Drogoul et al. [52].

GALATEA es un software de simulación que propone integrar, en una misma plataforma computacional, las herramientas conceptuales y concretas para simulación de eventos discretos, sistemas continuos y sistemas multiAgentes; de forma distribuida e interactiva. Este software de simulación fue desarrollado en los lenguajes de programación como JAVA [90] y SWI-Prolog [156], y proporciona a modelistas y simulistas una herramienta computacional para la simulación de sistemas por eventos discretos, continuos y multiAgentes. El software de simulación está basado en el formalismo general del modelado y simulación de eventos discretos, Discrete Event System Specification (DEVS) Zeigler et al. [172], Wainer [162], y en una teoría de Simulación de Sistemas multiAgentes que puede consultarse en Dávila et al. [44]. El estudio y la extensión de esta teoría de simulación para incluir sistemas de BD, es el objeto en este trabajo.

GALATEA es heredera de la semántica y buena parte de la sintaxis de la plataforma de simulación GLIDER: Gate, Line, Input, Decision, Exit, Resource (GLIDER)[2], desarrollado por Domingo et al. [51], que contiene un simulador DEVS, Zeigler et al. [172], Wainer [162]. Partiendo de GLIDER, GALATEA incorpora a los agentes en la semántica del lenguaje de simulación, Dávila et al. [41].

1.2 JUSTIFICACIÓN

La principal motivación para realizar esta tesis doctoral es la de proponer un modelo de referencia que sirva como base para la descripción de sistemas que utilizan la tecnología de agentes con el objeto de prestar servicios inéditos como los que requieren grandes

1 está en: http://galatea.sourceforge.net/Principal.htm
2 GLIDER fue desarrollado en el lenguaje de programación PASCAL

almacenes de conocimiento, los simuladores de sistemas complejos y los sistemas basados en ontologías.

1.3 ANTECEDENTES

Como trabajos similares a esta propuesta se puede mencionar los realizado por Beynon and Cartwright [12], Beynon et al. [13], en su propuesta de Modelado Empírico, Empirical Modelling (EM), que es un enfoque para el modelado de sistemas de simulación computacionales basado en observaciones empíricas. El grupo de investigación, EM [55], hace énfasis en el desarrollo de herramientas experimentales como base para el modelado de sistemas, y en ese sentido ha propuesto una serie de herramientas de software orientadas con ese fin. Aunque no es un antecedente relacionado directamente con la investigación, sí lo es la ídea de proponer un modelo que sirva como guía y especificación al desarrollo de sistemas de simulación computacionales complejos

También podemos mencionar los trabajos de Bailey et al. [5] y de Kowalski et al. [104], Kowalski [98], Kowalski and Sadri [101, 102], Kowalski [99], Kowalski and Sadri [103], Kowalski [100], que proporcionan una visión, desde el modelo de agentes, de los sistemas de BD y establecen la similitud entre ambos paradigmas. Estos trabajos nos permiten establecer que es posible proponer un modelo integrador de sistemas de BD y agentes que integre estos conceptos en un modelo formal más amplio, como el que se propone en este trabajo de investigación.

También como antecedente a la propuesta de extender el simulador GALATEA con una extensión geográfica, está el trabajo de Díaz [46], un experimento limitado donde se presentó un ejercicio de integración de la plataforma GALATEA con el software para SIG, Geographic Resources Analysis Support System (GRASS), GRASS Development Team [75].

1.4 PREGUNTAS DE INVESTIGACIÓN

Esta tesis doctoral pretende analizar los siguientes aspectos referidos a la geosimulación multiAgentes:

- Con respecto al modelo formal : ¿Puede extenderse la teoría de simulación de GALATEA para integre estructuras de datos complejas como las BD? si es así, entonces:
  - ¿Cuál sería el modelo formal de un geosimulador multi-Agente?
  - ¿Cómo se describen formalmente una Base de Datos Activas (BDA) y una Base de Datos Deductiva (BDD)? ¿Están las bases de datos dentro de los agentes? ¿Son los

agentes un tipo de sistema manejador de BD? ¿Cuál es la dinámica de los agentes y las BD? ¿Están conectados? ¿Se complementan?

- Con respecto a una aplicación de ejemplo: ¿Cuál es la dinámica de los sistemas de simulación con los agentes y las BD?¿Qué debe tener en cuenta un desarrollador de aplicaciones de software para construir sistemas de geosimulación y agentes? ¿Puede el modelo formal de geosimulación multiagentes, ser usado como soporte en el proceso de diseño de sistemas complejos? ¿Cuáles servicios se pueden ofrecer con sistemas de simulación integrados a SIG y agentes?

- Con respecto a los agentes y Sistema Multi Agentes (SMA): ¿Cómo las metodologías de desarrollo de SMA describen a los agentes y a los SMA? ¿Qué lenguajes utilizan las diferentes metodologías para describir un SMA? ¿Se puede establecer un modelo general de agentes que incluya las diversas caracterizaciones que presentan las metodologías para SMA?

- Con respecto a la teoría de simulacion multiAgentes: ¿Cuál es el modelo formal en el que se basa el software para simulación GALATEA?

- Con respecto a la geosimulación multiAgentes: ¿Existen un modelo formal que describa un geosimulador multiAgente? ¿Qué tipo de limitaciones se busca superar al integrar sistemas de simulación con SIG y agentes?

- Con respecto a la computación orientada a servicios (Service Oriented Computing (SOC)): ¿Qué es la SOC? ¿Cuáles son los conceptos asociados a la SOC? ¿Están los SMA relacionados con la SOC?

1.5 OBJETIVOS

Objetivo General:

Diseñar un Modelo de Referencia para un sistema de simulación integrado con agentes y bases de datos.

Objetivos Específicos:

En términos generales se exploran las áreas de conocimiento involucradas en la investigación. Esto proporciona el soporte apropiado para integrar estos conceptos en un modelo común. Luego, se propone un modelo formal que abarque estos conceptos. Por último se

analiza y diseña una aplicación representativa del tipo de situación que se quiere abordar con el modelo propuesto.

A continuación se formalizan los objetivos específicos:

- Establecer una caracterización de los agente de software.

  Investigar el fundamento teórico de los agentes de software para conocer los conceptos que caracterizán la arquitectura de un agente. Establecer un modelo general que caracterize a los agentes de software.

- Determinar cómo caracterizan a los agentes las diversas metodologías para desarrollo de SMA.

  Se evaluan metodologías reconocidas para determinar como estas metodologías para desarrollo de SMA definen a los agentes y que tipo de lenguaje usan para su formalización. Se comparan las metodologías para desarrollo de SMA con el modelo propuesto en el objetivo anterior con el objeto de establecer el caracter general de la caracterización realizada.

- Investigar teorías multiAgentes para sistema de simulación.

  Consiste en estudiar la base teórica que fundamenta un sistema de simulación, especificamente el simulador GALATEA propuesta por Dávila and Uzcátegui [39], y su descripción formal, para determinar si los modelos asociados con agentes y BD están explicados en el modelo formal de un sistema de simulación.

- Investigar los avances en la integración de SIG, agentes de software y sistemas de simulación.

  Consiste en revisar los avances que ha habido en el área de de los SIG y agentes de software, aplicaciones que se han desarrollado, software utilizado, entre otros aspectos relevantes.

- Proponer un modelo de referencia que incluya los conceptos asociados a agentes, bases de datos y sistemas de simulación.

  Presentar el modelo de referencia para un geosimulador que explique como este modelo incluye los modelos formales elaborados previamente.

- Demostrar que el modelo de referencia del geosimulador explica los modelos formales que describen a las BDA y BDD.

  - Determinar qué modelos formales se usan para describir BDA y BDD con el propósito de determinar la posibilidad de integrar ambos modelos en un modelo común que las caracterize.

- Comparar este modelo formal de base de datos con el modelo formal del geosimulador para establecer la factibilidad de que el modelo formal del geosimulador sea un modelo más general.

- Evaluar el modelo de referencia propuesto en este trabajo.

  Se diseña una aplicación que se usa para determinar si el modelo de referencia propuesto explica los conceptos involucrados en la aplicación. El resultado de esta evaluación será muy importante porque serve para identificar las ventajas y desventajas del modelo de referencia, y también, el planteamiento de futuras líneas de investigación que puedan derivarse para mejorar este trabajo.

  Se diseña un servicio asociado a la aplicación para evaluar que tipo de tecnología puede usarse para el desarrollo de aplicaciones orientadas a servicios que contemplen simuladores, SIG y agentes.

1.6 DESCRIPCIÓN BREVE DE CADA CAPÍTULO

A continuación se hará un breve descripción de los capítulos que conforman esta memoria.

- El capítulo 2 tiene tres secciones: *Agentes, Metodologías para desarrollo de sistemas multiAgentes, Teoría de Simulación multiAgentes y Geosimulación multiAgentes.*

  Agentes inicia con el estudio de estudia la base teórica de los agentes de software y se define un MRA.

  Luego en *Metodologías para desarrollo de sistemas multiAgentes,* se usa el MRA como un marco comparativo para evaluar metodologías industriales, tratando de mostrar como cada concepto está representado por la metodología. La meta (ambiciosa de por sí) es el de aproximarse a una meta-metodología, basada en una teoría multiagentes general (Dávila [33], Dávila and Tucci [37], Dávila and Uzcátegui [38, 39], Dávila et al. [43], Dávila and Uzcátegui [40], Dávila et al. [42, 44]) que permita explorar estrategias concretas para diseñar, generar y controlar sistemas multiagentes basados en conocimiento.

  Las metodologías bajo escrutinio han sido:

  - AAII propuesta por Kinny et al. [96], Kinny [95]
  - GAIA propuesta por Zambonelli et al. [171]
  - MaSE propuesta por Mark [113]
  - Prometheus propuesta por Padgham and Winikoff [127]
  - MESSAGE/UML propuesta por Caire et al. [21]

- INGENIAS propuesta por Gómez-Sanz and Pavón [74]
- Tropos propuesta por Bresciani et al. [18]
- MAS-CommonKADS propuesto por Iglesias et al. [83], y
- O-MaSE propuesta por Garcia-Ojeda et al. [68].

En *Teoría de Simulación multiAgentes* se presenta la teoría de simulación usada en la construcción de GALATEA. La teoría de simulación multiAgentes propuesta por Dávila et al. [44] es un modelo formal construidos sobre trabajos previos en IA (Ferber and Muller [61]) y Simulación (Dávila and Uzcátegui [39, 40], Dávila et al. [43, 42]). Esta teoría ha sido plasmada en el desarrollo del sistema simulador multiAgentes, GALATEA propuesta por Dávila and Uzcátegui [39], Dávila et al. [44].

En *Geosimulación multiAgentes,* se hace referencia a esta área de investigación que integra tres tecnologías: modelado y simulación, agentes de software y SIG. La integración de la tecnología de agentes con los SIG esta siendo propuesta como una solución para superar las limitaciones de los SIG. Entre estas limitaciones atribuidas a los SIG están limitados para el uso en modelado de sistemas dinámicos, el manejo del tiempo, necesidad de incorporar fenómenos de comportamiento humano y datos proveniente de las ciencias sociales em modelos espaciales. Finalmente, se hace referencia a trabajos de investigación en geosimulación multiAgentes, ellos son:

- SIG Y MODELADO BASADO EN AGENTES de Westervelt [163], se establecen criterios para evaluar la integración de los SIG y sistemas de simulación
- UNA VISIÓN ORIENTADA A AGENTE EN UN SIG CON SISTEMAS URBANOS Y AMBIENTALES de Jiang and Gimblett [92], donde explora el paradigma basado en agente para modelar sistemas y ambientes urbanos.
- AGENTES MÓVILES CON INTELIGENCIA ESPACIAL presentado por Itami [87], y donde se describe el sistema llamado Recreation Behavior Simulation (RBSIM), un programa de computadora que simula el *comportamiento recreativo* de los humanos en ambientes naturales.
- SIMULACIÓN BASADO EN AGENTES PARA LA TOMA DE DECISIONES Y CAMBIO DE USO DEL SUELO donde Lim et al. [107] presenta el sistema de simulación LUCITA: Land-Use Change in the Amazon (LUCITA), un sistema de simulación multiAgentes compuesto de dos sub-modelos que interactúan a través de un ambiente raster referenciado espacialmente.
- INFRAESTRUCTURA DE SIMULACIÓN MULTIAGENTE PARA PLANIFICACIÓN, donde Blecic et al. [14] presenta a MA-

GI: Multi-Agent Geosimulation Infrastructure (MAGI), un meta-modelo que representa una teoría formal de una geografía con agentes y objetos en ella. Posteriormente, usaremos MAGI para ampliar la teoría de simulación de GALATEA.

Y por último, en *Computación Orientada a Servicios*, se presentan los conceptos básicos asociados a este paradígma.

- El capítulo 3 está compuesto de dos secciones: *El Modelo Formal para geosimulación multiAgentes* y *Bases de Datos*.

  En *El Modelo Formal para geosimulación multiAgentes*, se propone un modelo formal derivado de la integración de la teoría multiagentes para simulación de GALATEA, presentada por Dávila et al. [44], y la teoría de MAGI presentada por Blecic et al. [14] para producir una teoría multiAgentes para geosimulación, la cual explica la relación entre agentes y SIG; como una herramienta para simular sistemas espaciales complejos.

  En la siguiente sección, *Bases de Datos*, se dedica a clarificar, por medio del modelo de referencia, la relación entre sistemas multiagentes y un modelo formal de BD que incluya las características de las BDA y BDD para demostrar que el modelo formal de geosimulación multiAgentes es un modelo más general que contiene BD y agentes.

- En el capítulo 4 se realiza la validación experimental del Modelo Formal de Geosimulación multiAgentes.

  En este cápitulo se presenta el diseño de un sistema de simulación con BD, SIG y agentes, con el objeto de mostrar como el modelo formal expuesto en el capítulo Capítulo 3 es un soporte para el proceso de modelado de un sistema complejo.

  El marco experimental se ha pensado como un proceso planificado de reuso de sistemas completos y arquitecturas enteras. El uso de sistemas y arquitecturas existentes permite asegurar la validez científica en los resultados obtenidos, así como los aspectos referidos al control y la verificación de los sistemas.

  Se inicia este capítulo presentando el dominio de la aplicación que es el área de la GDRR, Valdés [159], Cushla and Ochoa [32], Lavell [106]. Luego se presenta el diseño general del sistema para GDRR y los objetivos establecidos para el desarrollo del sistema. Siguiendo la metodología de la *programación extrema* se construyó una historia de usuario y con ayuda del modelo de formal de geosimulación multiAgentes, se establecieron tres objetivos para la construcción del sistema: (1) Desarrollar una interfaz al usuario que muestre al PSAP, (3) Codificar un agente que en la simulación sea el encargado de gestionar la salida del

estanque de almacenamiento y (4) Diseño de la interfaz gráfica para la descripción del escenario climático.

- (1) En la fase *Desarrollar una interfaz al usuario que muestre al* PSAP , se establecieron tres sub-objetivos: (a) *Implementar el modelo de simulación de la dinámica del acueducto,* (b) *Integración del* PSAP *en un* SIG, y (c) *Evaluar la construcción de un servicio con* SIG *y el* PSAP.
  - En *Implementar el modelo de simulación de la dinámica del acueducto* se construye el modelo de simulación para el Acueducto "La Ceibita", usando el software de simulación Environmental Protection Agency NETwork (EpaNet), EpaNet [56] . Luego, se generan dos modelo de simulación del Acueducto "La Ceibita" .
  - En *Integración del* PSAP *en un* SIG, se ha construido una interfaz gráfica usando el software gvSIG, gvSIG [77], usando servicios remotos de tipo WMS del IGVSB, IGVSB [84]; y usando, también, la versatilidad del software para la construcción de la capa vectorial para señalizar los poblados que se surten del acueducto. Esta interfaz gráfica forma parte de la extensión en gvSIG para el PSAP que fue desarrollada en esta fase. La extensión permite al usuario consultar cuanto es la demanda promedio semanal de agua potable, medida en lts/seg, que el acueducto proporciona a una determinada población. La extensión PSAP para gvSIG versión 1.11 puede descargarse desde `https://simulants.svn.sourceforge.net/svnroot/simulants/` y las instrucciones para su instalación pueden verse en Sección D.3.
  - En *Evaluar la construcción de un servicio con* SIG y el PSAP, se estableció que existe dos maneras de desarrollar un servicio en la Web con gvSIG:
    1. La integración del PSAP con otras tecnologías como Service Oriented Architecture (SOA) SOA [155], JavaScript Flanagan [65] y Extend Contextual Query Language (ECQL) este último es una versión extendida del estándard Contextual Query Language (CQL) CQL [30].
    2. Desarrollar un servicio usando la extensión que ofrece gvSIG para desarrollo de cliente móviles, gvSIG Mobile. Este desarrollo, servicio PSAP móvil, puede descargarse desde `https://simulants.svn.sourceforge.net/svnroot/simulants/` y las instrucciones para su instalación pueden verse en Sec-

ción D.2. EL servicio PSAP móvil se ejecuta en un emulador.

- (3) En la fase *Diseñar un agente que en la simulación sea el encargado de gestionar la salida del tanque de almacenamiento* se consideró que la integración de diversos simuladores, como el caso de EpaNet y GALATEA requeriría una plataforma de integración que puede ser proporcionada por el estándard para arquitecturas de alto nivel como HLA, DoD [48]. Bajo estas consideraciones, se establecieron los siguientes sub-objetivos para esta fase:
  - En el sub-objetivo *Construcción de la federación*, se evaluaron diversas implementaciones del estándard HLA y se decidio usar PoRTIco, PoRTIco [132], para crear una federación constituida de dos federados: *Mi Acueducto* y *Mi Administrador*. En el federado *Mi Acueducto* está contenido el modelo del Acueducto "La Ceibita", y el federado *Mi Administrador* es la estructura que servirá al agente gestionador de la salida del tanque de almacenamiento.
  - En el sub-objetivo *Integrar de la federación en el* SIG, se integra la federación, desarrollada en el paso anterior, en el software gvSIG.
  - En el sub-objetivo *Construcción del agente gestionador de la salida del tanque de almacenamiento*, se diseña y construye el agente que se ha llamado agente regulador. El agente regulador es un agente reactivo que observa su medio ambiente y actúa en consecuencia.
- (4) En la fase *Diseñar un agente que gestione los escenarios de simulación del acueducto*, se diseña la interfaz gráfica le permite al usuario escoger un escenario climático y, a partir del escenario climático seleccionado, un agente consultor construye los escenarios de simulación que van a generar diversos datos que se utilizan como entrada al proceso de simulación del acueducto "La Ceibita". Para integrar los datos seleccionado por el usuario con la plataforma construida se ha diseñado el federado *Mi Consultor* que se integra con la federación *Mi Federación*.

  El agente consultor tiene como función la de gestionar el escenario climático construido.

- El capítulo 5 se presentan las conclusiones que se han obtenido y las líneas de investigación que surgen de este trabajo. Éstas son el resultado, tanto del estudio de los agentes de software, de las metodologías para desarrollo de SMA, de la teoría de simulación multiAgentes, de la geosimulación multiAgentes y

de la SOC. La conclusión principal del trabajo es que *El Modelo Formal de Geosimulación multiAgente es un soporte en el proceso de desarrollo de sistemas de simulación complejos que integren a los* SIG *y los agentes de software.*

Adicionalmente, se presentan los trabajos futuros y las líneas de trabajo que se abren para complementar el alcance de esta investigación.

Parte II

MARCO TEÓRICO

2

FUNDAMENTO TEÓRICO

Las claves para el planteamiento del modelo de referencia que permite describir un sistema de simulación con agentes y BD son:

- Los agentes de software. Es un recorrido por diversas metodologías que nos permite evaluar como definen a los agentes de software, que métodos usan para desarrollar SMA y que tipo de lenguaje utilizan para formalizar el diseño del sistema.
- Teoría de simulación. Específicamente se revisa el modelo formal propuesto por Dávila et al. [44] para sustentar el desarrollo del sistema simulador GALATEA .
- GeoSimulación. Es una revisión de los avances en esta área que integra a los SIG, agentes de software y Sistemas de modelado y simulación; los tipos de aplicaciones desarrolladas, metodologías y el tipo de lenguaje utilizado para formalizar el sistema.
- SOC. Es una revisión de los conceptos asociados a los servicios para establecer si hay relación con los agentes de software.

2.1 AGENTES.

La noción de agentes ha llegado a ser extremadamente popular en el mundo tecnológico en tiempos recientes. La IA gira alrededor de este concepto Russell and Norvig [147] y existen proyectos para desarrollar un nuevo paradigma orientado a agentes con el fin de reforzar el paradigma de la orientación a objetos, OO, Rumbaugh et al. [146]. En ese contexto, Russell and Norvig [147] concibe a una agente como un objeto con una interface a su ambiente a través del cual llegan entradas al objeto y hay salidas desde el objeto. Es una cápsula en el sentido usual de la OO.

Lo que hace a un agente un objeto activo y especial es su dinámica interna, la cual opera sobre sus estados internos y conecta significativamente sus entradas con sus salidas. Cuando esta conexión produce cierto tipo de comportamiento, entonces, Russell and Norvig [147] hablan acerca de *agentes inteligentes* Se puede decir

que el objetivo del proyecto total de la IA es definir ciertos tipos de comportamientos y encontrar la manera de generarlos.

Para hacer que un agente sea un objeto de un tipo específico, el paradigma de la orientación a agentes (Kowalski and Sadri [101], Bratman [17], Shoham [151], Russell and Norvig [147]) prescribe un conjunto de estructuras para ese estado interno. A continuación se mostrará un conjunto de definiciones de esas estructuras internas y externas (con respecto al agente) descritas por Russell and Norvig [147], Bradshaw [16] y Dávila [33]; y que definen a un SMA, a partir de un modelo de un agente racional. Es importante destacar que la estructura es un esquema de un modelo que se ha descrito en varias formalizaciones por Kowalski and Sadri [101, 102], Kowalski [99], Kowalski and Sadri [103], Kowalski [100].

MRA:

- Estructuras de estado interna :
  - creencias: lo que el agente conoce acerca de su ambiente y de otros agentes
  - metas: objetivos o situaciones que al agente o a su diseñadora le gustaría lograr o causar, usualmente por medio de la ejecución de un plan. Estas pueden clasificarse en:
    - metas de mantenimiento, que representan una relación permanente entre el agente y su ambiente, en la forma de reglas condicionales.
    - metas de logro, son objetivos particulares que el agente trata de lograr en algún punto del tiempo.
  - intenciones: metas.
  - preferencias: un distinguido conjunto de metas. Las preferencias de un agente para cierto estado pueden ser incorporadas como parte de una función de utilidad, a la cual se le asigna valores para expresar cuan deseable es cada estado o meta.
  - compromisos: las obligaciones (metas) adquiridas por un agente o acordadas con otro agente y a las cuales el agente está sujeto.
  - planes: las sequencias de acciones que un agente puede ejecutar con el objeto de lograr sus metas.
  - historia: el agente guarda información concerniente a su propio registro de percepciones.
- Dinámica Interna:
  - mecanismo de actualización.
  - mecanismo de activación del agente.

- mecanismo de planificación y ejecución del agente, el cual incluye un motor de inferencia y un mecanismo de toma de decisiones.

- Estado Externo:
  - roles: funciones organizacionales realizadas por el agente en un sistema multiagente. Usualmente estan representados por medio de las metas.
  - casos de uso: descripción del comportamiento del agente.
- Interfaz:
  - aptitudes: el agente posee la funcionalidad e información correcta con el objeto de ser capaz de interrelacionarse con el medio ambiente que lo rodea. Está definido por dos atributos:
    - habilidades: lo que el agente sabe que puede hacer como resultado de la combinación de sus percepciones y creencias
    - capacidades: el conjunto de acciones que el agente puede realizar, bajo ciertas precondiciones proporcionadas.

Se ha usado este MRA como un marco comparativo para evaluar metodologías industriales, tratando de mostrar como cada concepto está representado por la metodología. La meta (ambiciosa de por sí) es el de aproximarse a una meta-metodología, basada en una teoría multiagentes general con base en los trabajos de Dávila [33], Dávila and Uzcátegui [38], Dávila and Tucci [37], Dávila and Uzcátegui [39], Dávila et al. [42, 43], Dávila and Uzcátegui [40], Dávila et al. [44], que permita explorar estrategias concretas para diseñar, generar y controlar sistemas multiagentes.

2.1.0.1 *Conclusión.*

El MRA propuesto es un modelo amplio que incluye las diversas caraterizaciones asociados a los agentes de software. En la siguiente sección se demostrará como este MRA propuesto es un modelo que incluye las definiciones establecidas por diversas metodologías para desarrollo de sistemas multiAgentes.

2.2 METODOLOGÍAS PARA DESARROLLO DE SISTEMAS MULTIAGENTES.

En esta parte se describen un conjunto de metodología reconocidas con el objeto de establecer si los conceptos descriptores de un agente, descritos en 2.1, están contemplados dentro de las metodologías.

Las metodologías bajo escrutinio han sido: AAII propuesta por Kinny et al. [96], Kinny [95], GAIA propuesta por Zambonelli et al. [171], MaSE propuesta por Mark [113] , Prometheus propuesta por Padgham and Winikoff [127], MESSAGE/UML propuesta por Caire et al. [21], INGENIAS propuesta por Gómez-Sanz and Pavón [74], Tropos propuesta por Bresciani et al. [18], MAS-CommonKADS propuesto por Iglesias et al. [83], y finalmente O-MaSE propuesta por Garcia-Ojeda et al. [68].

2.2.1 *AAII.*

La metodología AAII fue propuesta por Kinny et al. [96], Kinny [95]. AAII modela agentes del tipo Belief, Desire e Intention (BDI) elaborado por Rao and Georgeff [140, 139, 138]. El modelo de agente BDI tiene su base filosófica en las teorías de la intención de Bratman [17]. La metodología AAII ofrece una teoría lógica que define *creencias*, *deseos* e *intenciones* usando lógica modal.

Para la metodología AAII las *creencias* representan la información acerca del ambiente que puede tener el agente, los *deseos* corresponden con el conjunto de las metas del agente y los eventos que puede responder, y las *intenciones* son los planes para alcanzar una meta, todos estos atributos están incluidos en el MRA descrito en 2.1.

AAII describe a los agentes a través de dos modelos:

- Modelo Externo: consiste en la descomposición del sistema en agentes y sus interacciones. Se definen los roles que cumplen los agente, sus responsabilidades y los servicios que prestan. Para ello se construye un:
  - Modelo de Agentes que describe la relación jerárquica entre agentes y las relaciones entre agentes.
  - Modelo de Interacción que describe las responsabilidades, servicios e interacciones entre agentes y sistemas externos.
- Modelo Interno: el modelo interno esta relacionado con las creencias, deseos e intenciones del agente. Los agentes BDI están definidos través de tres modelos:
  - Modelo de Creencias: describe las información del agente acerca del ambiente y el estado interno que un agente puede mantener, y las acciones que puede realizar.
  - Modelo de Metas: describe las metas que puede perseguir el agente, así como los eventos que un agente puede responder.
  - Modelo de Planes: describe los planes que un agente usa para alcanzar sus metas.

Los pasos de la metodología son los siguientes:

- Con los roles (funcionales y organizacionales) relevantes en el dominio de la aplicación, se identifican los agentes, y se desarrolla una jerarquía de clases de agentes.
- Se identifican las responsabilidades asociadas con cada rol, los servicios requeridos y suministrados por el rol y se determinan las metas asociadas con cada servicio.
- Para cada meta, se determina el plan que se puede usar para lograrlas y las condiciones del contexto bajo el cual cada plan es apropiado.
- Se construye las creencias del sistema analizando los contextos y las condiciones que controlan la ejecución de actividades y acciones, y se descomponen en partes de las creencias del sistema.

Estos pasos se repiten en un proceso de interactivo e incremental con realimentación, hasta obtener un modelo más refinado y elaborado.

La metodología AAII usa diagramas provenientes de las técnicas de modelado de objetos, (Object Modelling Technique) de Jacobson [89], para describir el MODELO DE CREENCIAS. Las clases de agentes tiene asociado creencias, metas y planes, pero no poseen operaciones. El MODELO DE CREENCIAS extiende el MODELO DE AGENTES al agregarle un *conjunto de creencias* y uno o más *estados de creencias*. Los conjuntos de creencias son predicados, cuyos argumentos son términos de un universo predefinidos, y símbolos de funciones definidas por el usuario. Los predicados y las funciones se derivan de la definición de clases e instancias. Un estado de creencias es un conjunto de diagramas de instancias que definen una instancia particular de un conjunto de creencias.

El MODELO DE PLANES usa diagramas de Harel [79] al cual se le añade la noción de falla del plan. Este modelo son grafos con tres nodos: *estado inicial*, *estado final* (con dos estado: *pasa* o *falla*) y *estado interno* (caracterizado como *estado pasivo* y como *actividad* si es un estado activo). La metodología usa el lenguaje de comunicación de agentes KQML de Finin and Fritzson [63], Kinny [95] para documentar el protocolo de interacción entre agentes.

Conclusión.

La metodología AAII caracteriza al agente por medio de un MODELO EXTERNO y un MODELO INTERNO. En el MODELO EXTERNO se definen *roles*, *responsabilidades*, y *servicios*. En el MRA descrito en 2.1, los roles son parte del estado externo del agente; las responsabilidades son obligaciones del agente, que están definidad en su estado interno; y

los servicios forman parte de la definición de la interfaz del agente. El MODELO INTERNO hace énfasis a tres aspectos que contempla el nivel interno del MRA: *creencias* que representan la información acerca del ambiente que puede tener el agente, *deseos* que corresponden con las metas del agente, y las *intenciones* como los planes que el agente puede emplear para lograr sus metas.

2.2.2 *GAIA.*

GAIA es un metodología con un alto nivel de abstracción propuesta por Zambonelli et al. [171]. Para modelar SMA, la metodología GAIA utiliza el concepto de *Metáfora Organizacional* , esquematizado en la figura 2.1, para describir a los agentes inmersos en una o varias organizaciones, jugando uno o varios roles, con múltiples interacciones entre los agentes, e interactuando con su medio ambiente.

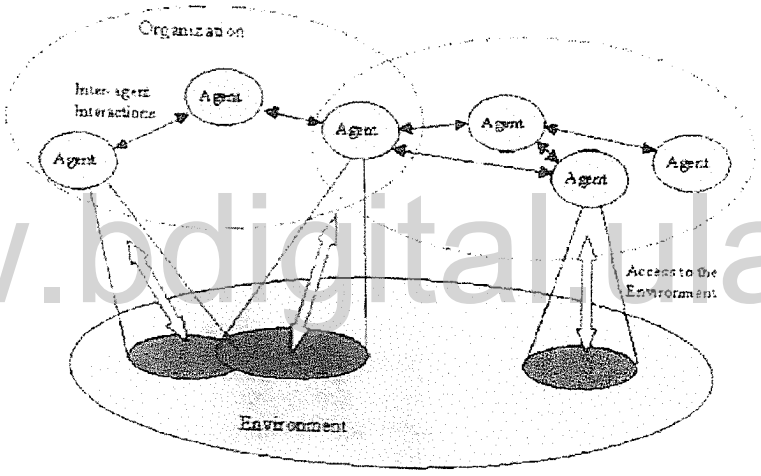


Figura 2.1: Sistemas multiAgentes como Organización Computacional (tomado de Zambonelli et al. [171]Jennings [91])

Un rol se define por cuatro atributos: *responsabilidad*, *permisos*, *actividades* y *protocolos*:

- La *responsabilidad* determina el comportamiento esperado del rol y es el atributo clave asociado con el rol. Esta caracterizado por: *propiedad de vitalidad* y *propiedad de seguridad* Manna and Pnueli [112].
  - La *propiedad de vitalidad* describe aquellos estados que un agente debe causar, bajo ciertas condiciones (estados donde *algo bueno pasa*).

- La *propiedad de seguridad*, son aquellos estados que permanecen invariantes, estados donde *nada malo pasa*.

- *Los permisos* que son los derechos asociados con el rol para cumplir con las responsabilidades.
- Las *actividades* de un rol son cálculos asociados al rol que definen las acciones de un agente *sin* interactuar con otros agente.
- *Los protocolos* establecen la interacción entre los agentes.

La metodología comprende la etapas de *Análisis* y *Diseño de la Arquitectura* y *Diseño Detallado*. La relación entre los modelos que se producen en cada etapa de la metodología GAIA pueden verse en la figura 2.2.

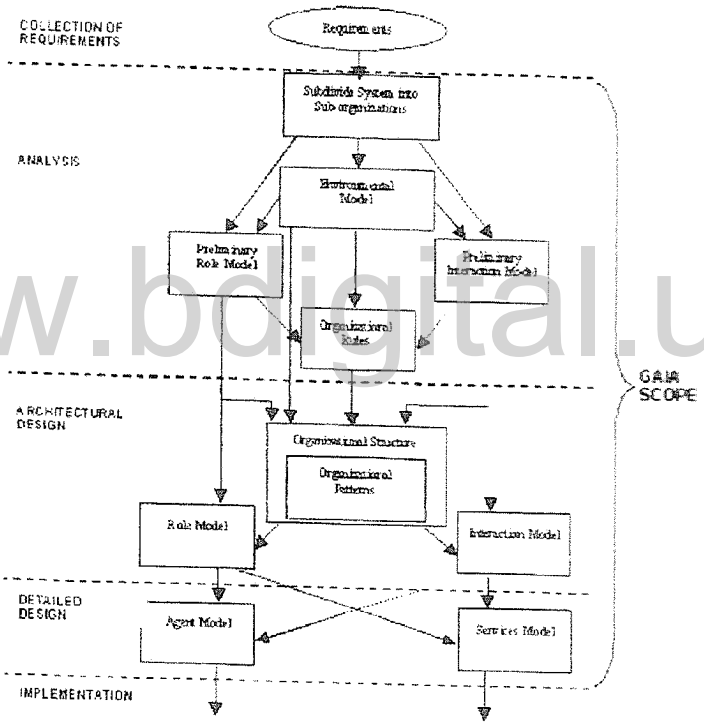


Figura 2.2: Modelos de la metodología GAIA y su relación en el proceso GAIA. (tomado de Zambonelli et al. [171])

Los modelos por etapa son los siguientes:

- Análisis: en esta etapa se construyen las especificaciones de los modelos que conforman el sistema MultiAgente:
  - Modelo Organizacional describe la organización (u organizaciones) que co-existen en el sistema

- Modelo Ambiental modela el ambiente en el cual esta inmerso el sistema multiagente
- Modelo Preliminar de Roles describe los roles a través de sus atributos descriptores
- Modelo Preliminar de Interacción muestra las dependencias e interacciones entre los diferentes roles del sistema Multiagente
- Reglas Organizacionales establecen una relación más general entre roles, protocolos y entre ambos; se clasifican como reglas organizacionales de supervivencia y seguridad.

- Diseño de la Arquitectura: incluye la definición de la estructura organizacional del sistema; y la culminación del Modelo Preliminar de Roles y del Modelo Preliminar de Interacción .

- Diseño Detallado: en esta etapa se diseña el Modelo de Agentes y el Modelo de Servicio.
  - Modelo de Agentes se identifican las *clases* de Agentes y las *instancias de agentes* que serán instanciadas a partir de esa clase.
  - Modelo de Servicio que són las servicios requeridos, entendidos como un conjunto de actividades, para que un agente cumpla con el rol asignado.

La metodología no provee un lenguaje gráfico propio como apoyo al modelaje del sistema multiAgentes, pero sí una serie de formalismos, además del lenguaje natural, para describir los modelos del sistema. Estos formalismos provienen de FUSION de Coleman et al. [28] y de la lógica temporal. GAIA sugiere usar esquemas gráficos diseñados por el usuario para documentar el sistema y la notación Agent Unified Modeling Language (AUML)[1] Odell et al. [123, 124] para definiciones más detalladas del Modelo de Interacción.

Conclusión.

GAIA modela un agente por medio de dos modelos con un alto nivel de abstracción: el Modelo de Agentes y el Modelo de Servicios. La metodología se enfoca en definir los agentes que integran el sistema y como estos interactan. No detalla la arquitectura interna del agente y la interfaz del Agente. GAIA contempla el concepto de roles al cual se le asocian atributos descriptivos como:

1 está en: http://www.auml.org/auml/main.shtml.
Los diagramas de AUML ya están contenidos en *UML 2.1*.

responsabilidad, *permisos*, *actividades* y *protocolos*. El MRA descrito en 2.1 preveé que el concepto de rol está descrito a través de metas del agente, que a su vez se describen por un conjunto de planes que tienen actividades y por reglas de condición-acción; por lo que los *permisos* y las *actividades del rol* de GAIA están incluidas en el modelo. Los *protocolos* se describen en el MRA como casos de usos en el estado externo. Las *responsabilidades del rol* de GAIA estan definidad en MRA como parte del estado interno del agente.

2.2.3 *MaSE.*

La metodología MaSE de Mark [113], inicia a partir de los requerimientos iniciales del sistema y produce una serie de documentos de diseño formales en un estilo gráfico. La figura 2.3 se refleja una visión breve de la metodología y sus modelos.

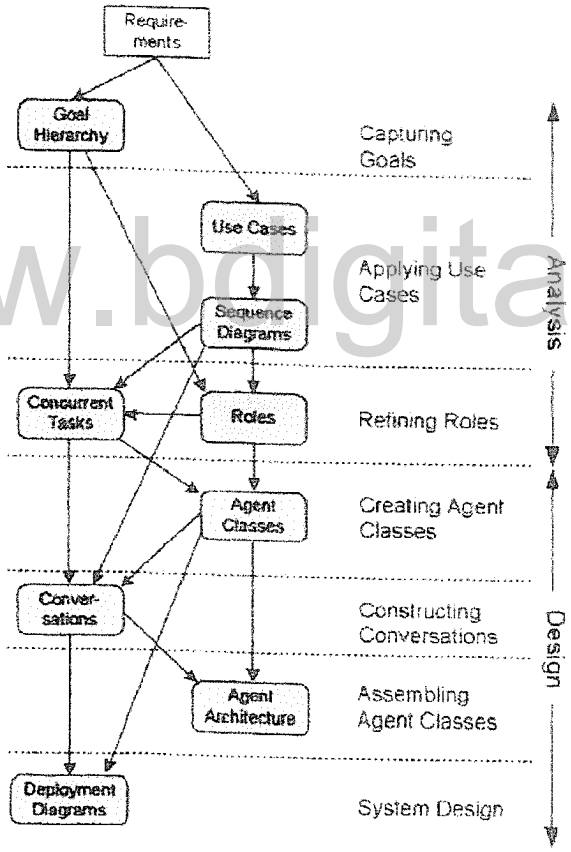


Figura 2.3: La metodología MaSE.
(tomado de Wood and DeLoach [166])

La metodología tiene dos etapas: *Análisis* y *Diseño*.

- Análisis: consta de tres fases:
  - Capturando Metas que transforma las especificaciones iniciales del sistema, producto del Análisis de Requerimientos, en un conjunto de *metas estructuradas* del sistema.
  - Casos de Usos, en esta fase se extraen el conjunto de *casos de usos* desde el contexto inicial y se crean un conjunto de *diagramas de sequencia de Rumbaugh et al. [146]* para ayudar al analista de sistema en la identificación del conjunto inicial de *roles* y formas de comunicación dentro del sistema.
  - Refinando Roles, su objetivo es la transformación de las *metas estructuradas* y *diagramas de sequencia* en *roles* y en *tareas* asociadas a los roles.
- Diseño: comprende cuatro fases:
  - Creando Clases de Agentes donde se identifican las clases de agentes a partir de los componentes de un rol, y las conversaciones entre los agentes.
  - Construyendo Conversaciones, se define el protocolo de coordinación entre dos agentes.
  - Ensamblando Clases de Agentes. El diseñador define la arquitectura del agente y sus componentes internos. Se tiene la opción de crear su propia arquitectura o usar plantillas pre-definidas de estilo de arquitecturas proporcionadas por Robinson [142]:
    - *Creencias-Deseo-Intenciones*(BDI)
    - *Reactivos*
    - *Planificadores*
    - *Basado en Conocimiento*
  - *Diseño del Sistema* las clases de agentes son *instanciada* como agentes.

Los diagramas utilizados por MaSE provienen de Kendall en Kendall and Zhao [94], Kendall et al. [93], y de la Técnicas de Modelado de Objetos, (Object Modelling Technique, OMT) de Rumbaugh et al. [146]. La metodología MaSE proporciona una herramienta para desarrollo del software, AgentTool desarrollados por DeLoach [45], que da soporte a los pasos de la metodología. Cada paso de la metodología genera los siguientes productos:

- Capturando Metas, las metas del sistema se representan en el *Diagrama Jerárquico de Metas* de Kendall and Zhao [94], creandose una jerarquía de metas donde cada nivel de la jerarquía contiene una meta y las submetas relacionadas.

- APLICANDO CASOS DE USOS, se dibujan los *Casos de Usos* que se usan para construir los *Diagramas de Secuencia* Rumbaugh et al. [146], en el cual se representa la secuencia de mensajes entre los diferentes roles de los agentes.

- REFINANDO ROLES, se construye un *Modelo de Rol Tradicional* de Kendall and Zhao [94] que proporciona una visión de alto nivel del sistema donde cada rol se diagrama con sus metas asociadas. Luego, se agrega las tareas de los roles. Para culminar esta fase, se construye el *Diagrama de Tareas* que es un Diagrama de Estado de Rumbaugh et al. [146] donde se esquematiza las comunicaciones de las tareas asociadas a los roles.

- CREANDO CLASE DE AGENTES, se presenta por medio de un *Diagrama de Clase de Agentes* que representa las clases de agentes y las conversaciones entre ellos.

- CONSTRUYENDO CONVERSACIONES, se construyen dos *Diagramas de Clases de Comunicación*: uno para el que inicia la conversación, y otro para el que responde. Un *Diagrama de Clases de Comunicación* es un par de *Diagrama de Máquina de Estado Finito* que define el estado de la conversación de las dos clases de agentes participantes.

- ENSAMBLANDO CLASES DE AGENTES, el componente interno del agente puede representarse como diagramas de estado que representen los eventos que se pasan entre componentes.

- DISEÑO DEL SISTEMA, las clases instanciadas se diagraman en un *Diagrama de Desarrollo* que muestra números, tipos y ubicación de los agentes dentro del sistema.

Conclusión.

La metodología MaSE modela distintos tipos de agente proporcionando *plantillas* de estilos de arquitectura. El MRA descrito en 2.1, en su estado interno considera una serie de conceptos por medio del cual pueden definirse diferentes tipos de agente. El estado externo del agente definido en roles y casos de uso, de igual manera, MaSE lo modela por medio de *roles y casos de uso*. El agente que se modela con la metodología MaSE está definido dentro del modelo de agente propuesto en MRA.

2.2.4 *Prometheus.*

La metodología Prometheus propuesta por Padgham and Winikoff [127], consta de tres fases: ESPECIFICACIÓN DEL SISTEMA, DISEÑO DE LA ARQUITECTURA y DISEÑO DETALLADO. Una relación entre los pasos

de la metodología y sus productos se muestran en la figura 2.4. Cada fase se describe como sigue a continuación.

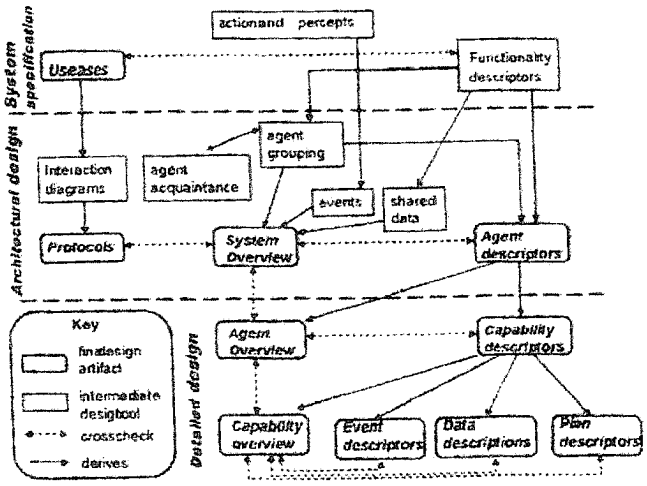


Figura 2.4: Metodología Prometheus
(tomado de Padgham and Winikoff [128])

- ESPECIFICACIÓN DEL SISTEMA: esta fase inicia con la identificación de los *perceptos* y las *acciones* de los agentes. El autor define, de acuerdo con Russell and Norvig [147], *perceptos* como la información que proviene del ambiente, y *acciones* como los mecanismos para afectar al ambiente. Establece, a partir de Winikoff et al. [165], la distinción entre *perceptos* y *eventos*: un *evento* es una ocurrencia significante para el sistema de agente mientras que los *perceptos* es información en bruto disponible para el sistema de agentes. Luego se identifica las *funcionalidades* del agente, en términos de *nombre, descripción, acciones, perceptos, datos usados* y *producidos*, e *interacciones*, con otras funcionalidades del sistema.

- DISEÑO DE LA ARQUITECTURA: usa la salida de la fase anterior para definir los tipos de agentes, diseñar la estructura del sistema y definir las interacciones entre los agentes. En esta etapa del diseño también se identifican que *eventos* serán generados como resultado de la información proveniente del medio ambiente, la información que será enviada entre los mensajes que intercambia los agentes y los *objetos de datos compartidos*. Estos últimos son estructuras de datos.

- DISEÑO DETALLADO: se desarrolla la estructura interna de cada agente y como éste cumple sus tareas dentro del sistema. Esto

se logra mediante la definición de las *capacidades*, definidas como módulos dentro del agente, *eventos internos*, *planes* y las *estructuras de datos detalladas* para cada tipo de agente identificado en la etapa previa.

La metodología Prometheus puede ser soportada por dos herramientas: The JACK Development Enviroment (JDE)[2] y The Prometheus Design Tool (PDT)[3]. PDT, desarrollado por los autores de la metodología, permite ingresar y editar un diseño en términos de los conceptos que maneja Prometheus, y genera, además de los diagramas de acuerdo a la metodología, una descripción del diseño LaTeX con sus descriptores, y un diccionario de datos.

La metodología utiliza varias notaciones como lenguaje gráfico:

- Especificación del Sistema: las funcionalidades del sistema se diagraman mediante *casos de uso* y *escenarios de casos de uso* de Unified Modeling Language (UML).
- Diseño de la Arquitectura: construye cuatro diagramas:
  - *Diagrama de los Agentes Conocidos*: se dibujan los agentes y los enlaces entre los agentes que interactuan.
  - *Diagrama General del Sistema*: une a los *agentes*, *eventos* y *objetos de datos compartidos* en un solo esquema.
  - *Diagramas de Interacción:* se diagrama la interacción entre agentes.
  - *Protocolos de Interacción.*:se construyen los protocolos de interacción usando AUML.
- Diseño Detallado: en esta etapa se construyen dos diagramas:
  - *Diagrama General de Agentes* , que suministra una visión de alto nivel de las internalidades de los agentes.
  - *Diagrama de Capacidad* donde se describen el detalle de cada capacidad que poseen los agentes.

Conclusión.

Prometheus describe el sistema desde dos niveles: Interno y Externo. El nivel Interno especifica las *capacidades* del agente, que son *eventos internos*, *planes* y las *estructuras de datos detalladas*, y puede incluir *preceptos* y *acciones*. En MRA descrito en 2.1 los *perceptos* son las creencias del agente, que es lo que el agente conoce acerca de su ambiente y de otros agentes. Las *acciones y eventos internos* están

2 JDE está en www.agent-software.com
3 PDT está en http://macr.cis.ksu.edu/index.php

incluidas dentro de los planes del agente definidas en la estructura interna del MRA. En el nivel Externo, Prometheus, usa casos de uso para definir las conversaciones del agente; de igual manera, en el MRA este concepto esta definido en el nivel externo del agente.

2.2.5 *MESSAGE/UML.*

MESSAGE/UML aparecida en Caire et al. [21], conceptualiza un SMA categorizandolo en dos niveles: *nivel de datos* y *nivel de conocimiento.* El *nivel de conocimiento* esta compuesto de *entidades* a las cuales se le asocian conceptos.

Las entidades a *nivel de conocimiento* se clasifican en *Entidades Concretas, Actividades* y *Entidades de Estado Mental.* A a continuación se detallan los conceptos asociados a cada entidad:

- *Entidades Concretas*:
  - *Agentes:* es una entidad autónoma y atómica que es capaz de realizar alguna función. La capacidad funcional está descrita en el *servicio* y la motivación del agente está representada en el atributo *propósito.*
  - *Organización*: es un grupo de agentes que trabajan juntos para un propósito común. Está representado a través de las relaciones de subordinación o jerarquía y los mecanismos de comportamiento-coordinación establecidos por medio de las interacciones entre los agentes.
  - *Rol:* describe las características externas de un agente en un contexto en particular.
  - *Recursos:* representa entidades no-autónomas como bases de datos o programas externos usados por agentes.
- Actividades:
  - *Tareas*: es una unidad de actividades a nivel de conocimiento, con ejecuciones sencillas.
  - *Interacción y Protocolos de Interacción*: el concepto de interacción, MESSAGE/UML lo toma de la metodología GAIA de Wooldridge et al. [167]. El Protocolo de Interacción define un patrón de intercambio de mensaje asociado con la interacción.
- Estado Mental:
  - *Metas*: una meta asocia un agente con una situación. Las metas pueden ser intrínsicas, que persisten en la vida del agente, ó tácticas, que no persisten durante la vida del agente. Las metas son expresada en términos de una función de utilidad y su función objetivo se estima de manera de que se maximice la utilidad.

La figura 2.5 proporciona una visión informal centrada en el agente, de como se interrelacionan los conceptos de MESSAGE/UML , y su relación con el concepto de agente.

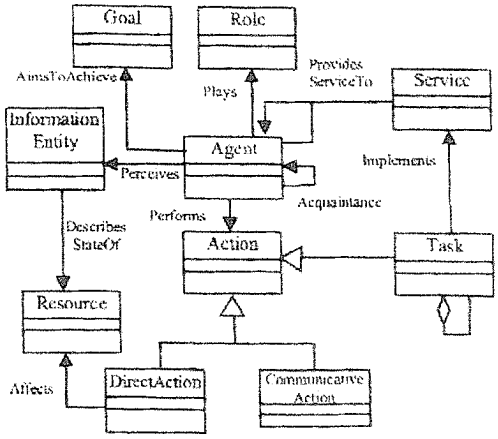


Figura 2.5: Conceptos de MESSAGE/UML
(tomado de Caire et al. [22])

MESSAGE/UML define un conjunto de vistas o perpectivas, que proporciona un mayor nivel de entendimiento del sistema. Este conjunto es similar al usado en la metodología MAS-CommonKADS de Iglesias et al. [83]:

- VISTA DE LA ORGANIZACIÓN: muestra entidades concretas en el sistema y su ambiente y las relaciones entre ellos (agregación, poder, y relaciones conocidas).
- VISTA META/TAREA: muestra *metas*, *tareas*, *situaciones* y las dependencias entre ellos.
- VISTA AGENTE/ROL: se enfoca en especificar a los agentes y sus roles asociados.
- VISTA INTERACCIÓN: por cada interacción entre agentes-roles, se muestra el *iniciador*, el *colaborador*, el *motivador*, la información relevante suministrada/lograda por cada participante, eventos que activó la interacción, entre otros aspectos.
- VISTA DEL DOMINIO: muestra dominio y relaciones que son relevantes para el sistema en desarrollo.

Durante el *Proceso de Análisis* se genera un modelo del sistema y su ambiente, utilizando refinamiento de manera gradual. El nivel más alto de descomposición esta referido como un nivel 0. En este nivel el proceso de modelaje comienza con la construcción de las VISTAS DE ORGANIZACIÓN, VISTA METAS/TAREAS, VISTA AGENTE/ROL y VISTA

DEL DOMINIO. Finalmente, la VISTA DE INTERACCIÓN se construye usando los otros modelos. A partir de los modelos de nivel 0 se crean los modelos de nivel 1, donde se definen las estructuras y el comportamiento de entidades como *Organización, Agentes, Tareas, Metas, Dominios*. No se sugiere alguna estrategia en particular para refinar los modelos de nivel 0, va a depender de diferentes enfoques para analizar las propiedades del sistema.

MESSAGE/UML usa el diagrama de clases de UML y lo extiende con los conceptos de orientación a Agentes para construir todas las vista del sistema. El diagrama de Clases se le agregan tres conceptos: Recursos, Organización y Rol; cada uno representado con un ícono en particular.

Conclusión.

MESSAGE/UML construye un agente con estado interno, cuyos atributos internos son *propósitos y metas*, y considera que el estado externo del agente esta descrito a través de *roles*. Ambos estado internos y externos, descritos en MESSAGE/UML estan previstos en el MRA descrito en 2.1.

2.2.6 *INGENIAS.*

La metodología INGENIAS propuesta por Gómez-Sanz and Pavón [74], re-usa el trabajo de la metodología MESSAGE/UML para establecer su propuesta. INGENIAS ofrece tres elementos de ayuda para el desarrollo de un sistema multiAgentes:

- Un lenguaje visual: utiliza GOPRR (Graph, Object, Property, Relationship, and Role (GOPRR)) de Lyytinen and Rossi [109] como lenguaje de meta-modelos.
- Integración con el ciclo de vida de desarrollo de software: la metodología está embebida en un proceso de desarrollo industrial, Unified Software Development Process (USDP) de Jacobson et al. [88]
- Herramienta de desarrollo: usa una herramienta, METAEDIT+ de Lyytinen and Rossi [109], como ambiente de análisis/diseño.

Para describir un sistema multiAgentes, INGENIAS proporciona los siguientes modelos:

- MODELO DE AGENTES: describe al agente con sus *metas, tareas, estado mental inicial,* (Shoham [152]) , y *roles jugados*.
- MODELO DE INTERACCIÓN: describe la interacción entre los Agentes.

- MODELO DE METAS Y TAREAS: describe las relaciones entre *metas* y *tareas, estructuras de las metas* y *estructuras de las tareas.*
- MODELO ORGANIZACIONAL: describe como se agrupan los componentes del sistema, las metas que comparten y las restricciones existentes en la interacción entre agentes.
- MODELO DE AMBIENTE: define la percepción del agente en términos de los elementos existentes en el sistema.

La metodología contempla dos etapas: *Análisis* y *Diseño.*
En la etapa de *Análisis:*

- genera los casos de uso y se identifican las acciones de esos casos de uso con el modelo de interacción;
- se esquematiza la arquitectura del sistema con el MODELO ORGANIZACIONAL
- se genera el MODELO AMBIENTAL.

Durante la etapa de *Diseño,* se genera un prototipo usando una herramienta de desarrollo de aplicaciones tales como agentTool de DeLoach [45] o ZEUS de Nwana et al. [122].

INGENIAS proporciona una herramienta para el modelado visual: INGENIAS IDE[4] que generar los modelos que conforman la metodología y UML para los diagramas de interacción que describen el MODELO DE INTERACCIÓN.

Conclusión.

En el estado interno, INGENIAS modela un agente como una entidad con un estado mental, con *creencias, compromisos, metas y planes.* El agente tiene un estado mental inicial y éste estado mental puede variar dependiendo de las creencias del agente y de las percepciones que el agente tenga de su ambiente. El estado externo del agente, INGENIAS lo describe a través de *roles* y *caso de uso.* Ambas descripciones, a saber estado interno y estado externo del agente, están descritos de igual manera en el MRA descrito en 2.1.

2.2.7 *Tropos.*

La metodología Tropos presentada por Bresciani et al. [18], toma como base el trabajo de Yu [168] acerca del modelo *i**, que ofrece modelar actores, metas y dependencia entre actores como conceptos primitivos. La metodología Tropos usa los siguientes conceptos:

4 está en http://ingenias.sourceforge.net

- *Actor:* modela una entidad que tiene metas estratégicas e intencionalidad dentro de un sistema o valores organizacionales. Un *actor* representa un agente de software ó un agente físico ó un agente social, así como también, un *rol* ó una *posición*.
  - Un rol es una caracterización abstracta del comportamiento de un actor social dentro de algún contexto especializado o dominio.
  - Una posición representa un conjunto de roles jugado por un agente.
- *Metas*: representa intereses estratégicos del *actor*. Se clasifican según su relación o nó con los requerimientos funcionales como *metas duras* y *metas blandas*.
- *Plan:* es una manera de hacer algo. La ejecución del plan puede ser un medio para satisfacer una *meta*.
- *Recursos*: representa una entidad física o una entidad informacional.
- *Dependencia*: entre dos actores, indica que un actor depende, por alguna razón, de otro para lograr una meta, ejecutar algún plan, o distribuir algún recurso.
- *Capacidad*: representa la habilidad de un actor de definir, escoger y ejecutar un plan para cumplir con una meta.
- *Creencia*: representa el conocimiento del actor acerca del mundo.

La metodologia Tropos consiste de cinco fases:

- Requerimientos Tempranos: se construye el *Modelo de Actor*, el *Modelo de Dependencia de Metas*, el *Modelo de Metas* y el *Modelo de Plan*.
- Requerimientos Finales: extiende los modelos creado en la etapa previa e incluye un nuevo actor, el sistema, y las dependencias con otros actores del ambiente.
- Diseño de la Arquitectura, extiende el Modelo de Actores, identificando las capacidades y agrupandolas para forMRA los Agentes.
- Diseño detallado: se especifican los agentes y se construye un *Modelo de Capacidad*.
- Implementación: utiliza una plataforma BDI, como JACK Intelligent Agents de Buseta et al. [20], para la implementación del sistema multiAgentes.

Para las fases de REQUERIMIENTOS TEMPRANOS, REQUERIMIENTOS FINALES y DISEÑO DE LA ARQUITECTURA, la metodología Tropos proporciona una notación gráfica propia. En la fase de DISEÑO DETALLADO se usan los los diagramas de actividad de UML y los diagramas de interacción de AUML para representar las capacidades y los planes.

Conclusión.

Tropos propone un agente con un estado interno descrito por *metas, intenciones, capacidades y creencias*. Los atributos *metas, intenciones y creencias* del estado interno del agente Tropos, de igual manera están incluidos en el MRA descrito en 2.1. El atributo *capacidad*, Tropos lo define como habilidad de un actor de definir, escoger y ejecutar un plan. El atributo *capacidad*, el MRA lo define como el atributo preferencias del agente, que son un conjunto de metas o de reglas que pueden seleccionarse. El estado externo del agente de Tropos esta descrito por medio de *roles* al igual que el estado externo del MRA.

2.2.8 *MAS-CommonKADS.*

La metodología MAS-CommonKADS propuesto por Iglesias et al. [82] extiende la metodología Common Knowledge Acquisition Design System (CommonKADS)[5] de Schreiber et al. [148], añadiendo aspectos relacionados con el modelado de SMA. Se definen los SMA mediante los siguientes modelos:

- MODELO DE AGENTES: Especifica las carácteristicas de los agentes: capacidad de razonamiento, habilidades, servicios, grupos y jerarquía de agentes.
- MODELO DE TAREAS: describe las tareas que un agente puede realizar
- MODELO EXPERTO: describe el conocimiento necesario del agente para lograr su tarea.
- MODELO DE ORGANIZACIÓN: describe la organización en la cual el SMA se va a incluir, y la organización social de la sociedad de agente
- MODELO DE COORDINACIÓN: describe la conversación entre agentes: sus interacciones, protocolos y capacidades requeridas.
- MODELO DE COMUNICACIÓN: detalla la interacción entre el agente software-humano, y el factor humano para desarrollar estas interfases de usuario

5 está en http://www.commonkads.uva.nl

- MODELO DE DISEÑO: consiste de tres submodelos: *modelo de red*, el *diseño del agente* y la *plataforma de diseño*.

La metodología sigue las siguientes fases para desarrollar los modelos descritos: *Conceptualización*, *Análisis* y *Diseño*.

- *Conceptualización*. Por medio de *análisis de los requerimientos* y *casos de usos* se obtiene una primera descripción del problema.
- *Análisis*. Se especifican los requerimientos, mediante el desarrollo de los siguientes pasos:
  - *Modelado de Agentes:* se identifican y describen los agentes mediante el desarrollo de una instancia inicial del modelo de agentes.
  - *Modelado de Tareas*: descomposición de las tareas y determinación de metas y tareas
  - *Modelado de Coordinación*: describe la interacción y el protocolo de coordinación entre agentes.
  - *Modelado del Conocimiento*: modela el conocimiento de agente (conocimiento para cumplir la tareas y su comportamiento proactivo) y su ambiente (creencias e inferencias del mundo).
  - *Modelado de la Organización*: se desarrolla el modelo de la organización de los agentes que muestra la relación estructural o estática entre los agentes.
- Diseño. Durante esta fase se desarrolla el modelo especificado en las etapas anteriores. Esta fase consiste en
  - *Diseño de la red de Agentes*: se determinan las facilidades de redes, conocimiento y coordinación de la infraestructura del SMA
  - *Diseño del Agente*: se determina la arquitectura más adecuada para cada agente
  - *Diseño de la plataforma*: selección del software y hardware necesario para el sistema.

El proceso de desarrollo de la metodología combina la visión de manejo de riesgos con la visión basada en componentes, de manera que los componentes definidos sean candidatos para el reuso.

La metodología MAS-CommonKADS inicia la etapa de Conceptualización del sistema con diagramas UML de *Casos de Uso* para recolectar los requerimientos del sistema y luego, se construye Message Sequence Charts (MSC) - Rudolph et al. [145] para formalizar las interacciones.

En la etapa de Análisis, no hay herramientas gráficas para el modelo de agentes, sino más bien un conjunto de estrategias para

analizar los requerimientos y determinar los agentes. La metodología no presenta una estructura gráfica para el MODELO DE TAREAS. El MODELO DE COORDINACIÓN usa MSC para describir los escenarios entre agentes, y Specification and Description Language (SDL) que aparece en Z100 [169], para modelar cada interacción.

El MODELO DE CONOCIMIENTO, la metodología extiende las Técnicas de Modelado de Objetos, (Object Modelling Technique, OMT) para representar el Dominio del Conocimiento. Y finalmente, para el MODELO DE LA ORGANIZACIÓN se usa el modelo de Objeto de la Técnicas de Modelado de Objetos, (Object Modelling Technique, OMT).

Conclusión.

MAS-CommonKADS modela el estado interno del agente con los atributos de *capacidades de razonamiento, metas e intenciones.* La capacidad de razonamiento se modela durante la fase de MODELAJE DEL CONOCIMIENTO, donde se define también el mecanismo de inferencia del dominio y el mecanismo de inferencia del ambiente. Como atributos de la capacidad de razonamiento del agente están los métodos de resolución de problemas, carácter del agente, creencias, conocimiento del mundo y de otros agentes.

El estado externo del agente modelado en MRA descrito en 2.1, establece un estado interno que contempla los atributos de creencias, metas e intenciones. Así como un conjunto de mecanismo de inferencia que son parte de la dinámica interna del agente en MRA.

MAS-CommonKADS modela el estado externo del agente por medio de las *conversaciones* entre los agentes y los *protocolos entre agente software-humano, y el factor humano.* Además considera el modelado de creencias e inferencias sobre el mundo. El agente de MRA describe en su estado externo las *convesaciones entre agentes y* la interrelación *agentes software-humano y el factor humano.* Y en la interfaz del agente del agente MRA se considera el conocimiento del agente acerca de su medio ambiente.

En consecuencia, el MRA y MAS-CommonKADS coinciden, salvo porque en el MRA no hay previsión explícita del carácter del agente y de su conocimiento sobre otros agentes.

2.2.9 *O-MaSE.*

La metodología O-MaSE de Garcia-Ojeda et al. [68], extiende MaSE con la visión de la Ingeniería de Métodos presentado en Brinkkemper [19] y visualizan la construcción del SMA a partir de fragmentos de métodos, los cuales están basados en un metamodelo común. Para definir el *armazón de procesos* de O-MaSE, toman como referencia al

OPEN Process Framework (OPF)[6] de Firesmith and Henderson-Sellers [64] . Definen, de manera similar los niveles del sistema: En el nivel M2 está el metamodelo OPF; y el nivel M1 contiene la definición de O-MaSE en la forma del *Metamodelo O-MaSE*, *Fragmentos de Métodos* y las *Guías*.

- *Metamodelo O-MaSE*: presenta la *Organización* a través de las entidades: *Metas, Roles, Agentes, Dominio, Modelo y Políticas.*
- *Fragmentos de Métodos*: O-MaSE define las siguientes *unidades de trabajo*: Actividad, Tareas, Técnicas, Productos de Trabajo, Producción y Lenguaje. Se definen tres tipos de *actividades: Ingeniería de Requerimientos*, *Análisis* y *Diseño*. Cada *actividad* tiene asociada *tareas, técnicas* para realizar la actividad, *productos de trabajo* como resultado,método de *producción* y *lenguaje* para describir la actividad.

Las actividades constan de:

- Ingeniería de Requerimientos:
  - las tareas asociadas son: *(i)*la realización del *modelo de metas* y *(ii)* el *Refinamiento de las metas.*
    - el producto de trabajo es un *Modelo de Metas para Sistemas dinámicos.*
    - la producción de esta actividad es el *Modelador de Metas.*
  - Análisis:
    - las tareas *son: (i)* Modelado de la Interface Organizacional,*(ii)* Modelado de Roles,*(iii)* Definición del Roles y *(iv)* el Modelado de Dominio.
    - la producción de esta actividad son: *Modelador Organizacional*, el *Modelador de Roles*, y el *Dominio Experto.*
    - el producto de trabajo son: *Modelo Organizacional, Modelo de Roles y el Modelo de Dominio*
  - Diseño:

6 OPF está en http://www.opfro.org/
En Garcia-Ojeda et al. [68], lo definen así: "es una visión de un estandar industrial para Ingeniería de Métodos aplicados a la producción de procesos. **OPF !** (OPF !)usa un armazón integrado basado en metamodelos que permite a los diseñadores seleccionar fragmentos de métodos de un repositorio y construir el proceso usando construcciones identificadas y guias adaptadas. El armazón basado en metamodelos está soportado por un esquema de tres capas. La capa M2 incluye el metamodelo OPF, el cual es un metamodelo de procesos genéricos que define los tipos de fragmentos de métodos que pueden usarse en M1". En el nivel Mo están las instancias de los procesos.

- Las tareas asociadas son: *(i)* Modelado de Clases de Agentes,*(ii)* Modelado de Protocolos, *(iii)* Modelado de Planes, *(iv)* Modelado de Políticas, *(v)* Modelado de Capacidades, *(vi)* Modelado de Acciones y *(vii)* Modelado de Servicios.
- Los productores son: *Modelador de Clases de Agentes, Modelador de Protocolos, Modelador de Planes, Modelador de Políticas, Modelador de Capacidades, Modelador de Acciones* y *Modelador de Servicios.*
- El producto de trabajo son: *Modelo de Clases de Agentes, Modelo de Protocolos, Modelo de Planes, Modelo de Políticas, Modelo de Capacidades, Modelo de Acciones* y *Modelo de Servicios*

- Guías: describe como se pueden combinar los *fragmentos de métodos* para obtener los *procesos*.

La metodología O-MaSE proporciona una herramienta para desarrollo del software, AgentToolIII de Garcia-Ojeda et al. [68], que da soporte a los pasos de la metodología y proporciona los diagramas que apoyan cada etapa. Los lenguajes utilizado son:

- *lenguaje natural*
- UML y AUML para los diagramas de interacción, protocolos de interacción y modelos específicos
- notación O-MaSE desarrollada en la herramienta AgentToolIII
- lenguaje formal, para las especificaciones formales de las propiedades del sistema.

Conclusión.

La metodología O-MaSE proporciona un marco de procesos para construir SMA. O-MaSE plantea el estado interno del agente de manera similar que el agente de MaSE. Pero en O-MaSE toma en consideración el ambiente en el cual está inmerso el agente definiendo los objetos que lo conforman y las interacciones entre ellos y le asigna políticas o reglas organizacionales a las organizaciones para modelarlas. Como se explicó en la parte 2.2.3, el agente de MRA descrito en 2.1 contempla los aspectos relacionados al estado interno del agente y su estado externo. Sin embargo, los aspectos relacionados con las políticas y reglas organizacionales, no están consideradas en el MRA.

Hemos revisado un conjunto de metodologías conocidas y se han comparado con el MRA propuesto en 2.1. Esta comparación

ha permitido establecer que el MRA es un modelo general de un agente que abarca las distintas visiones de los SMA que ofrecen las metodologías revisadas.

En la siguiente sección exploraremos como el MRA esta inscrito dentro de la teoría de simulación multiAgentes propuesta por Dávila et al. [44].

2.3 TEORÍA DE SIMULACIÓN MULTIAGENTES.

Una teoría de simulación, Dávila et al. [43] la conceptualizan como "...una explicación general de lo que es un sistema, sus componentes y sus reglas de transición, establecido todo como un conjunto de proposiciones matemáticas formalizadas. El objetivo es el de proporcionar a los desarrolladores de sistemas de simulación con una especificación que dice lo que un simulador debe hacer y cómo debe comportarse para simular un sistema."

La teoría de simulación multiAgentes propuesta por Dávila et al. [44] es un modelo formal construidos sobre trabajos previos en IA de Ferber and Muller [61] y Simulación de Dávila and Uzcátegui [39], Dávila et al. [42, 43], Dávila and Uzcátegui [40]. Esta teoría ha sido plasmada en el desarrollo del sistema simulador multiAgentes, GALATEA propuesta por Dávila and Uzcátegui [39], Dávila et al. [44].

En la figura 2.6 se esquematiza de manera general al simulador multiAgentes GALATEA. Los agentes de GALATEA durante la simulación, observan su medio ambiente o área de influencia, toman decisiones y ejecutan acciones que pueden afectar su área de influencia. El conjunto de operadores *observa-piensa-actua* constituye el comportamiento del agente. El simulador registra las acciones de los agentes y lleva el sistema de un estado al próximo cronológicamente.

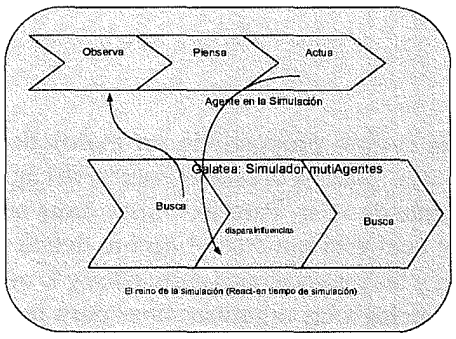


Figura 2.6: Esquema del Simulador GALATEA

En esta sección se presenta el modelo formal del sistema multi-Agentes GALATEA que fueron descritos en el MRA de la sección 2.1.

2.3.1 *Una primera aproximación a un modelo formal de un sistema multiagentes.*

A continuación se introducirá la descripción de la dinámica completa de un sistema multiagentes propuesto por Dávila et al. [44], Dávila and Tucci [37], Dávila and Uzcátegui [38] .

Las formulas 2.1 a 2.9 constituyen la descripción matemática completa de un sistema poblado por muchos agentes, descrito a partir de la función *Cycle* la cual *transporta* el sistema desde un estado global al próximo cronológicamente. El estado global se caracteriza, como es usual en simulación, como un conjunto de variables de estados y sus valores, representado en esta formalización por σ (y similares).

Sin embargo, tal y como fue propuesto en Ferber and Muller [61], el estado global también incluye un conjunto conocido como las influencias, γ (y similares), que representan todas las acciones que el agente ha ejecutado y que están actualmente tratando de ejecutar.

Se usará γ_i para representar la influencia producida por el agente$_i$. El único mecanismo adicional que se necesita es un vector de s_i valores, cada uno de los cuales se descompone en metas y creencias, provenientes del conjunto de todos los posibles estados mentales de todos los agentes (S), para producir la definición formal de la función evolución de un sistema:

$$Evolution : S \otimes \Im \otimes \Sigma \otimes \Gamma \rightarrow \epsilon \tag{2.1}$$

donde $s_i \in S$, $\sigma \in \sum$, $\gamma \in \Gamma$ y $t \in \Im$

La función *Evolution* de un sistema se define como una función recursiva, tal que:

$$Evolution(< s_1, s_2, .., s_n >, t, \sigma, \gamma) =$$

$$Evolution(Cycle(< s_1, s_2, .., s_n >, t, \sigma, \gamma)) \tag{2.2}$$

y *Cycle* reune todas las influencias producidas por los agentes, la cual esta determinada por la conducta del agente mismo, y la dinámica del ambiente, y se define como:

$$Cycle : S \otimes \Im \otimes \Sigma \otimes \Gamma \rightarrow S \otimes \Im \otimes \Sigma \otimes \Gamma :$$

$$Cycle(< s_1, s_2, .., s_n >, t, \sigma, \gamma) =<< s'_1, s'_2, .., s'_n >, t', \sigma', \gamma' > \tag{2.3}$$

La reacción ambiental a las influencias de los agentes están definidas por:

$$React : \Lambda \otimes \beta \otimes \Im \otimes \Sigma \otimes \Gamma \tag{2.4}$$

donde Λ representa las leyes de cambio del sistema, y β es el conocimiento contextual sobre el sistema, entonces,

$$< \sigma', \gamma' >= React(\Lambda, \beta, t, \sigma, \gamma \cup_i \gamma_i) \tag{2.5}$$

y la conducta del agente se expresa como:

$$< s_i', \gamma_i >= Behaviour_a(t, r_a, k, g, \gamma) =< k, g', \gamma' > \quad (2.6)$$

donde, cada s' es una abreviación de $< k_i, g_i' >$, la base de conocimiento y metas del agente i, y,

$$\gamma = Scan^*(Network, \xi) \quad (2.7)$$

$$\xi = NextEvent(\gamma) \quad (2.8)$$

$$t' = TimeOf(\xi) \quad (2.9)$$

son los elementos usuales en un simulador DEVS Zeigler et al. [172], Wainer [162]. *Network* representa los componentes del sistema simulado, cuyos eventos asociados (uno de los cuales es ξ) impulsan la dinámica completa.

2.3.2 *El comportamiento de un agente como una función matemática.*

Para completar la descripción de la sección anterior caracterizamos un agente a como una función matemática:

$$Behaviour_a : \Im \otimes \Re \otimes Ka \otimes Ga \otimes \Gamma \rightarrow Ka \otimes Ga \otimes \Gamma \quad (2.10)$$

que relaciona los recursos de cómputo, el estado interno del agente y el conjunto de influencias con un nuevo estado interno y a un conjunto de influencias producidas por el agente.

La función $Behaviour_a$se define como sigue:

$$Behaviour_a(t, r_a, k, g, \gamma) =< k', g', \gamma' > \quad (2.11)$$

donde

$$k' = Update_a(t, Perception_a(\gamma), k) \quad (2.12)$$

$$< \gamma', g' >= Planning_a(t, r_a, k', g) \quad (2.13)$$

La función $Update_a$ añade el conjunto de perceptos observado por el agente a su base de conocimiento. En particular, $obs(P, t)$ representa el hecho de que un agente observa la propiedad P en el tiempo t.

La función $Planning_a$ especifica una máquina de inferencia que transforma las metas g en metas g' e influencias γ', usando las reglas e información en k', comenzando en el tiempo t y tomando no más que r_a unidades de tiempo para hacerlo.

2.3.3 *Agentes reactivos y racionales.*

Un agente es descrito por seis elementos:

$$< P_a, K_a, G_a, Perception_a, Update_a, Planning_a > \qquad (2.14)$$

donde $P_a$es el conjunto de posibles perceptos para un agente a y $Perception_a$ explica como actualmente el agente percibe su ambiente. Un agente racional tiene una base de conocimiento, K_a, y un conjunto de metas (o intenciones), G_a, que, conjuntamente, caracteriza su estado interno.

La función

$$Update_a : \Im \otimes P_a \otimes K_a \rightarrow K_a \qquad (2.15)$$

es el mecanismo de memorización y garantiza la adición de nueva información, y preserva la estructura interna de la base de conocimiento (y su consistencia) porque K_a es una colección de formulas lógicas con sintaxis y semántica bien definidas.

La función

$$Planning_a : \Im \otimes \Re \otimes Ka \otimes Ga \rightarrow Ga \otimes \Gamma \qquad (2.16)$$

representa el razonamiento que deriva nuevas metas e influencias, tomando en cuenta las metas previas y la base de conocimiento. Nótese que ambas $Update_a$ y $Planning_a$ introducen un argumento (con dominio $\Im$, el conjunto de todos los puntos de tiempo posibles) para indicar el tiempo de (re)inicio de cada proceso (actualización y planificación).

Con $Planning_a$, se modela el proceso por medio del cual un agente deriva, a partir de un conjunto de metas de alto nivel, un conjunto de metas de bajo nivel Kowalski [100], Kowalski and Sadri [101], Dávila [33], algunas de las cuales son acciones que pueden ser tratadas de ejecutar.

Las ecuaciones comprendidas entre 2.14 y 2.16 representan la descripción formal del MRA descrito en 2.1, donde las *estructuras de estado interna* son el conjunto K_a y G_a; la *dinámica interna* esta representada en $Update_a$ y $Planning_a$; y la *interfaz* es la ecuación 2.14 que describe al agente.

Se usará este modelo formal de un agente como la base de la propuesta presentada en este documento.

2.3.4 *Un agente con racionalidad acotada.*

Racionalidad acotada se refiere al hecho de que un agente tiene recursos limitados, tipicamente tiempo o espacio de memoria, para razonar. El concepto fue usado por Simon [154] como parte de un intento de modelar personas como agentes en una economía. El autor dice que el hombre racional perfecto, propuesto por modelos

económicos tradicionales, difícilmente representa el comportamiento preciso de los seres humanos reales porque estos sencillamente no razonan y actuan de esa manera. Los seres humanos están influenciados por un número de variables diversas, tales como la oportunidad, y normalmente no muestran el comportamiento matematicamente perfecto de esos modelos.

Lo que se propone en esta teoría de simulación es parte del modelo de un agente presentado en Dávila [33]. Éste básicamente dice que un agente debe intercambiar entre razonamiento y acción, así que debe haber un límite de tiempo (o espacio) para el razonamiento y, entonces, puede ser que el agente actúe antes de completar sus razonamiento. Teniendo más tiempo para razonar, el agente podría haber tomado otro curso de acción. La idea clave es que ese tipo de agente estará listo para *(re)accionar* más rápido que un agente que trata de completar su proceso de razonamiento. El precio que paga es que el agente *reactivo* podría no tomar el *mejor* curso de acción.

Formalmente, se ha transladado ese límite a una restricción en el tiempo utilizado por el agente para razonar. Una vez alcanzado ese límite, se debe cambiar al *lugar de control* y tratar cualquier otra acción que haya decidido (si la hubiere) despues del proceso de razonamiento *truncado*.

Nótese que el segundo argumento de la función de planificación:

$$Planning_a : \Im \otimes \Re \otimes Ka \otimes Ga \rightarrow Ga \otimes \Gamma$$

es un número real ($\Re$), que es el tiempo utilizado para el proceso de planificación. En una primera aproximación, sin embargo, ese número es un entero y contabiliza el número de *pasos de razonamiento* realizados durante el último espacio de tiempo concedido para razonar. Esta extensión, junto con la estructura de la función que describe el comportamiento del agente es una primera propuesta al modelo de agente con racionalidad acotada.

2.3.5 *El* MRA *y El Modelo Formal de Agentes.*

La caracterización que conforma el MRA está plenamente formalizada en el modelo del agente expresado en la teoría de simulación multiAgente para GALATEA. Los conceptos relacionados con la estructura de estado interno del agente en la sección 2.1 para el MRA están expresados por medio de las metas. En la teoría de simulación, en la ecuación 2.14, las metas forman parte de la formalización del agente. Los mecanismos que conforman la dinámica interna del agente en el MRA están plasmados en las funciones $Perception_a$, $Update_a$ y $Planning_a$ del modelo formal del agente. En el estado interno del agente del MRA, los roles se expresan de igual manera en metas. Y finalmente, la interfaz del agente en el MRA, las percepciones y creen-

cias son las P_a y las K_a , respectivamente; y las capacidades del agente están formalizadas en la función $Behaviour_a$.

Por consiguiente , el MRA establecido en la sección 2.1 esta formalizado en la teoría de simulación de GALATEA.

Conclusión.

En esta sección se ha expuesto la teoría de simulación mutiAgentes propuesta por Dávila and Uzcátegui [38] y Dávila and Tucci [37], que describe matemáticamente un sistema poblado por muchos agentes, sus componentes y sus reglas de transición; así como la caracterización interna de los agentes que conforman el sistema.

Se ha mostrado que el MRA establecido en la sección 2.1 está incluido en la formalización de la teoría de simulación de GALATEA.

La descripción del sistema de simulación expresada en la teoría de simulación de GALATEA describe la conformación de un sistema poblado de agentes, pero no contempla la ubicación de los agentes en un espacio geográfico.

2.4 GEOSIMULACIÓN.

Los SIG tiene varias definiciones según los diversos puntos de vistas de los autores de estos conceptos, Sendra [150]. Desde el punto de vista informático, Bracken and Webster [15] definen un SIG como "...un tipo especializado de BD, que se caracteriza por su capacidad de manejar datos geográficos, es decir, espacialmente referenciados, los cuales se pueden representar gráficamente como imágenes ".

Gimblett [72] establece que los SIG han usado exitosamente la tecnología de Autómata Celular (AC) para implementar modelo ecológicos espaciales dinámicos en espacios discretos y que la mayoría de los estudios se han enfocado en el modelado de interacciones biofísicas.

Por otra parte, [71] define a los AC como aquellos que " ... modelizan un mundo en el cual el espacio es representado como una cuadrícula uniforme, el tiempo avanza por periodos discretos, y las "leyes" del mundo son representadas por un conjunto unitario de reglas que permite calcular el estado de cada celda a partir de su propio estado previo y los estados de sus vecinas inmediatas".

La tecnología AC ha resultado ser muy útil para modelar sistemas dinámicos. Pero en la unión de las tecnologías de AC y SIG para el modelado de sistemas dinámicos, se ha tenido "... diversos grados de éxito debido a que los SIG son inherentemente procesos estáticos, ellos están limitados para el uso en modelado de sistemas dinámicos en la actualización asíncrona de datos celulares y por su naturaleza celular implícita", Gimblett [72].

Otra de las limitación de los SIG para ser usado en el modelado de sistema dinámicos, es el manejo del tiempo. Esta limitante ha sido resumida por Maidment [111] e Itami [86] en sendos artículos de la siguiente manera: " ... como la necesidad de incorporar alguna forma de estructuras de datos dependiente del tiempo si ésta es capaz de registrar la evolución de fenómenos espaciales".

Además, Gimblett [72] dice que "...muy poco se ha hecho en:

- incorporar datos de las ciencias sociales en modelos explicitamente espaciales de fenómenos del comportamiento humano; e
- incorporar investigaciones en sistemas inteligentes y técnicas de programación OO para simulación "

Investigadores, como Jiang and Gimblett [92], estiman que existen propiedades en los sistemas naturales, sociales, entre otros, que pueden ser modeladas como propiedades "que emergen" de interacciones individuales. Este último concepto es muy afín a las características de los agentes. Consideran que la incorporación de la tecnología de agentes en los sistemas de simulación con SIG, pueden resultar una buena alternativa para modelar la dinámica espacio-tiempo.

Es por ello que existe un gran interés por parte de la comunidad científica (Gimblett [73], Gilbert and Troitzsch [71], Batty et al. [7], Murgante et al. [121], Torrens [158]) en utilizar avanzadas tecnologías informáticas para el desarrollo de sistemas integrados de modelado y simulación orientado a agentes, integrando SIG, con el objeto de adquirir una comprensión más profunda de la complejidad de los sistemas naturales, sistemas sociales y de planificación urbana.

La integración de estas tres tecnologías, modelado y simulación, agentes de software y SIG, fue llamado por Benenson and M.Torrens [10] *geosimulación,* reconociendolo además como un nuevo campo de investigación y una oportunidad para contribuir con el desarrollo de nuevas herramientas. Más recientemente, Blecic et al. [14] ha dicho que la *geosimulacion multiAgente* es una técnica de simulación para modelado de los fenómenos que tienen lugar en zonas geográficas a través del uso de enfoque basado en agente en modelos de alta resolución espacial.

A continuación se hará referencia a los trabajos de investigación en geosimulación multiAgentes que se han revisado:

- SIG Y MODELADO BASADO EN AGENTES (Westervelt [163]), se establecen criterios para evaluar la integración de los SIG y sistemas de simulación
- UNA VISIÓN ORIENTADA A AGENTE EN UN SIG CON SISTEMAS URBANOS Y AMBIENTALES de Jiang and Gimblett [92], donde explora el paradigma basado en agente para modelar sistemas y ambientes urbanos.

- AGENTES MÓVILES CON INTELIGENCIA ESPACIAL presentado por Itami [87], y donde se describe RBSIM, un programa de computadora que simula el *comportamiento recreativo* de los humanos en ambientes naturales.

- SIMULACIÓN BASADO EN AGENTES PARA LA TOMA DE DECISIONES Y CAMBIO DE USO DEL SUELO donde Lim et al. [107] presenta el sistema de simulación LUCITA, un sistema de simulación multiAgentes compuesto de dos sub-modelos que interactúan a través de un ambiente raster referenciado espacialmente.

- INFRAESTRUCTURA DE SIMULACIÓN MULTIAGENTE PARA PLANIFICACIÓN, donde Blecic et al. [14] presenta a MAGI un metamodelo que representa una teoría formal de una geografía con agentes y objetos en ella. Posteriormente, usaremos MAGI para ampliar la teoría de simulación de GALATEA.

2.4.1 *SIG y modelado basado en agentes*

En Westervelt [163] se establecen criterios para evaluar la integración de los SIG y sistemas de simulación, con las fortalezas de las bases de datos y la simulación. Estos criterios, el autor los definen como:

- Velocidad de integración: el tiempo de programación para establecer el enlace entre los programas
- Experticia del programador: el nivel de experticia del programador del software
- Evitar autorías múltiples: ¿Debe el producto de software desarrollado por un programador ser modificado por el programador que realizará la integración? Si es así, ¿cuanto?
- Velocidad de ejecución: ¿Que tan rápido se ejecuta el software integrado?
- Ejecución simultáneas: ¿Deben los componentes de un sistema correr simultáneamente y comunicarse con otros? ¿Pueden los componentes operar en plataformas separadas?
- Depuración: ¿Que tán difícil resulta la depuración de errores en los sistemas integrados?

Y establece que las soluciones de integración de software caen en tres tipos de interacción: Débil, Moderado y Fuerte. Brevemente, cada interacción esta descrita de la siguiente manera:

- Débil: representa una conexión débil entre los programas de computadoras, típicamente incluye:

- la operación asíncrona de cada módulo por sí mismo, y
- el intercambio de información usando archivos ASCII sencillos

- Moderado: es una conexión lograda por medio de archivos comunes en un formato estándar que es conocido por los programas. Ejemplo de esta visión, es el uso que hacen los SIG de los archivos raster, vectorial o puntos.

- Fuerte: se caracteriza por operaciones simultáneas de los modelos, permitendo que durante la ejecución de los módulos, se comuniquen entre ellos. Una variante de este tipo de interacción es compilar todos los diferentes componentes de los modelos en un único programa. Otra variante de esta interacción entre modelos esta representada con el PD; esta variante se descibre en 2.4.1.

En el cuadro 2.1 se muestra los valores de cada criterio para las categorías que se describieron en 2.4.1. Las cajas coloreadas indican las características más deseables en el proceso de integración de los SIG y sistemas de simulación.

		Tipos de Integración		
		Débil	Moderado	Fuerte
Atributos para comparar integraciones	Velocidad de Integración	rápido	medio	lento
	Experticia del Programador	lento	alto	alto
	Evitar múltiple autoría	alto	alto	bajo
	Velocidad de ejecución	lento	medio	rápido
	Ejecución simultánea	lento	bajo	alto
	Depuración	fácil	moderado	difícil

Cuadro 2.1: Comparación rápida de tres visiones para integración de software. (Aparece en Westervelt [163])

Westervelt [163] agrega dos tipos de interacción para la integración de modelos:

- PD: Esta es una variante de la interacción fuerte. La diferencia está en que en vez de compilar todos los componentes de los modelos en un único programa, los componentes del modelo corren en programas separados y se comunican con los otros durante la ejecución de la simulación. Esta solución implica construir canales de comunicación entre los modelos haciendo uso de tecnologías como Common Object Request Broker Architecture (CORBA) [7], HLA (DMSO [47]) o Dynamic Information Architecture System (DIAS)(Christiansen [26]).

7 está en http://www.corba.org/

- OO: la programación OO esta considerada como una " visión agregada" ya que puede usarse con cualquiera de las interacciones descritas en el cuadro 2.1 y en 2.4.1. Esta visión es tomada en cuenta debido a que "... los software SIG están basados en desarrollos con software no-OO...".

En el cuadro 2.2 se muestran los valores que le ha asignado Westervelt [163] a los criterios de evaluación para los tipos de interacción adicionales.

	PD	OO
Velocidad de Integración	lento	incrementa
Experticia del Programador	alto o muy alto	poca
Evitar múltiple autoría	bajo	incrementa
Velocidad de ejecución	alto	decrece
Ejecución simultánea	alto	sin afectar
Depuración	difícil	eficiencia mejorada

Cuadro 2.2: Comparación de dos visiones PD y OO según la clasificación de Westervelt [163]

2.4.2 *Una visión orientada a agente en un SIG con Sistemas urbanos y ambientales.*

Jiang and Gimblett [92], explora el paradigma basado en agente para modelar sistemas y ambientes urbanos. Los SIG basado en celdas son herramientas utilizada para análisis y modelado espacial. Pero su dificultad es la habilidad para considerar la dimension temporal puesto que está circunscrito a la celda y no toma en cuenta la dinámica espacio-tiempo. Establece que existen varios contribuciones en el campo de la tecnología AC para usar la dinámica espacio-tiempo. Clasifica estas contribuciones como:

- AC para dinámica urbanas y regionales (White [164]).
- AC para modelos ambientales (Camara et al. [23], Clarke and Olsen [27]).
- AC para integración con SIG (Couclelis [29])

Sin embargo, estima que existen propiedades que pueden ser modeladas como propiedades "que emergen" de interacciones individuales. Desde ese punto de vista, el autor considera que los agentes pueden resultar una buena alternativa para modelar la dinámica espacio-tiempo.

La simulación multiAgente, Jiang and Gimblett [92] la describen como una tupla con cuatro elementos:

< agentes, objetos, ambiente, comunicaciones >

donde los *agentes* representa a los agentes en el SMA, *objetos* son las entidades pasivas, *ambiente* es el espacio donde los agentes están localizados y *comunicaciones* las definen como el conjunto de todas las categorías de comunicaciones. SMA puede ser considerado una combinación de AC y agentes autónomos al establecer que el atributo *ambiente* esta representado por un AC. Luego, presenta dos software para desarrollar simulaciones SMA: SWARM[8] (en Minar et al. [115]) y STARTLOGO[9]. Finalmente los autores presentan la descripción general de un SMA que titula "Pedestrian Movement in Urban Systems".

2.4.3 *Agentes móviles con inteligencia espacial.*

En Itami [87] se describe RBSIM, un programa de computadora que simula el *comportamiento recreativo* de los humanos en ambientes naturales. El *comportamiento recreativo* esta definido como un conjunto de actividades que un humano podría realizar en un ambiente, tales como: bicicleta, escaladas y paseos en "jeep". RBSIM usa un SIG para representar el ambiente, y agentes autónomos para simular el comportamiento humano dentro del espacio geográfico. El autor clasifica dos maneras de integrar SIG con agentes:

- El programa de simulación trabaja independientemente del SIG: el simulador pasa parámetros al SIG para que ejecute operaciones propias y devuelva resultados en forma de mapas o variables.
- Acceso a los mapas contenidos en el SIG directamente desde el simulador y encapsulando solo aquellas funciones que son necesarias.

La primera opción tiene como la ventaja de requerir poca programación y ganar la funcionalidad del SIG; y la desventaja gira en torno a la adecuación de las funciones estándar del SIG al modelo de simulación multiAgentes y a la baja velocidad con que operan estas funciones. La segunda opción tiene como desventaja el aumento del tamaño y la complejidad del programa de simulación debido a la necesidad de soportar operaciones Entrada/Salida (E/S) para leer y escribir los mapas de los SIG; y la ventaja es que las pueden escribirse funciones de usuario optimizando el código para asegurar un mejor rendimiento en la ejecución de la simulación.

8 Está en http://www.swarm.org/index.php/Main_Page
9 Está en : http://education.mit.edu/starlogo/

2.4.4 *Simulación basado en agentes para la toma de decisiones y cambio de uso del suelo.*

Lim et al. [107] presenta el sistema de simulación LUCITA, un sistema de simulación multiAgentes compuesto de dos sub-modelos que interactúan a través de un ambiente raster referenciado espacialmente. Los sub-modelos se utilizan para capturar uno, la dinámica ecológica y el otro, la dinámica humana. El sub-modelo ecológico, basado en ecuaciones de regresión múltiples, se derivan del modelo KPROG2 de Fearnside [60], que estima las capacidades de transporte humano a través de la carretera TransAmazónica.

El sub-modelo ecológico modela el impacto de la deforestación en las propiedades del suelo, la relación entre la fertilidad del suelo y los campos de cultivos exitosos, y los efectos de la fertilidad del suelo y la relación con la reforestación natural. El sub-modelo del sistema humano describe la arquitectura de un agente propietario de tierra autónomo. Cada agente propietario representa el núcleo familiar de unos colonos, la labor del jefe de familia y el capital disponible. LUCITA fue desarrollado usando el sistema de simulación SWARM (Minar et al. [115])

2.4.5 *Infraestructura de simulación multiAgente para planificación.*

El meta-modelo presentado por Blecic et al. [14], representa una teoría formal de una geografía con agentes y objetos en ella. Sirve, al igual que la teoría de Galatea Dávila and Uzcátegui [39], como una especificación formal que guía la implementación de un sistema geocomputacional llamado MAGI actualmente usado como un geosimulador.

En la teoría MAGI el ambiente *Env*, esta representado por las 3-tuplas:

$$Env =< P_G, F_G, \mathcal{L} > \tag{2.17}$$

donde $\{P_G\}$ representa el conjunto de todos los posibles pares *parámetro* = *valor* definiendo atributos del sistema y $\{F_G\}$el conjunto de todas las posibles funciones globales operando sobre esos parámetros y valores, mientras $\mathcal{L}$ es el conjunto de las posibles capas que describen una geografía. Cada capa , $L \in \mathcal{L}$, a su vez esta definida por una tupla-3:

$$L =< P_L, F_L, \mathcal{A} > \tag{2.18}$$

donde$\{P_L\}$ and $\{F_L\}$ son las contrapartes locales de $\{P_G\}$ y $\{F_G\}$, y $\mathcal{A}$ representa el conjunto de entidades (objetos y agentes) que pueblan el sistema. Noten que, en esta solución, el número de agentes no está definido (ni, por tanto, acotado). Los agentes son, a su vez, descritos

por un doble registro: el agente mismo (o un agente particular de tipo τ) se describe con una tupla-3:

$$a_\tau =< s, g, C > \tag{2.19}$$

donde s es el estado interno del agente, g representa los atributos geo-espaciales del agente (atributos descriptivos del cuerpo del agente) y C representa un contexto espacial: el conjunto de referencias a objetos y agentes observadas por este agente y sujeto a sus acciones.

El correspondiente tipo de agente τ se formaliza por una tupla-6:

$$\tau =< S_\tau, Shapes_\tau, \sum_\tau, \Theta_\tau, \delta_\tau, \gamma_\tau > \tag{2.20}$$

dondeS_τ es el conjunto de todos los posibles estados internos del agente, $Shapes_\tau$ representa el conjunto de capas admisibles para este tipo de agente, $\sum_\tau$ representa el conjunto de todas las posibles acciones, Θ_τ es el conjunto de las funciones de percepción, δ_τ, el conjunto de las funciones de decisión y γ_τ el conjunto de las funciones de negociación por medio de la cual el agente coopera con otros agentes.

Es claro que esto corresponde a las características de un sistema multiagente, debido al hecho que los agentes tienen atributos bien definidos para su representación y ubicación en un espacio físico, aún cuando no hay previsiones particulares para la dinámica interna de cada agente.

Conclusión.

En el campo de investigación de la geosimulación, la integración de los SIG y los agentes de software representa una oportunidad para superar la limitante que presentan los SIG con el manejo del tiempo.

MAGI representa una teoría formal de un geosimulador con agentes y objetos, pero no contempla un registro del tiempo. Esta falta puede ser subsanada al integrar la teoría MAGI con la teoría de simulación para GALATEA.

Los resultados de la integración de la teoría MAGI y de la teoría de simulación para GALATEA, y que es el objetivo principal de esta investigación, se verá a continuación en el capítulo Capítulo 3.

2.5 COMPUTACIÓN ORIENTADA A SERVICIOS.

En la sección 2.2 veíamos que varias de la metodologías hacen mención a conceptos como servicios, tareas y procesos. La metodología AAII, sección 2.2.1, considera que los *servicios* son parte de la definición de la interfaz del agente y lo asocia a las tareas que el agente

debe realizar para cumplir un rol. En GAIA, sección2.2.2, se describe a los *servicios* como un conjunto de actividades, para que un agente cumpla con el rol asignado y se considera la construcción de un Modelo de Servicio como parte del modelado del SMA. La metodología MaSE, sección 2.2.3, no hace referencia directa a los *servicios* pero le asocia a los agentes *roles* y *tareas* asociadas a esos roles.

MESSAGE/UML, sección 2.2.5, conceptualiza los *servicios* como la capacidad, de los agentes, de realizar alguna función útil y describe al *servicio* como un conjunto de tareas. La metodología MAS-CommonKADS, sección 2.2.8, considera que los *servicios* están asociados a los agentes y representan funciones de utilidad que son requeridas bajo demanda. Para la metodología O-MaSE, 2.2.9, se define un *servicio* como un conjunto de actividades que el agente lleva a cabo con el objeto de cumplir con su rol asignado.

En el MRA no se hace una referencia explicita al concepto de *servicios*, pero al igual como lo refieren gran parte de la metodologías mencionadas, los *servicios* están relacionados con los roles del agente y se componen de una serie de actividades y tareas para cumplir ese rol.

También, el tema de los servicios orientados a la *web* es recurrente, sobre todos en aquellos, como [144, 110, 131], que buscan definir con claridad su significado y establecer las diferencias entre servicios, servicios web y las distintas arquitectura orientada a servicio. Es por eso que es necesario conocer los conceptos fundamentales y establecer las relaciones, o diferencias, que tienen cada concepto en el área de conocimiento específico de la SOC.

2.5.1 *Computación Orientada a Servicios.*

La SOC [155], es un paradígma de desarrollo usado para representar una nueva generación de plataformas de computación distribuidas. Este paradigma prioriza el reuso y la interoperabilidad de las aplicaciones y servicios para la construcción de aplicaciones de usuario en ambientes distribuidos heterogéneos mediante la composición y el ensamblado de funcionalidades existentes, o servicios.

2.5.2 *¿Que es un Servicio?*

En W3C [160] definen un servicio como un recurso abstracto que representa una capacidad de realizar tareas que constituyen una funcionalidad coherente desde el punto de vista de los proveedores de las entidades y de las entidades solicitantes.

De manera más amplia un servicio es un conjunto de actividades que busca dar respuesta a las necesidades de un cliente. Específicamente, en [155] lo definen como una unidad lógica que proporciona una respuesta y a la cual se le ha aplicado la *orientación a servicio* en

su significado extendido. La *orientación a servicio* en [155], es un paradigma de diseño destinado a la creación de unidades lógicas que se diseñan de manera individual para que puedan ser utilizada colectiva y repetidamente, en apoyo de la realización de un conjunto específico de objetivos estratégicos y con los beneficios relacionados a SOA y SOC.

Es esta aplicación de los principios de diseño orientados a servicios lo que distingue a la unidad lógica como un *Servicio* comparado con la unidad logica que puede existir solo como componentes de un objeto de software. El servicio representa en el paradigma de la SOC lo que el objeto es en el paradigma de la OO.

2.5.3 *Arquitectura orientada a Servicios.*

La SOA[155], es un concepto de arquitectura de software para soluciones orientadas a servicios con características distintivas en apoyo a la realización de aplicaciones orientadas a servicios y de las metas estratégicas asociadas con la computación orientada a servicios. Una implementación SOA puede consistir de una combinación de tecnologías, productos, Application Programming Interface (API) y, extensiones de infraestructura de apoyo, entre otros.

Las características de SOA, según Erl [58] es que es orientada a componentes y objetos distribuidos, débilmente acoplada, orientada al reuso de software, con BD dedicadas y con mínima representación de estados.

2.5.4 *¿Que es un Servicio Web?*

Un *Servicio Web* [155]es un conjunto de unidades lógicas que ofrecen un contrato técnico, físicamente desacoplado, y que consiste en una definición expresada en un lenguaje específico como el Web Service Description Language (WSDL), una definición de esquemas, eXtensible Markup Language (XML) y, posiblemente, una definición de políticas del servicio. Este contrato de servicios expone las funciones públicas o disponibles (llamadas operaciones) y por lo tanto, es comparable a una API.

Conclusión.

El concepto del *servicio* está relacionado con las funciones que un agente, o un conjunto de ellos, realiza en un SMA. Las metodologías para SMA, en general, consideran los *servicios* como un conjunto de tareas y actividades que un agente debe realizar para cumplir con un rol. De manera similar, se consideran los *servicios* en el MRA y en la SOC.

La SOC es un paradígma de desarrollo que se enfoca en el acoplamiento débil de los elementos que la integran. La SOC se enfoca en la minimización de dependencias innecesarias entre sistemas y elementos de software mientras se mantiene la funcionalidad de los elementos, promoviendo la reutilización de software.

3

UNA NUEVA APROXIMACIÓN A LA TEORÍA DE SIMULACIÓN MULTIAGENTES.

La importancia de un modelo de referencia para geosimulación multiAgentes es que posibilita la comprensión y descripción de un producto de software y le proporciona al analista de sistema una herramienta para asegurar la correctitud de sistemas espaciales complejos.

Este capítulo está compuesto de las secciones: El Modelo Formal para geosimulación multiAgentes y Bases de Datos. En la primera sección se presenta una nueva visión de la teoría multiagentes para simulación de GALATEA, presentada por Dávila et al. [44].

En la siguiente sección se dedica a clarificar, por medio del modelo de referencia, la relación entre sistemas multiagentes y un modelo formal de BD que incluya las características de las BDA y BDD para demostrar que el modelo formal de geosimulación multiAgentes es un modelo más general.

3.1 EL MODELO FORMAL DE GEOSIMULACIÓN MULTIAGENTES.

La teoría multiagentes para simulación de GALATEA, se conecta con la teoría MAGI de Blecic et al. [14] para producir una teoría multiAgentes para geosimulación, la cual explica la relación entre agentes y SIG; como una herramienta para simular sistemas espaciales complejos.

La teoría MAGI es un complemento perfecto para la teoría de simulación y el modelo de referencia puesto que

1. define el cuerpo físico de cada agentes y su ubicación en una geografía

2. establece una relación cuidadosamente detallada de la estructura de datos y funciones asociadas requeridas para que un sistema de información geográfica calcule eficientemente las respuestas a consultas.

La teoría original de Galatea (Dávila and Uzcátegui [39]) no contempla esos elementos. Un tercer efecto colateral de la combinación de teorías es la posibilidad de dar cuenta de la creación tanto de objetos

como de agentes. Por su lado, la teoría MAGI obtiene de GALATEA un registro explícito del tiempo.

En la figura 3.1 se muestra el esquema del geosimulador multi-Agentes. La zona sombreada comprende la definición de la teoría de simulación propuesta por Dávila et al. [44], expresada en el simulador GALATEA. Esta zona describe un sistema poblado de agentes descrito a través de la función $Step_{in}$, la cual *transporta* el sistema desde un estado global al próximo cronológicamente.

Tal y como se expresó en la teoría de simulación de Dávila et al. [44], el estado global se caracteriza como un conjunto de variables de estados y sus valores, representado en esta formalización por σ (y similares); y el estado global también incluye un conjunto conocido como las influencias, γ (y similares), que representan todas las acciones que el agente ha ejecutado y que están actualmente tratando de ejecutar.

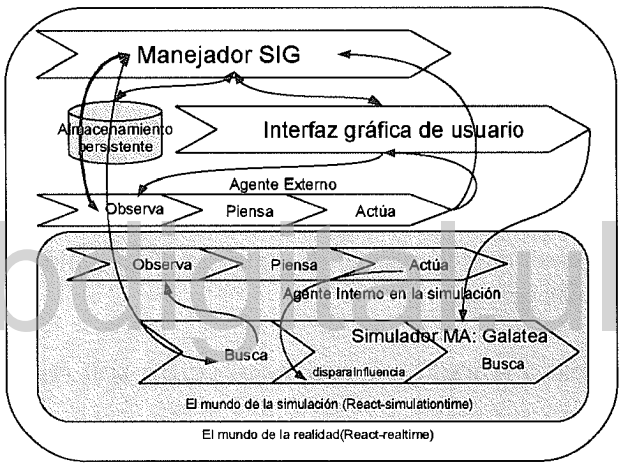
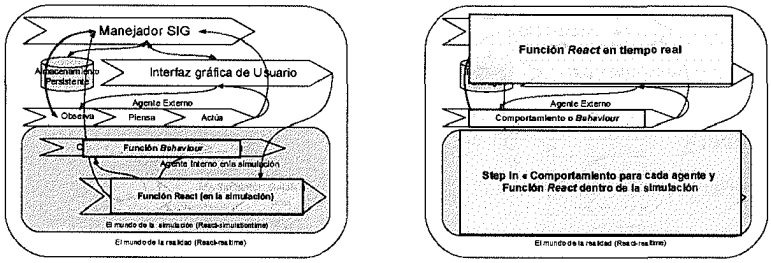


Figura 3.1: Esquema del Geosimulador GALATEA

Esta zona descrita por la función $Step_{in}$, se ha llamado el *mundo de la simulación* y evoluciona al paso del *tiempo de simulación*. El comportamiento del agente en *tiempo de simulación* se describe con la función $Behaviour_{inter}$ y la reacción del ambiente a las influencias del agente se expresa con la función $React_{simulationtime}$ (ver figura 3.2a). Ambas funciones están circunscritas al simulador.

A la teoría de simulación de GALATEA se ha agregado agentes externos que están en una geografía específica, definida en un SIG. Esta zona describe una geografía poblada de agentes descrito a través de la función $Step_{out}$, la cual *transporta* el sistema desde un estado global al próximo cronológicamente. La función $Step_{out}$, se ha llamado el *mundo de la realidad* para diferenciarlo del *mundo de la simulación*, y evoluciona al paso del *tiempo real*.



(a) Funciones en el mundo de la simulación

(b) Funciones en el mundo real

Figura 3.2: El mundo Real y el mundo de la Simulación en el Geosimulador

Los agentes interactúan con la simulación a través de las estructuras de datos proporcionadas por el SIG o estructuras de datos externas; y los agentes interactúan con los usuarios por medio de la interfaz gráfica del SIG. Esta interacción de los agentes con el SIG y con el usuario se muestra en la figura 3.1 con las flechas que entran/salen del agente.

De la misma manera y como se observa en la figura 3.2b, en el *tiempo real*, el comportamiento del agente se describe con la función $Behaviour_{ext}$ y la reacción del ambiente a las influencias del agente en *tiempo de simulación* se expresa con la función $React_{realtime}$.

En esta propuesta, la función Env^*, es una representación más general de la evolución del sistema que contiene las funciones $Step_{in} \bigcup Step_{out}$ (figura 3.3).

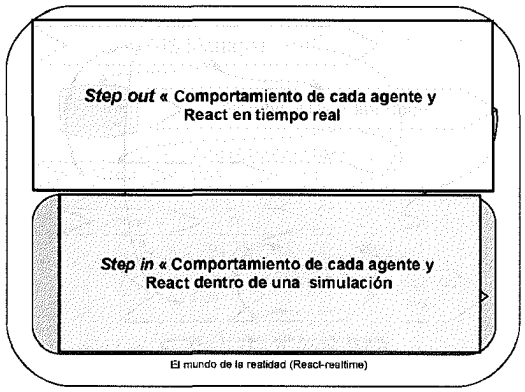


Figura 3.3: Función Evolución en el Geosimulador

A continuación, veamos el modelo formal que se expresó en la figura3.1.

3.1.1 *Teoría de Geosimulación multiAgentes*

Se preserva la notación de la teoría MAGI solo parcialmente en el siguiente replanteamiento de la teoría de simulación multiagentes.

La función evolución de un sistema es una función recursiva y se describe:

$Evolution : \Im \otimes Env^* \rightarrow \epsilon$

$$Evolution(t, env) = Evolution(Step_{out}(t, env)) \tag{3.1}$$

donde $\Im$ representa el tiempo y $env \in Env^*$ es una estructura que contiene todos los estados globales, como elementos de Env^*, pero también las influencias en el sistema:

$$Env^* =< P_G, F_G, \mathcal{L}, \Gamma > \tag{3.2}$$

donde P_G representa el conjunto de todos los posibles pares *parámetro* = *valor* que definen los atributos del sistema; F_G el conjunto de todas las posibles funciones globales operando sobre esos parámetros y valores; $\mathcal{L}$ es el conjunto de las posibles capas que describen una geografía y Γ representa el conjunto de todas las posibles acciones que los agente han ejecutado y que están actualmente tratando de ejecutar.

$Step_{out}$ reune todas las influencias producidas por los agentes, la cual esta determinada por la conducta del agente mismo, y la dinámica del ambiente, por lo tanto:

$Step_{out} : \Im \otimes Env^* \rightarrow \Im \otimes Env^*$:

$$< t'', env'' >= Step_{out}(t, env) \tag{3.3}$$

donde,

$$env'' =< p'', f, l'', \gamma'' >$$

y

$$env =< p, f, l, \gamma >$$

donde p representa el conjunto de todos los posibles pares *parámetro* = *valor* que definen los atributos de una capa en particular; f el conjunto de todas las posibles funciones definidas para esa capa en particular y que operan sobre esos parámetros y valores; l es una capa que describe una geografía y γ representa el conjunto de todas las posibles acciones que los agente han ejecutado y que están actualmente tratando de ejecutar sobre esa capa.

La reacción ambiental a las influencias de los agentes está definida por:

$$< p'', f, l'' >= React_{realtime}(t', p', f, l', \gamma') \tag{3.4}$$

y

$$< \gamma'' >= Exec(p', f, l', \gamma' \cup_i \gamma_i') \tag{3.5}$$

y la conducta del agente se expresa como:

$$< s_i'', \gamma_i' >= Behaviour_{ext}(t', r_{ext}, s_i', \gamma) \tag{3.6}$$

donde cada s' es una abreviación de $< k_i, g_i' >$, la base de conocimiento y metas del agente i.

$Step_{in}$ reune todas las influencias producidas por los agentes, la cual esta determinada por la conducta del agente mismo, y la dinámica del ambiente, en *tiempo de la simulación*, por lo tanto:

$Step_{in} : S \otimes \Im \otimes Env^* \otimes \Gamma \rightarrow S \otimes \Im \otimes Env^* \otimes \Gamma$:

$$Step_{in}(< s_1, s_2, .., s_n >, t, < p, f, l >, \gamma) = << s_1', s_2', .., s_n' >, t', < p', f, l' >, \gamma' > \tag{3.7}$$

donde, a su vez, la reacción ambiental a las influencias de los agentes en la simulación está definida por:

$React_{simulationtime} : \Lambda \otimes \beta \otimes \Im \otimes Env^* \otimes \Gamma$:

$$<< p', f, l' >, \gamma' >= React_{simulationtime}(<\lambda_1 | | ... | | \lambda_m>, t, < p, f, l >, \gamma \cup_j \gamma_j) \tag{3.8}$$

y la conducta del agente, en *tiempo de simulación*, se expresa como:

$$< s_j', \gamma_j >= Behaviour_{inter}(t, r_{inter}, s_j', \gamma) \tag{3.9}$$

y $l \in \mathcal{L}$, $l =< p_l, f_l, A >$, es una capa que representa una geografía con parámetros locales, funciones locales y agentes sobre ella,

y A es el conjunto de agentes en esa capa .

3.1.1.1 *Modelo Formal de Agentes*

Un agente se describe con cuatro elementos:

$$a_\tau =< k, goals, georefs, Context > \tag{3.10}$$

donde $k \in K_\tau$, es la base de conocimiento del agente, $goals \in G_\tau$, son las metas del agente $georefs \in Shapes_\tau$ es la forma del agente y $Context$ es el contexto espacial.

Nótese que, como lo indica el subíndice, este agente está asociado a un tipo: τ el cual, a su vez, se formaliza por ocho elementos:

$$\tau =< K_\tau, G_\tau, Shapes_\tau, \sum_\tau, P_\tau, Perception_\tau, Update_\tau, Planning_\tau > \tag{3.11}$$

donde K_τ es el conjunto de bases de conocimiento posibles para este tipo de agente, G_τ es el conjunto de metas posibles para este tipo de agente, $Shapes_\tau$ es el conjunto de formas admisibles que el cuerpo del agente (de este tipo) puede adoptar, $\sum_\tau$ es el conjunto de acciones posibles que el agente puede ejecutar, P_τ es el conjunto de observaciones posibles que el agente puede hacer, y *Perception*, *Update* y *Planning* son, como se explicó antes, funciones para modelar las conexiones entre percepción y acción para este tipo de agente.

Todas estas estructuras especificadas para el agente o para el tipo de agente pueden conectarse con el resto del sistema por la función modificada del comportamiento del agente, como se indicó anteriormente en las ecuaciones.

La conducta del agente se expresa como:

$$Behaviour_a : \Im \otimes \Re \otimes K_\tau \otimes G_\tau \otimes \Gamma \rightarrow K_\tau \otimes G_\tau \otimes \Gamma \tag{3.12}$$

$$< k', goals', \gamma_a >= Behaviour_a(t, r_a, k, goals, \gamma) \tag{3.13}$$

donde

$$k' = Update_a(t, Perception_a(\gamma), k) \tag{3.14}$$

$$< \gamma_a, goals' >= Planning_a(t, r_a, k', goals) \tag{3.15}$$

donde$< k', goals', \gamma_a >$, es la base de conocimiento del agente, las metas del agente a y las influencias,γ_a, que este agente está enviando a su ambiente como acciones que intenta ejecutar.

Como antes, el estado global, constituido por parámetros globales, funciones y capas, es recorrido adecuadamente para activar sus eventos y cambios, con:

$$< p, f, l >= Scan^*(env, \xi) \tag{3.16}$$

$$\xi = NextEvent(\gamma) \tag{3.17}$$

$$t' = TimeOf(\xi) \tag{3.18}$$

que son, como se indicó anteriormente, los elementos usuales del simulador DEVS [172, 162]. La más alta estructura *env* es ahora el sujeto de la búsqueda para identificar los componentes actuales donde ocurrirá el próximo evento ξ y las instrucciones asociadas. Así, esto es probablemente una forma diferente de recorrido, más específico a los atributos geográficos del sistema.

Conclusión.

En la primera sección se presenta una nueva visión de la teoría multiagentes para simulación de GALATEA, presentada por Dávila et al. [44]. Se adapta la teoría multiagentes para simulación de GALATEA, y se conecta con la teoría MAGI de Blecic et al. [14] para producir una teoría multiAgentes para geosimulación, la cual explica la relación entre agentes y SIG, como una herramienta para simular sistemas espaciales complejos.

La teoria multiAgentes para geosimulación además de describir un sistema poblado por muchos agentes, sus componentes y sus reglas de transición, describe físicamente el sistema geográfico, define el cuerpo físico de cada agentes y su ubicación en una geografía, da cuenta de la creación tanto de objetos como de agentes, y proporciona un registro explícito del tiempo.

3.2 BASES DE DATOS

Las primeras bases de datos fueron colecciones de tipos de datos estructurados y predefinidos Elmasri and Navathe [54], Paton and Diaz [130] y las aplicaciones estaban enfocadas a dominios particulares de aplicaciones, tales como sistemas administrativos.

Desde 1980, se comenzaron a desarrollar nuevos y más exigentes dominios de aplicaciones (Paton and Diaz [130], Bernstein et al. [11], Silberschatz et al. [153]) los cuales necesitaron manejar tipos de datos complejos y no estructurados, y aplicaciones orientadas a otros dominios, como los datos geográficos, multimedia, entre otros.

Para dar respuesta a estos requerimientos, se han desarrollados diferentes modelos de bases de datos. Dos modelos prominentes son las BDD y las BDA.

Elmasri and Navathe [54] dice que las BDD están conformada por tres elementos: hechos o información extensional, reglas deductivas y Restricciones de Integridad (RI). Una regla deductiva define la información intensional, que se define como la información que no está explícitamente guardada dentro de la base de datos (Teniente [157]). Una regla tiene la forma *cabeza :- cuerpo* y, por lo general, la cabeza de la regla tiene un solo predicado, y el cuerpo de la regla lo conforman uno o más predicados. Una base de datos deductiva es

capaz de inferir informacion a partir de informacion explicitamente guardada.

Los modelos de BDA Elmasri and Navathe [54] los define como un sistema de BD capaz de detectar situaciones de interés y tomar acciones en consecuencia por medio de las reglas Evento, Condición, Acción (ECA). Las reglas ECA tiene tres componentes:

- Evento que activa la regla: los eventos son operaciones de actualizacion de base de datos que se aplican sobre la base de datos.
- Condición que determina si la acción de la regla debe ejecutarse: una vez que se ha producido el evento, puede evaluarse una condición opcional, y si la evaluación es verdadera entonces se ejecutará la acción de la regla.
- Acción: es un conjunto de sentencias en SQL, o una transacción de base de datos o un programa externo que se ejecutará automáticamente.

Sin embargo, a pesar de los esfuerzos para extender las funcionalidades de las bases de datos (Geppert and Dittrich [70]), algunos investigadores (Agrawal et al. [1]) insisten en la oportunidad de explorar nuevos paradigmas que permitan a las BD enfrentarse a los requerimientos actuales, en áreas como manejos de datos complejos y grandes, integración de datos estructurados y no-estructurados, aplicaciones para dispositivos móviles y redes sociales.

A continuación se describe un modelo genérico para un sistema de BD que contempla las características de las BDA y las BDD, y se muestra como están incluidos ambos conceptos dentro del modelo de referencia de geosimulación multiAgentes.

Tomando como base el modelo formal de una BDD propuesto por Gallaire et al. [67], se propone un modelo formal que intenta dar cuenta de las BDA. Los componentes estáticos de una base de datos deductiva se pueden describir por tres elementos:

$$< T, IC, Q >$$

donde T es una teoría que incluye los axiomas de dominio cerrado, de nombres únicos, de igualdad y de completitud, hechos elementales y reglas deductivas; IC es el conjunto de las RI y las reglas ECA; y Q representa el conjunto de consultas posibles.

El modelo de agentes subsume este modelo al considerar que $T \subseteq K_a$ e $IC \subseteq G_a$, siendo K_a y G_a la base de conocimiento y las metas del agente, respectivamente, como fue descrito en la sección 3.1.1.1.

Nótese también que $Q \subseteq P_a$, es decir, Q corresponde al conjunto de posibles entradas que una BD puede tener. Se puede asociar la base

de datos con un tipo de agente geofísico (de type τ) si se usa a, K_τ, G_τ y P_τ.

3.2.1 *Modelo Formal de una Base de Datos y el Modelo Formal del Agente.*

Las similitudes entre modelo genérico de una BD y el modelo de referencia de un geosimulador multiAgente pueden ser esquematizada como se indica en la figura 3.4. La BD está subsumida por la estructura de creencias. Cada tabla en la BD corresponde a la definición de un predicado del conjunto de creencias del agente.

El conjunto de consultas, RI, y reglas ECA pueden ser vistas como metas. Todas ellas tienen en común que comparten la misma semántica, pero difieren en su objetivo.

Como lo establece Kowalski and Sadri [103], mientras las RI se asocian a las consultas persistentes, que equivalen a las metas de mantenimiento en el modelo de agente; las consultas *ad hoc* se relacionan a un estado determinado de la base de datos y pueden ser vistos como metas de logro.

Las reglas ECA son un tipo de metas, que pueden provenir de las metas de alto nivel. Gallaire et al. [67] dice que las reglas deductivas sirven para dos propósitos: como reglas para definir datos o reglas de razonamiento hacia adelante, y que han sido integradas a los hechos de la base de datos en la figura 3.4.

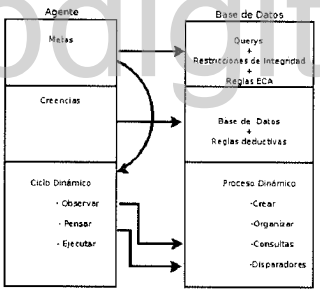


Figura 3.4: Agentes y Bases de Datos

El aspecto dinámico de la BD, expresados en el modelo de ejecución, y la máquina de inferencia, pueden incluirse en los mecanismos *Update* y *Planning* del agente definido en el modelo de referencia de un geosimulador multiAgente. Además, el modelo de referencia de un geosimulador multiAgente proporciona un mecanismo de activación que no es encontrado en modelos de BD estándares.

Este mecanismo proporciona al agente la habilidad de desarrollar el comportamiento reactivo y proactivo para la prosecución de sus metas. El comportamiento reactivo está asociado con cambios en la BD, disparado por actualizaciones a traves de las RI. En el modelo de referencia, el comportamiento reactivo es una característica

que se construye por la naturaleza abierta de cada agente que constantemente observa su ambiente y actua en consecuencia.

El agente está inmerso en un proceso dinámico llamado ciclo *observa-piensa-ejecuta,* el cual monitorea constantemente el mundo que lo rodea y cuando detecta que alguna de sus observaciones concuerdan con las condiciones de sus reglas, entonces se ejecutan las acciones correspondientes.

El ciclo del agente *observa-piensa-ejecuta* se reduce al proceso de *observa-piensa* en una BD, donde *observa* corresponde a las consultas enviadas a la BD y *piensa* podría corresponder al mecanismo de ejecución o disparo de las reglas. Es importante notar que el ciclo dinámico del modelo de referencia multiagente se ocupa también de las actividades de la base de datos *crea* y *organiza* , que están definidas en el ciclo dinámico del modelo de la base de datos (figura 3.4).

3.2.2 *El Modelo de Formal del Agente y otros modelos de Bases de Datos.*

Así, el modelo de referencia de sistemas multiagentes puede subsumir otros modelos de BD tal y como es el modelo de Base de Datos Orientada a Objeto Activa (BDOOA) (Beeri and Milo [9]). En las BDOOA, los objetos están separados en objetos pasivos, o recipientes, y objetos activos los cuales representan a los agentes. El intercambio de mensajes entre objetos Kowalski [99] los clasifica como:

1. *objeto pasivo-objeto pasivo,* que representa una observación,
2. *objeto activo-objeto pasivo,* también observaciones,
3. *objeto activo-objeto pasivo con cambio de atributos en el receptor*, que representa una acción, y
4. *objeto activo- objeto activo,* que es una acción para el emisor y una observación para el receptor.

También, los *disparadores* en las BDOOA son un tipo de método. El modelo de agente descrito puede representar métodos que son *disparados* por un conjunto de observaciones (y eventos). También el modelo puede representar otras variantes de disparadores en la cual se ejecutan las reglas que se activan después de que son usadas las restricciones de integridad para producir metas y sub-metas intermedias, reduciendolas hasta acciones.

Así, el modelo de referencia de geosimulación multiAgente es un modelo más general que toma en cuenta, no solo la eventual respuestas a consultas o el comportamiento al activarse las reglas ECA, sino también por presentar un comportamiento autónomo, persistente y auto-sostenido.

3.2.3 *La Relación entre agentes y bases de datos.*

La relación entre agentes y BD pueden ser exploradas respondiendo las interrogantes del estudio. ¿Están las BD dentro de los agentes? Los agentes poseen una base de conocimiento o almacenes de creencias que pueden contener una BD tradicional y, también una BDD. Más aún, usando sus metas como un datos almacenados, el agente puede ser visto como que contiene completamente una BDA.

¿Son los agentes una especie de sistema de BD? De hecho, como contenedor y administrador de sus propias metas y creencias, los agentes son, también, gestores de sus propios datos. Pero van más alla de la *consulta-respuesta* de sistema de gestión de BD tradicional al mediar con actualizaciones válidas que deben registrarse y acciones derivadas a partir de los cambios producidos en los datos.

¿Cual es la dinámica de los agentes y las BD? Los agentes pueden ser vistos como inmersos en un ciclo permanente de *observa-piensa-actua*. Este ciclo es una generalización de la dinámica tradicional de una BD que está limitada a *consultas-respuesta* (una actividad "pensante").

¿Están conectados? La relación entre modelos de BD y modelos de agente (figura 3.4) ya fue discutida. El modelo de agente mayormente subsume el modelo de BD. La representación del modelo de BD en el modelo de geosimulación multiAgentes incluye el proceso de creación BD , considerando que el modelo de referencia da cuenta de la creación tanto de objetos como de agentes.

¿Se complementan? El ciclo de vida de una BD puede ser subdividido en la fase de creación y la fase de uso. El último corresponde a una dinámica de régimen estable en la cual actualmente el modelo de agente subsume el modelo de BD tomando en cuenta que los cambios y actualizaciones o estimulos en el modelo de agentes provienen del medio ambiente.

Conclusión.

Se ha mostrado que el modelo de agentes establecido en el modelo formal de geosimulación multiAgentes subsume el modelo formal que describe a BDA , BDD y BDOOA

Por tanto, el modelo de referencia de geosimulación multiAgente es un modelo más general que toma en cuenta, no solo la eventual respuestas a consultas o el comportamiento al activarse las reglas ECA, sino también por presentar un comportamiento autónomo, persistente y auto-sostenido.

Parte III

APLICACIÓN PRÁCTICA

4

VALIDACIÓN EXPERIMENTAL DEL MODELO FORMAL DE GEOSIMULACIÓN MULTIAGENTES

En este cápitulo se presenta el diseño de un sistema de simulación con BD, SIG y agentes, con el objeto de demostrar como el modelo formal expuesto en el capítulo Capítulo 3 puede servir como especificación y guía metodológica para el proceso de modelado y construcción de un sistema complejo.

El marco experimental se ha pensado como un proceso planificado de reuso de sistemas completos y arquitecturas enteras. El reuso de sistemas permite asegurar la validez científica en los resultados obtenidos, como muy bien lo afirma Anderson [2] " ... mientras es ciertamente posible construir completamente ambientes especializados desde cero para un nuevo proyecto ..., idealmente nos gustaría una herramienta que sea lo suficientemente flexible para soportar un gran variedad de agentes y su entorno. La razón de esto no es solamente el deseo de evitar reinventar la rueda. Aspectos como el control y la verificabilidad, por ejemplo, están también involucrados."

Se inicia este capítulo presentando el dominio de la aplicación que es la GDRR, Lavell [106], Cushla and Ochoa [32], Valdés [159]. Luego se presenta el diseño general del sistema para GDRR y los objetivos establecidos para el desarrollo del sistema. Se ha empleado la metodología de la *programación extrema Beck [8]* para la implementación de los módulos que componen el sistema a partir de las plataformas base: HLA y los simuladores EpaNet y GALATEA.

4.1 EL DOMINIO DE LA GESTIÓN DE DESASTRES Y REDUCCIÓN DE RIESGO

Los desastres ocupan un espacio cada vez más importante del debate público debido, desde luego, a las pérdidas lamentables de vidas humanas y de infraestructura, pero también porque sus efectos se acumulan para influir negativamente sobre las posibilidades de desarrollo de los países y, en general, en la defensa de los derechos humanos.

Sobre la GDRR, Valdés [159] dice "... como se ha ido definiendo durante el Decenio, tiene muchas definiciones y matices, pero se entiende como el conjunto de actividades que se realizan para

eliminar o reducir los elementos expuestos a un posible fenómeno destructor de origen natural o socio-natural y mitigar su impacto. Se refiere a las acciones para reducir las causas o mitigar el impacto, mejorar la capacidad de responder y actuar, y, sobre todo, mejorar las estrategias para reducir la vulnerabilidad y condiciones de riesgo".

A los términos especializados que hacen explícitos los conceptos vinculados a los desastres, Valdés [159] los presenta así:

- Un peligro (amenaza) representa la potencial ocurrencia de un suceso, que se manifiesta en un lugar específico, con una intensidad, magnitud y duración determinada. Se suelen clasificar en: de origen natural (hidrometeorológico o geológico), socio-natural (deterioro ambiental, incendios forestales) o provocados por el ser humano.

- La vulnerabilidad es el resultado de la conducta humana, y se puede definir como un sujeto o sistema expuesto a una amenaza, que corresponde a su disposición intrínsica a ser dañado. Aspectos físicos, sociales, económicos, educativos, políticos y culturales, entre otros, contribuyen a la conformación o acumulación de vulnerabilidad. Como ejemplos se pueden citar el grado de conciencia colectiva de los peligros, el estado físico de los asentamientos humanos y su infraestructura, la calidad de las políticas y de la gestión pública, la capacidad de organización en todos los campos de manejo de los desastres.

- El riesgo se define como la probabilidad de daños sociales, ambientales y económicos, en un lugar dado y durante un tiempo de exposición determinado.

En este contexto, diseñar una solución supone enfrentarse no sólo al modelado de un sistema con una complejidad de multiples facetas, relacionadas con la naturaleza del fenomeno y las dinámicas humanas, sino que implica también el desarrollo de herramientas flexibles, adaptables, amigables, computacionalmente eficientes al servicio de información que soporte la solución.

Nuestra hipótesis fundamental de trabajo es que la mejor manera de construir tales herramientas es a partir de una conceptualización de nociones como BD, SIG y agentes adaptable a este dominio de aplicación y servicio.

4.2 DISEÑO GENERAL DEL SISTEMA PARA LA GESTIÓN DE DESASTRES Y REDUCCIÓN DE RIESGO.

A continuación se presenta el diseño general de un sistema de información para servicio público que simulará la ocurrencia de cambios en la dinámica de lluvias sobre una geografía determinada que podrían significar períodos de sequía.

Una versión preliminar de este sistema de simulación para GDRR se presentó en Padilla and Dávila [129]. El sistema le permitirá a un usuario anticipar las secuelas de tal tipo de desastre sobre el servicio de suministro de agua potable a comunidades que son atendidas por un acueducto ubicado en la región de estudio.

El escenario de estudio, basado en un trabajo hecho por Ramírez [136], es el acueducto "La Ceibita" ubicado en el sector Buena Vista, municipio Santos Marquína del Estado Mérida. El acueducto le proporciona servicio de agua potable a los sectores Buena Vista, La Ceibita, La Travesía y La Trinidad; todas pertenecientes a la comunidad "El Murciélago" y con una población de aproximadamente novecientos (900) habitantes.

El sistema del acueducto "La Ceibita" esta catalogado como un acueducto de tipo rural [1] y opera por gravedad. Es abastecido por la "Quebrada La Zarza", una fuente de tipo manantial[2]. El caudal en la fuente "Quebrada La Zarza" es de 25,67 lts/seg medido por medio de un aforo puntual realizado por Ramírez [136] el 7 de mayo de 2005. El caudal de entrada al acueducto es de 5 lts/seg.

La metodología de la *programación extrema Beck [8]* establece el diseño de *historias de usuarios y la construcción de estas historias.* Siguiendo esta metodología se ha construido la siguiente historia de usuario para un sistema de GDRR.

4.2.1 *Historia de Usuario*

Nombre del Problema: El PSAP

Usuario: Habitante de un sector perteneciente al municipio Santos Marquina, Estado Mérida.

A continuación el diálogo entre el usuario y el servicio es el siguiente:

Usuario: Vivo en un pequeño pueblo. A continuación identifico mi ubicación en el mapa.

1 acueducto rural: atiende a menos de 2000 habitantes

2 que proviene de aguas subterraneas por lo que ha sido difícil precisar su cuenca hidrográfica

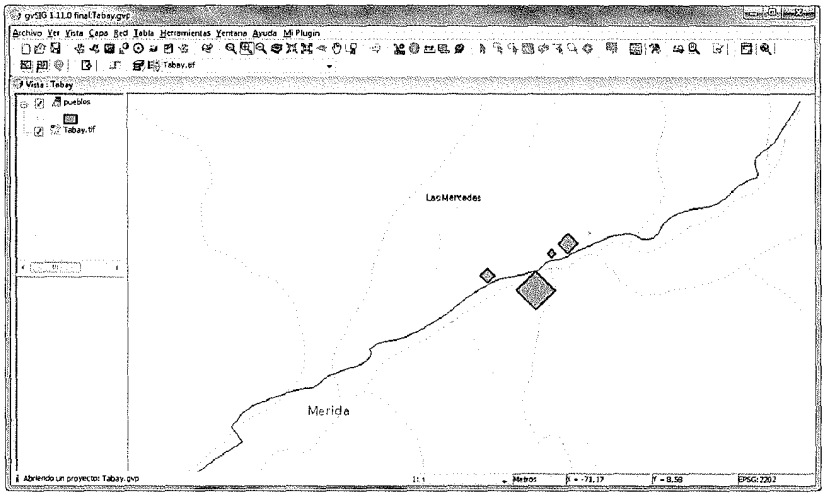


Figura 4.1: Poblaciones del Municipio Santos Marquina, Mérida.

Usuario: El servicio de agua potable en mi sector no ha sido constante, sobre todo en épocas con poca lluvia. Durante el año 2001 la lluvias fueron escasas y el servicio de agua se vio bastante disminuido. La empresa que administra el acueducto ha asegurado a la comunidad que con las ampliaciones realizadas al Acueducto el servicio mejorará. ¿Si ocurriera un periodo de sequía como el ya acontecido, que cantidad de agua llegará a mi casa para un mes determinado?

Servicio: Este sería la cantidad de agua que dispondría en su vivienda

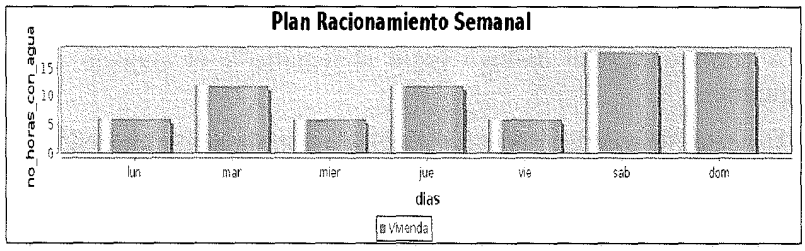


Figura 4.2: Plan de suministro.

4.2.2 *El Modelo de Geosimulación multiAgentes y el Predictor de Servicio de Agua Potable.*

El Modelo Formal de Geosimulación multiAgentes es un soporte para el diseño del sistema para GDRR que representa la historia de usuario descrita. Usaremos el Modelo Formal de Geosimulación multiAgentes para establecer los objetivos de sistema.

El sistema de GDRR puede ser estructurado mediante las funciones $Step_{in}$, y $Step_{out}$, establecidas en el Capítulo 3, y que se representan en el dibujo superior de la figura 4.3

La función $Step_{in}$, Ecuación 3.7, es el *mundo de la simulación* y evoluciona al paso del *tiempo de la simulación*. En este mundo de la simulación se inserta el simulador para el PSAP, que representa la reacción ambiental a las influencias de los agentes, y los agentes que evolucionan en el *tiempo de la simulación*. La función $Step_{in}$, representado en el dibujo inferior izquierdo de la figura 4.3, puede establecerse en los objetivos 1 y 2 que se explican a continuación.

En el objetivo 1 se construye el modelo de simulación para el PSAP, y se integra a la interfaz de usuario en el SIG. El objetivo 1 se enuncia a continuación:

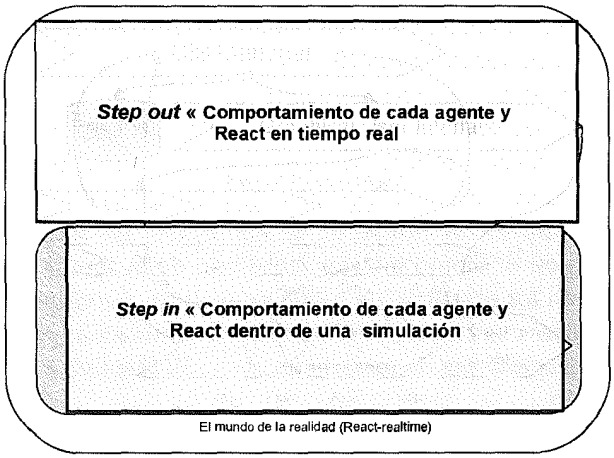
- 1. *Desarrollar el modelo de simulación para el* PSAP.

En el objetivo 2 se implementa el comportamiento de un agente en tiempo de simulación. .

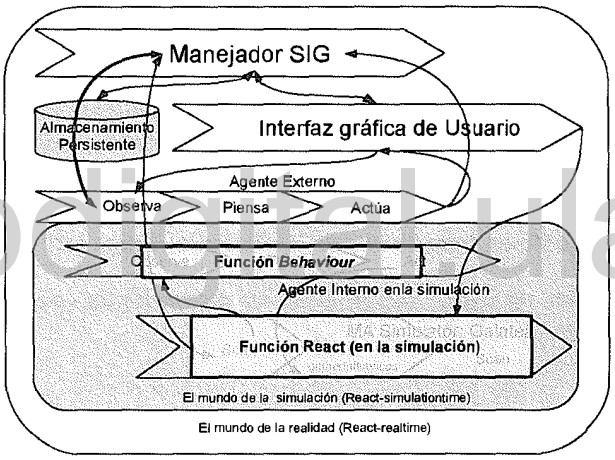
- 2. *Diseñar un agente que en la simulación sea el encargado de gestionar la salida del acueducto.*

La función $Step_{out}$ es el *mundo de la realidad,* y evoluciona al paso del *tiempo real*. En el *mundo de la realidad* están los agentes que observan la simulación y actúan sobre ella por medio de la interfaz gráfica, ver dibujo inferior derecho de la figura 4.3 El objetivo 3 se dedica al diseño del agente en tiempo que se ha llamado *tiempo real:*

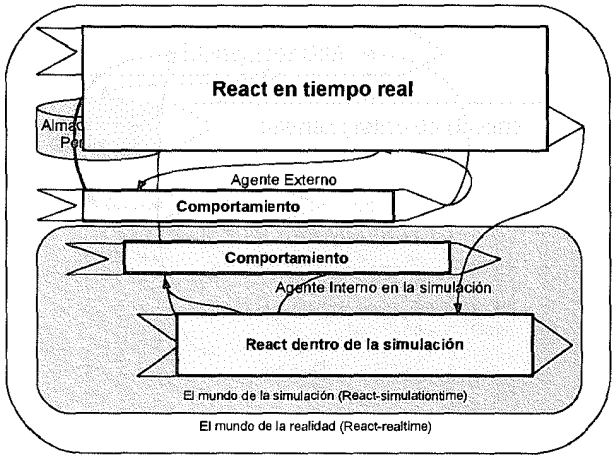
- 3. *Diseñar un agente que gestione los escenarios de simulación del acueducto.*



(a) Funciones Step In y Step Out.



(b) Comportamiento Agente y función React en tiempo de simulación.



(c) Comportamiento Agente y función React en tiempo real

Figura 4.3: Diseño del sistema para GDRR

Conclusión.

En esta sección se presentó el diseño general de un sistema de información para servicio público que simulará la ocurrencia de cambios en la dinámica de lluvias sobre una geografía determinada que podrían significar períodos de sequía en el dominio de la GDRR. La metodología que se utilizó para el desarrollo de la aplicación fue la *programación extrema* y en base a esta decisión se construyó la *historia de usuario* para el PSAP.

El Modelo Formal de Geosimulación multiAgentes se ha utilizado como guía para analizar y establecer la estructuración del sistema para GDRR. A partir del análisis de los componentes del sistema se han establecido los objetivos que en las siguientes secciones se desarrollan.

4.3 DESARROLLAR EL MODELO DE SIMULACIÓN PARA EL PREDICTOR DE SERVICIO DE AGUA POTABLE.

En esta etapa además de la construcción del modelo de simulación para el PSAP, se utiliza las facilidades que proporciona un SIG con el manejo de los mapas para construir una interfaz gráfica usando capas raster y vectoriales relacionadas con la zona de estudio.

En el desarrollo de la interfaz gráfica se consideraron tres aspectos:

1. Implementar el modelo de simulación del acueducto, que describe la dinámica del acueducto "La Ceibita".
2. Integración del PSAP en un SIG. Este paso lo detallaremos a continuación en la sección 4.3.2.
3. Evaluar la construcción del PSAP en un servicio con SIG. Este paso lo detallaremos a continuación en la sección 4.3.3.

4.3.1 *Implementar el modelo de simulación del acueducto.*

Para representar el modelo de simulación de la dinámica del acueducto La Ceibita se usó el sistema de simulación hidráulico EpaNet, Rossman [143].

El software EpaNet Rossman [143] lo definen como "un programa de ordenador que realiza simulaciones en periodos extendidos del comportamiento hidráulico y de la calidad del agua en redes de distribución a presión. En general, una red consta de tuberias, nudos (conexiones entre tuberías), bombas, válvulas y tanques de almacenamiento o depósitos. EpaNet determina el caudal que circula por cada una de los ductos, la presión en cada uno de los nudos, el nivel de agua en cada tanque y la concentración de diferentes

componentes químicos a través de la red durante un determinado período de simulación analizado en diferentes intervalos de tiempo."

En Rossman [143], se propone una metodología para modelar un sistema de distribución de aguas usando el software de simulación EpaNet, que consta de los siguientes pasos:

1. Dibujar un representación de la red del sistema de distribución o importar una descripción básica de la red en un archivo de texto.
2. Editar las propiedades de los objetos que conforman el sistema.
3. Describir como trabaja el sistema.
4. Determinar las opciones de análisis.
5. Iniciar un análisis hidráulico.
6. Obtener los resultados del análisis.

Se seguirán estos pasos para construir el sistema de simulación para el Acueducto "La Ceibita", según los datos aportados por Ramírez [136].

4.3.1.1 *Dibujar un representación de la red del sistema de distribución o importar una descripción básica de la red en un archivo de texto.*

Con EpaNet versión 2.0 se construyó la representación gráfica del Acueducto "La Ceibita". Esta representación se muestra en la figura 4.4.

El Acueduto "La Ceibita" esta conformado por un estanque que es el depósito fuente alimentado por la "Quebrada La Zarza". En la cota 2.055 Metros Sobre el Nivel del Mar (MSNM) hay un desarenador con una demanda de 5 lts/seg que esta representado por una conexión o nodo. Luego, estan ubicados los tanques de almacenamiento (etiquetados como T1, hasta T6) y por último, el estanque de almacenamiento. Los tanques de almacenamiento están interconectados por medio de tuberías. La red de distribución para las poblaciones que se surten del acueducto "La Ceibita" no está detallada en la figura 4.4. Usaremos este modelo en la sección 4.4, donde un agente administrará la salida del estanque de almacenamiento, determinando como va a ser el suministro a las zonas pobladas.

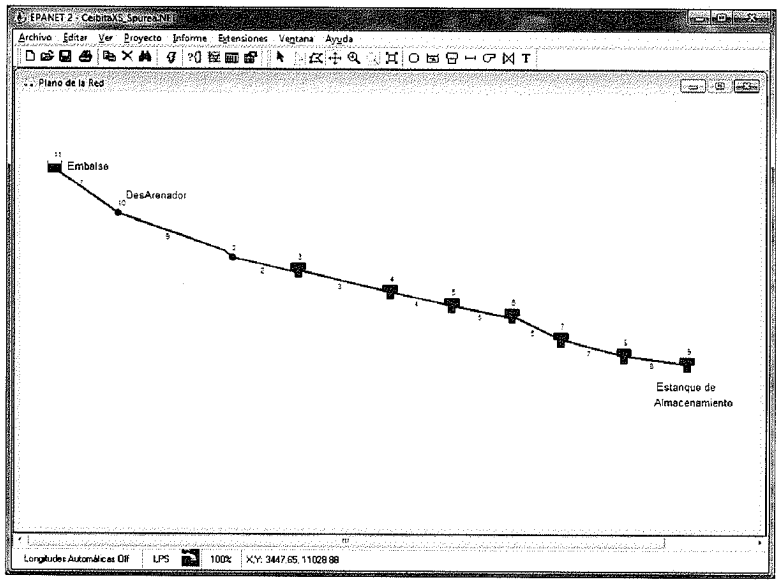


Figura 4.4: Acueducto La Ceibita.

No obstante, con el objeto de obtener un análisis más completo del comportamiento del acueducto, se agregó una red de distribución con depósitos de almacenamientos. Esta extensión, que llamaremos Acueduto "La Ceibita" extendido, se muestra en la figura 4.5. Esta modificación en el modelo de simulación proporciona información acerca de cual es el comportamiento del acueducto con respecto a la distribución de agua hacia los sectores poblados.

El modelo de simulación del Acueducto "La Ceibita" extendido figura 4.5 se usará en la siguiente sección 4.3.

El acueducto "La Ceibita" le proporciona servicio de agua potable a los sectores Buena Vista, La Ceibita, La Travesía y La Trinidad; pero los detalles de la red de distribución para estos urbanismos no está incluída por simplicidad.

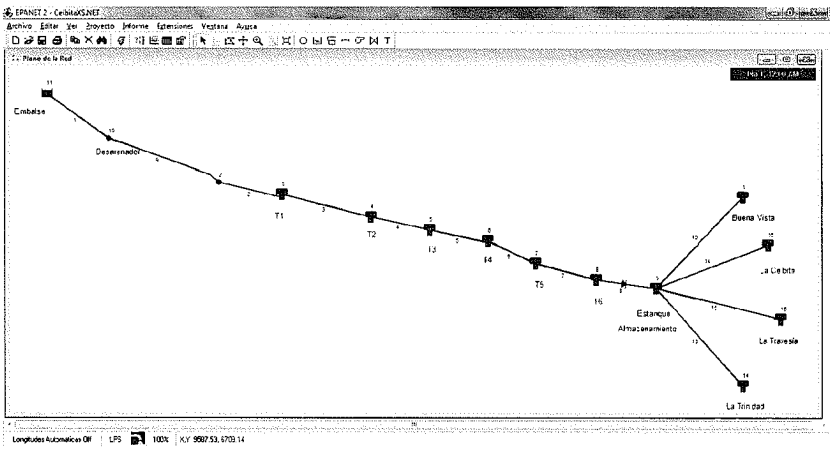


Figura 4.5: Acueducto La Ceibita extendido.

4.3.1.2 *Editar las propiedades de los objetos que conforman el sistema.*

Las propiedades de los objetos que conforman el sistema se especifican en las siguientes tablas. Los objetos marcados con un asterísco (*) representan los objetos que se agregaron a la representación de la red del sistema de simulación, el Acueducto "La Ceibita" extendido , y que se mostró en la figura 4.5.

En el cuadro 4.1 se detallan las características de los nodos y embalse: nodo 2 que es un nodo de conexión, nodo 10 que representa el desarenador y el objeto 11 es el embalse.

Objeto	Altura	Demanda(lts/seg)
2	2030	0
10	2055	-5
11	2061.48	

Tabla 4.1: Características de los Nodos y Embalse del Acueducto La Ceibita.

En el cuadro 4.2 están las caraterísticas de las tuberías que conforman el acueducto, con la longitud en metros, el diámetro y la rugosidad, ambos en milímetros.

Conducción	Longitud (m)	Diámetro (mm)	Rugosidad (mm)
2	8.9	75	0.1
3	75	75	0.1
4	800	75	0.1
5	2.7	75	0.1
6	5	75	0.1
7	1	75	0.1
8	1	110	0.1
9	1175	100	0.1
1	25	75	0.1
10*	1000	75	0.1
14*	1000	75	0.1
15*	1000	75	0.1
13*	1000	75	0.1

Cuadro 4.2: Características de las Conducciones del Acueducto La Ceibita.

En el cuadro 4.3 se describe las características de la red de depósitos que conforman el acueducto. Los depósitos se describen por un identificador llamado Depósito, su descripción, la altura en MSNM, niveles inicial , máximo y mínimo en metros, y diámetro del tanque en metros.

Las características de los tanques agregados, que se resaltan con un asterísco, se calcularon de acuerdo a la demanda, en litros/seg, requerida por las poblaciones involucradas y que fueron establecidos por Ramírez [136]. Estos requerimientos se muestran en el cuadro 4.4

Depósito	Descripción	Altura (MSNM)	Nivel Inicial	Nivel Mínimo	Nivel Máximo	Diámetro (m)
3	T1	2028	0.4	0.4	0.8	2.85
4	T2	2020	0.5	0.5	1.8	1.58
5	T3	2010	0.1	0.1	0.5	1.13
6	T4	2009	0.5	0.5	1	1.45
7	T5	2008.5	0.1	0.1	0.6	2.25
8	T6	2008	0.2	0.2	1	1.5
9	Estanque Almacenamiento	2007.95	2	1	2.7	7
1*	BuenaVista	2005	1.75	0.5	2	4
15*	La Ceibita	2005	2.2	0.5	2.5	6
16*	La Travesía	2005	0.6	0.2	1	2
14*	La Trinidad	2005	0.88	0.5	1	3

Cuadro 4.3: Características de los Depósitos del Acueducto La Ceibita.

	año 2010		2035	
	no. hab.	Demanda (lts/seg)	no. hab.	Demanda (lts/seg)
Buena Vista	210	1,68	299	2,37
La Ceibita	600	4,79	853	6,77
La Travesía	30	0,24	43	0,34
La Trinidad	60	0,47	85	0,68

Cuadro 4.4: Demanda de la comunidad "El Murciélago". Tomado de Ramírez [136].

4.3.1.3 *Describir como trabaja el sistema.*

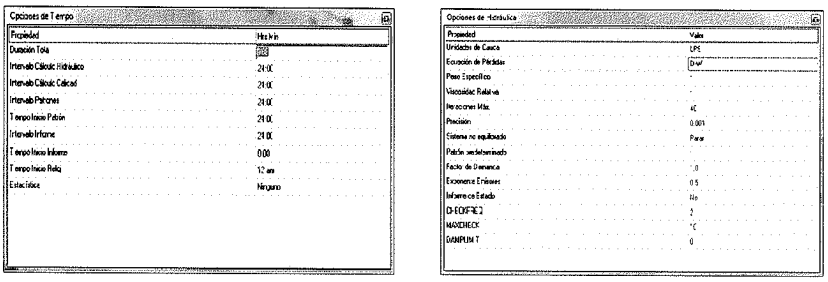
En esta sección se describe como trabajan los objetos no visibles: curvas, patrones de tiempo, controles, editor de demandas y editor de fuentes de calidad. En este caso no se establecieron estos tipos de objetos, debido a que el modelo de simulación para este acueducto no los usa. Por ejemplo, las curvas se usan para describir el comportamiento de un motor; el sistema de simulación del acueducto no tiene motores. Los controles que se ejercen sobre el Acueducto "La Ceibita" es a través de operadores.

4.3.1.4 *Determinar las opciones de análisis.*

Existen cinco categorías de opciones que controlan como EpaNet analiza un sistema: hidráulicas, de calidad, de reacción, de tiempo, y de energía.

En la figura 4.6 se detallan las opciones de tiempo e hidráulicas que se establecieron para el sistema de simulación del Acueducto la Ceibita.

La duración total de la simulación es de una semana, 168 horas, y los intervalos de cálculo es de 24 horas. En las opciones hidráulicas se seleccionó la ecuación Darcy-Weisbach (D-W), que es la ecuación con la cual se calculan las pérdidas en las tuberías.



(a) Opciones de Tiempo (b) Opciones Hidráulicas

Figura 4.6: Opciones de Análisis para sistema simulación Acueducto La Ceibita

4.3.1.5 *Iniciar un análisis hidráulico.*

Con los objetos del sistema descritos y las opciones de análisis establecidas se da inicio al análisis hidráulico, el cual se ejecuta satisfactoriamente. Los resultados se observan en la siguiente sección.

4.3.1.6 *Obtener los resultados del análisis.*

EpaNet proporciona diversos resultados dependiendo del tipo de análisis que se necesite. En este caso en particular, se usa la demanda promedio en los nodos, que es un valor que indica el consumo promedio en la conexión medido en litros de agua por unidad de tiempo. Los resultados de la simulación para el Acueducto "La Ceibita" extendido (figura 4.5) se muestra en la figura 4.7.

En la gráfica muestra la demanda de los nodos: Buena Vista (Nodo 1), La Ceibita (Nodo 15), La Travesía (Nodo 16) y La Trinidad (Nodo 14). Para el depósito de Buena Vista (Nodo 1) se observa que el caudal neto entrante es de cero (0), esto significa que el flujo entrante y el flujo de salida son iguales. Para el resto de los nodos se observa un caudal neto entrante positivo indicando que el flujo entrante es mayor que el flujo de salida.

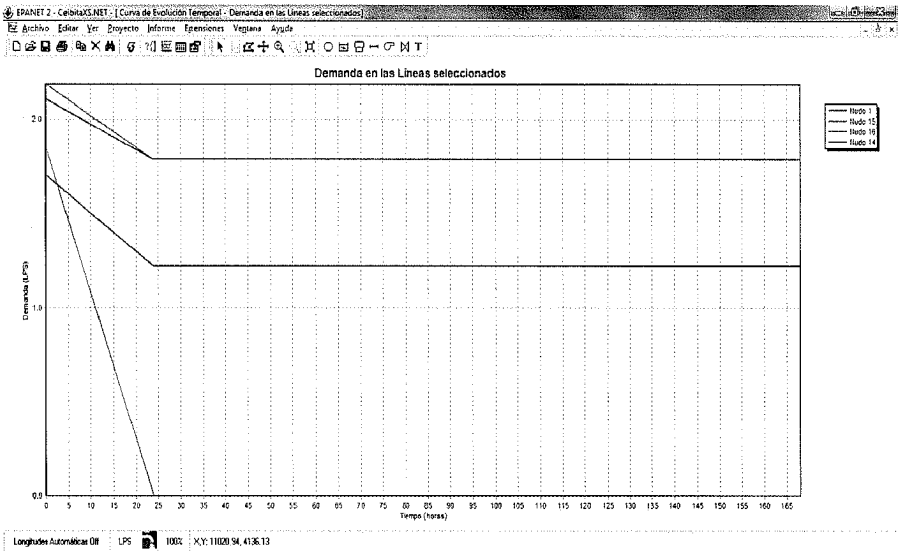


Figura 4.7: Resultados del simulador EpaNet

Conclusión

En esta parte se ha construído el modelo de simulación para el Acueducto "La Ceibita", usando la metodología que proporciona Rossman [143] para modelar un sistema de distribución de aguas usando el software de simulación EpaNet.

EpaNet también genera, en un archivo de texto, el modelo de simulación del Acueducto "La Ceibita" que se ha presentado. El archivo de texto con el modelo de simulación del Acueducto "La Ceibita", llamado *Ceibita.inp* y que representa la red de la figura 4.4, puede verse en la Sección C.1. El archivo de texto con el modelo de simulación extendido que incluye los nodos, llamado *Ceibita_Nodo.inp* y que representa la red de la figura 4.5, puede verse en la Sección C.2.

4.3.2 *Integración del PSAP en un SIG .*

Luego de una evaluación de software para SIG, se decidió usar gvSIG versión 1.11, gvSIG [77], para desarrollar la interfaz gráfica al usuario que muestre el gráfico de salida de agua esperable para una vivienda. Esta evaluación de software para SIG puede consultarse en la Sección B.1.

Las razones que nos ha inclinado por el uso de gvSIG son:

- Atractiva interfaz gráfica,
- La posibilidad de desarrollo de aplicaciones para equipos móviles.
- Construido en JAVA, un lenguaje multi plataforma.

- Grupos de usuarios comprometidos con el desarrollo y mejora del software.

El entorno de desarrollo usado fue el Integrated Development Environment (IDE) Eclipse Juno, ECLIPSE [53]. Para configurar el IDE Eclipse Juno con gvSIG versión 1.11, ir a Sección D.1, donde se detalla el proceso de configuración.

Para el desarrollo de una extensión del PSAP en gvSIG 1.11, se siguieron los siguientes pasos:

1. Diseñar diálogo del usuario con la interfaz gráfica.
2. Cargar capas raster que identifiquen la zona de estudio: La comunidad "El Murciélago" del municipio Santos Marquina del Estado Mérida.
3. Construir una capa vectorial con polígonos que identifican las poblaciones de Buena Vista, La Ceibita, La Travesía y La Trinidad.
4. Construir una extensión en gvSIG que consulta la información relacionada con las poblaciones.
5. Construir una extensión en gvSIG que consulta la demanda promedio estimada de agua potable para cada población que atiende el Acueducto "La Ceibita".

A continuación, se detalla el desarrollo de estos pasos.

4.3.2.1 *Diseñar diálogo del usuario con la interfaz gráfica.*

A partir de la historia de usuario definida en 4.2.1, se han establecido dos tipos de interacciones del usuario con el SIG: *Consultar datos de las poblaciones* y *Consultar la demanda promedio estimada* para una población específica.

La consulta de datos de las poblaciones arroja la información contenida en la BD, que fue referida en el cuadro 4.4.

La consulta de la demanda promedio estimada para una población específica informa cuanto es el estimado de agua potable que el acueducto puede entregar a esa población.

4.3.2.2 *Cargar capas raster que identifiquen la zona de estudio.*

Antes de cargar las capas raster, se crea un proyecto que llamaremos TABAY. En propiedades del proyecto, las pantallas que se muestran en la figura 4.8, se coloca la proyección geodésica que identifica la zona de estudio. La proyección geodésica, de tipo European Petroleum Survey Group (EPSG), EPSG [57], que identifica la zona oeste venezolana es EPSG=2202 [3].

3 Importante: una vez abierta la ventana del proyecto de gvSIG, en la zona inferior derecha de la figura debe estar indicado la proyección que se seleccionó.

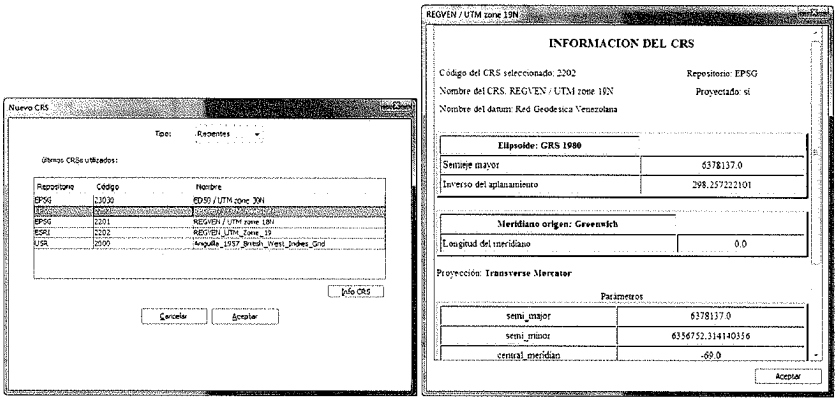


Figura 4.8: Edición de las propiedades del proyecto en gvSIG

Las capas raster relacionadas con la zona de estudio son proporcionadas por el servicio WMS que brinda el IGVSB, IGVSB [84], en la siguiente dirección `http://igvsb.geoportalsb.gob.ve/cgi-bin/mapserv`.

Añadir una capa y seleccionar la ventana WMS. Registrar las dirección del WMS del IGVSB, realizar la conexión y seleccionar los mapas que se necesiten.

Finalmente la capa raster de la zona se muestra en la figura 4.9.

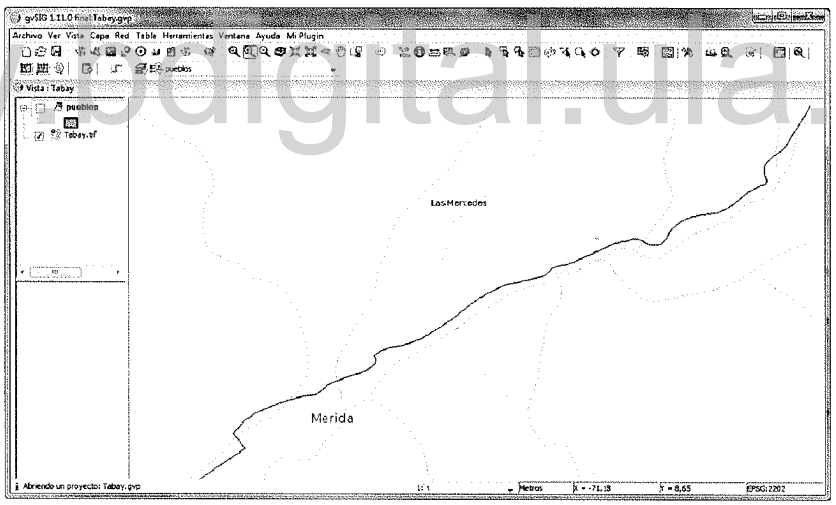


Figura 4.9: Capa Raster proporcionada por el servicio WMS del IGVSB.

4.3.2.3 *Construir una capa vectorial con polígonos que identifiquen a las poblaciones de Buena Vista, La Ceibita, La Travesía y La Trinidad.*

Para la construcción de la capa de polígonos se usa las facilidades que ofrece gvSIG para la manipulación de objetos vectoriales.

Usando como guía la capa raster obtenida por el servicio WMS del IGVSB, figura 4.9, se dibujaron unos polígonos para representar los sectores de población que son parte del área de estudio.

A cada polígono se le asoció el conjunto de datos que se presentaron en el cuadro 4.4. Estos datos se almacenaron en una BD PostGreSQL 8.3 PostGreSQL [134] con extensión PostGIS 1.5 PostGIS [133], que son las versiones que soporta gvSIG. En la figura 4.10 se muestra la capa vectorial que se construyó.

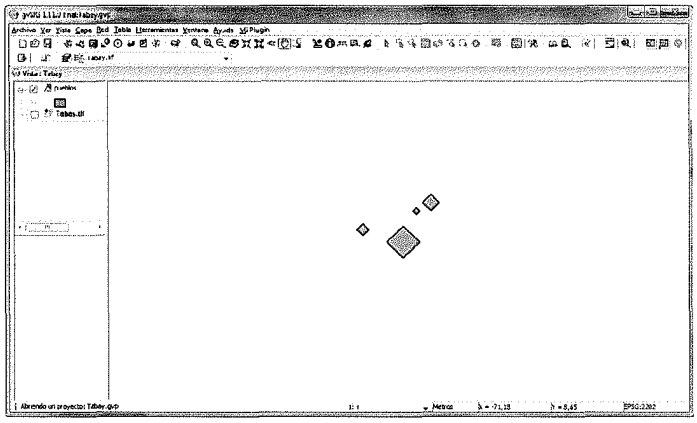


Figura 4.10: Capa vectorial con los sectores poblacionales

Finalmente la interfaz grafica de usuario esta compuesta por la superposición de las capas de las figura 4.9 y figura 4.10, y se muestra en la figura 4.1.

4.3.2.4 *Construir una extensión en gvSIG que consulte los datos de una población.*

Las extensiones para gvSIG siguen normas en cuanto a nombres y a estructura. Los paquetes de las extensiones deben comenzar con el prefijo *com.iver.cit.gvsig* y la estructura de la extensión debe tener esta forma, figura 4.11:

```
ExtMyPlugin
   +src
   -config
        config.xml
    build.xml
    text_en.properties
    text.properties
```

Figura 4.11: Estructura de una extensión en gvSIG

Se crea la extensión ExtMyPlugin con la estructura indicada arriba. En la carpeta *src* se crea la clase principal *PoblacionInfoExtension.java*

que hereda la clase *Extension*. Se utiliza el componente *MapControl* que permite la manipulación de los mapas que se cargan en la interfaz de gvSIG. *MapControl* tiene asociado un *Listener* que advierte si se seleccionó alguno de los polígonos del mapa para ejecutar, como consecuencia de la elección del polígono, los eventos. El algoritmo *PoblacionInfoExtension.java* se muestra en el Programa 4.1

```
...
 private InfoByPointListener listener = null;
        public void execute(String actionCommand) {
                View view = (View)PluginServices.getMDIManager().
                    getActiveWindow();
                MapControl mc = view.getMapControl();
                if (listener == null){
                            listener =  new InfoByPointListener(
                                mc);
                            mc.addMapTool("poblacionInfo",new
                                PointBehavior(listener));
                }
                        mc.setTool("poblacionInfo");
...
```

Programa 4.1: Clase PoblacionInfoExtension.java

El algoritmo *InfoByPointListener.java, Programa 4.2,* tiene el acceso a la BD donde se recupera la información relacionada con los polígonos y la despliega en un panel que proporciona gvSIG.


```
...
public void point(PointEvent event) throws BehaviorException {
        Point2D pReal = event.getPoint();
        Point2D mapPoint =      mapControl.getViewPort().
            toMapPoint((int)pReal.getX(),(int)pReal.getY());
        System.out.println("El punto seleccionado es: " +
            mapPoint.getX() + " " +mapPoint.getY()
        double tol = mapControl.getViewPort().toMapDistance(1);
        com.iver.cit.gvsig.fmap.layers.FLayers lyrs =
            mapControl.getMapContext().getLayers();
        com.iver.cit.gvsig.fmap.layers.FLyrVect lyrPueblos = (com
            .iver.cit.gvsig.fmap.layers.FLyrVect)lyrs.getLayer("
            pueblos");
        FBitSet selection;
        try {
                selection = lyrPueblos.queryByPoint(mapPoint,tol)
                    ;
                if (!selection.isEmpty()){
                        DataSource ds = ((AlphanumericData)
                            lyrPueblos).getRecordset();
                        ds.start();
                        idField = ds.getFieldIndexByName("id");
                        ...
```

```
        ds.stop();
        if (strNombre != null){
                PoblacionPanel panel = new PoblacionPanel(
                    strNombre.toString()+" = "+strPoblacion.
                    toString()+ " habitantes");
                PluginServices.getMDIManager().addWindow(panel);
...
```

Programa 4.2: Clase InfobyPointListener.java

El archivo *config.xml* permite asociar una opción de menú a la extensión. El menú para consulta de datos se configura como se muestra en la figura 4.12. La opción *active=true,* permite que la extensión se muestre en el menú. La extensión, que se llamó *Mi Plugin,* estará ubicada en el panel de opciones superior del menú principal de gvSIG.

```
...
<extension class-name="com.iver.cit.gvsig.myplugin.
    PoblacionInfoExtension"
    description="Consultar cantidad de habitantes."
    active="true"
    priority="2">
   <menu text="Mi Plugin/consulta Población" action-command="
       PoblacionInfo" />
</extension>
...
```


Figura 4.12: config.xml

El archivo *build.xml* compila los fuentes y mueve el archivo compilado y sus dependencias al repositorio de extensiones de gvSIG.

Finalmente, la consulta de los datos de la población arroja la pantalla que se muestra en la figura 4.13.

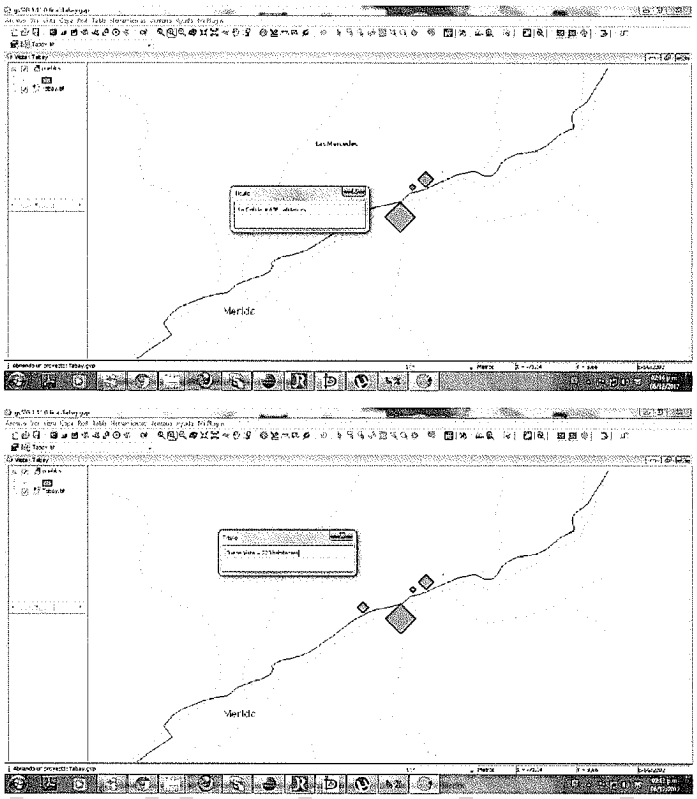


Figura 4.13: Consulta los datos de una población.

4.3.2.5 *Construir una extensión en gvSIG que consulte la demanda promedio estimada para una población.*

La extensión de gvSIG para la consulta de la demanda promedio tiene la misma estructura que la explicada en 4.3.2.4. Sólo que el archivo asociado al *Listener* ejecuta las acciones adecuadas a esta consulta.

Para el cálculo de la demanda promedio se usa una versión del software EpaNet escrito en el lenguaje Java, Baseform-Epanet-Java Library [6]. Esta versión se ha incorporado al programa *InfoByDemandaListener;* parte de este archivo se muestra en *Programa* 4.3

```
...
public void point(PointEvent event) throws BehaviorException {
// Lee el archivo con  modelo de simulación
   try  {
    inFile = new File("C:/Users/Virginia/planos/CeibitaXS.inp");
...
// corre la simulación
   hydFile = File.createTempFile("hydSim", "bin");
   consoleLog("START_RUNNING");
   HydraulicSim hydSim = new HydraulicSim(net, log);
   hydSim.simulate(hydFile);
```

```
...
// Lee de la BD los datos de la población seleccionada
selection = lyrPueblos.queryByPoint(mapPoint,tol);
if (!selection.isEmpty()){
   DataSource ds = ((AlphanumericData)lyrPueblos).getRecordset();
   ds.start();
   idField = ds.getFieldIndexByName("id");
...
// Se obtiene la demanda para la población seleccionada
...
HydraulicReader hydReader2 = new HydraulicReader(new
    RandomAccessFile(hydFile, "r"));
...
for (long time = pMap.getRstart(); time <= pMap.getDuration();
    time += pMap.getRstep()) {
    AwareStep step = hydReader2.getStep((int) time);
    if (targetTimes.size() > 0 && !targetTimes.contains(time))
        continue;
    for (Node node : net.getNodes()) {
        if (targetNodes.size() > 0 && !targetNodes.contains(node.
            getId())) continue;
        for (NodeVariableType nodeVar : nodesVariables) {
        if (strNombre.toString().equals(node.getComment())   &
            nodeVar.name().equals("DEMAND")){
                val= nodeVar.getValue(net.getFieldsMap(), step,
                    node, i);
        if (time > 0 & (time/pMap.getRstep() - k == 0.0)){
                dataset.setValue(val, "Suministro de Agua",
                    names[k]);
...
```

Programa 4.3: Clase InfoByDemandaListener.java

las acciones que ejecuta *InfoByDemandaListener* pueden resumirse como:

- leer el archivo *Ceibita_Nodo.inp* (Sección C.2), que describe el modelo de simulación del Acueducto "La Ceibita" extendido,
- ejecutar la simulación,
- recuperar de la BD la información asociada al polígono seleccionado,
- obtener el consumo de agua potable para la población seleccionada.

La consulta de la demanda promedio para la población "La Ceibita" se muestra en la figura 4.14.

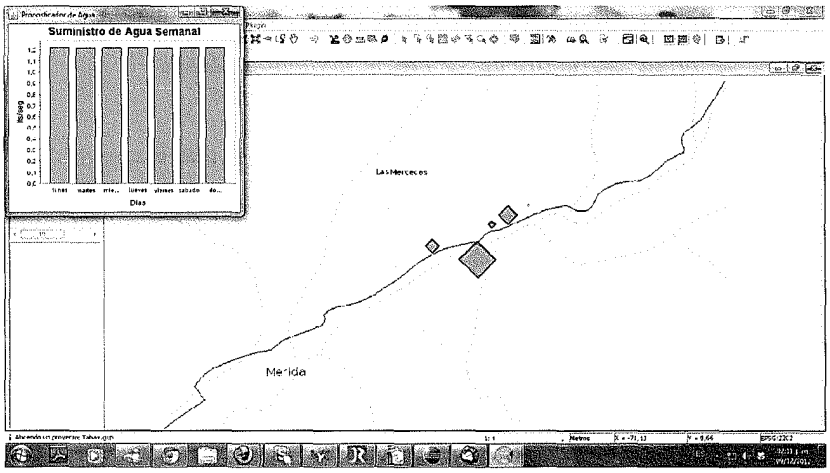


Figura 4.14: Consulta Demanda Promedio

Conclusión.

Se ha construido una interfaz gráfica en un SIG, que permite al usuario consultar cuanto es la demanda promedio semanal de agua potable medida en lts/seg, que el acueducto proporciona a una determinada población. La extensión PSAP para gvSIG 1.11 puede descargarse desde `https://simulants.svn.sourceforge.net/svnroot/simulants/` y las instrucciones para su instalación pueden verse en Sección D.3.

4.3.3 *Evaluar la construcción de un servicio con el SIG y el PSAP.*

Hay dos maneras de desarrollar un servicio con gvSIG:

1. Publicar la interfaz gráfica desarrollada en la sección 4.3.2 que se muestra en la figura 4.1 en un servidor de mapas. Se ha evaluado esta opción instalando la capa de polígono, 4.10, en un servidor de mapas GeoServer GeoServer [69].

   Con GeoServer es posible configurar un servicio Open Geospatial Consortium (OGC), por lo que el acceso al servicio WMS del IGVSB puede establecerse. De esta manera la interfaz gráfica de la figura 4.1 puede ser replicada en el servicio de mapas de GeoServer.

   La integración del PSAP para acceder al servicio de mapas de GeoServer requiere el uso de otras tecnologías como Representational State Transfer (REST) Pautasso et al. [131], JavaScript Flanagan [65] y ECQL, este último es una versión extendida proporcionada por GeoServer del estándard CQL CQL [30] para la búsqueda de información en colecciones de datos.

Sin embargo, para la implementación de una plataforma distribuida con el objeto de construir un servicio en la Web con PSAP, debe extenderse EpaNet para el intercambio de datos con el formato XML.

2. Desarrollar un servicio usando la extensión que ofrece gvSIG para desarrollo de cliente móviles.

   Para la integración del PSAP en un servicio con un gvSIG se ha optado por explorar la posibilidad de integrar el PSAP con el servicio móvil que ofrece gvSIG por medio de gvSIG Mobile versión 0.3.

   La interfaz gráfica se construye de la siguiente manera. La capa raster se carga al dispositivo usando el servicio WMS del IGVSB. La capa vectorial esta guardada en la memoria del dispositivo.

   El programa *DemandaPointListener.java*, cuyo extracto se muestra en Programa 4.4, realiza las acciones que se describen en el párrafo 4.3.2.5.

```
public class DemandaPointListener  implements PointToolListener {

public void point(PointEvent event) {
...

// Lee los datos del  polígono seleccionado
     fea = infoByPoint(item_layer, event.getPoint(),
        tol, null, mapCtrl.project.getProjection().getAbrev());
....

//Lee archivo con el modelo de simulación
  inFile = new File("C:/Users/Virginia/planos/CeibitaXS.inp");

...
// ejecuta la simulación
   hydFile = File.createTempFile("hydSim", "bin");
   consoleLog("START_RUNNING");
   HydraulicSim hydSim = new HydraulicSim(net, log);
      hydSim.simulate(hydFile);
...

// calcula la demanda para el polígono seleccionado
...

       for (NodeVariableType nodeVar : nodesVariables) {
         if (String.valueOf(fea.getValues()[0]).equals("1.0")  &i
              == 10& j ==4 ){
               val = val + nodeVar.getValue(net.getFieldsMap(),
                   step, node, i);
                if (time > 0 & (time/pMap.getRstep() - k == 0.0))
                     {
```

```
                dataset.setValue(val, "Suministro de Agua",
                    names[k]);

...
```

Programa 4.4: Clase DemandaPointListener.java

La versión de gvSIG Mobile con la extensión del PSAP descrita en esta sección puede descargarse desde `https://simulants.svn.sourceforge.net/svnroot/simulants/`. Los pasos para configurar el entorno de desarrollo IDE Eclipse con gvSIG Mobile versión 0.3 y la instalación de la extensión PSAP en este entorno se encuentra en la Sección D.2.

Conclusiones.

Las conclusiones de esta sección tiene dos vertientes:

1. Con respecto al uso de un servidor de mapas para publicar la interfaz gráfica y a partir de esto construir un servicio en la Web tenemos las siguientes consideraciones. Con la tecnología que proporciona GeoServer se puede desarrollarse un servicio en la Web. La única consideración es que se debe extender el simulador EpaNet para compartir datos en el lenguaje XML.
2. Con respecto aL servicio PSAP móvil, éste se ha ejecutado en un emulador. Se ha tenido inconveniente al intentar instalar la extensión PSAP en equipos de tipo Personal Digital Assistant (PDA) que aceptan la aplicación gvSIG Mobile 0.3. La Java Virtual Machine (JVM) que usa gvSIG Mobile en los PDA: la plataforma PHONEME, disponible en `http://davy.preuveneers.be/phoneme/?q=node/33` para equipos móviles con Windows CE y Android, es una implementación de PHONEME open source J2ME de Sun Microsystems, Inc. (ahora Oracle). Para la adaptación del servicio PSAP móvil se hace necesario re-escribir el simulador para que use los métodos que tiene disponible la JVM , y que usa gvSIG Mobile en los PDA.

   Esta limitante también se debe considerar en el proceso de considerar la instalación de algún tipo de PD para integrar otras plataformas de simulación.

4.4 DISEÑAR UN AGENTE GESTIONADOR DE LA SALIDA DEL ACUEDUCTO

El simulador GALATEA se usa para la implementación del agente que gestiona la salida del tanque de almacenamiento o *agente regulador*. La incorporación de otro simulador al sistema requiere que

el proceso de diseño se piense como un proceso de integración de diversas simulaciones computacionales. Es por ello que para integrar un agente que gestione la salida del tanque de almacenamiento con el SIG y la extensión del PSAP se han considerado las soluciones de integración de software referidas en Subsección 2.4.1, y se ha decidido usar un tipo de interacción fuerte, especificamente el PD.

Entre las arquitecturas usadas para la implementación de soluciones de integración de software con PD, y es la que se usa para el desarrollo de este trabajo, destaca la HLA , DoD [48], DMSO [47]. HLA es una arquitectura de propósito general para sistemas de simulación computacionales distribuidos. Con HLA las simulaciones computacionales pueden interactuar con otros simulaciones computacionales sin importar la plataforma computacional. En el anexo Sección A.1 se encuentra una breve introducción a los estándares HLA 1.3, IEEE HLA 1516 , IEEE HLA 1516-Evolved.

Existen varias implementaciones del estándard HLA. Estas implementaciones se han evaluado y se ha decidido el uso de PoRTIco, PoRTIco [132]. La razón principal para escoger PoRTIco es que su implementación no requiere que se ejecute el Run-Time Infrastructure (RTI) antes de la ejecución de los federados; los federados operan de una manera *punto-a-punto*. Esta característica de no necesitar la ejecución previa de un RTI añade una simplificación al proyecto.

Se usará el estándard HLA 1.3 debido a que el intercambio de datos entre los simuladores se realiza a través de un archivo de texto; en el estándard IEEE HLA 1516 y el IEEE HLA 1516-Evolved, se usan archivo XML para el intercambio de datos.

En el anexo Sección A.3 se encuentra la evaluación de los software que implementan el estándard HLA. En HLA, el sistema de simulaciones combinadas se llama federación y los componentes de la federación son los federados.

Entonces, la fase de la codificación del agente de software comprende los siguientes pasos:

- Construcción de la federación,
- Construcción del agente gestionador de la salida del tanque de almacenamiento, e
- Integrar la federación en el SIG.
- Ejecución de la federación con el agente regulador.

A continuación se detallan estos pasos.

4.4.1 *Construcción de la federación.*

Scrudder et al. [149] ha definido un proceso de desarrollo de una federación, Federation Development and Execution Process (FEDEP), que consta de los siguientes pasos:

- Definir los requerimientos de la federación
- Desarrollo del modelo conceptual de la federación
- Diseño y desarrollo de la federación
- Integración y pruebas de la federación
- Ejecución y análisis de resultados de la federación

A continuación y siguiendo los pasos de FEDEP, se muestra el proceso de desarrollo de una federación que se ha llamado *Mi Federación.*

4.4.1.1 *Definir los requerimientos de la federación.*

Los requerimientos de la federación, que se ha llamado *Mi Federación* se ha formulado por medio de un objetivo que se enuncia a continuación: "Crear una federación que tenga como objetivo la generación de un sistema de simulación del Acueducto "La Ceibita" que incluya las acciones que realiza el operador humano en la administración del estanque de almacenamiento para el suministro de agua a los pueblos que conforman la comunidad "El Murciélago"".

La federación *Mi Federación* está constituida por dos federados: *Mi Acueducto* y *Mi Administrador*. El federado *Mi Acueducto* tiene como objetivo la ejecución del proceso de simulación del Acueducto "La Ceibita" y el objetivo del federado *Mi Administrador* es la ejecución del agente administrador quien gestiona el servicio de agua potable para las poblaciones .

4.4.1.2 *Desarrollo del modelo conceptual de la federación.*

Este paso permite crear una representación apropiada del dominio del mundo real aplicada al espacio del problema particular. De esta forma, es necesario identificar las entidades conceptuales y las interacciones entre éstas. El modelo conceptual de la federación usa la especificación llamada marco de modelos de objetos ú OMT, Object Model Template (OMT), para definir la estructura del modelo de objetos de la federación o FOM, Federation Object Model (FOM).

El FOM define el nombre de las cosas y las ocurrencias que los federados comparten. Esta especificación usa un grupo de tablas, llamadas tablas OMT, para recoger y clasificar la información. A continuación se presentan las tablas OMT que se usaron para especificar la federación *Mi Federación*. Es importante acotar que, la especificación OMT proporciona otras estructuras de tablas que no se utilizaron en este desarrollo.

Identificación del modelo de objeto	
Categoría	Identificación
Nombre	Mi_Federación
versión	1.0
fecha	1 de noviembre de 2012
propósito	Modelo de Objeto para Mi_Federación
dominio de la aplicación	Operaciones para el suministro de Agua Potable en el Municipio Santos Marquina, Mérida

Cuadro 4.5: Identificación del modelo de objeto.

TABLA IDENTIFICACIÓN DEL MODELO DE OBJETO. 4.5

TABLA ESTRUCTURA DE CLASE DE OBJETOS. El cuadro con la estructura de clase de objetos contiene dos subclase: Acueducto y Administración, ver 4.6 La subclase Acueducto se ha catalogado como publicable (P), significa que el federado usa el servicio de publicación de objetos (*Publish Object Class*) y puede publicar la clase de objeto especificada.

Estructura de la clase de objetos	
ObjectRoot	
Principal(S)	Acueducto(P)

Cuadro 4.6: Estructura de clase de objetos.

TABLA ESTRUCTURA DE CLASE DE INTERACCIÓN. Una interacción es una acción hecha por un objeto en un federado que puede tener efectos en objetos de otros federados. El cuadro 4.7 con la estructura de la clase interacción contiene una interacción que se ha llamado Acueducto_Administracion_Transaccion(I). La clase Simul(I), catalogada como I para indicar que inicia y envia la interacción especificada.

Estructura Clase de Interacción	
Acueducto_ Administración_ Transacción(I)	Simul(I)

Cuadro 4.7: Estructura de clase de interacción.

TABLA DE ATRIBUTOS. Las clases de objeto que pertenecen al dominio de la simulación se caracterizan por un conjunto fijo de tipos

de atributos. Los valores para estos tipos de atributos para el objeto Acueducto se muestran en el cuadro de atributos, 4.8.

Tabla de Atributos

Objeto	Atributo	Tipo-Dato	Cardi-nalidad	Unidad	Resolu-ción	Precisión	Condición-Precisión	Tipo Actuali-zaciones	Condición Actuali-zaciones	T/A	U/R	espacio enruta-miento
Acueducto	nodo	string	1	N/A	N/A	N/A	N/A	periodico	con la simulación	T/A	U/R	N/A
	dotación	float	1	lts/seg	1	perfecto	siempre	periodico	con la simulación	T/A	U/R	N/A
	presión	float	1	mts	1	perfecto	siempre	periodico	con la simulación	T/A	U/R	N/A

Cuadro 4.8: Tabla de Atributos de la Federación.

TABLA DE PARÁMETROS. En el cuadro 4.9 se especifica el parámetro asociado con la clase interacción definida anteriormente en el cuadro de estructura de clase de interacción.

Tabla de Parámetros									
Interacción	Parámetros	Tipo-Dato	Cardi-nalidad	Unidad	Resolu-ción	Precisión	Condición-Precisión	espacio enrutamiento
Simul(I)	EndSimulation	boolean	1	N/A	N/A	N/A	N/A	N/A

Cuadro 4.9: Tabla de Parámetros de la Federación.

TABLA DE LÉXICO. El cuadro de léxico Simulation Object Model (SOM)/FOM proporciona información semántica necesaria para entender la información contenida en las tablas OMT. El Léxico SOM/FOM construido para la federacion *Mi Federación* está en el cuadro 4.10.

Definición Clase de Objetos	
Término	Definición
nodo	nombre del nodo
dotación	dotación promedio del nodo
presión	presión en el nodo
Definición Clase de Interacción	
Término	Definición
EndSimulation	Inicio/fin de la simulación

Cuadro 4.10: Léxico SOM/FOM.

DATOS PARA LA EJECUCIÓN DE LA FEDERACIÓN. La estructura del FOM de una federación está predeterminada. El FOM está constituido por la declaración de un objeto principal, *ObjectRoot*, del cual dependen todas las clases que se declaren en la federación. La federación *Mi federación* tiene una clase definida en su FOM: Acueducto. El diagrama de clases de la federación se observa en la figura 4.15.

La clase Acueducto tiene el atributo nodo que identifica un nodo específico del acueducto, en este caso el Estanque de Almacenamiento. Los atributos dotación y presión indican la dotación promedio y la presión, en el Estanque de Almacenamiento calculado ambos valores por la simulación.

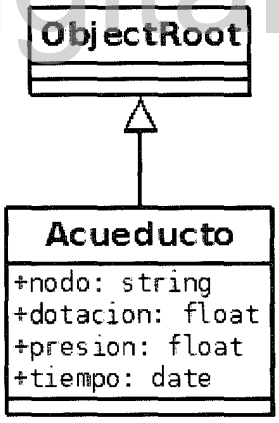


Figura 4.15: Clase de Objetos de Mi_Federación

La federación *Mi federación* tiene una clase de interacción para indicar que la simulación ha terminado. El diagrama de interacción de la federación se observa en la figura 4.16. Todas las interacciones de una federación son sub-clases de *InteractionRoot*.

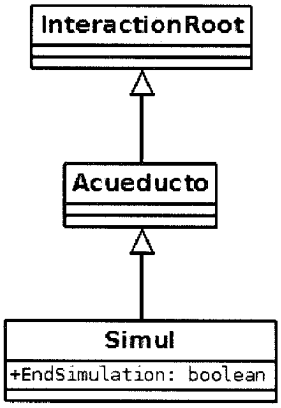


Figura 4.16: Clase Interacción de Mi_Federación

En el estándard HLA 1.3 los datos que se necesitan para la ejecución de la federación se definen en un archivo Federation Execution Data (FED). El archivos de tipo FED se construye con la información especificada en las tablas OMT.

El archivos de tipo FED construido para la federación *Mi Federación* se muestra en la figura 4.5.

```
;; A comment in the test file, just to show I'm cool ;;
(FED
  (Federation Portico-Test)
  (FEDversion v1.3)
  (spaces
    (space TestSpace
      (dimension TestDimension)
    )
    (space OtherSpace
      (dimension OtherDimension)
    )
  )
  (objects
    (class ObjectRoot
      (attribute privilegeToDelete reliable timestamp)
      (class RTIprivate)
      (class Principal
          (class Acueducto
        (attribute nodo reliable timestamp TestSpace)
        (attribute dotacion reliable timestamp TestSpace)
        (attribute presion reliable timestamp TestSpace)
        (attribute tiempo reliable timestamp TestSpace)
        ) ;;edd-class Acueducto
        )
    )
  )
  (interactions
    (class InteractionRoot reliable timestamp
      (class RTIprivate reliable timestamp)
```

```
      (class Simul reliable timestamp TestSpace
        (parameter EndSimulation)
       )
       (class X reliable timestamp TestSpace
        (parameter xa)
       )
    )
  )
)
```

Programa 4.5: Archivo suministro.fed

4.4.1.3 *Diseño y desarrollo de la federación.*

Mi Federación tiene dos federados: *Mi Acueducto* y *Mi Administrador*. En la figura 4.17 se muestra un esquema de *Mi Federación* con sus federados.

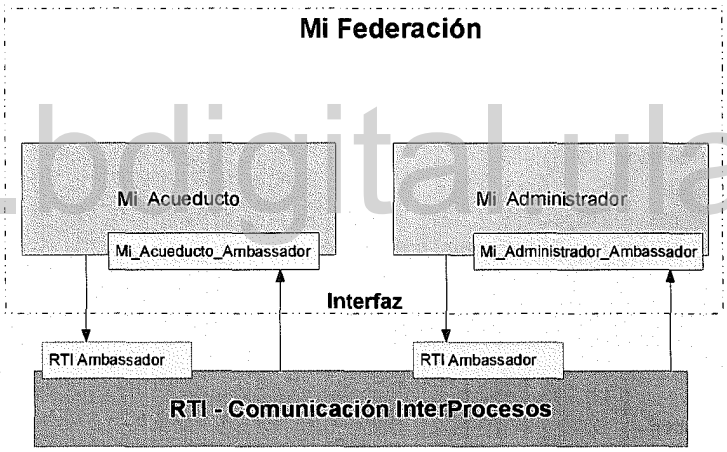
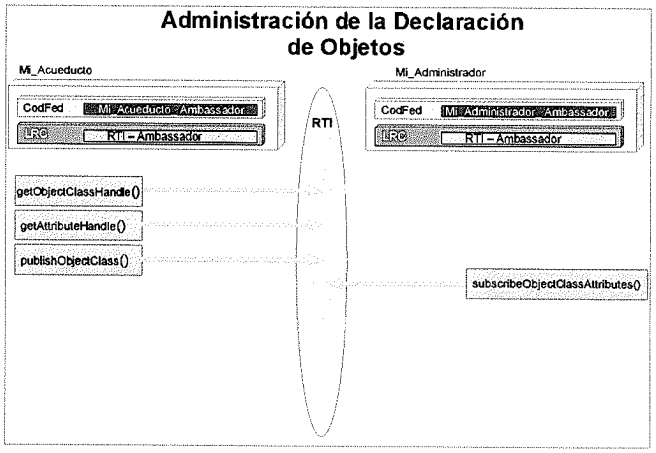


Figura 4.17: Mi_Federación. Tomado de Kuhl et al. [105]

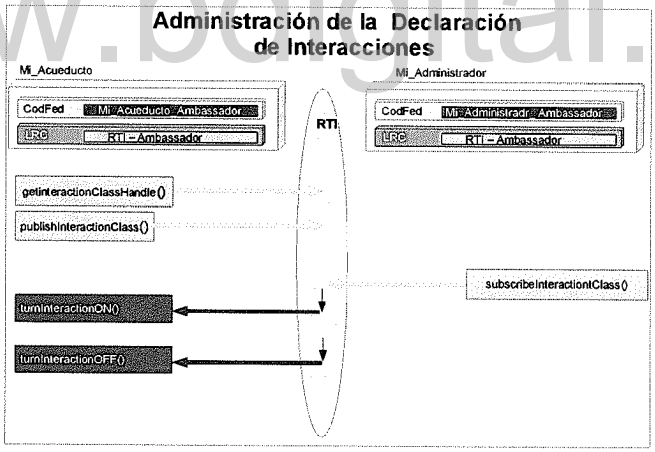
En el federado *Mi Acueducto* se ejecuta la simulación del Acueducto "La Ceibita". Para esta simulación se usa el archivo con el modelo de simulación del acueducto llamado *CeibitaXS_Nodo.inp*, que contiene la descripción de la red mostrada en la figura 4.4 y que puede verse en el anexo Sección C.1.

El federado *Mi Acueducto* publica la salida de este proceso, que son los datos asociados al *Estanque de Almacenamiento* proporcionado por la simulación. En la figura 4.18 las responsabilidades de los federados en el proceso de declaración de objetos para *Mi Federación*.

Para el federado *Mi Administrador* se ha definido como entrada al proceso del federado la información proporcionada por el federado *Mi Acueducto*, esto es, los datos asociados al *Estanque de Almacenamiento* proporcionado por la simulación. Por ello, el federado *Mi Acueducto* se suscribe a esos objetos, ver figura 4.18. Los datos aportados al federado *Mi Administrador* son usados por el agente para gestionar el tanque de almacenamiento.



(a) Declaración de Objetos



(b) Declaración de Interacciones

Figura 4.18: Diseño de los Federados. Tomado de DoD [50].

MANEJO DEL TIEMPO. Los federados definidos para la federación *Mi Federación*, operan con *Sincronización Conservativa*. Esta forma de operacion es aquella en la cual los federados avanzan en el tiempo solo cuando se garantiza que no se recibirá ningún evento del pasado. Los eventos son de tipo TSO, es decir están ordenados o "marcados"

en el tiempo. El RTI garantiza la distribución ordenada de todos los eventos de tipo TSO.

CICLO DE VIDA DE LA FEDERACIÓN *mi federación.* El ciclo de vida de la federación *Mi Federación* se ha dibujado en la figura 4.19. El federado *Mi_Acueducto* crea la federación, *createFederationExecution*, y se une a la misma, *joinFederationExecution*.

Posteriormente, el federado *Mi_Administrador* se une a la federación, *joinFederationExecution*. Cuando *Mi_Acueducto* termina la simulación, renuncia a la federación, *resignFederationExecution*. Luego, el federado *Mi_Administrador*, renuncia a la federación y por último la destruye, *destroyFederationExecution*.

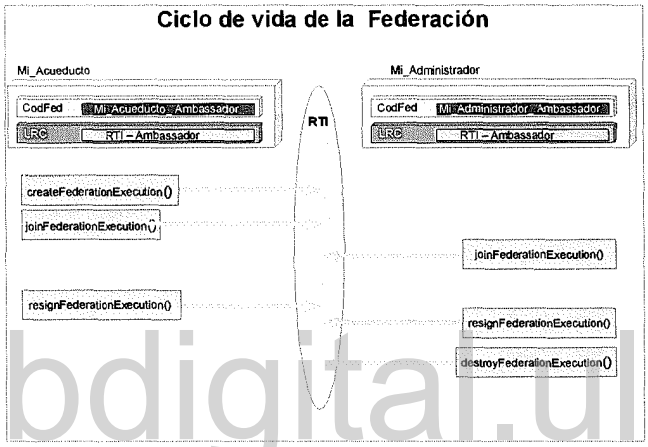


Figura 4.19: Ciclo vida de la federación. Tomado de DoD [50].

4.4.1.4 *Integración y pruebas de la federación*

En este paso se verifica que la federación pueda ser creada, la ejecución de la federación, la incorporación de los federados y los puntos de sincronización definidos.

4.4.1.5 *Ejecución y análisis de resultados.*

Una vez iniciada la ejecución de la federación, se obtienen los resultados. El federado *Mi_Administrador* recopila los datos que proporciona el federado *Mi_Acueducto*. Estos datos serán usados posteriormente por el agente para realizar evaluación. A partir de esta evaluación, es posible introducir mejoras o cambiar políticas, pueden plantearse nuevos escenarios y ejecutarse nuevas simulaciones.

Conclusión

En esta sección se ha construido un modelo de simulación, llamado *Mi Federación*, usando el estándard HLA 1.3. Este modelo esta constituido por dos federados, *Mi_Administrador y Mi_Acueducto.*

El federado *Mi_Acueducto* incluye el modelo del PSAP, descrito en la figura 4.4, construido con EpaNet. El federado *Mi_Administrador* obtiene los datos de la demanda promedio y de la presión en el nodo Estanque de Almacenamiento, proporcionado por la simulación, y que van a ser las entradas al proceso de simulación que realiza GALATEA. En la siguiente sección se detalla este aspecto.

El uso del estándard HLA proporciona una base para extender el modelo de simulación con otros simuladores.

4.4.2 *Construcción del agente regulador.*

En la Subsubsección 3.1.1.1 se estableció que los agentes pueden describirse a través de las ecuaciones Ecuación 3.10 y Ecuación 3.11, y que se reproducen a continuación. Un agente se define con cuatro elementos : $a_\tau =< k, goals, georefs, Context >$, donde $k \in K_\tau$, es la base de conocimiento del agente, $goals \in G_\tau$, son las metas del agente $georefs \in Shapes_\tau$ es la forma del agente y *Context* es el contexto espacial. Y el tipo de agente τ se formaliza con $\tau =< K_\tau, G_\tau, Shapes_\tau, \sum_\tau, P_\tau, Perception_\tau, Update_\tau, Planning_\tau >$ donde K_τ es el conjunto de bases de conocimiento posibles para este tipo de agente, G_τ es el conjunto de metas posibles para este tipo de agente, $Shapes_\tau$ es el conjunto de formas admisibles que el cuerpo del agente puede adoptar,$\sum_\tau$ es el conjunto de acciones posibles que el agente puede ejecutar, P_τ es el conjunto de observaciones posibles que el agente puede hacer, y *Perception*, *Update* y *Planning* son, funciones para modelar las conexiones entre percepción y acción para este tipo de agente.

Con la ayuda de este Modelo Formal de Agente determinaremos los elementos que se necesitan para diseñar y construir el agente regulador.

Los pasos para modelar el agente que gestione la salida del estanque de almacenamiento pueden clasificarse en dos: diseño y construcción del agente.

En la fase de diseño del agente se describe el tipo de agente y se establece sus metas, base de conocimiento, su forma y contexto espacial. También debe definirse el conjunto de observaciones que el agente puede hacer, y las funciones que conectan las observaciones con las acciones del agente. Entonces, para diseñar y construir un agente se debe definir:

- Diseño del Agente:

- Descripción del agente
- Establecer el conjunto de observaciones y acciones posibles del agente.
- Establecer las metas del agente.
- Definir la forma del agente.
- Definir el contexto espacial del agente.
- Construir la base de conocimiento.
- Diseñar las funciones para modelar las conexiones entre percepción y acción para este tipo de agente.

- Construcción del agente
- Ejecución de la federación con el agente regulador.

4.4.2.1 *Diseño del Agente.*

A continuación se describe los pasos seguidos para diseñar el agente regulador.

DESCRIPCIÓN DEL AGENTE. El agente regulador tiene como función la de gestionar la salida del estanque de almacenamiento del acueducto "La Ceibita" y proporciona el plan de suministro de agua potable para el escenario simulado.

Para realizar esta gestión, el agente debe obtener los valores en el estanque de almacenamiento, datos que son proporcionados por la salida de la simulación del acueducto. El agente, también, debe tener la información relacionada con las poblaciones que se surten del acueducto, el número de habitantes y la dotación promedio para que el agente pueda calcular el plan de servicio de agua potable.

El agente regulador observa los niveles de agua en el estanque de almacenamiento del acueducto, la entrada de agua al estanque de almacenamiento y los datos de la población y toma la decisión de cuanta agua potable se va a distribuir a las poblaciones. En la figura 4.20 se observa el caso de uso para el agente regulador.

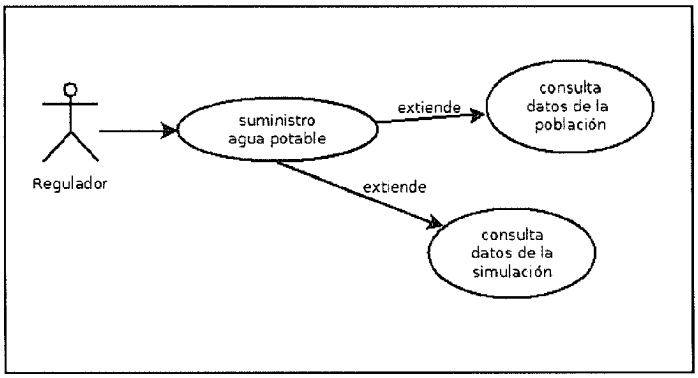


Figura 4.20: Caso Uso del agente Regulador

ESTABLECER EL CONJUNTO DE OBSERVACIONES DEL AGENTE. El agente regulador observa los siguientes atributos que se detallan en el cuadro 4.11. Los atributos provienen de dos fuentes: el proceso de simulación del acueducto "La Ceibita" y la geografía donde actúa el agente. Del proceso de simulación el agente observa la identificación, la demanda y la presión del nodo. Y de la geografía, el agente tiene conocimiento del nombre de la población, el número de habitantes y la dotación promedio para un año determinado.

	atributo	unidad	formato
Proceso simulación acueducto "La Ceibita"	id_nodo		string
	demanda	lts/seg	float
	presión	mts	float
Geografía donde actúa el agente	nombre_pob		string
	número_hab		int
	dotación_prom	lts/seg	float
	año		date

Cuadro 4.11: Observaciones del agente

ESTABLECER LAS METAS DEL AGENTE. Para este ejercicio se han definido las siguientes metas:

- Si existe caudal neto entrante en el estanque de almacenamiento, se suministra el agua a las poblaciones.

```
if ingresa_volumen(Litros) then  calcula_distribuye(Litros).
```

DEFINIR LA FORMA DEL AGENTE. No se ha definido forma alguna para "corporizar" al agente regulador puesto que no hace falta.

DEFINIR EL CONTEXTO ESPACIAL DEL AGENTE. El contexto espacial define el espacio en el cual el agente "corporizado" se desenvuelve. Pero en este ejemplo, no se ha definido un "cuerpo" al agente por lo que su conexto espacial no tiene sentido definirlo.

DISEÑO DE LA BASE DE CONOCIMIENTO. En el proceso de diseño de la base de conocimiento del agente regulador se definen el vocabulario que identifica los predicados y las funciones, y los procedimientos de inferencia. Para este ejemplo se ha usado *galatea.gloria* que es una extensión de GALATEA que proporciona estos procedimientos de inferencia.

Para especificar las metas del agente regulador se usan los lenguajes ACTILOG, lenguaje para escribir reglas de tipo *condición* $\rightarrow$ *meta* (DÁVILA [35]) y OPENLOG, lenguaje de programación de lógica de agente para un razonador abductivo (Dávila [34]).

La base de conocimiento del agente regulador se explica a continuación, cuadro 4.12. La regla de la base de conocimiento es *calcula_distribuye* que se activa con la regla:

$$if\ ingresa\_volumen\ then\ calcula\_distribuye$$

La regla *calcula_distribuye* se reduce a dos submetas: *reserva* y *disponible*.

La submeta *reserva* corresponde al cálculo de la cantidad mínima de volumen de agua que debe mantener el estanque de almacenamiento. El volumen mínimo de agua se calcula con el radio del estanque de almacenamiento y la presión mínima que se debe mantener.

La submeta *disponible* corresponde al cálculo de la cantidad disponible para distribuir a la población, esto es la diferencia entre los predicados *ingresa_volumen* y *reserva*.

to *calcula_distribuye*(*Litros*) **do** *reserva*(*Litros*, *R*), *disponible*(*R*)

Cuadro 4.12: Base de Conocimiento del Agente Regulador.

DISEÑAR LAS FUNCIONES PARA MODELAR LAS CONEXIONES ENTRE PERCEPCIÓN Y ACCIÓN PARA ESTE TIPO DE AGENTE. El Modelo Formal de Agente describe las funciones *Perception*, *Update* y *Planning* para designar las funciones necesarias para establecer las conexiones entre lo que percibe el agente y las acciones que debe realizar para cumplir con la meta de alto nivel que le fue asignada.

El agente regulador al ser de tipo reactivo no tiene funciones *Update* que actualizen su base de conocimiento. Para este tipo de agente están definidas funciones de tipo *Perception*, que describe como el agente percibe su ambiente y *Planning* donde se modela el proceso por medio del cual un agente reduce las metas de alto nivel

al conjunto de acciones que se van a ejecutar. La función *Perception* se implementa en el Programa 4.6.

```
public class Delta extends Node {
    public PrologAg AgenteA ;

    @SuppressWarnings("static-access")
    public Delta() {
        super("Delta", 'A');
        Glider.nodesl.add(this);

        String Path = "C:/Users/Virginia/ejemplo/Mi_Ejemplo";
        AgenteA  = new PrologAg("agenteA");
        AgenteA.demoHome = Path;
        AgenteA.initProlog();

        GInterface.agentList.add(AgenteA);

        System.out.println("Este es un Ejemplo de Simulacion de
            un Agente Administrador del Acueducto");
        System.out.println("");

    }

    @Override
    @SuppressWarnings("empty-statement")
    public void fact(){
        it(1);

        Glider.setTime(24);

        // Actualiza el reloj del agente
        AgenteA.clock = Glider.getTime();

        // Transmite lo que el agente debe ver
         ActualizarSensores(AgenteA);

        // Activa el razonado del agente
        AgenteA.cycle();

        System.out.println("Simulador : Comienzo a procesar las
            influencias");
        GInterface.gatherInfluences_test();
        GInterface.process_test();

        AgenteA.inputs.clear();
    }

    @SuppressWarnings({ "empty-statement", "unchecked" })
    public void ActualizarSensores(PrologAg agente){
          agente.inputs.add("volumen("+AgenteAdministrador.
              volumen+")");
```

```
        System.out.println("Este agente observara " + agente.
            inputs);
    }
}
```

Programa 4.6: Archivo Delta.java

4.4.2.2 *Construcción del agente.*

En esta fase se construye el agente siguiendo lo especificado en la etapa de diseño. Para la construcción del agente regulador se usa el simulador GALATEA y la base de conocimiento el lenguaje Prolog, especificamente la distribución SWI-Prolog SWI-Prolog [156].

Como paso final de esta fase, el agente se integra a la plataforma HLA en gvSIG. Esto se hace cuando se incluye el agente en el federado *Mi Administrador* que se construyó en 4.4.1.

Conclusión.

Se ha construido un agente regulador que administra la salida del estanque de almacenamiento. El agente se ha integrado en el federado *Mi Administrador*. El modelo del agente establecido en el Modelo Formal de Geosimulación multiAgentes se ha utilizado como guía para desarrollar el agente regulador.

4.4.3 *Integración de la federación en el SIG.*

La integración de la federación en el SIG se hace de manera similar que lo realizado para la construcción de una extensión en gvSIG, ver 4.3.2.4. Es importante resaltar que se debe tomar en cuenta que el archivo de datos suministro.fed, 4.5, y las dependencias asociadas al RTI deben estar incluidas en el proceso de construcción del archivo ejecutable de la extensión.

El archivo *config.xml* se ha extendido para incluir las opciones de menú de la extensión en gvSIG que llaman a los federados *Acueducto* y *Administrador*. En la figura 4.21 se muestra el archivo *config.xml*.

```
<extension class-name="com.iver.cit.gvsig.myplugin.Acueducto"
     description="Mi Federacion."
     active="true"
     priority="1">
     <menu text="Mi Plugin/Federacion/Acueducto" action-command="
         AcueductoExtension" />
</extension>

<extension class-name="com.iver.cit.gvsig.myplugin.Administrador"
         description="Mi Federacion."
     active="false"
     priority="1">
 <menu text="Mi Plugin/Federacion/Administrador" action-command="
     AdministradorExtension" />
</extension>
```

Figura 4.21: El archivo config.xml con las opciones de la federación

La extensión tiene la siguiente estructura descrita en la figura 4.22.

```
src
        com.iver.cit.gvsig.myplugin
                AcuedutoExtension.java
                AdministradorExtension.java

        com.iver.cit.gvsig.myplugin.tools
        Mi_AcueductoAmbassador.java
                InfoByAdministradorListener.java
                Mi_AdministradorAmbassador.java
```


Figura 4.22: Estructura extensión gvSIG para Mi Federación

El federado Acueducto esta constituido por *AcueductoExtension.java* y *Mi_AcueductoAmbassador.java*.

A su vez, el federado Administrador esta constituido por *AdministradorExtension.java*, que conforma el esqueleto de la extensión y que llama al código que ejecuta la simulación. *InforByAdministradorListener.java* y *Mi_AdministradorAmbassador.java*, se activan cuando el usuario selecciona uno de los polígonos que representan las poblaciones que están surtidas por el acueducto "La Ceibita".

Con esto, la integración de HLA en gvSIG está hecha.

Conclusión

En esta sección se hace referencia a los pasos que se deben seguir para integrar la versión de HLA, PoRTIco en gvSIG. Con la integración de PoRTIco en gvSIG se ha establecido una plataforma para integrar al modelo de simulación existente, agentes y otros simuladores.

4.4.4 *Ejecución de la federación con el agente regulador.*

Para mostrar la salida de la simulación se han diseñado los siguientes escenarios para el acueducto "La Ceibita":

1. El primer escenario es el que ya fue descrito en la figura 4.4. En la figura 4.23 se muestra el comportamiento del nodo DesArenador, nodo ubicado en la entrada del acueducto. Para este escenario, la evolución en el tiempo de los valores de presión y demanda para el nodo DesArenador indican que existe una entrada constante de agua al sistema. El Estanque de Almacenamiento se comporta de la manera que se muestra en la figura 4.24.

   El agente regulador proporciona la misma cantidad de agua para los pobladores de los distintos poblados que atiende el acueducto "La Ceibita". La salida de la simulación se muestra en la figura 4.25 para las poblaciones de La Ceibita y Buena Vista. Se puede observar que sólo para el primer día, el suministro es mayor que 55 litros/día-persona. El resto de los días, el suministro varía entre 40 y 45 litros/día-persona, indicando ya un déficit al estar por debajo de los 50 litros/día-persona que es la cantidad mínima de agua necesaria para uso doméstico sugerido por la Organización Mundial de Salud[4].

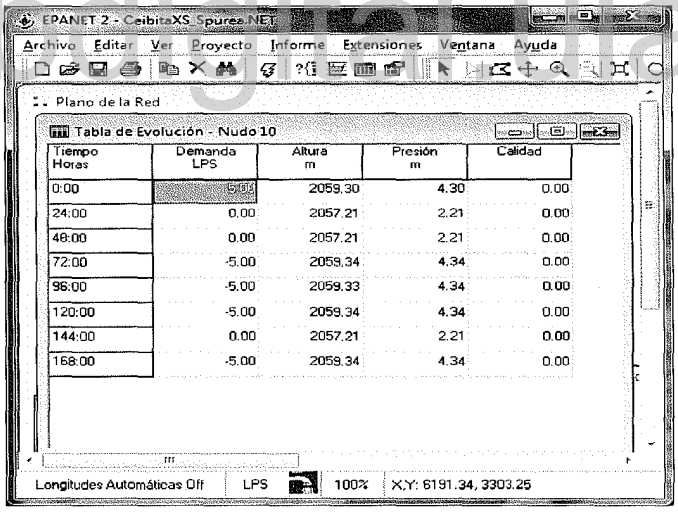


Figura 4.23: Evolución en el tiempo del Nodo DesArenador. Escenario1.

4 está en http://preparativosyrespuesta.cridlac.org/XML/spa/doc18188/doc18188-contenido.pdf

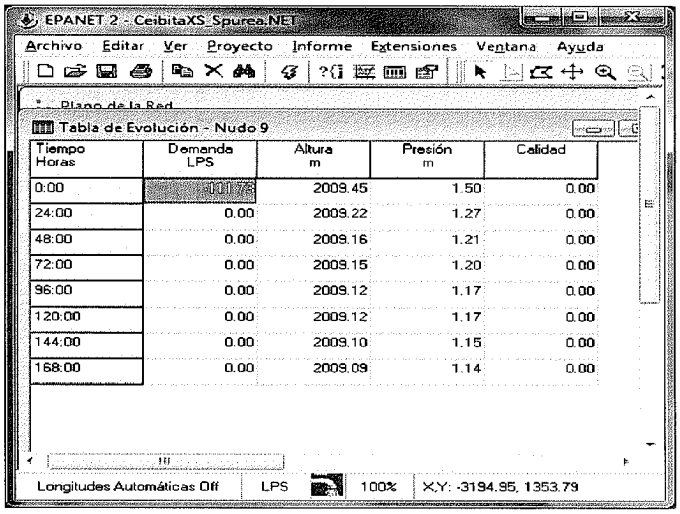
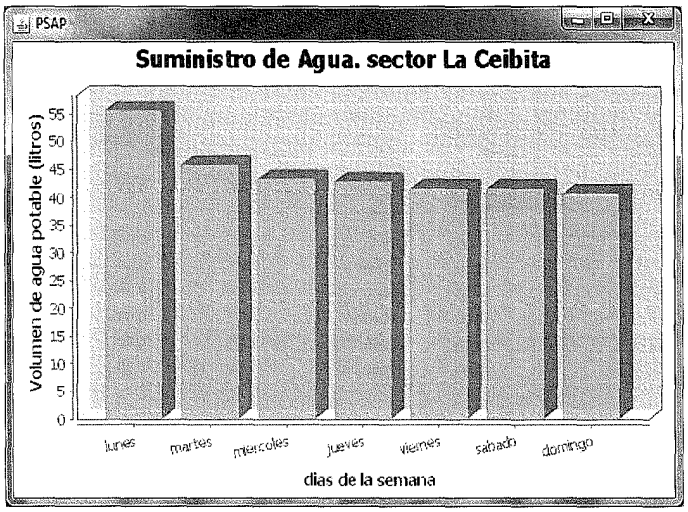
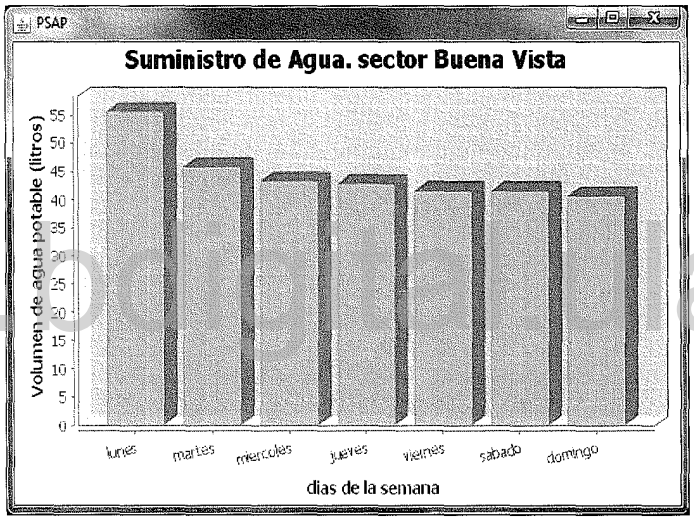


Figura 4.24: Evolución en el tiempo del Estanque de Almacenamiento. Escenario 1.



(a) Sector La Ceibita



(b) Sector Buena Vista

Figura 4.25: Salida Simulación Escenario 1.

2. Para construir el segundo escenario se ha cambiado el valor de la demanda base del nodo DesArenador con el objeto de generar distintos valores para las demandas y presiones en el nodo mencionado. En la figura 4.26 se observa el comportamiento a lo largo del tiempo de la presión y la demanda para el nodo DesArenador. La presión en el nodo varía entre valores positivos como negativos, indicando, la presencia y ausencia de agua en el nodo, respectivamente. Y en la figura 4.27, se muestra la evolución en el tiempo de la demanda y la presión en el Estanque de Almacenamiento para este escenario. La salida de la simulación, ver la figura 4.28,

muestra que el suministro el primer día es de 12 litros/día-persona y el resto de los días es un poco más de 35 litros/día-persona, ambos valores muy por debajo de los 50 litros/día-persona.

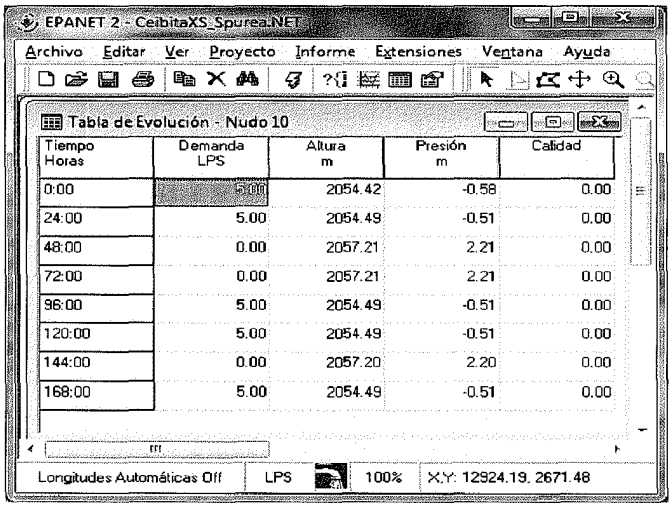


Figura 4.26: Evolución en el tiempo del Nodo DesArenador. Escenario 2.

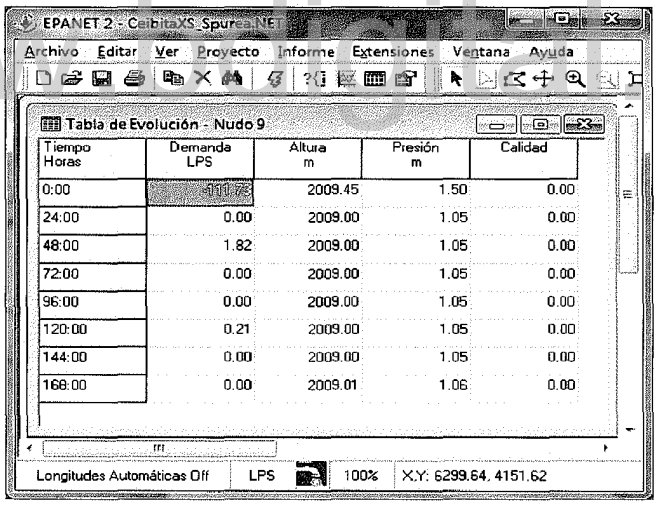
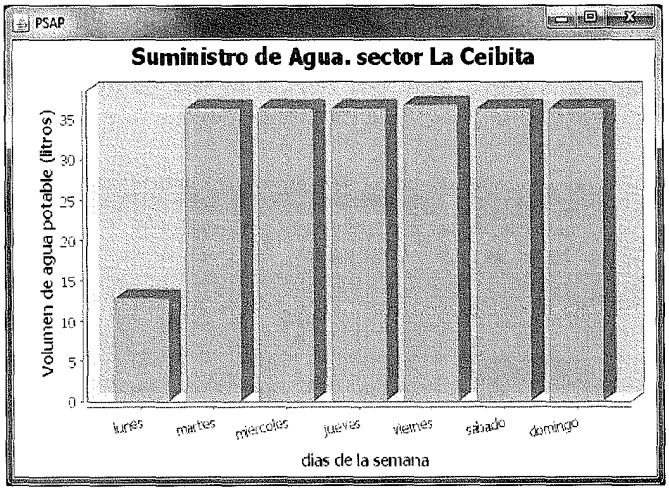
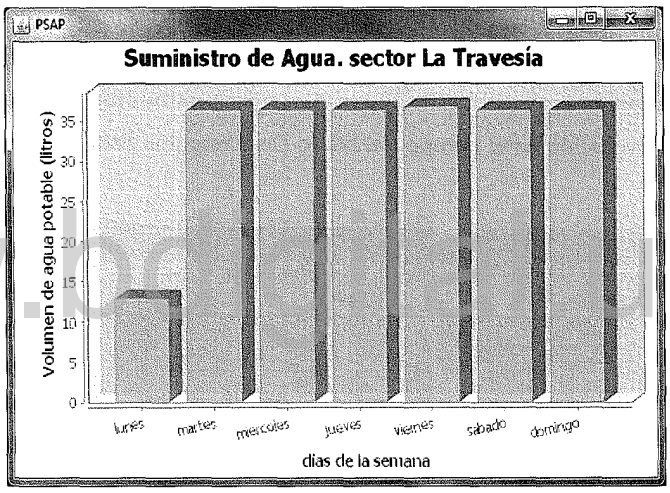


Figura 4.27: Evolución en el tiempo del Estanque de Almacenamiento. Escenario 2.



(a) La Ceibita



(b) La Travesía

Figura 4.28: Salida Simulación Escenario 2.

4.4.4.1 *Pruebas de rendimiento del software.*

Con el propósito de determinar el tiempo de respuesta de la aplicación se realizaron pruebas de rendimiento del software. Las pruebas de rendimiento consistieron en medir los tiempos de respuesta de los federados y del agente regulador.

En el cuadro 4.13 se muestra los resultados de esta prueba. En la columna *Escenarios* está el tiempo de simulación definido para el modelo del Acueducto "La Ceibita". En las columnas *Federado Mi_Acueducto* y *Federado Mi_Administador* se muestra la contabilización del tiempo de ejecución de los federados en mili

segundos (mseg). Y en la columna *Agente Regulador* está el promedio del tiempo de respuesta del agente, estimado con el número de llamadas que se realiza al agente.

Por ejemplo, para el escenario de *1 semana* los federados *Federado Mi_Acueducto* y *Federado Mi_Administador* tienen un tiempo de respuesta de 15.268 mseg y 15.526 mseg, respectivamente. El agente regulador es invocado en siete oportunidades, por cada día de la semana, y el promedio de su tiempo de respuesta es de 1.243 mseg.

Estos resultados muestran que con el incremento del tiempo de simulación en el modelo del acueducto, se aumenta sensiblemente el tiempo de respuesta de los federados.

Escenarios	Federado Mi_Acueducto	Federado Mi_Administrador	Agente Regulador
1 semana	15.268	15.526	1.243
2 semanas	20.739	21.205	1.092
4 semanas	45.420	45.716	1.223
24 semanas	211.809	212.245	1.199

Cuadro 4.13: Medidas del tiempo de la simulación.

4.5 DISEÑAR UN AGENTE QUE GESTIONE LOS ESCENARIOS DE SIMULACIÓN DEL ACUEDUCTO.

En esta sección se presenta el diseño del agente consultor que tiene como objetivo gestionar los escenarios de simulación. Es importante acotar que esta fase está en etapa de construcción.

Una interfaz gráfica le permite al usuario escoger un escenario climático y, a partir del escenario climático seleccionado, el agente consultor construye los escenarios de simulación que van a generar diversos datos que se utilizan como entrada al proceso de simulación del acueducto "La Ceibita". El proceso de diseño del agente consultor cumple con los siguientes pasos: (1) Construcción de la interfaz gráfica para el escenario climático, (2) Construcción de un nuevo federado, y (3) Construcción del agente consultor. A continuación se detalla los pasos para diseñar el agente consultor.

4.5.1 *Construcción de la interfaz gráfica para el escenario climático.*

La interfaz presenta la opción de modelar el escenario de amenazas. En la figura 4.29 se muestra la interfaz para el escenario climático. La pantalla que se despliega muestra una gráfica con el promedio histórico de lluvias de la zona y una barra, ubicada en la zona inferior de la figura referida, para escoger por cada mes del año un valor *bajo*, *medio* o *alto*, establecidos como predeterminados. *Bajo*, es el valor de precipitación más bajo para ese mes, entre los valores históricos

registrados; *medio*, el valor de la precipitación es igual al promedio; y *alto*, el valor de precipitación más alto para ese mes, entre los promedios calculados para la zona de estudio.

Los valores históricos de precipitaciones que se utilizaron para construir el promedio de lluvias, están detallados en la opción *Historia* del menú. Esta opción despliega la gráfica derecha de la figura 4.29, que muestra los valores de las precipitaciones mensuales desde el año 2.000 hasta 2.008. Los valores de precipitaciones para la zona de estudio se calcularon siguiendo el método de los polígonos de Thiessen Ramírez [137], Fetter [62], Chow et al. [25] usando los datos de la estación metereológica *Mucujún* Andressen [3], Guada [76], Méndez [114], que forma parte de la red bioclimática del estado Mérida, REDBC [141] y de la estación climatológica *La Mucuy, INIA [85],* de la red bioclimática del Instituto Nacional de Investigaciones Agrícolas (INIA).

Una vez hecha la selección de los niveles de lluvia, se despliega el escenario de lluvia en el cual el gráfico muestra los niveles de precipitación escogidos por el usuario. El nivel de lluvia en la zona inciden sobre la cantidad de agua que ingresa al acueducto y esta influencia se visualiza en la parte baja de la figura 4.29. Como una primera aproximación, para estimar cuanto aportan las lluvias al acueducto se han tomado los valores puntuales de medida del caudal realizados en Ramírez [136], y con interpolación aritmética se han generado los aportes mensuales al acueducto para el nivel de precipitaciones estimado.

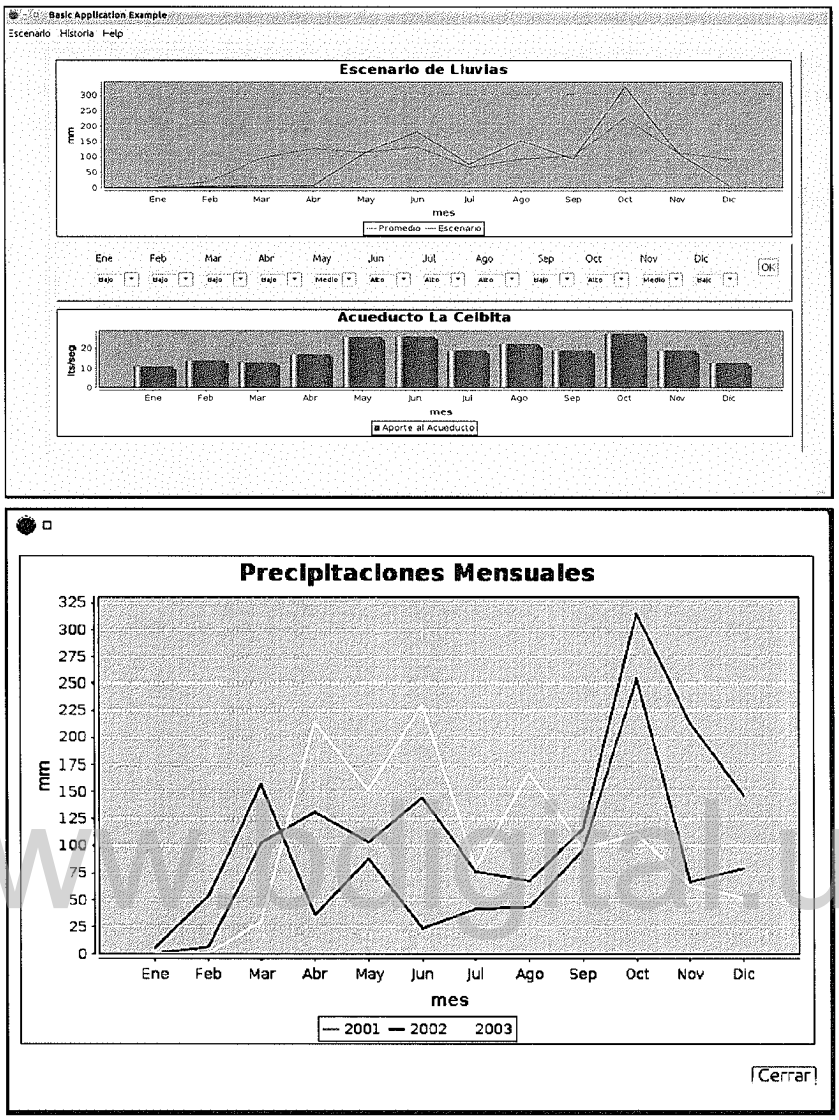


Figura 4.29: Escenario climático.

En el programa de simulación EpaNet los embalses proporcionan el agua que necesita el acueducto según lo establecido por el diseñador. Para gestionar los aportes al acueducto sustituiremos la definición del embalse de la figura 4.4 por un nodo con un patrón de demanda variable. Con este cambio la entrada al acueducto puede administrarse según los requerimiento de usuario.

Conclusión.

Se ha descrito la interfaz gráfica para la escogencia del escenario climático. La interfaz gráfica es administrada por el agente consultor, que se incorpora a la federación por medio del federado *Mi Consultor*.

En el anexo Sección A.2 se documenta el diseño de la infraestructura que se necesita para incorporar este nuevo federado a la federación *Mi Federación*. Este proceso de diseño muestra con que facilidad el sistema puede ser extendido al integrarse otros simuladores El federado *Mi Consultor* tendria como objetivo proporcionar a la federación, los datos asociados al escenario climático.

Parte IV

CONCLUSIONES Y RECOMENDACIONES

5

CONCLUSIONES Y TRABAJOS FUTUROS.

El desarrollo de sistemas en el área de la geosimulación multiAgentes es una actividad que puede catalogarse en el ámbito de los sistemas complejos. El diseño de un sistema complejo involucra una serie de procesos y de componentes que requieren de una definición apropiada y un tratamiento adecuado. Este tipo de sistemas ha generado mucho interés en los últimos años, tanto en las comunidades científicas como en la industria. Este interés es porque esta clase de sistemas sirven para solucionar problemas que se presentan en diversos entornos.

Los principales aportes en el área de la geosimulación multiAgentes han estado orientados hacia la construcción misma del sistema, pero no se ha dedicado mucho esfuerzo en el desarrollo de propuestas para la especificación de este tipo de sistemas. Es por esto, que se considera importante contar con un modelo formal de geosimulación multiAgentes y que permite explorar la relación entre los agentes en un sistema multiAgente y los SIG, y establecer una especificación de sus componentes.

A continuación se presentan las conclusiones más relevantes que se han obtenido después de realizar esta investigación. Éstas son el resultado, tanto del estudio de los agentes de software, de las metodologías para desarrollo de SMA, de la teoría de simulación multiAgentes, de la geosimulación multiAgentes y de la SOC.

Adicionalmente, se presentan los trabajos futuros y las líneas de trabajo que se abren para complementar el alcance de esta investigación.

5.1 CONCLUSIONES GENERALES.

La conclusión más relevante de este trabajo es que el modelo formal de geosimulación multiAgente que se propone en esta investigación, sirve para el desarrollo de sistemas de simulación complejos que integren a los SIG y los agentes de software.

El Modelo Formal de Geosimulación multiAgente es un soporte en el proceso de desarrollo de sistemas de simulación complejos que integren a los SIG *y los agentes de software.*

Para llegar a esta conclusión se desarrolló un proceso paulatino de investigación en el cual se fueron analizando diversos aspectos relacionados con los agentes de software, las metodologías para desarrollo de SMA, la teoría de simulación multiAgentes, la geosimulación multiAgentes y la SOC.

La investigación inicio con el estudio de los agentes de software, su conceptualización y, la determinación y definición de las diversas estructuras que definen el estado interno y externo que se le atribuye a los agentes. Este estudio permitio establecer un modelo más amplio, el MRA que incluye las diversas caracterizaciones asociada a los agentes de software.

Se revisaron un conjunto de metodologías conocidas y se compararon con el MRA. Esta comparación ha permitido establecer que el MRA es un modelo general de un agente que abarca las distintas visiones de los SMA que ofrecen las metodologías revisada.

El MRA es un modelo general de agente que contiene las diversas estructuras internas y externas que se le atribuyen a los agentes de software.

Se estudió la teoría de simulación mutiAgentes en la que se sustenta el software para simulación GALATEA que describe matemáticamente un sistema poblado por muchos agentes, sus componentes y sus reglas de transición; así como la caracterización interna de los agentes que conforman el sistema. Se mostró que el MRA establecido en la sección 2.1 esta incluido en la formalización de la teoría de simulación de GALATEA. También se estableció que la teoría de simulación de GALATEA no contempla la ubicación de los agentes en el espacio geográfico.

La teoría de simulación de GALATEA es un modelo formal de un sistema de simulación multiAgentes pero no contempla la ubicación de los agentes en un espacio geográfico.

En el área de investigación que estudian la integración de sistemas de simulación, SIG y los agentes de software, se investigó el concepto de geosimulación, los tipo de deficiencias o limitantes que se buscan subsanar con la integración de sistemas de simulación, SIG y los agentes de software; la existencia de modelos formales en el área de geosimulación multiAgentes y los métodos que se usan para el desarrollo de geosimulaciones multiAgentes.

Se determinó que la integración de los SIG y los agentes de software representa una oportunidad para superar la limitante que presentan los SIG con el registro del tiempo. Además, de que las características "emergentes" como respuesta a los eventos de los agentes de software,

proporcionan un modelo para el diseño de los sistemas naturales, sociales, entre otros, que pueden ser modeladas como propiedades "que emergen" de interacciones individuales.

También, se conoció la teoría de simulación de MAGI que es una teoría formal de un geosimulador con agentes y objetos en ella, pero que no contempla un registro del tiempo. Se estableció la factibilidad de la integración de las teorías de simulación MAGI y de GALATEA. La teoría resultante recibe de la teoría MAGI, la definición del cuerpo físico de cada agentes y su ubicación en una geografía, y de la teoría GALATEA obtiene un registro explícito del tiempo.

La teoría de geosimulación multiAgentes MAGI *es un modelo formal de un sistema de simulación que incluye un sistema de información geográfica y sistemas multiAgentes en ella pero carece del registro explícito del tiempo.*

Con el objeto de determinar la posibilidad de desarrollar un servicio en el área de la geosimulación multiAgentes se investigó los conceptos relacionados con SOC, un paradigma de desarrollo que se enfoca en el acoplamiento débil de los elementos que la integran, promoviendo la reutilización de software. Se estableció que las metodologías de sistemas multiAgentes y la SOC definen, de manera similar, el concepto de servicio y que los servicios web tienen asociados unos lenguales específicos para describir el servicio y el esquema para intercambio de información.

Los servicios están relacionados con los roles del agente y se componen de una serie de actividades y tareas que se deben realizar para cumplir ese rol. Los servicios web tienen asociado lenguajes específicos para describir un servicio.

La teoría multiagentes para simulación de GALATEA, se ha conectado con la teoría MAGI para producir una teoría multiAgentes para geosimulación, la cual explica la relación entre agentes y SIG, como una herramienta para simular sistemas espaciales complejos. La teoria multiAgentes para geosimulación además de describir un sistema poblado por muchos agentes, sus componentes y sus reglas de transición, describe físicamente el sistema geográfico, define el cuerpo físico de cada agentes y su ubicación en una geografía, da cuenta de la creación tanto de objetos como de agentes, y proporciona un registro explícito del tiempo.

La teoría de geosimulación multiAgentes propuesta es un modelo formal de un sistema de simulación que incluye un sistema de información geográfica, un sistemas multiAgentes con representaciones explícitas del tiempo.

El modelo formal de geosimulación multiAgentes se ha comparado con el modelo formal que describe a BDA y BDD, mostrándose que el modelo formal de geosimulación multiAgentes subsume a la descripción formal de las BDA y BDD. También se comparó el modelo formal de geosimulación multiAgentes con el modelo de BDOOA demostrándose que éste modelo está descrito en el modelo de formal de geosimulación multiAgente.

Por tanto, el modelo de formal de geosimulación multiAgente es un modelo más general que toma en cuenta, no solo la eventual respuestas a consultas o el comportamiento al activarse las reglas ECA, sino también se caracteriza por especificar un comportamiento autónomo, persistente y auto-sostenido de los agentes y módulos con conocimiento.

La teoría de geosimulación multiAgentes propuesta es un modelo formal que subsume el modelo formal de las BDA *y* BDD*, además del modelo de* BDOOA.

Para comprobar que el modelo formal de geosimulación multi-Agente es una especificación apropiada para el proceso de modelado y construcción de un sistema complejo, se presentó el diseño general de un sistema de información para servicio público en el dominio de la GDRR, que simula la ocurrencia de cambios en la dinámica de lluvias sobre una geografía determinada que podrían significar períodos de sequía. La metodología que se utilizó para el desarrollo de la aplicación fue la *programación extrema* y siguiendo los preceptos de esta metodología, se construyó la *historia de usuario* para el PSAP, nuestra prueba del concepto.

El Modelo Formal de Geosimulación multiAgentes fue utilizado como guía para analizar y establecer la estructuración del sistema y los objetivos para desarrollar el PSAP. Los objetivos establecidos fueron: 1. *Desarrollar una interfaz al usuario que muestre al* PSAP, 2. *Diseñar un agente que en la simulación sea el encargado de gestionar la salida del acueducto, y 3. Diseñar un agente que gestione los escenarios de simulación del acueducto.*

En el objetivo *Desarrollar una interfaz al usuario que muestre al* PSAP, se construyó el modelo de simulación del acueducto "La Ceibita" usando el software de simulación EpaNet. Se ha construido una interfaz gráfica usando gvSIG, que permite al usuario consultar la salida de la simulación.

La construcción de la interfaz gráfica en gvSIG *del* PSAP *ha mostrado las potencialidades que tiene el software* gvSIG *para la creación de extensiones que permitan adecuarlo a un geosimulador multiAgentes.*

En el objetivo *Diseñar un agente que en la simulación sea el encargado de gestionar la salida del acueducto*, se usó el estándard HLA que

proporciona una base para extender el modelo de simulación con otros simuladores. Se integró HLA en gvSIG y se construyó una federación constituida por dos federados.

La integración de HLA *en* gvSIG *proporciona la infraestructura necesaria para integrar otros modelos de simulación. Además el uso de* HLA *en* gvSIG *proporciona al* SIG *una base para el registro del tiempo de ocurrencia de eventos.*

Se construyó un agente regulador que administra la salida del estanque de almacenamiento. El agente se integró en el federado *Mi Administrador*. El modelo formal de geosimulación multiAgentes se ha utilizado como guía para desarrollar el agente regulador.

Para la construcción del agente regulador se ha usado como guía el modelo formal de geosimulación multiAgentes. Además, con la construcción del agente regulador se ha integrado el software de simulación GALATEA *en* gvSIG

En *Diseñar un agente que gestione los escenarios de simulación del acueducto* se mostró como puede extenderse el sistema para integrar otros agentes. Para ello se integró otro federado en la federación. Se construyó una interfaz gráfica y se diseñó el agente consultor.

El sistema ha sido extendido, con el uso de HLA, *al integrar otro federado en la federación. El agente consultor aún está en proceso de diseño.*

5.2 CONTRIBUCIONES PRINCIPALES.

Nuestra hipótesis fundamental de trabajo es que la mejor manera de construir tales herramientas es a partir de una conceptualización de nociones como BD, SIG y agentes adaptable a este dominio de aplicación y servicio. Es por ello que el objetivo principal de esta investigación es la proposición de un modelo formal de geosimulación multiAgentes que formaliza estos conceptos. Así, que de acuerdo con esto y con los objetivos planteados también al comienzo, se describirán las principales contribuciones de esta tesis:

- Contribución en relación al modelo formal de geosimulación multiAgentes.

  Se ha construido un modelo formal de geosimulación multi-Agentes y se ha demostrado su uso en el diseño de una aplicación en el área de la GDRR. El modelo formal de geosimulación multiAgentes se usó para establecer la conformación del sistema y para el diseño de los agentes.

- Contribución en relación en el área de la interfaz para sistemas de geosimulación multiAgentes.

  En relación con el software para SIG, gvSIG y el software de simulación EpaNet. Se ha integrado el software de simulación EpaNet al gvSIG, extendiendo las funcionalidades del software gvSIG. Esta integración se evidencia en la extensión del PSAP para gvSIG, construida.

- Contribución en relación a la construcción de un servicio.

  Se ha construido un servicio móvil usando gvSIG Mobile versión 0.3 y el PSAP. Esta versión de gvSIG Mobile con la extensión del PSAP puede descargarse desde `https://simulants.svn.sourceforge.net/svnroot/simulants/`. y los pasos para su instalación están en el anexo Sección D.2.

- Contribución en relación al diseño del agente gestionador de la salida del acueducto.

  Se ha mostrado el uso del modelo formal de geosimulación multiAgentes para el proceso de diseño del agente regulador.

  Se le ha proporcionado al software gvSIG una plataforma de sismulación distribuida usando el estándard HLA, lo que le permite integrar al software gvSIG otros sistemas de simulación.

  Con la integración del estándard HLA en el software gvSIG se ha proporcionado un método para que el SIG tenga un registro del tiempo en el cual ocurren los eventos.

  Se ha integrado el software de simulación GALATEA en gvSIG extendiendo las funcionalidades del software gvSIG al tener a su disposición un sistema de simulación multiAgentes.

5.3 TRABAJOS FUTUROS.

El trabajo futuro planeado incluye desarrollar, implementar y evaluar con la comunidad el sistema descrito en el documento. Se pretende ampliar las funcionalidades del agente consultor para que interactue con el usuario, asistiéndolo en el proceso de construir distintos escenarios de amenazas e incorporando datos geográficos y metereológicos disponibles libremente.

Se proponen las siguientes líneas de trabajo:

- En el área de la Interfaz para sistemas de geosimulación multiAgentes.

  Se propone desarrollar extensiones gráficas en gvSIG para construir agentes de software, integradas a la interfaz gráfica que

ofrece gvSIG. Esto le proporcionaría al simulador GALATEA una interfaz gráfica en un SIG para ensamblar agentes de software.

Se propone desarrollar extensiones de software para gvSIG que permitan manipular el software de simulación EpaNet, desde la interfaz gráfica que ofrece gvSIG. De esta forma, la integración de EpaNet en gvSIG puede tener una interfaz que le facilite al usuario la construcción de modelos de simulación usando las facilidades gráficas que ofrece gvSIG.

- En el área de la construcción de un servicio.

  En el área de la contrucción de un servicio móvil con gvSIG Mobile se puede considerar las siguientes áreas de trabajo:

  - trabajar con la actualización de la JVM para dispositivos móviles. Esto posibilita el uso de sistemas de simulación en PDA u otros dispositivos móviles.

  En el área de la contrucción de un servicio Web con servidores de mapas usando el PSAP se puede considerar las siguientes áreas de trabajo:

  - crear una extensión en los simuladores involucrados para el intercambio de datos en el lenguaje XML.

  - construir una plataforma de PD para simulaciones distribuidas en la Web usando el estándard HLA 1516 ó HLA 1516-e. Integrar un servidor de mapas, como GeoServer, con HLA, siguiendo las pautas que se han descrito para el PSAP.

- En el área del diseño y construcción de agentes.

  Extender el diseño del agente consultor para incorporar fuentes de datos climáticas externas. Culminar el desarrollo del agente consultor.

  Se propone trabajar en la construcción de agentes con forma físicas y que perciban por medio de efectores el contexto espacial en el que se desenvuelven, aspecto que está considerado en el modelo formal de geosimulación multiAgentes pero que no fue estudiado en este trabajo.

Parte V

APÉNDICES

ARQUITECTURA DE ALTO NIVEL

A.1 BREVE INTRODUCCIÓN A LA HLA.

HLA es una arquitectura de propósito general para sistemas de simulación distribuidos. Con HLA se pueden integrar diversos sistemas de simulación sin importar la plataforma computacional.

HLA proporciona un marco general dentro del cual los desarrolladores de simulaciones pueden estructurar y describir su aplicación de simulación.

La especificación HLA fue desarrollada por la oficina de Defensa de Modelado y Simulación (Defense Modeling and Simulation Office-DMSO) del Departamento de Defensa de EEUU DoD [48]. En febrero de 1.998, la especificación 1.3 fue liberada y la comunidad de modelado y simulación inicio el conocimiento y uso de este estándard. Para septiembre de 2.000, HLA se volvio un estándard de la IEEE conocido como *IEEE 1516*,IEEE1516 [81] y ha venido siendo desarrollado como tal desde entonces, Flanagan [66]. Luego, en marzo de 2010, fue aprobado una nueva versión llamada la IEEE 1516e que se describe en Evolved [59].

A continuación se enuncian los conceptos básicos de la arquitectura HLA 1.3.

A.1.1 *La Arquitectura de HLA .*

A continuación hay un compendio acerca de HLA y sus componentes. Para realizar esta recopilación se utilizaron estos documentos: Möller [116], Möller and Löfstrand. [119], DMSO [47], DoD [50], Kuhl et al. [105], Crosbie and Zenor [31], DoD [49, 48]

A.1.1.1 *Terminología común de la HLA*

- Federado (ver figura A.1) : un programa componente de la simulación que pertenece a HLA, más un SOM, (ver SOM en A.1.2.1). Cada Federado puede modelar cualquier número de objetos en una simulación.
- Federación: una simulación compuesta de múltiples Federados conectadas vía los servicios de RTI, más un FOM, ver A.1.2.1.
- Objeto: colección de datos relacionados enviados entre simulaciones. Los objetos persisten en el tiempo y tiene atributos que pueden ser actualizados.
- Tipos de Datos: describe la semántica y la representación técnica de los atributos de una clase de objeto y parámetros de una interacción.
- Interacción: eventos enviados entre entidades de simulaciones. Las interacciones no persisten en el tiempo. Usualmente poseen algún parámetro.

- Parámetro: campo de datos de una interacción

A.1.1.2 *¿Cuales servicios proporciona HLA?*

- Los servicios de información de HLA se basan en *publicar* y *suscribir.*
- Los servicios de sincronización incluyen el manejo del *tiempo lógico, sincronización de puntos* y *guardar/restaurar.*
- Los servicios de coordinación incluyen mantener el registro de los federados y de la federación, transferir la responsabilidad del modelado, e inspecciones avanzadas y administración de la federación.
- El estándard consiste de tres documentos: Reglas, Especificación de la Interfaz y el OMT (ver A.1.2)

A.1.1.3 *Componentes de HLA*

Consiste de tres componentes, DoD [50]:

- Especificación de la Interfaz (A.1.1.8), que define como las simulaciones en HLA interactua con el RTI. RTI proporciona una libreria de programación y una interfaz de programación de aplicaciones de acuerdo a la especificación de la interfaz.
- Marco para OMT A.1.2 , que especifica que información es comunicada entre simulaciones, y como es documentada. También establece el formato de los módulos claves: FOM, SOM y Management Object Model (MOM)
- Reglas de la federación (A.1.1.4), que las simulaciones deben obedecer de manera que se cumpla con el estandar. Asegura una apropiada interacción de las simulaciones en una federación y describe las responsabilidades de la simulación y de los federados.

A.1.1.4 *Reglas:*

Las Reglas de HLA estan divididas en reglas de las Federaciones y reglas de los Federados.

- Reglas de la Federaciones: establece las reglas bases para la creación de una federación
  1. Requerimientos de Documentación: las federaciones deben tener un FOM documentado de acuerdo con el OMT de la HLA
  2. Representación del Objeto: las representaciones de objetos en el FOM deben estar en los federados, no en el RTI

3. Intercambio de Datos: durante la ejecución de una federación, todo el intercambio de datos del FOM entre federados debe ocurrir a través del RTI
4. Requerimientos de la Interfaz: durante la ejecución de una federación, los federados deben interactuar con el RTI de acuerdo con la especificación de la interfaz de HLA
5. Posesión del atributos: durante la ejecución de una federación, un atributo de una instancia de un objeto debe ser poseido por solo un federado en cualquier instante de tiempo.

- Reglas de los Federados

1. Documentación: federados deben tener un SOM, documentado de acuerdo con el OMT del HLA
2. Control de atributos : federados deben ser capaces de actualizar y/o reflejar cualquier atributo de objetos en su SOM y enviar y/o recibir interacciones de objetos del SOM externamente, como se especifica en su propio SOM
3. Transferencia de atributos: federados deben ser capaces de transferir y/o aceptar la propiedad de un atributo dinámicamente durante la ejecución de una federación, como se especifica en su propio SOM
4. Actualizaciones de atributos: federados deben ser capaces de variar las condiciones bajo la cual ellos proveen actualizaciones de atributos de objetos, como se especifica en su propio SOM.
5. Manejo del tiempo: federados deben ser capaces de manejar el tiempo local de manera que permitirá coordinar el intercambio de datos con otros miembros de una federación.

A.1.1.5 *¿Que es el RTI?*

El RTI es un software que conforma la especificación, pero no es por si mismo parte de ella. Proporciona los servicios de software que son necesarios para dar soporte a la simulación HLA. Uno de ellos es enviar los datos correctos al receptor correcto.

- Servicios del RTI
  - Separa simulación de comunicación
  - Facilita la construcción y destrucción de federaciones
  - Soporte a declaraciones y administraciones de objetos entre federados
  - Asistencia con el manejo del tiempo de las federaciones

- Proporciona comunicación eficiente a grupos lógicos de federados, una libreria de programación y una interfaz de programación de aplicaciones de acuerdo a la especificación de la interfaz.

COMPONENTES DEL RTI. La figura A.1 muestra una vista lógica de una federación HLA en ejecución.

- FedExec:
  - Un proceso en ejecución por cada ejecución de la federación
    - Se crea por el primer federado al unirse exitósamente a la federación
  - Administra que múltiples federados entren y salgan de la federación en ejecución
    - Asigna manejadores (*handles*) únicos para cada federado
  - Facilita el intercambio de datos entre federados
  - Interfaz de consola para operaciones manuales
- rtiExec:
  - Gestiona la creación y destrucción de múltiples federación en ejecuciones (con nombres diferentes)
  - Proceso global que se ejecuta en una plataforma
  - Escuchas a un puerto bien conocido
  - Interfaz de consola para operaciones manuales
- libRTI
  - Hace que los métodos de servicio de HLA estén disponible para los federados.

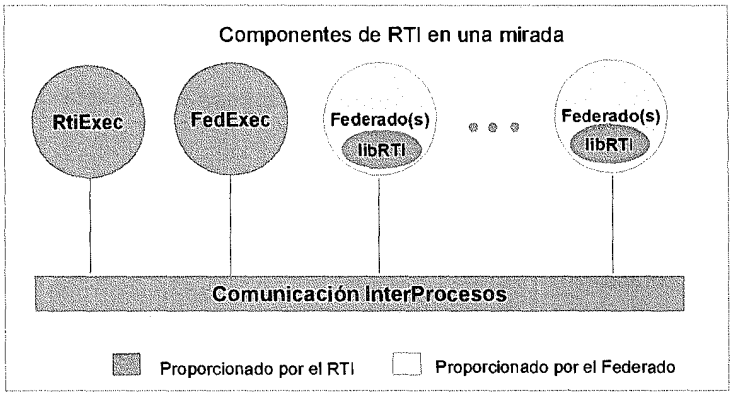


Figura A.1: Visión Lógica de los Componentes del RTI. Tomado de Crosbie and Zenor [31].

INTERFAZ ENTRE EL RTI Y LOS FEDERADOS En la figura A.2 puede verse la interfaz entre el RTI y varios tipos de federados. Nótese que:

- Los Federados no hablan entre sí. Los Federados se conectan al RTI y se comunican con otros federados usando los servicios que proporciona el RTI
- Los Federados pueden ser de varios tipos.
- Cada Federado usa aquellos servicios del RTI apropiado para sus propósitos.
- Cada Federado tiene solo un punto de contacto con el RTI.

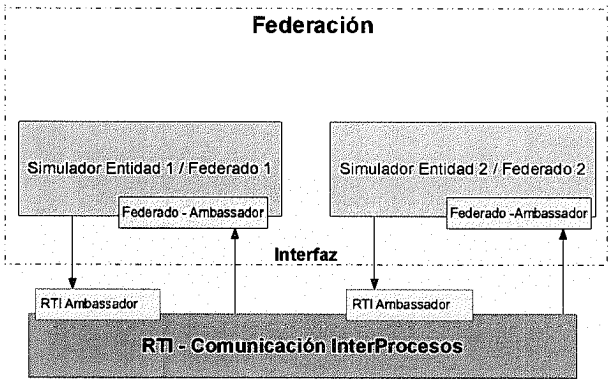


Figura A.2: Interfaz entre el RTI y los Federados. Tomado de Kuhl et al. [105].

A.1.1.6 *Componentes de un Federado.*

Los componentes de un federado pueden verse en la figura A.3.

- RTI Ambassador: las solicitudes de servicios en el código federado del RTI que llaman funciones de la clase RTIAmbassador, que están contenida en libRTI.
- Federado Ambassador: el RTI envía mensajes y respuestas al código federado por medio de llamadas a funciones implementadas en el federado. Estas funciones son conocidas como funciones de devolución de llamada *(callback)* y se implementan en el federado como una subclase de la clase FederateAmbassador. FederateAmbassador está contenida en LibRTI, y contiene funciones virtuales puras para cada *callback* posible. Estos rutinas son simplemente "marcadores de posición" *(place holders)* que no pueden ser llamados. El código federado debe crear una clase derivada de esta clase que contiene la implementación de cada una de estas funciones *callback*.

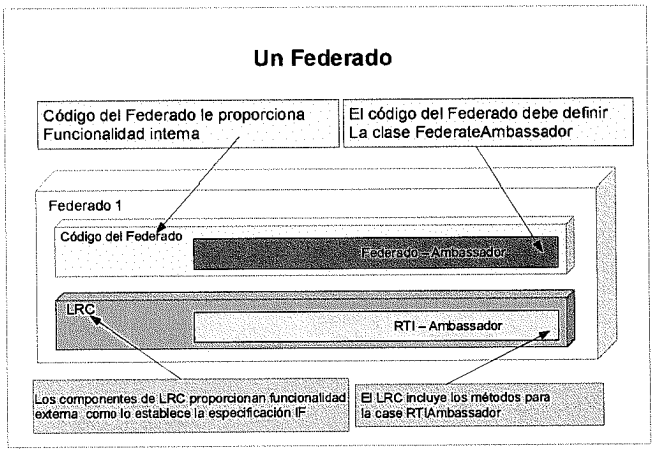


Figura A.3: Componentes de un Federado. Tomado de DoD [50].

A.1.1.7 *División de responsabilidades entre el RTI y los Federados.*

La figura A.4 ilustra la separación de responsabilidades para el código federado proporcionado por el usuario y el código proporcionado por libRTI.

- libRTI contiene la clase RTIAmbassador, que se utiliza para proporcionar servicios RTI a la federación. También contiene la clase abstracta FederateAmbassador que define todas los interfaces para las funciones *callback.*, pero que no se pueden utilizar hasta que los federados proporcionen las implementaciones de estas funciones.

- El código del federado contiene el código de la simulación el cual define y crea diversos objetos de simulación (o federados) y solicitudes de servicio al RTI mediante el uso de la clase RTIAmbassador. El código del federado también crea una clase derivada de la clase FederateAmbassador . Esta clase derivada contiene la implementaciones para cada una de las funciones *callback* de manera que el federado puede recibir mensajes y las respuestas originadas por la RTI.

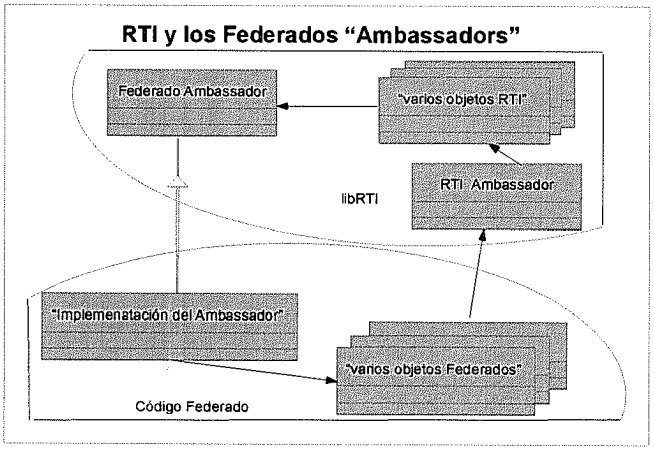


Figura A.4: Responsabilidad del código. Tomado de Crosbie and Zenor [31]

A.1.1.8 *Especificación de la Interfaz .*

La especificación de la interfaz es orientada a objetos. Esta dividida en grupo de servicios:

1. Administración de la Federación: maneja una federación de dos maneras: definiendo la ejecución de una federación en términos de existencia y membresía; y acompañando las operaciones de la federación (*checkpoints* y *restore*)
2. Administración de la Declaración: es la manera en que los federados declaran su intención de producir (*publish*) o consumir (*subscribe to*) datos.
3. Administración de Objetos: son usados para el intercambio de datos. A.1.2
4. Administración de Propiedad: implementa en HLA la noción de la responsabilidad para la simulación de una entidad. Permite que la responsabilidad sea compartida o transferida entre federados.
5. Administración del tiempo: el orden de los eventos es expresado en *tiempo lógico.*
6. Administración de la Distribución de Datos: controla la relación *producción-consumo* entre federados.

A.1.2 *OMT*

La especificación OMT es parte del estándard HLA. Proporciona un marco común para la comunicación entre simulaciones HLA. Möller

and Löfstrand. [119]. Facilita el diseño y la aplicación de herramientas comunes para el desarrollo de modelos de objetos HLA.

A.1.2.1 *Modelos de Objetos*

FOM:
La función primaria del FOM es especificar, en un formato común estándarizado, la naturaleza del intercambio de datos entre federados. Estos datos incluye una enumeración de todos las clases de objetos e interacciones pertinentes a la federación, además de la especificación de los atributos o parámetros que caracterizan esa clase.

- describe los objetos, tipos de datos e interacciones compartidas para toda la federación.
- Es el vocabulario de intercambio de datos a través de la RTI para la ejecución de una federación.
- Es uno por federación

SOM: Un SOM es una especificación de las capacidades intrínsicas que una simulación individual puede suministrar a una federación. El SOM proporciona una indicación de cuan adecuado es un sistema de simulación para la participación en una federación en particular.

- Es uno por federado
- Describe las características salientes de un federado
- Presenta Objetos e interacciones que pueden ser usadas externamente
- Se enfoca en las operaciones internas del federado

MOM: El MOM es una parte estandarizada y obligatoria de cada FOM que le permite al RTI describir y administrar el estado de la federación.

- Definición universal
- Identifica objetos e interacciones usadas para manejar una federación

COMPONENTES DEL OMT. Los siguientes componentes son los requeridos por la especificación OMT:

1. Tabla identificación del modelo de objeto
2. Tabla estructura de la clase de objetos
3. Tabla estructura de la clase de interacción

4. Tabla de atributos
5. Tabla de parámetros
6. Tabla de espacio de enrutamiento
7. Lexico FOM / SOM

La especificación OMT requiere que las federaciones y las simulaciones individuales usen todas estas siete estructuras componentes de OMT, aunque alguna de las tablas puedan estar vacias.

A.1.2.2 *Tabla identificación del modelo de objeto.*

El propósito del cuadro con la identificación del modelo de objeto es proporcionar la identificación clave de la información acerca de la federación o de los federados. En A.1 se muestra la estructura de esta tabla y ejemplo de su contenido.

Identificación del modelo de objeto	
Categoría	Identificación
Nombre	Mi_Federación
versión	1.0
fecha	1 de noviembre de 2012
propósito	proporciona modelo de Objeto para Mi_Federación
dominio de la aplicación	Operaciones para el suministro de Agua Potable en el Municipio Santos Marquina, Mérida

Cuadro A.1: Ejemplo de la Identificación del modelo de objeto.

A.1.2.3 *Tabla estructura de la clase de objetos*

El cuadro con la estructura de la clase de objetos contiene información acerca de la jerarquía clase-subclase de la clase de objetos. El cuadroA.2 muestra un ejemplo del contenido de esta tabla.

Estructura de la clase de objetos	
ObjectRoot	
Principal(S)	Acueducto(PS)
	Administración(PS)

Cuadro A.2: Ejemplo estructura de clase de objetos.

Cada clase de objeto está clasificado de acuerdo a su capacidad de publicación y suscripción. Las letras P,S y N indican *publicable*, *suscribible* y *ni-publicable* y *ni-suscribible*, respectivamente.

Publicable significa que un federado que usa el servicio de Clase Objeto Publicable, puede publicar la clase de objeto especificada. *Suscribible* significa que un federado es capaz de utilizar y reaccionar a la información del objeto de una clase dada.

A.1.2.4 *Tabla estructura de la clase de interacción*

Una interacción es una acción tomada por un objeto (o conjunto de objetos) en un federado que puede tener efectos en objetos pertenecientes a otros federados. Estas interacciones están definidas en el cuadro estructura de la clase de interacción, ver A.3.

Estructura Clase de Interacción	
Acueducto_ Administración_ Transacción(I)	Simulación(I)

Cuadro A.3: Ejemplo de estructura de clase de interacción.

Al igual que el cuadro Estructura de Clase de Objetos, Tabla estructura de clase de interacción identifica las capacidades federado/federación con respecto a sus clases interacción. Las categorás I,S,R y N indican que un federado es capáz de:

- I - iniciar y enviar una interacción dada
- S - detectar la interacción suscribiendose a ella y utilizando su información sin necesariamente ser capaz de efectuar modificaciones a los objetos afectados (S)
- R - reaccionar a la interacción mediante la suscripción a la misma y reaccionar a ella correctamente efectuando la cambios apropiados en cualquier atributo de propiedad de los objetos afectados (R)
- N - ni iniciar, ni detectar, ni reaccionar a esta clase de interacción .

A.1.2.5 *Tabla de atributos*

Cada clase de objetos en el dominio de la simulación se caracteriza por un conjunto fijo de tipos de atributos. Estos atributos son algunas partes de las variables de estado del objeto cuyos valores se pueden cambiar con el tiempo. Las actualizaciones de los valores de los atributos de la clase de objeto **HLA!** (HLA!) se publican a través del RTI y estos valores se le proporciona a los otros federados en un federación.

En el cuadro A.4 hay un ejemplo de una tabla de atributos. Para tipos de datos más complejos o tipos enumerados están disponibles tablas de formatos adicionales que pueden consultarse en IEEE [80].

Tabla de Atributos

Objeto	Atributo	Tipo-Dato	Cardinalidad	Unidad	Resolución	Precisión	Condición-Precisión	Tipo Actualizaciones	Condición Actualizaciones	T/A	U/R	espacio enrutamiento
Acueducto	nodo	string	1	N/A	N/A	N/A	N/A	periodico	con la simulación	T/A	U/R	N/A
	dotación	float	1	lts/seg	1	perfecto	siempre	periodico	con la simulación	T/A	U/R	N/A
	presión	float	1	mts	1	perfecto	siempre	periodico	con la simulación	T/A	U/R	N/A

Cuadro A.4: Tabla de Atributos.

A.1.2.6 *Tabla de parámetros*

Para cada clase de interacción identificada en el cuadro estructura de la clase de interacción, el conjunto de parámetros asociados con esa clase interacción se especifica en el cuadro de parámetros.

La entrada a las tablas consiste de las clases de interacción relevantes, los nombres de parámetros asociados con las clases de interacción e información especifica del tipo de datos, unidad, resolución, precisión, condición de precisión y espacio de enrutamiento. El cuadro A.5 muestra un ejemplo.

Tabla de Parámetros

Interacción	Parámetros	Tipo-Dato	Cardinalidad	Unidad	Resolución	Precisión	Condición-Precisión	espacio enrutamiento
Simul(I)	EndSimulation	boolean	1	N/A	N/A	N/A	N/A	N/A

Cuadro A.5: Tabla de Atributos.

A.1.2.7 *Tabla de espacio de enrutamiento*

Las federaciones usan los servicios Declaration Management (DM) para permitir, pero también para limitar, el flujo de atributos de objetos y datos de interacción entre los federados, y los Data Distribution Management (DDM) para limitar el flujo de datos entre los federados. En estos casos se usa un espacio de enrutamiento. Los

espacios de enrutamiento se especifican en el cuadro de espacio de enrutamiento que pueden consultarse en IEEE [80]

A.1.2.8 *Léxico SOM/FOM.*

El Léxico SOM/FOM se incluye en el OMT para suministrar información semántica necesaria para entender la información contenida en las tablas OMT. El cuadro A.6 muestra un ejemplo

Definición Clase de Objetos	
Término	Definición
nodoA	nombre del nodo
dotaciónA	dotación del nodo

Definición Clase de Interacción	
Término	Definición
Simulation	Inicio/fin de la simulación

Cuadro A.6: Léxico SOM/FOM.

A.1.2.9 *Formatos OMT DIFy archivos FED.*

El formato OMT de tipo DIF, Data Interchange Format (DIF) es una formato estándard de intercambio de achivos usado para guardar y transferir HLA SOMs y FOMs entre constructores SOM/FOM por medio de archivos de tipo FED.

A.1.3 *Actualizaciones e Interacciones*

RTI proporciona un conjunto de servicios con el objeto de manipular las actualizaciones e Interacciones de datos. Entre estos servicios:

- *Actualizar valores de atributos (Update Attribute Values):* con este servicio el RTI comunica los cambios en el estado de un objeto a otros federados en la federación mediante el envío de un mensaje de actualización a la federación.
- *reflejar valores de atributos (Reflect Attribute Values):* cuando el RTI llama esta rutina, otros federados recibirán los valores de los atributo actualizado
- *Activar/Desactivar Switch Aviso Atributo Relevante (Enable/Disable Attribute Relevance Advisory Switch):* mecanismo de control para evitar el envío de mensajes innecesarios.
  - *encender/apagar (Turn Updates On/Off):* al encender/apagar las llamadas del RTI los federados serán notificados si hay o no federados que puedan recibir las actualizaciones.

  - *apagar actualizaciones (Turn Updates Off):* El envío de actualizaciones será suspendido

- *Activar/Desactivar Switch Aviso Atributo Alcance (Enable/Disable Attribute Scope Advisory Switch):* los federados que estén interesado en recibir actualizaciones serán notificados si hay o no cualquier federado capaz de enviar la actualización de atributos

  - *atributos Fuera de Alcance (Attributes Out of Scope):* Si ningún otro federado es actualmente capaz de enviar la actualización de atributos
  - *atributos en Alcance (Attributes in Scope):* Si un *Attributes Out of Scope* fue llamado por el RTI, significa que la rutina *Reflect Attribute Values* no será llamada hasta que se llame a *Attributes in Scope*

- *Enviar Interacción (Send Interaction):* usando este servicio el RTI, si una simulación genera eventos, los cuales causan un inmediato cambio de estado, se puede enviar un mensaje que contiene información sobre el evento a otros federados en la Federación

- *Recibe interacción (Receive Interaction):* otros federados puede recibir este mensaje cuando el RTI llama esta rutina.

- *Activar/Desactivar Switch Asesor Relevancia interacción (Enable/Disable Interaction Relevance Advisory Switch) :* mecanismo para controlar el tráfico de mensajes innecesario para las interacciones

- *Activar Interacciones On/Off (Turn Interactions On/Off):* El RTI indica si otros federados son capaces de recibir esta interacción.

A.1.4 *Mecanismo de Publicación y Suscripción*

Los mecanismos de *publicación* y *suscripción* son utilizado por el HLA para gestionar la distribución de los mensajes entre los federados en una federación.

Los Federados definen que datos han de ser publicados usando las siguientes rutinas :

- *Publicar clase objeto (Publish Object Class)*
- *Publicar clase interacción (Publish Interaction Class)*

Los Federados declaran a la federación su interés en recibir mensajes usando las siguientes rutinas :

- *Suscribirse clase atributo (Subscribe Object Class Attribute)*
- *Suscribirse clase interacción (Subscribe Interaction Class)*

Los Federados envian al RTI mensajes usando las llamadas de "publicación":

- *Registrar instancia de objeto (Register Object Instance):* federado envia información al RTI de la instancia de un objeto.
- *Descubre Instancia de Objeto (Discover Object Instance):* federado recibe información de la existencia de nuevos objetos, si está suscrito a esta clase de objeto

Los Federados pueden manifestar al RTI interés en que administre el envio de mensajes innecesarios con:

- *Habilite Switch Relevancia Atributo (Enable Attribute Relevance Advisory Switch)*
- *Habilite Switch Relevancia Interaccione (Enable Interaction Relevance Advisory Switch)*

Cuando un federado se suscribe a datos publicado por otro federado, el RTI enviará al federado las rutinas:

- *Encienda actualizaciones para instancias de objetos (Turn Updates On For Object Instance)*
- *Encienda Interacciones (Turn Interactions On)*

El RTI llama a estas rutinas para controlar las llamadas de registro de objetos, si previamente el federado ha llamado *Enable Class Relevance Advisory Switch.*:

- *inicio/parada de registro de Clase de objetos (Start/Stop Registration for Object Class)*

A.1.5 *TSO y recepción de actualizaciones ordenadas*

Los mensajes que deben ser entregados en el orden de tiempo correcto, se dice que se entregarán por orden cronológico estampado (TSO). El RTI garantiza la entrega de mensajes bajo dos modalidaddes:

- TSO :
  - Mensajes se entregan a los federados en el orden TSO.
  - El RTI garantiza que ningún mensaje del pasado será recibido

Receive Order (RO):

- mensajes se entregan al federado en el orden que se reciben

.

A.1.6 *Regulación del Tiempo y restricción del tiempo en los federados*

- Para recibir un mensaje TSO en el orden de tiempo marcado, federado deben declararse como de restringuido en el tiempo *(Time Constrained)*
- Para enviar mensaje TSO, federado debe declararse como regulado en el tiempo (*Time Regulating*)
- Por defecto, federados no son ni restringuido en el tiempo ni regulado en el tiempo.
- *Enable Time Constrained* para iniciar la restricción en el tiempo
- *Enable Time Regulation* , para iniciar la regulación en el tiempo

A.1.6.1 *Lookahead, Lower Bound Time Stamp (LBTS), Tiempo Lógico, Tiempo Lógico Efectivo*

- *Lookahead*: para cada federado restringuido en tiempo, es la mínima marca de tiempo que será usada para cualquier mensaje futuro
- LBTS: El máximo tiempo que un federado restringuido en el tiempo, puede avanzar. Restringuido en el tiempo en el sentido que el avance del tiempo lógico por el federado esta restringuido por el manejo del tiempo al no avanzar más alla que el último tiempo de un mensaje sellado en tiempo que puede posiblemente ser recibido de otro federado
- Tiempo Lógico: es el valor local del tiempo para un federado. Se asume como cero cuando el federado se une a la simulación
- Tiempo Lógico Efectivo: es igual al tiempo lógico más el *lookahead*

A.1.6.2 *Avance del tiempo y Recepción de mensajes*

Para el avance en el tiempo, HLA proporciona las siguientes rutinas:

- Simulación continua:
  - *Requiere Avance en Tiempo (Time Advance Request)*: el federado llama al RTI para aumentar su tiempo lógico .
  - *Avance en Tiempo (Time Advance Grant):* el RTI notificará cuando avanzar en el tiempo
- Simulación Discreta:
  - *Requiere Próximo Evento (Next Event Request):* avance del tiempo hasta el tiempo del evento próximo

- *Avance en Tiempo (Time Advance Grant):* el RTI notificará cuando avanzar en el tiempo

Los mensajes son solo recibidos cuando hay un *estado de avance en el tiempo*:

- Un federado se coloca en *estado de avance en el tiempo (time-advancing state)* por:
  - *Requiere Avance en Tiempo (Time Advance Request)* o
  - *Requiere Próximo Evento (Next Event Request)*
- Para habilitar la recepción de mensajes RO en otros tiempos:
  - *Habilitar distribución asíncrona (Enable Asynchronous Delivery)*

A.1.7 *Declaración y Administración de Objetos.*

Por medio de varias ilustraciones, se mostrará los métodos que se usan para la declaración y administración de objetos en las simulaciones que usan HLA.

A.1.7.1 *Publicación y suscripción de Objetos.*

La figura A.5 ilustra las responsabilidades de los federados en la publicación y suscripción de objetos.

Para expresar interés en publicar o suscribirse a una clase de objetos, se requieren los pasos siguientes, por cada clase de objeto que se publicará:

1. Obtener el identificador (*handle*) de la clase de objeto actual.
2. Crear una asignación con *AttributeHandleSet* utilizando el método create() de la clase *AttributeHandleSetFactory*
3. Para cada atributo de la federación que se va a publicar:
   a) Obtener el identificador (*handle*) del atributo actual.
   b) Añadir el identificador (*handle*) al *AttributeHandleSet*
4. Publicar/Suscribirse al *AttributeHandleSet* para la clase de objeto.

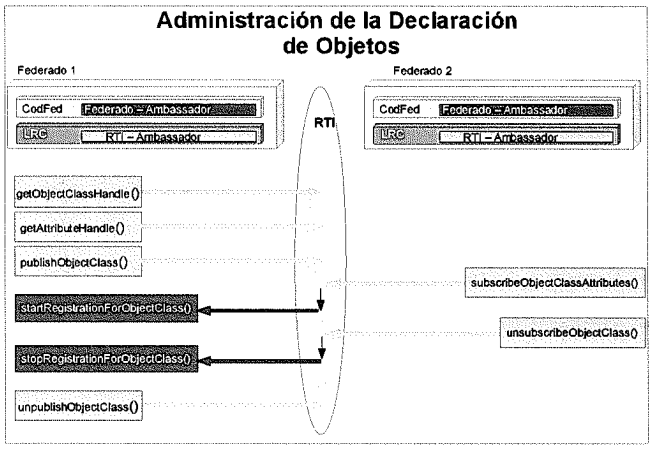


Figura A.5: Administración de Declaración de Objetos. Tomado de Crosbie and Zenor [31], DoD [50].

A.1.7.2 *Publicación y suscripción de Interacciones.*

El registro de la publicación y suscripción para las clases de interacción se ilustra en la figura A.6.

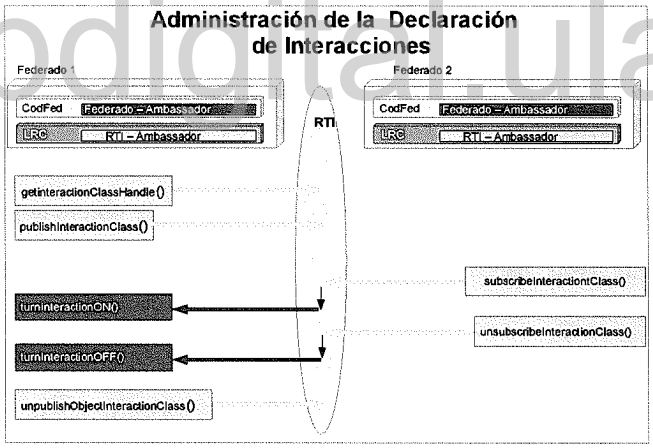


Figura A.6: Administración de Declaración de Interacciones. Tomado de Crosbie and Zenor [31], DoD [50].

A.1.7.3 *Registrar, descubrir y borrar instancias de Objetos.*

La figura A.7 ilustra las interacciones que se requieren para registrar y descubrir instancias de objetos. Las siguientes reglas deben cumplirse para que un federado pueda crear, publicar y descubrir objetos:

- Para crear un objeto, el federado deberá haber publicado esa clase de objeto usando el servicios de gestión de la declaración del RTI.
- El federado registra la nueva instancia del objeto
- Para descubrir el objeto, otros federados deben suscribirse a esa clase de objeto

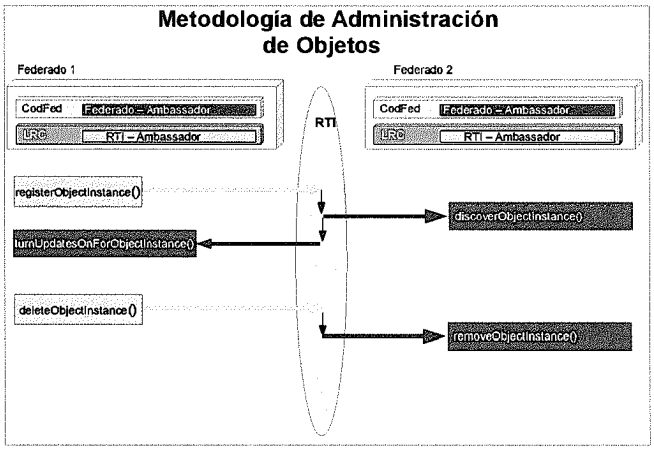


Figura A.7: Metodología para Administración de Objetos. Tomado de Crosbie and Zenor [31], DoD [50].

A.1.7.4 *Actualizando y reflejando atributos de los Objetos.*

La figura A.8 muestra las interacciones que se requieren para descubrir y reflejar las actualizaciones de los valores de los atributos en instancias de objetos externas al federado.

Para expresaractualizar uno o más atributos asociados con una instancia de objetos registrada, se requieren los pasos siguientes, por cada tipo de atributo:

1. Crear una asignación con *AttributeHandleValuePairSet* utilizando el método create() de la clase *AttributeSetFactory* . Este método identifica el conjunto de atributos y sus valores
2. Para actualizar el atributo:
   a) Obtener el identificador (*ObjectHandle*) del atributo actual.
   b) Actualizar el atributo con el método *updateAttributeValues*. Este método requiere el identificador (*ObjectHandle*), un *AttributeHandleValuePairSet* y una cadena de carateres descriptiva.

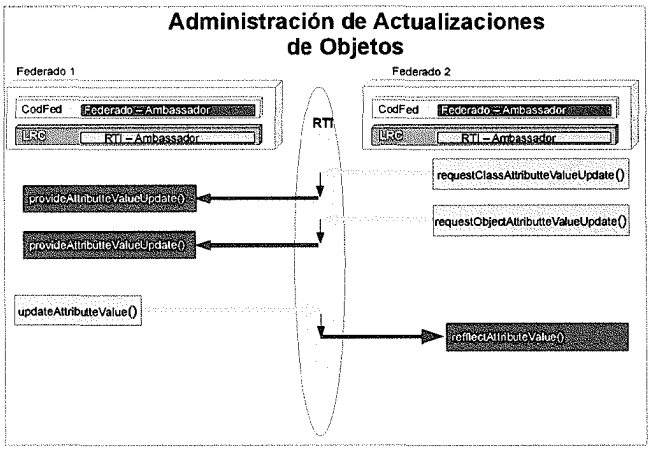


Figura A.8: Administración de Actualizaciones de Objetos. Tomado de Crosbie and Zenor [31], DoD [50].

A.1.7.5 *Intercambio de Interacciones*

Las actualización de interacciones es administrada de manera similar que las actualizaciones de los atributos. La figura A.9 muestra los métodos involucrados en la generación y consumo de interacciones. Se debe recordar que los objetos persisten en el tiempo, las interacciones no. Cada interacción es construida, enviada y olvidada.

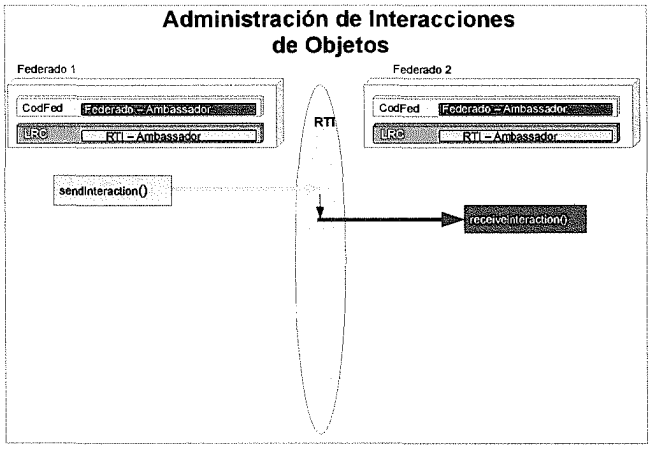


Figura A.9: Administración de Interacciones de Objetos. Tomado de Crosbie and Zenor [31], DoD [50].

A.1.8 *Administración del Tiempo en Simulaciones HLA*

HLA es permisivo en como un federado maneja el tiempo. Más aún, HLA proporciona unos esquemas para el manejo del tiempo, que se nombran a continuación:

1. No hay administración del tiempo, en la cual cada federado avanza en el tiempo a su propio paso.
2. Sincronización Conservativa, en la cual los federado avanzan en el tiempo solo cuando se garantiza que no se recibirá ningún evento del pasado.
3. Sincronización Optimista, en la cual un federado es libre de avanzar en su tiempo lógico, pero está preparado para regresar ese tiempo lógico si se recibe algún evento del pasado.
4. Búsqueda Activa, en la cual los federados proceden a través de periodos , y durante ese tiempo, ellos intercambian mensajes hasta que están de acuerdo en avanzar juntos al próximo tiempo lógico.

En todos los casos el tiempo lógico de un federado tiene restricciones:

1. Tiene un valor inicial.
2. Su valor no está atado a ninguna unidad del sistema.
3. Está bien ordenado. El RTI puede determinar cual de los dos tiempos es mayor.
4. Es siempre mayor que el tiempo inicial.
5. El tiempo es discreto.
6. Este puede asumir un valor de infinito positivo que es siempre mayor que cualquier otro valor.

Cada federado determina su grado de participación en el proceso de administración del tiempo. Los federados pueden escoger:

- No participar en la administración del tiempo. No están ni regulados en el tiempo ni restringuidos en el tiempo.
- Regulados en el tiempo, en cuyo caso pueden ser capaces de generar eventos TSO.
- Restringuidos en el tiempo, en cuyo caso pueden ser capaces de recibir eventos TSO.
- Regulados en el tiempo y Restringuidos en el tiempo, en cuyo caso pueden ser capaces de generar/recibir eventos TSO.

Para federados restringuidos en el tiempo y que operan con sincronización conservativa:

1. Los eventos TSO son distribuidos a los federados en el orden de la marca de tiempo, independiente al orden en el cual fueron enviados los eventos originales.
2. Ningún evento será distribuido a los federados con una marca de tiempo menor que el tiempo lógico actual del federado.

A.1.9 *Estándard IEEE 1516*

HLA también esta definida bajo el estandard IEEE 1516, IEEE1516 [81]. Estos son los documentos que pertenecen a este estándar:

- IEEE 1516–2010 – Estándard para Modelado y Simulación de Arquitectura de Alto Nivel – Marco y Reglas
- IEEE 1516.1–2010 – Estándard para Modelado y Simulación de Arquitectura de Alto Nivel – Especificación de la Interfaz de los Federados
- IEEE 1516.2-2010 – Estándard para Modelado y Simulación de Arquitectura de Alto Nivel – Especificación OMT.
- IEEE 1516.3-2003 – Practica Recomendada para Desarrollo de Federaciones y Ejecución de Procesos en la Arquitectura de Alto Nivel – FEDEP
- IEEE 1516.4-2007 – Practica Recomendada para Verificación, Validación, and Acreditación de una Federación como una superposición al Desarrollo de Federaciones y Ejecución de Procesos en HLA

A.1.10 *Estándard IEEE 1516-Evolved*

El estandard IEEE 1516 ha sido revisado por el Grupo de Desarrollo de Productos Simulation Interoperability Standards Organization (SISO) HLA-*Evolved* y fue aprobado el 25 de Marzo de 2010 por el equipo de estándares de la IEEE. El estandard 1516-2010 Evolved [59] incluye la interpretación tradicional del estándard del DoD [48], y del Evolved Dynamic Link Compatible (EDLC) API, que es una versión extendida del estandar SISO **DLC!** (**DLC!**) API .

IEEE 1516-Evolved presenta las siguientes mejoras:

- Soporte XML extendido para FOM/SOM, tales como esquemas y extensibilidad
- Servicio de soporte a tolerancia de fallas

- Soporte a WSDL
- FOM modulares
- Reducción de las tasas de actualizaciones
- Soporte extendido para transportes adicionales
- Representación del tiempo estandarizado

Además, debo destacar la documentación para dar soporte a los procesos de migraciones de HLA 1.3 hasta IEEE 1516 e IEEE 1516-Evolved. Entre ellas: Möller and Löf. [118], Möller and Dahlin. [117], Möller and Olsson [120].

A.2 CONSTRUCCIÓN DE UN FEDERADO PARA EL ESCENARIO CLIMÁTICO.

A la federación *Mi Federación* se le agregará el federado *Mi Consultor*. El federado *Mi Consultor* contendria al simulador del escenario climático que generaría las variables de entrada para el proceso de simulación del acueducto "La Ceibita" y al agente Consultor que gestionaría este escenario climático. La simulación del escenario climático se construirá con el simulador GALATEA. A continuación se extiende el modelo conceptual desarrollado para la federación *Mi federación* con el objeto de tener la infraestructura necesaria que permita la incorporación del federado *Mi Consultor*.

A.2.0.1 *Tabla estructura de clase de objetos.*

El cuadro con la estructura de clase de objetos se agrega la subclase Escenario como se observa en el cuadro A.7. La subclase Escenario se ha catalogado como publicable (P), significa que el federado usa el servicio de publicación de objetos (Publish Object Class) y puede publicar la clase de objeto especificada.

Estructura de la clase de objetos	
ObjectRoot	
Principal(S)	Acueducto(P)
	Escenario(P)

Cuadro A.7: Estructura de clase de objetos.

A.2.0.2 *Tabla estructura de clase de interacción.*

El cuadro A.8 con la estructura de la clase interacción se ha agregado la interacción que se ha llamado Acueducto_Consultor_Transaccion(I), con la clase Config(I). Con esta interacción el federado *Mi Consultor* le comunica al Federado *Mi Acueducto* que debe ejecutar una nueva simulación con los escenarios construidos por el usuario.

Estructura Clase de Interacción	
Acueducto_ Administración_ Transacción(I)	Simul(I)
Acueducto_ Consultor_ Transacción(I)	Config(I)

Cuadro A.8: Estructura de clase de interacción.

A.2.0.3 *Tabla de atributos.*

Los atributos para el objeto Escenario se muestran en el cuadro de atributos de A.9. El objeto Escenario tiene tres atributos: el nodo, la dotación y el tiempo de la simulación. Los atributos nodo y dotación ya están definidos en el objeto Acueducto por lo que no es necesario agregarlos. Entonces, el objeto Escenario tiene como atributo el *tiempo_dot*, una variable que indica el tiempo de duración de la simulación.

Tabla de Atributos

Objeto	Atributo	Tipo-Dato	Cardinalidad	Unidad	Resolución	Precisión	Condición-Precisión	Tipo Actualizaciones	Condición Actualizaciones	T/A	U/R	espacio enrutamiento
Acueducto	nodo	string	1	N/A	N/A	N/A	N/A	periodico	con la simulación	T/A	U/R	N/A
	dotación	float	1	lts/seg	1	perfecto	siempre	periodico	con la simulación	T/A	U/R	N/A
	presión	float	1	mts	1	perfecto	siempre	periodico	con la simulación	T/A	U/R	N/A
Escenario	tiempo_dot	date	1	seg	1	perfecto	siempre	periodico	Agente Consultor	T/A	U/R	N/A

Cuadro A.9: Tabla de Atributos.

A.2.0.4 *Tabla de parámetros.*

En el cuadro A.10 se especifica el parámetro asociado con la clase interacción definida anteriormente en el cuadro de estructura de clase de interacción.

Tabla de Parámetros									
Interacción	Parámetros	Tipo-Dato	Cardinalidad	Unidad	Resolución	Precisión	Condición-Precisión	espacio enumeramiento
Simul(I)	EndSimulation	boolean	1	N/A	N/A	N/A	N/A	N/A
Config(I)	Configuration	boolean	1	N/A	N/A	N/A	N/A	N/A

Cuadro A.10: Tabla de Parámetros.

A.2.0.5 *Tabla de Léxico.*

El cuadro de léxico SOM/FOM, tablaA.11, se le añaden los atributos del federado Mi Consultor.

Definición Clase de Objetos	
Término	Definición
nodo	nombre del nodo
dotación	dotación promedio del nodo
presión	presión en el nodo

Definición Clase de Interacción	
Término	Definición
EndSimulation	Inicio/fin de la simulación
Configuration	Envia nueva configuración

Cuadro A.11: Léxico SOM/FOM.

A.2.0.6 *Datos para la ejecución de la federación.*

Mi federación tiene dos clases definidas en su FOM: Acueducto y Escenario. El diagrama de clases de la federación se observa en la figura A.10. La clase Escenario tiene el atributo *tiempo_dot* y hereda de la clase Acueducto los atributos *nodo* y *dotación*.

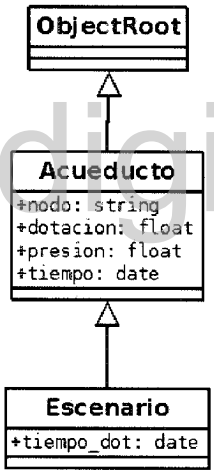


Figura A.10: Clase de Objetos de Mi_Federación

El diagrama de interacción de la federación se observa en la figura A.11, con la clase de interacción definida para Escenario.

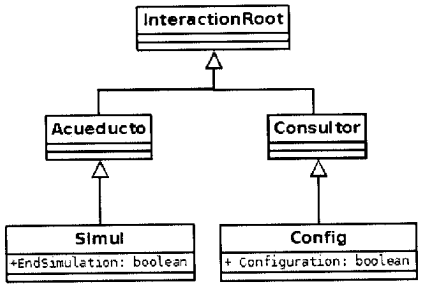


Figura A.11: Clase Interacción de Mi_Federación

El archivo de tipo FED modificado para la federación *Mi Federación* se muestra en la figura A.1.

```
(FED
  (Federation Portico-Test)
  (FEDversion v1.3)
  (spaces
    (space TestSpace
      (dimension TestDimension)
    )
  )
  (objects
    (class ObjectRoot
      (attribute privilegeToDelete reliable timestamp)
      (class RTIprivate)
      (class Principal
          (class Acueducto
                          (attribute nodo reliable timestamp
                              TestSpace)
                                 (attribute dotacion reliable
                              timestamp TestSpace)
                    (attribute presion reliable timestamp
                        TestSpace)
                    (attribute tiempo reliable timestamp
                        TestSpace)
                                 (class Escenario
                                         (attribute tiempo_dot
                                             reliable timestamp
                                             TestSpace)


                                 )  ;; end class Escenario
                )  ;; end class Acueducto
        )    ;; end class Principal
    )       ;; end class ObjectRoot
  )         ;; end class Object
  (interactions
    (class InteractionRoot reliable timestamp
      (class RTIprivate reliable timestamp)
      (class Inicio reliable timestamp TestSpace
            (class Simul reliable timestamp
               (parameter EndSimulation)
```

```
        )    ;; end  class Simul
            (class Config reliable timestamp ;;
               (parameter Configuration)
        )    ;; end  class Config
      )      ;; end class Inicio
    )        ;; end class InteractionRoot
  )          ;; end class Interaction
)           ;; end FED
```

Programa A.1: Archivo suministro.fed modificado

A.2.0.7 *Diseño y desarrollo de la federación.*

A la federación *Mi Federación* se añade un nuevo federado: *Mi Consultor*. El federado *Mi Consultor* publica los datos asociados al nodo que representa el Embalse. Estos datos son: el nombre del nodo, la dotación en el nodo y el tiempo relacionado con la dotación.

El federado *Mi Acueducto* usa los datos publicado por el federado *Mi Consultor*, para modificar el modelo de simulación del acueducto, *Ceibita.inp*, que se utiliza para ejecuta la simulación.

A.2.1 *Construcción del agente consultor.*

El agente consultor tiene como función la gestión del escenario climático construido. La función del agente consultor se extenderá a la asistencia al usuario en el proceso de construir distintos escenarios de amenazas. Estos escenarios deben poder construirse con la búsqueda de distintos datos geográficos y metereológicos disponibles W3C [161] en la web y la incorporación de estos datos a la aplicación. Para lograr esta capacidad en el agente, es necesario que pueda extraer información de fuentes de datos estructurados y no-estructurados. Para ello se están evaluando un conjunto de herramientas, Zaccak [170], Liu et al. [108], Ashish and Knoblock [4], Hammer et al. [78] para su incorporación.

A.3 EVALUACIÓN DE SOFTWARE PARA HLA.

Existen varios productos de software para implementar HLA. Entre los software que implementan HLA, se evaluaron los siguientes: CERTI, CERTI [24] con licenciamiento de tipo GNU, GNU is Not Unix (GNU) General Public License (GPL), PoRTIco PoRTIco [132], con licencia Common Development and Distribution License (CDDL) y OPEN-HLA Open HLA [126], con licencia de tipo APACHE.

El cuadro (A.12) muestra los criterios que se usaron para la evaluación. En esta comparación de software se puede destacar lo siguiente:

- Debido a que CERTI no tiene la totalidad de los estándares instalados, se descartó.
- poRTIco no soporta el estándard IEEE 1516e, pero ya se ha anunciado en la lista de usuario, el desarrollo y la fase de prueba del estándard IEEE 1516e
- PoRTIco tiene documentación para la instalación y desarrollo de federaciones y federados en HLA 1.3
- Con PoRTIco no es necesario ejecutar un RTI central. Todos los federados operan de una manera *punto-a-punto (peer-to-peer).*

Debido a esta última característica, se decidio usar PoRTIco porque esto representa una simplificación importante para la implementación de este proyecto en particular.

criterios	PoRTIco	OPEN-HLA	CERTI
Estándard	HLA 1.3 IEEE 1516	HLA 1.3 IEEE 1516 IEEE 1516e	HLA 1.3 parcial IEEE 1516 parcial
Lenguaje	C++, JAVA	JAVA	C++, Fortran90, JAVA, Matlab, Python
Documentación:			
- HLA 1.3	si	no	si
- IEEE1516	si	no	si
- IEEE 1516e	-	no	-
- Listas de usuarios	si	si	si
Soporte a Usuarios	si	no	si
Forma de Ejecución del RTI	p2p	iniciar RTI	iniciar RTI

Cuadro A.12: Comparación software para HLA

B

EVALUACIÓN DE SOFTWARE PARA SISTEMA DE INFORMACIÓN GEOGRÁFICA.

B.1 EVALUACIÓN DE SOFTWARE PARA SIG.

El objetivo de este reporte es la evaluación de software para SIG con el objeto de integrarlos en la plataforma de simulación GALATEA.

Esta evaluación fue realizada a mediados del año 2010, con los siguientes softwares: GRASS 6.4 [75], gvSIG 1.9 [77] y QUANTUM GEOGRAPHIC INFORMATION SYSTEM (QGIS) 1.4.0[135]. La razón de la selección de estos software para su evaluación fue que todos ellos poseen licencia de tipo GNU GPL.

Los parámetros de evaluación se muestran a continuación en el cuadro B.1.

Los puntos se asignaron de acuerdo a las siguientes consideraciones:

- software cumple totalmente con el criterio: 10
- software cumple parcialmente con el criterio: 5
- software no cumple con el criterio: 0

Los siguientes puntos resaltan en la evaluación:

- El programa de desarrollo es diferente (C,C++, TCL) vs JAVA
- Interfaz Gráfica
  - QGISposee una versátil interfaz gráfica para manipulación de datos vectoriales.
  - GRASS utiliza la interfaz gráfica vectorial de QGIS.
  - GRASSproporciona varios tipos de de interfaz gráfica.
  - La interfaz gráfica de gvSIG es una sola y resulta muy agradable visualmente por su sencillez y facilidad de uso.
- Formatos de archivos
  - GRASS usa los módulos de Geospatial Data Abstraction Library (GDAL)[1] para importar archivos raster y vectorial de diferentes formatos.
  - gvSIG maneja un menor número de formatos de archivos vectoriales y raster.
- GRASS posee herramienta para visualización 3D. gvSIG lo proporciona como un módulo adicional, desarrollado por la comunidad de usuarios de gvSIG.
- GRASS tiene algoritmos para análisis de modelo en 3D, redes y modelos de elevación digital. gvSIG se apoya en SEXTANTE [125] para algoritmos de redes y modelos de elevacion digital.

1 está en: http://www.gdal.org/

- Servicios OGC
  - gvSIG tiene implementado servicios de acceso remoto OGC: WMS, Web Feature Service (WFS) y Web Coverage Service (WCS).
  - GRASS y QGIS tienen implementado el servicio WMS
- Servicios móviles
  - GRASS y gvSIG poseen versiones del software para PDA.
  - gvSIG proporciona una versión gvSIG MINI como visor de mapas y otros servicios en teléfonos celulares.

El resultado de la evaluación ha proporcionado valores muy cercanos para gvSIG y GRASS.

Sin embargo, tres razones adicionales, además de su atractiva interfaz gráfica, nos ha inclinado por el uso de gvSIG :

1. La posibilidad de desarrollo de aplicaciones para equipos móviles.
2. La relación de la asociación gvSIG con los usuarios iberoamericanos: existe una creciente comunidad de usuarios que abarca paises de la región latinoamericana y del caribe, con diversos encuentros anuales para la divulgación de información científica.
3. Es un software de reciente aparición pero ha crecido rapidamente, indicando un grupo de usuarios comprometidos con el desarrollo y mejora del software.

B.1.1 *Evaluación revisitada:*

En el momento de la redacción de este tesis, junio 2013 , y revisando esta evaluación, han sucedido grandes cambios en el área de los SIG. Estos cambios apuntan a una integración de software para aprovechar los beneficios de los programas que se integran.

Por ejemplo,gvSIG ha incorporado los módulos de GRASS aumentando su versatilidad en el manejo de diferente tipos de formatos; QGIS incorpora a GRASS como plug-in aportando entre otras ventajas, los módulos de manejo raster de los que QGIS carecía antes. Además han surgidos nuevos SIG . En `http://es.wikipedia.org/wiki/Sistema_de_Informaci%C3%B3n_Geogr%C3%A1fica` existe un cuadro comparativo muy útil.

				Puntos		
Criterios	GRASS	gvSIG	OGIS	GRASS	gvSIG	OGIS
Version	6.4	1.9	1.4.0	-	-	-
programa de desarrollo	C,C++,TCL	java	C++	-	-	-
tiempo desarrollo	1985	2004	Mayo 2002	-	-	-
comunidades	variada	IberoAmericana	variada	-	-	-
Interfaz Grafica Raster	si	si	si	10	10	10
archivos Raster	GDAL library	algunos	no	10	5	0
Interfaz Grafica Vectorial	con qGIS	si	si	10	10	10
archivos Vectoriales	formatos de OGR	SHP, GML, DXF, DWG, DGN	formatos de OGR	10	5	10
atributos vectoriales	o sql-DBMS	PostGIS, MySQL,	Spatialite- Postgres	10	10	10
		ArcSDE, Oracle, JDBC, CSV.				
Visualizacion 3-D	si	gvSIG-3D.	no	10	10	0
Mod. elevacion Dig	si	SEXTANTE	no	10	10	0
Analisis de redes	si	SEXTANTE	no	10	5	0
Digitalizacion	si	no	no	10	0	0
Script	si	si	si	10	10	10
Servicios OGC: WMS WFS, WCS	parcial	si	parcial	5	10	5
PDA	IPAQ o Zaurus	cualquiera con SO Windows M		10	10	0
PDA - SO	Linux	Windows Mobile		5	10	0
PDA- Acceso Remoto		OGC (WMS)		0	10	0
Teléfonos	no	si	no	0	10	0
Integracion con Galatea	si	no	no	10	0	0
				130	125	55

Cuadro B.1: Comparacion de software para SIG

C

MODELOS DE SIMULACIÓN

C.1 MODELO DE SIMULACIÓN ACUEDUCTO LA CEIBITA

```
[TITLE]


[JUNCTIONS]
;ID              Elev            Demand          Pattern
 2               2030            0                               ;
 10              2055            -5                              ;

[RESERVOIRS]
;ID              Head            Pattern
 11              2061.48                         ;Embalse

[TANKS]
;ID              Elevation       InitLevel       MinLevel
     MaxLevel    Diameter        MinVol          VolCurve
 3               2028            0.4             0.4             0.8
     2.85        0                               ;T1
 4               2020            0.5             0.5             1.8
     1.58        0                               ;T2
 5               2010            0.1             0.1             0.5
     1.13        0                               ;T3
 6               2009            0.5             0.5             1
     1.45        0                               ;T4
 7               2008.5          0.1             0.1             0.6
     2.25        0                               ;T5
 8               2008            0.2             0.2             1
     1.5         0                               ;T6
 9               2007.95         2               1               2.7
     7           0                               ;Estanque de
Almacenamiento

[PIPES]
;ID              Node1           Node2           Length
     Diameter    Roughness       MinorLoss       Status
 2               2               3               8.9             100
     0.1         0               Open       ;
 3               3               4               75              75
     0.1         0               Open       ;
 4               4               5               800             75
     0.1         0               Open       ;
 5               5               6               2.7             75
     0.1         0               Open       ;
 6               6               7               5               75
     0.1         0               Open       ;
 7               7               8               1               75
     0.1         0               Open       ;
 8               8               9               1               110
     0.1         0               CV         ;
 9               10              2               1175            110
     0.1         0               Open       ;
 1               11              10              25              75
     0.1         0               Open       ;

[PUMPS]
```

```
;ID                  Node1           Node2           Parameters

[VALVES]
;ID                  Node1           Node2           Diameter        Type
     Setting         MinorLoss

[TAGS]
 NODE      11              Embalse
 NODE      3               T1
 NODE      4               T2
 NODE      5               T3
 NODE      6               T4
 NODE      7               T5
 NODE      8               T6

[DEMANDS]
;Junction            Demand          Pattern         Category

[STATUS]
;ID                  Status/Setting

[PATTERNS]
;ID                  Multipliers

[CURVES]
;ID                  X-Value         Y-Value

[CONTROLS]

[RULES]


[ENERGY]
 Global Efficiency   75
 Global Price        0
 Demand Charge       0

[EMITTERS]
;Junction            Coefficient

[QUALITY]
;Node                InitQual

[SOURCES]
;Node                Type            Quality         Pattern

[REACTIONS]
;Type      Pipe/Tank       Coefficient


[REACTIONS]
```

```
 Order Bulk              1
 Order Tank              1
 Order Wall              1
 Global Bulk             0
 Global Wall             0
 Limiting Potential      0
 Roughness Correlation   0

[MIXING]
;Tank                   Model

[TIMES]
 Duration               168
 Hydraulic Timestep     24:00
 Quality Timestep       0:24
 Pattern Timestep       24:00
 Pattern Start          24:00
 Report Timestep        24:00
 Report Start           0:00
 Start ClockTime        12 am
 Statistic              None

[REPORT]
 Status                 No
 Summary                No
 Page                   0

[OPTIONS]
 Units                  LPS
 Headloss               D-W
 Specific Gravity       1
 Viscosity              1
 Trials                 40
 Accuracy               0.001
 CHECKFREQ              2
 MAXCHECK               10
 DAMPLIMIT              0
 Unbalanced             Stop
 Pattern                1
 Demand Multiplier      1.0
 Emitter Exponent       0.5
 Quality                None mg/L
 Diffusivity            1
 Tolerance              0.01

[COORDINATES]
;Node                   X-Coord         Y-Coord
 2                      1252.35         7250.47
 10                     -1308.85        8267.42
 11                     -2737.84        9302.33
 3                      2702.45         6967.98
 4                      4755.18         6459.51
 5                      6129.94         6158.19
 6                      7467.04         5913.37
```

```
 7                  8559.32         5386.06
 8                  9971.75         5009.42
 9                  11365.35        4821.09

[VERTICES]
;Link               X-Coord         Y-Coord
 2                  2758.95         6892.66
 5                  7542.37         5856.87
 9                  1067.65         7420.72

[LABELS]
;X-Coord            Y-Coord          Label & Anchor Node
 2557.52            6584.07         "T1"
 4681.42            6159.29         "T2"
 6079.65            5823.01         "T3"
 7371.68            5592.92         "T4"
 7371.68            5592.92         "T4"
 8557.52            5008.85         "T5"
 9920.35            4619.47         "T6"
 11000.00           4495.58         "Estanque"
 10840.71           4053.10         "Almacenamiento"
 -1601.77           8053.10         "Desarenador"
 -3477.88           8991.15         "Embalse"

[BACKDROP]
 DIMENSIONS     0.00            0.00            10000.00        10000.00
 UNITS          Ninguno
 FILE
 OFFSET         0.00            0.00

[END]
```

C.2 MODELO DE SIMULACIÓN ACUEDUCTO LA CEIBITA EXTENDIDO

```
                                [TITLE]

[JUNCTIONS]
;ID                 Elev            Demand          Pattern
 2                  2030            0                               ;
 10                 2055            -5                              ;

[RESERVOIRS]
;ID                 Head            Pattern
 11                 2061.48                         ;Embalse

[TANKS]
;ID                 Elevation       InitLevel       MinLevel
     MaxLevel       Diameter        MinVol          VolCurve
 3                  2028            0.4             0.4             0.8
     2.85           0                               ;T1
 4                  2020            0.5             0.5             1.8
     1.58           0                               ;T2
 5                  2010            0.1             0.1             0.5
     1.13           0                               ;T3
 6                  2009            0.5             0.5             1
     1.45           0                               ;T4
 7                  2008.5          0.1             0.1             0.6
     2.25           0                               ;T5
 8                  2008            0.2             0.2             1
     1.5            0                               ;T6
 9                  2007.95         2               1               2.75
     7              0                               ;Estanque de
Almacenamiento
 1                  2005            1.75            0.5             2
     4              0                               ;Buena Vista
 14                 2005            0.88            0.5             1
     3              0                               ;La Trinidad
 15                 2005            2.2             0.5             2.5
     6              0                               ;La Ceibita
 16                 2005            0.6             0.2             1
     2              0                               ;La Travesia

[PIPES]
;ID                 Node1           Node2           Length
     Diameter       Roughness       MinorLoss       Status
 2                  2               3               8.9             100
     0.1            0               Open       ;
 3                  3               4               75              75
     0.1            0               Open       ;
 4                  4               5               800             75
     0.1            0               Open       ;
 5                  5               6               2.7             75
     0.1            0               Open       ;
 6                  6               7               5               75
     0.1            0               Open       ;
 7                  7               8               1               75
     0.1            0               Open       ;
```

```
 8                  8                  9                  1                  110
     0.1            0                  CV            ;
 9                  10                 2                  1175               110
     0.1            0                  Open          ;
 1                  11                 10                 25                 75
     0.1            0                  Open          ;
 10                 1                  9                  1000               75
     0.1            0                  Open          ;
 13                 9                  14                 1000               75
     0.1            0                  Open          ;
 14                 9                  15                 1000               75
     0.1            0                  Open          ;
 15                 9                  16                 1000               75
     0.1            0                  Open          ;

[PUMPS]
;ID                 Node1              Node2              Parameters

[VALVES]
;ID                 Node1              Node2              Diameter           Type
     Setting        MinorLoss

[TAGS]
 NODE      11              Embalse
 NODE      3               T1
 NODE      4               T2
 NODE      5               T3
 NODE      6               T4
 NODE      7               T5
 NODE      8               T6

[DEMANDS]
;Junction           Demand             Pattern            Category

[STATUS]
;ID                 Status/Setting

[PATTERNS]
;ID                 Multipliers

[CURVES]
;ID                 X-Value            Y-Value

[CONTROLS]



[RULES]



[ENERGY]
 Global Efficiency   75
 Global Price        0
```

```
 Demand Charge        0

[EMITTERS]
;Junction            Coefficient

[QUALITY]
;Node                InitQual

[SOURCES]
;Node                Type            Quality         Pattern

[REACTIONS]
;Type      Pipe/Tank       Coefficient


[REACTIONS]
 Order Bulk              1
 Order Tank              1
 Order Wall              1
 Global Bulk             0
 Global Wall             0
 Limiting Potential      0
 Roughness Correlation   0

[MIXING]
;Tank                Model

[TIMES]
 Duration            168
 Hydraulic Timestep  24:00
 Quality Timestep    0:24
 Pattern Timestep    24:00
 Pattern Start       24:00
 Report Timestep     24:00
 Report Start        0:00
 Start ClockTime     12 am
 Statistic           None

[REPORT]
 Status              No
 Summary             No
 Page                0

[OPTIONS]
 Units               LPS
 Headloss            D-W
 Specific Gravity    1
 Viscosity           1
 Trials              40
 Accuracy            0.001
 CHECKFREQ           2
 MAXCHECK            10
 DAMPLIMIT           0
 Unbalanced          Stop
```

```
 Pattern             1
 Demand Multiplier   1.0
 Emitter Exponent    0.5
 Quality             None mg/L
 Diffusivity         1
 Tolerance           0.01

[COORDINATES]
;Node                X-Coord         Y-Coord
 2                   1252.35         7250.47
 10                  -1308.85        8267.42
 11                  -2737.84        9302.33
 3                   2702.45         6967.98
 4                   4755.18         6459.51
 5                   6129.94         6158.19
 6                   7467.04         5913.37
 7                   8559.32         5386.06
 8                   9971.75         5009.42
 9                   11365.35        4821.09
 1                   13372.09        6913.32
 14                  13372.09        2558.14
 15                  13964.06        5813.95
 16                  14260.04        4101.48

[VERTICES]
;Link                X-Coord         Y-Coord
 2                   2758.95         6892.66
 5                   7542.37         5856.87
 9                   1067.65         7420.72

[LABELS]
;X-Coord           Y-Coord          Label & Anchor Node
 2557.52          6584.07          "T1"
 4681.42          6159.29          "T2"
 6079.65          5823.01          "T3"
 7371.68          5592.92          "T4"
 7371.68          5592.92          "T4"
 8557.52          5008.85          "T5"
 9920.35          4619.47          "T6"
 11000.00         4495.58          "Estanque"
 10840.71         4053.10          "Almacenamiento"
 13690.27         7044.25          "Buena Vista"
 14292.04         5823.01          "La Ceibita"
 14628.32         4070.80          "La Travesía"
 13761.06         2495.58          "La Trinidad"
 -1601.77         8053.10          "Desarenador"
 -3477.88         8991.15          "Embalse"

[BACKDROP]
 DIMENSIONS     0.00            0.00            10000.00        10000.00
 UNITS          Ninguno
 FILE
 OFFSET         0.00            0.00
```

[END]

C.3 MODELO DE SIMULACIÓN DEL AGENTE REGULADOR

```
public void AgenteAdministrador (String nodo , String dot
     , String pres,  LogicalTime tiem   ) throws
    ParseException {
presion = StringToDouble(pres);
    dotacion = StringToDouble(dot);

    // Calcula el volumen en el estanque de
        almacenamiento
        volumen = CalculaVolumen(presion, radio);

  // Calcula el volumen en las poblaciones que atiende el
      acueducto
        volumenPobA = CalculaAgua(noHabitantesA);
        volumenPobB = CalculaAgua(noHabitantesB);
        volumenPobC = CalculaAgua(noHabitantesC);
        volumenPobD = CalculaAgua(noHabitantesD);

    GInterface.init("net.cesimo.administra.Interfaz");
    Glider.setTime(0);
    Glider.setTsim(8);

 System.out.println("Inicio de simulacion GALATEA");

 // Traza de la simulaci?n en archivo
 Glider.trace("GALATEA.trc");
 Glider.act(Ambiente, 0);
 Glider.process();

        return;
}


public static  void Grafico ( double volumen )
    {
        array[i]= volumen ;
        i++;
        GUI demo = new GUI("PSAP",  array );
        demo.pack();
        demo.setPreferredSize(new Dimension(500, 270));
        RefineryUtilities.centerFrameOnScreen(demo);
        demo.setVisible(true);
        return;

}

public double StringToDouble (String val ) throws
    ParseException {
            NumberFormat format = NumberFormat.
                getInstance(Locale.FRANCE);
```

```
                Number number;
                number = format.parse(val);
                double numero = number.doubleValue();
                return numero;
            }

        public double CalculaVolumen(double presion, double radio
            ){
                // calcula el  volumen(en  litros) del estanque
                    de almacenamiento .
                double volumen;
                double pi=3.1416;
                volumen= (pi*radio*radio*presion)*1000;
                return volumen;
        }

        public double CalculaAgua(double noHab  ){

         double volumenPob = noHab*50 ;
         return volumenPob;
        }
```

Programa C.1: AgenteAdministrador.java

```
public class Delta extends Node {
    public PrologAg AgenteA ;

    @SuppressWarnings("static-access")
    public Delta() {
        super("Delta", 'A');
        Glider.nodesl.add(this);

        String Path = "C:/Users/Virginia/ejemplo/Mi_Ejemplo";
        AgenteA  = new PrologAg("agenteA");
        AgenteA.demoHome = Path;
        AgenteA.initProlog();

        GInterface.agentList.add(AgenteA);

        System.out.println("Este es un Ejemplo de Simulacion de
            un Agente Administrador del Acueducto");
        System.out.println("");

    }

    @Override
    @SuppressWarnings("empty-statement")
    public void fact(){
        it(1);

        Glider.setTime(24);
```

```
        // Actualiza el reloj del agente
        AgenteA.clock = Glider.getTime();

        // Transmite lo que el agente debe ver
         ActualizarSensores(AgenteA);

        // Activa el razonado del agente
        AgenteA.cycle();

        System.out.println("Simulador : Comienzo a procesar las
            influencias");
        GInterface.gatherInfluences_test();
        GInterface.process_test();

        AgenteA.inputs.clear();
    }

    @SuppressWarnings({ "empty-statement", "unchecked" })
    public void ActualizarSensores(PrologAg agente){
          agente.inputs.add("volumen("+AgenteAdministrador.
              volumen+")");

        System.out.println("Este agente observara " + agente.
            inputs);
    }
}
```

Programa C.2: Delta.java


```
public class GUI extends ApplicationFrame    {

    /** The time series data. */
    private TimeSeries series;

    /** The most recent value added. */
    private double lastValue = 100.0;



    public GUI(String s, Double[] array)
    {
        super(s);
        JPanel jpanel = createDemoPanel(array);
        jpanel.setPreferredSize(new Dimension(600, 370));
        setContentPane(jpanel);
    }

    private static CategoryDataset createDataset(Double[] array)
    {
```

```
        DefaultCategoryDataset defaultcategorydataset = new
            DefaultCategoryDataset();
                    defaultcategorydataset.addValue(array[0], "
                        Series 1", "lunes");
                    defaultcategorydataset.addValue(array[1], "
                        Series 1", "martes");
                    defaultcategorydataset.addValue(array[2], "
                        Series 1", "miercoles");
                    defaultcategorydataset.addValue(array[3], "
                        Series 1", "jueves");
                    defaultcategorydataset.addValue(array[4], "
                        Series 1", "viernes");
                    defaultcategorydataset.addValue(array[5], "
                        Series 1", "sabado");
                    defaultcategorydataset.addValue(array[6], "
                        Series 1", "domingo");
        return defaultcategorydataset;
    }


    /**
     * Creates a sample chart.
     *
     * @param dataset  the dataset.
     *
     * @return A sample chart.
     */

        private static JFreeChart createChart(CategoryDataset
            categorydataset)
        {
           JFreeChart jfreechart = ChartFactory.createBarChart3D
               ("Suministro de Agua. sector La Ceibita", "dias de
                la semana",
                "Volumen de agua potable (litros)",
                    categorydataset, PlotOrientation.VERTICAL,
                    false, false, false);
                      CategoryPlot categoryplot = (CategoryPlot)
                          jfreechart.getPlot();
                      CategoryAxis categoryaxis = categoryplot.
                          getDomainAxis();
                      categoryaxis.setCategoryLabelPositions(
                          CategoryLabelPositions.
                          createUpRotationLabelPositions(0.2D));
                      CategoryItemRenderer categoryitemrenderer =
                           categoryplot.getRenderer();
                      categoryitemrenderer.
                          setBaseItemLabelsVisible(true);
                      BarRenderer barrenderer = (BarRenderer)
                          categoryitemrenderer;
```

```
                    barrenderer.setItemMargin(200D);
            return jfreechart;
        }
        public static JPanel createDemoPanel(Double[] array)
        {

                    JFreeChart jfreechart = createChart(
                        createDataset(  array));
            return new ChartPanel(jfreechart);
        }



}
```

Programa C.3: GUI.java

```
public class Interfaz {

        public  void disponible (double t, PrologAg a,jpl.Float
            ahorro) {
         double valor =   ahorro.doubleValue() ;

        double volumen_total = AgenteAdministrador.volumenPobA+
            AgenteAdministrador.volumenPobB +
                                        AgenteAdministrador.
                                            volumenPobC +
                                            AgenteAdministrador.
                                            volumenPobD ;

                        if (AgenteAdministrador.nombreP.equals(
                            AgenteAdministrador.nombrePuebloA)) {
                        valor = valor - valor/volumen_total*((
                            AgenteAdministrador.volumenPobB +
                                AgenteAdministrador.volumenPobC +
                                    AgenteAdministrador.
                                    volumenPobD));
                         AgenteAdministrador.Grafico(valor/
                             AgenteAdministrador.noHabitantesA);
          }
    }

   }
```

Programa C.4: Interfaz.java

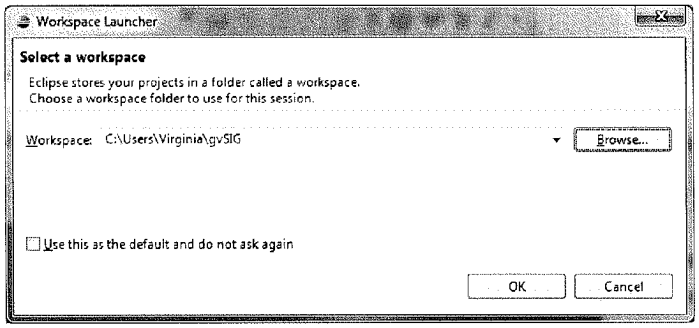
D

GUÍA DE INSTALACIÓN DEL PSAP

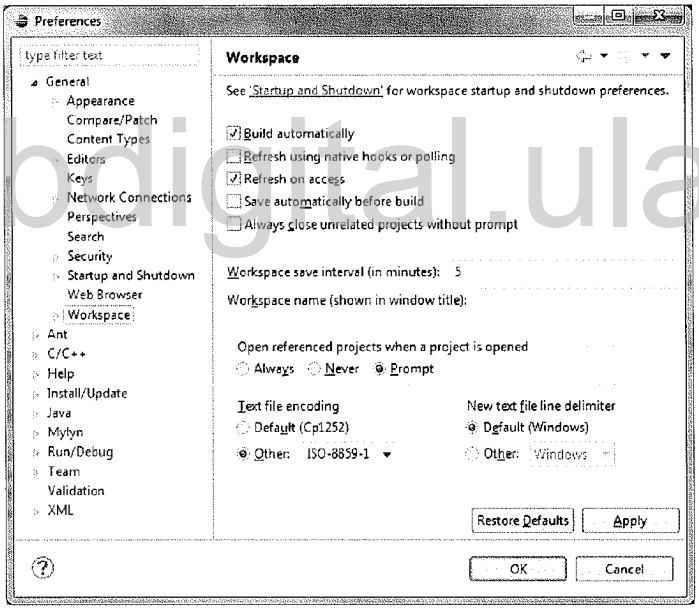
D.1 CONFIGURAR ENTORNO DE DESARROLLO PARA GVSIG 1.11 DESKTOP EN EL IDE ECLIPSE JUNO

D.1.0.1 *Preparar el espacio de trabajo*

- Crea un nuevo *workspace* con Eclipse

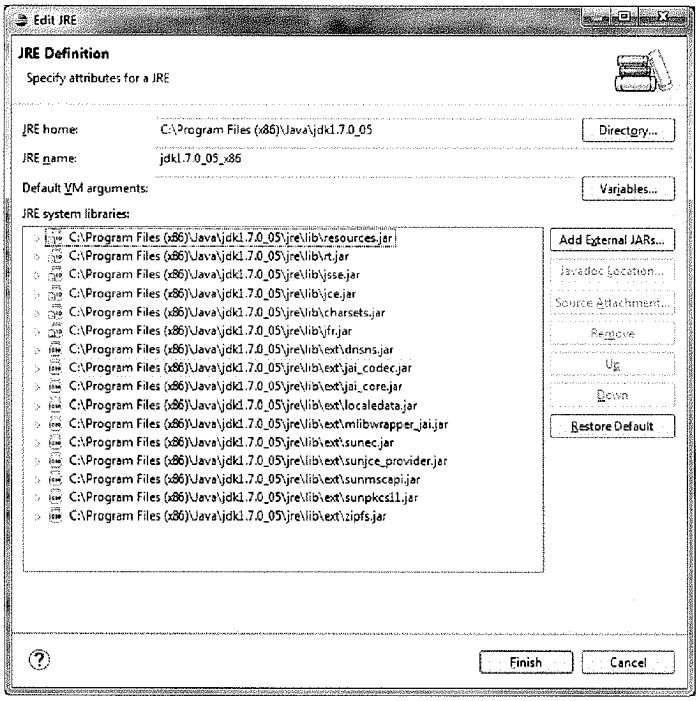


- Establecer la codificación de los archivos a ISO-8859-1.



- Verificar que la máquina virtual de JAVA es JAVA 1.5 SDK (o posterior) con *Java* Java Advanced Imaging (JAI)[1] y *Java* JAI *Input/Output* (I/O) en el *classpath*.

1 esta en: http://www.oracle.com/technetwork/java/current-142188.html

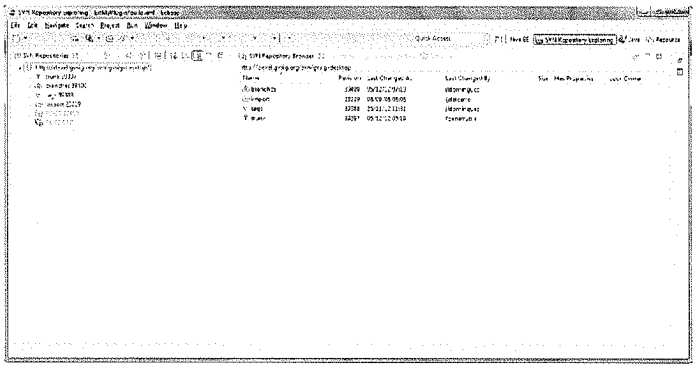


D.1.0.2 *Conectarse al repositorio SVN de gvSIG.*

- Se debe tener instalado el cliente Subclipse.
  - En el menú de Eclipse: **Help, Install New Software.**
  - En el dialogo **Available Software, Work with** añadir el repositorio `http://community.polarion.com/projects/subversive/download/eclipse/3.0/juno-site/`.
  - Escoger SVN CONNECTORS y SVN SOURCES, **Next** y seguir los pasos de instalación [2]
- Despues de re-iniciar Eclipse, desde el menu Window, seleccione Open Perspective, Other... Seleccione SVN Repository Exploring y click OK .
- Desde la opción SVN Repository Exploring añade un nuevo repositorio con la dirección `https://devel.gvsig.org/svn/gvsig-desktop`

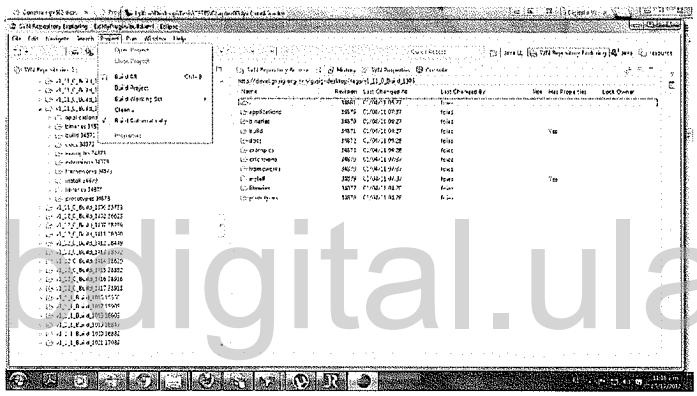
Para verificar la versión de Subversion que se debe usar para su versión de Eclipse leer las notas de `http://www.eclipse.org/subversive/downloads.php`

[2]
- Hay otra opción de instalación de subversión con el repositorio proporcionado por la instalación de Eclipse: Juno– http://download.eclipse.org/releases/juno. En Collaboration escoger SVN y seguir los pasos de instalación.



D.1.0.3 *Obtener el código fuente mínimo para gvSIG*

- Desactivar la propiedad *automatic building* para descargar el código.



- Escoger el directorios *tag* . En este caso se ha trabajado con la *versión 1.11 Build 1305*.

- Los directorios de los que obtendremos los proyectos son:

  - applications (aplicaciones que funcionan sobre Andami)

  - binaries (.dll o .so) extensions (todas las extensiones de Andami)

  - frameworks (Andami)

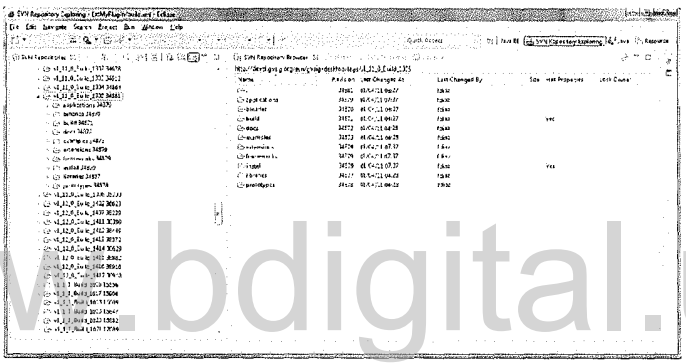
  - libraries (todas las bibliotecas usadas por las extensiones, Andami y appgvSIG)

- Los proyectos que se necesitan:

  - _fwAndami: Framework usado por gvSIG para la interfaz de usuario MDI
  - appgvSIG: La aplicación por encima de Andami que añade las funcionalidades "geo"

- binaries: Los enlaces JNI y las bibliotecas nativas para el sistema operativo, las necesitan varios componentes de gvSIG
- libCorePlugin: Un *skin* para Andami que implementa la apariencia final de la interfaz de usuario
- libExceptions: Biblioteca común para el tratamiento de excepciones de los fuentes de gvSIG
- libFMap: Biblioteca principal de gvSIG, trabaja con geodatos, pintado, etc.

- También, los proyectos que se necesitan para compilar gvSIG versión 1.11 puede consultarse en `https://gvsig.org/web/projects/gvsig-desktop/devel/gvsig/gvsig-desktop/devel/gvsig/1-11-0/projects`

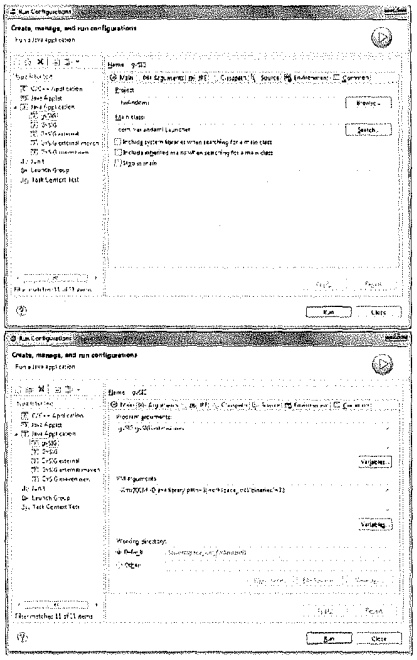


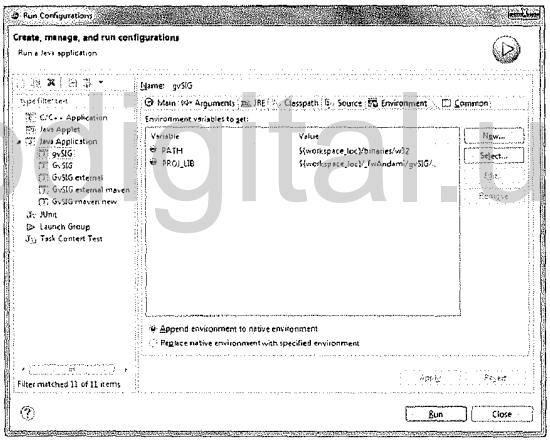
- Después de descargar todos los proyectos en el workspace cambiar la propiedad **Build Automatically**.
- Hay un task de Ant en **appgvSIG** llamado *install-all* que ejecuta todos los Ant de las diferentes extensiones y bibliotecas de gvSIG en el orden correcto y coloca todo bajo Andami. También puede ejecutar el task de Ant que hay en el proyecto binaries para copiar las bibliotecas apropiadas a ese directorio.

Puede que Eclipse olvide algunas referencias a bibliotecas, así que si faltan clases o jars *en un proyecto, intente borrar cualquier* jar *del conjunto de bibliotecas referenciadas y añádalo de nuevo.*

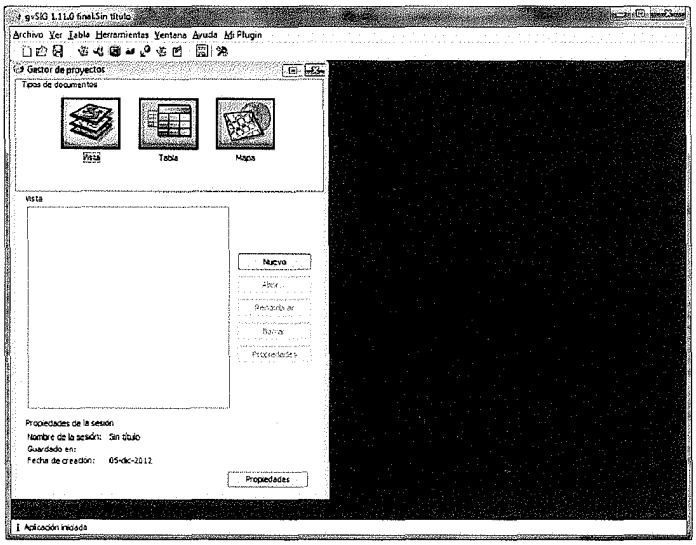
D.1.0.4 *Configurar el diálogo de ejecución*

- Ahora puede configurar el diálogo Run para ejecutar gvSIG.





- Ejecutar la aplicación



En la siguiente dirección se encuentra más información de como construir gvSIG con el IDE Eclipse: `https://gvsig.org/web/docdev/building-from-svn/building-gvsig-from-svn`

D.2 GUÍA DE INSTALACIÓN DEL PREDICTOR DE SERVICIO DE AGUA POTABLE EN GVSIG MOBILE.

Guía de Instalación

1 Instalación de Eclipse

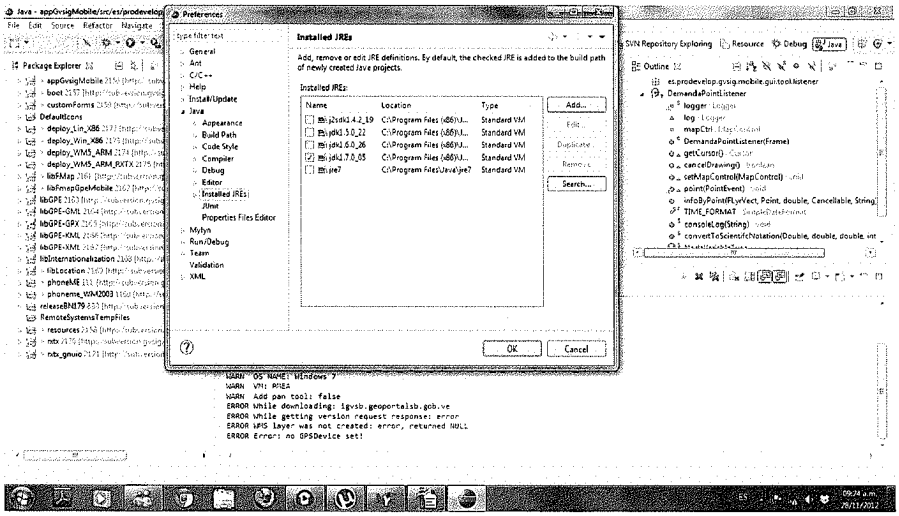
Para el desarrollo de la aplicación se usó del **IDE Eclipse** versión **Juno 4.2** , el paquete **Eclipse IDE for Java EE Developers**, http://www.eclipse.org/downloads/ . Aunque puede usarse el paquete Eclipse Classic- 4.2.1.

1. Bajar la versión de Eclipse seleccionada, descomprimir y listo.
2. Abrir eclipse y cerrar la ventana de Bienvenida.

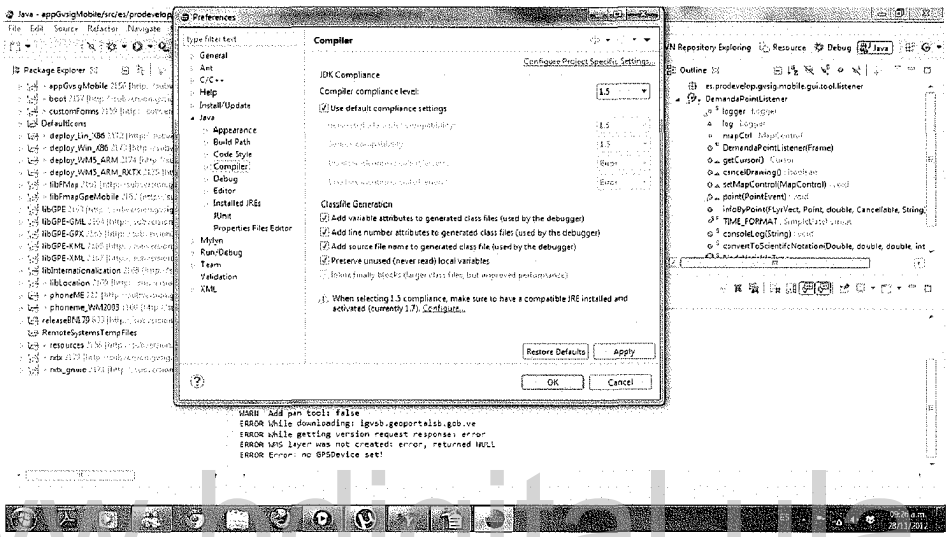
2 Configurar jdk en Eclipse

Se usó jdk1.7.0_05 , pero bien puede usarse jdk1.6.

1. En el menú escoger **Window, Preferences**
2. En el dialogo abrir la opción de **Java,** y click en **Installed JREs.** Se abre el dialogo que muestra los JREs instalados
3. Click en el boton **Add,** luego **Standard VM, Next.**
4. En el dialogo **JRE Definition , JRE home:,** dar click en **Directory** para escoger el directorio de instalación de java jdk. Y Next.
5. En el dialogo **JRE Definition,** seleccionar el jdk que se añadio. Click en **OK.**



6. Colocar el nivel de conformidad del jdk en el Workspace a 1.5 **:Window, Preferences, Java,Compiler.**



3 Instalación del paquete Subversive para Eclipse

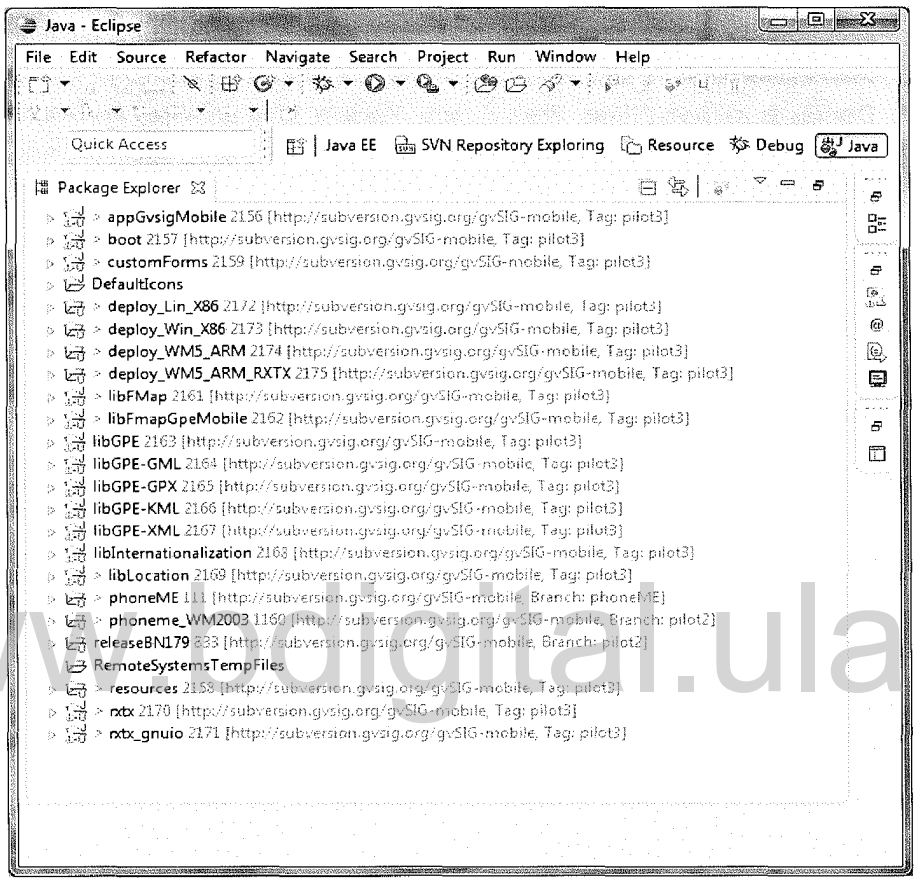
Para bajar y configurar **gvSIG Mobile,** se necesita instalar Subversive para Eclipse:

1. En el menú de Eclipse: **Help**, **Install New Software.**
2. En el dialogo **Available Software,** en **Work with** añadir el repositorio http://community.polarion.com/projects/subversive/download/eclipse/3.0/juno-site/. Escoger SVN CONNECTORS y SVN SOURCES, **Next** y seguir los pasos de instalación.

- **NOTA 1:** Para verificar la versión de Subversion que se debe usar para su versión de Eclipse leer las notas de http://www.eclipse.org/subversive/downloads.php .
- **NOTA 2:** Hay otra opción de instalación de subversión con el repositorio proporcionado por la instalación de Eclipse: **Juno–**

http://download.eclipse.org/releases/juno. En **Collaboration** escoger **SVN** y seguir los pasos de instalación.

3. Despues de re-iniciar Eclipse, desde el menu **Window,** seleccione **Open Perspective, Other...** Seleccione **SVN Repository Exploring** y click **OK** . Se debe abrir la ventana **SVN Repositories**
4. En la ventana **SVN Repositories,** escoger añadir un repositorio en **New Repository Location**. Añadir https://simulants.svn.sourceforge.net/svnroot/simulants
5. Seleccionar gvSIG_Mobile , right-click y **Check-out** para bajar la instalación
6. Al finalizar cambiarse a la perspectiva **Java EE .** Debe tener los siguientes proyecto que se ven en esta pantalla



4 Instalación de Librerias de Usuario en Eclipse

En la carpeta Readme está el jar de jfreechart, jcommon y BaseformEpaNetLib

1. En Eclipse, seleccionar **Window , Preferences.** Seleccione **Java, Build Path, User Libraries** .
2. Seleccione **New**, añada *jfreechart_version* correspondiente. Luego click en *jfreechart_version* y añada el jar asociado. Click en OK.
3. Repetir para estos pasos para añadir la definición correspondiente a *jcommon_version*

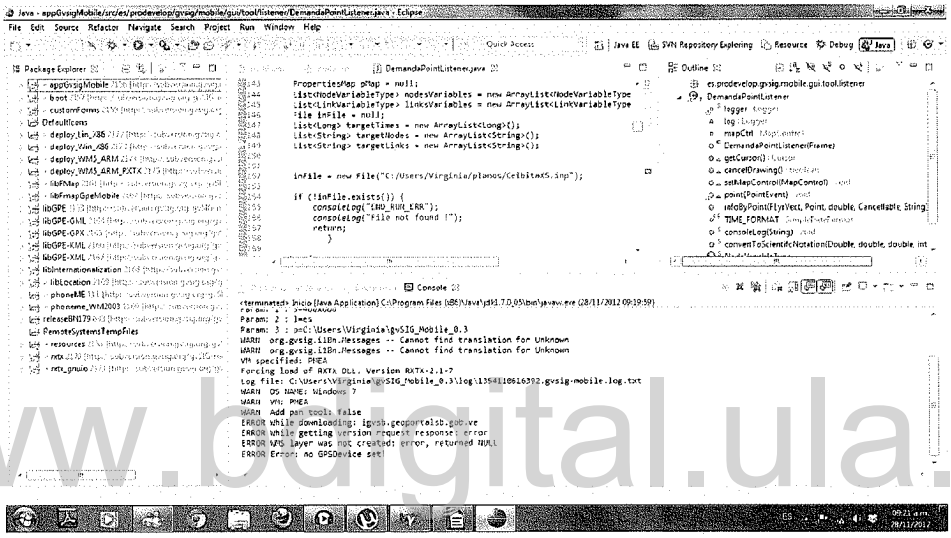
y BaseformEpaNetLib .

4. En *appGvsigMobile, src, es.prodevelop.gvsig.mobile.gui.tool.listener,* right-click en **DemandaPointListener** , **build Path, configure Build Path,** añada las librerías de usuarios definidas.
5. Refresque el espacio de trabajo: **menú, File, Refresh**

- **NOTA 1: Puede usar su propia instalación de jfreechart, apuntando al directorio correspondiente.**
- **NOTA 2: en** http://www.baseform.org/np4/tools/epanetTool.html **puede bajar los fuentes y binarios de EPANET**

5 Instalación de gvSIG_Mobile

En la perspectiva Java, **Project Explorer** revisar que las carpetas no estén marcadas con aspas en rojo. En el caso afirmativo, revisar los build path de los proyectos marcados. En esta carpeta, README, adjunto el .zip de librerías que usé.



Ya depurado todos los errores, construiremos gvSIG_Mobile. (Me basé en la última versión estable de gvSIG_Mobile 0.3) :

1. En **boot** construir el boot del proyecto (**build.xml->right click->Run As->Ant Build)**
2. Añadir el boot generado al JRE : **boot/deploy/boot.jar** al JRE (**Window->Preferences->Java->Installed JREs->Edit->Add External JARs...**).
3. Seleccionar todos los proyectos y limpiarlos (**Menu -> Project -> Clean**)
4. Construir el proyecto appGvsigMobile **(build.xml->right click->Run As->Ant Build)**.
5. Crear una configuración de ejecución para una aplicación java **(Run->Run...->Java Application)**

6. Seleccione el proyecto **appGvsigMobile** y la clase **es.prodevelop.gvsig.mobile.app.Launcher.**
7. En la pestaña argumento añadir:

- **Program arguments:** m=J9 s=480x600 l=es p="${workspace_loc}"
- **VM arguments**: -Xmx256M -Djava.library.path=${workspace_loc}/resources/lib/Win_X86

Cambie los siguiente valores de acuerdo a su configuración:

- m: virtual machine, J9 or PMEA (Phone ME Advance)
- s: window size, in pixels, in the example above 480x600
- l: language, in the example above spanish (es)
- Djava.library.path: should point to the native libraries -.dll or .so files- para SO, que puede ser Lin_X86 (Linux), Win_X86 (Windows) or WM5_ARM (Windows Mobile)

Ejecute la configuración. Debe obtener la siguiente pantalla.

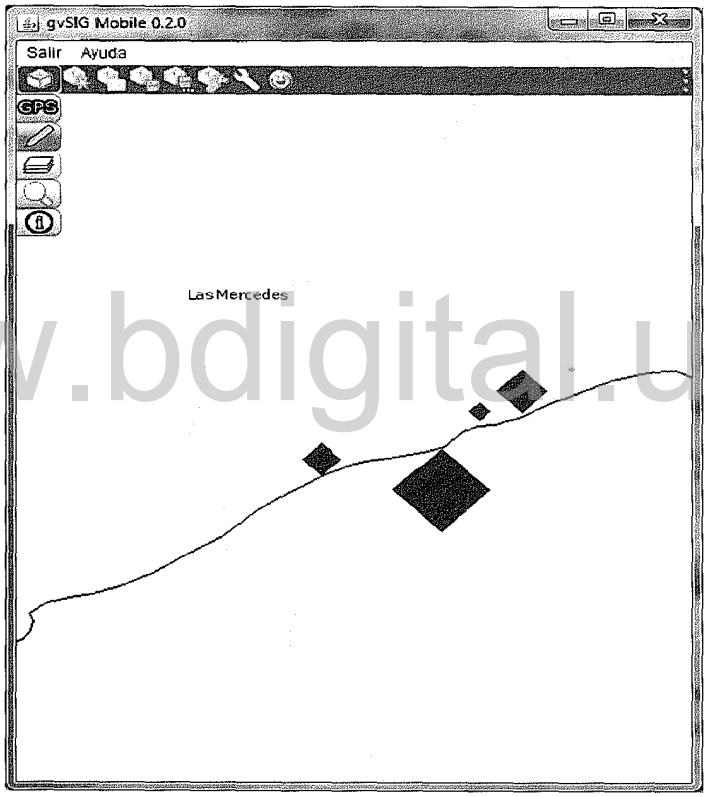
6 Para ejecutar la aplicación

En la carpeta planos está incluido la red de la Ceibita (*CeibitaXS.inp*) que se usó.

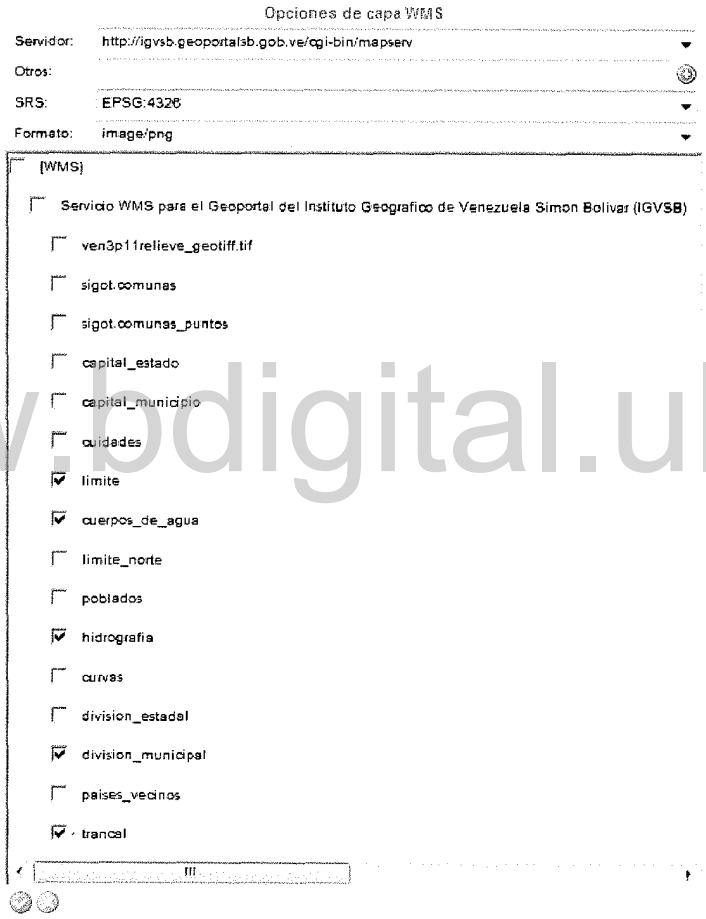
En la carpeta gvm_projects está *TabayMobile.gvm,* proyecto que puede usar para probar la aplicación.

Para ejecutar la aplicación.

1. Iniciar la aplicación
2. Abrir proyecto (*click* en icono "cajita con carpetas") : navegar hasta TabayMobile.gvm. El proyecto TabayMobile.gvm tiene ya configurado el acceso al WMS del Instituto Geográfico Simón Bolívar. Una vez cargado el proyecto tendrán una pantalla como la que se muestra a continuación:



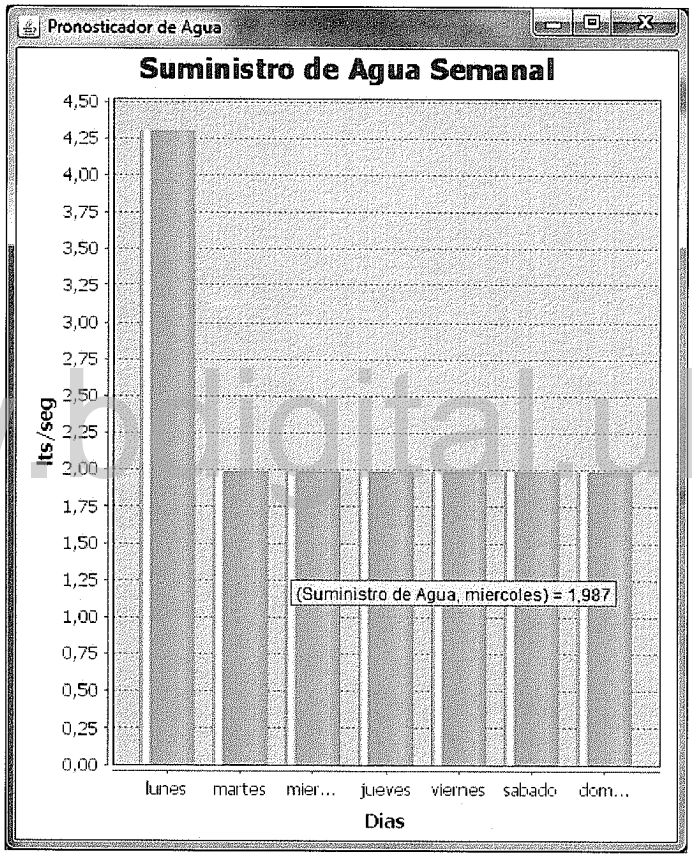
3. NOTA: En caso de que se desee cargar capas del Instituto Geográfico Simón Bolívar. Ir hasta el ícono de hojas superpuestas, escoger **WMS** y agregar la dirección : http://igvsb.geoportalsb.gob.ve/cgi-bin/mapserv. Escoger las opciones que se deseen agregar.



4. Click en opcion ***i***

5. *Click en* ***PS***
6. *Click* en en uno de los polígono: en la parte inferior se muestra la información de la población y esperar un poco para la salida.

Este es el suministro promedio de agua potable para el sector "La Trinidad".



NOTA:

- Error "Minor version 5.1" o algo parecido. Revisar las versiones de java
- Error "librería no existe" o algo parecido. Revisar versión de java y la compatibilidad con la

librería

- Cualquier duda, inquietud, sugerencia o si desean notificar algún error, por favor escribir a virginiapadillas@gmail.com.

Gracias por sus comentarios.

Virginia Padilla

D.3 GUÍA DE INSTALACIÓN DE EXTENSIÓN PSAP EN GVSIG DESKTOP

Para instalar esta extensión en gvSIG debe seguirse los siguientes pasos:

- Instalar el entorno de desarrollo para gvSIG 1.11 desktop en el IDE Eclipse Juno. Ver anexo Sección D.1.
- Bajar la extensión *ExtMyPlugin* desde `https://simulants.svn.sourceforge.net/svnroot/simulants/`.
- Incluir la extensión *ExtMyPlugin* en gvSIG 1.11 desktop.
- Ejecutar *build.xml*. Con esto se instala la extensión *ExtMyPlugin* en gvSIG
- Ejecutar gvSIG 1.11 desktop.

BIBLIOGRAFÍA

[1] Rakesh Agrawal, Anastasia Ailamaki, Philip A. Bernstein, Eric A. Brewer, Michael J. Carey, Surajit Chaudhuri, AnHai Doan, Daniela Florescu, Michael J. Franklin, Hector Garcia-Molina, Johannes Gehrke, Le Gruenwald, Laura M. Haas, Alon Y. Halevy, Joseph M. Hellerstein, Yannis E. Ioannidis, Hank F. Korth, Donald Kossmann, Samuel Madden, Roger Magoulas, Beng Chin Ooi, Tim O'Reilly, Raghu Ramakrishnan, Sunita Sarawagi, Michael Stonebraker, Alexander S. Szalay, and Gerhard Weikum. The Claremont report on database research. *SIGMOD Rec.*, 37(3):9–19, 2008. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/1462571.1462573. (Cited on page 64.)

[2] John Anderson. Providing a broad spectrum of agents in spatially explicit simulation models: The gensim approach. In *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity. Randy Gimblett, 2001. (Cited on page 71.)

[3] R Andressen. Datos estación climatológica mucujún. Technical report, Centro de Investigaciones Atmósfericas y del Espacio. Universidad de Los Andes, Se encuentra en http://www.cecalc.ula.ve/redbc/colecciones, 2000. Fecha de consulta: 15 de octubre 2010. (Cited on page 118.)

[4] N Ashish and C Knoblock. Wrapper generation for semi-structured internet sources. *ACM SIGMOD Record*, 26:4, 1997. (Cited on page 163.)

[5] James Bailey, Michael Georgeff, David Kemp, David Kinny, and Kotagiri Ramamohanarao. Active database and agent systems - a comparison. Technical report, Departament of Computer Science. University of Melbourne, 1995. (Cited on page 5.)

[6] Baseform-Epanet-Java Library. Baseform EpaNet Java Library, v 1.0. URL `https://github.com/Baseform/Baseform-Epanet-Java-Library`. (Cited on page 90.)

[7] Michael Batty, Andrew Crooks, and Christian Castle. Key challenges in agent-based modelling for geo-spatial simulation. *CASA Working paper 121*, 2007. URL `jmichaelbatty.wordpress.com/working-papers/`. (Cited on pages 3 y 46.)

[8] Kent Beck. *Extreme Programming Explained: Embrace Change*. Addison-Wesley, 1999. (Cited on pages 71 y 73.)

[9] Catriel Beeri and Tova Milo. A model for active object oriented databases. In *VLDB '91: Proceedings of the 17th International Conference on Very Large Data Bases*, pages 337–349, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. ISBN 1-55860-150-3. (Cited on page 66.)

[10] Itzhak Benenson and Paul M.Torrens. Geosimulation: object-based modeling of urban phenomena. *Computers, Environment and Urban Systems*, 28:1–8, 2004. URL `www.elsevier.com/locate/compenvurbsys`. (Cited on pages 3 y 46.)

[11] Phil Bernstein, Michael Brodie, Stefano Ceri, David DeWitt, Mike Franklin, Hector Garcia-Molina, Jim Gray, Jerry Held, Joe Hellerstein, H. V. Jagadish, Michael Lesk, Dave Maier, Jeff Naughton, Hamid Pirahesh, Mike Stonebraker, and Jeff Ullman. The asilomar report on database research. *SIGMOD Rec.*, 27(4): 74–80, 1998. ISSN 0163-5808. doi: http://doi.acm.org/10.1145/306101.306137. (Cited on page 63.)

[12] Maurice Beynon and R.I. Cartwright. Empiricall modelling principles for cognitive artefacts. *Proc. IEE Colloquium: Design Systems with Users in Mind: The Role of Cognitive Artefacts*, Digest No. 95/231:8/1–8/8, 1995. (Cited on page 5.)

[13] Meurig Beynon, Alan Cartwright, and Yun Pui Yung. Databases from an agent-oriented perspective. Technical report, Department of Computer Science, Coventry, UK., 1995. (Cited on page 5.)

[14] I Blecic, A Cecchini, and GA Trunfio. A multi-agent geosimulation infrastructure for planning. *Geocomputation and Urban Plannig, Editorial Springer Verlag*, 176:pp. 237–253, 2009. (Cited on pages 9, 10, 46, 47, 51, 57 y 63.)

[15] Bracken and Webster. *Information technology in Geography and Planning. Including principles of GIS.* Routledge, Londres, 1990. (Cited on page 45.)

[16] Jeffrey M. Bradshaw. An introduction to software agents. In Jeffrey M. Bradshaw, editor, *Software Agents*, pages 3–46. AAAI Press / The MIT Press, 1997. URL `citeseer.ist.psu.edu/bradshaw97introduction.html`. (Cited on page 18.)

[17] M Bratman. *Intention, Plans and Practical Reasoning*. Harvard University Press, Cambridge., 1987. (Cited on pages 18 y 20.)

[18] P Bresciani, P Giorgini, F Giunchiglia, J Mylopoulos, and A Perini. Tropos: An agent-oriented software development

methodology. *Autonomous Agents and Multi-Agent Systems,*, Vol. 8, no.3(No. 3):pp. 203–236, 2004. URL citeseer.ist.psu.edu/article/bresciani04tropos.html. (Cited on pages 9, 20 y 33.)

[19] S. Brinkkemper. Method engineering: Engineering of information systems development methods and tools. In *Information and Software Technology*, volume 38(4), pages 275–280, 1996. (Cited on page 37.)

[20] P. Buseta, R. Rönnquist, A. Hodgson, and A. Lucas. JACK intelligent agents - components for intelligent agent in java. Technical report, Agent Oriented Software Pty. Ltd, Melbourne, Australia, 1998. (Cited on page 34.)

[21] G Caire, W Coulier, F Garijo, J Gomez, J Pavon, F Leal, P Chainho, P E Kearney, J Stark, R Evans, and P Massonet. Agent oriented analysis using message/UML. *LNCS*, 2222: pp. 119–135, 2002. URL citeseer.ist.psu.edu/caire01agent.html. (Cited on pages 8, 20 y 30.)

[22] Giovanni Caire, Wim Coulier, Francisco J. Garijo, Jorge Gomez, Juan Pavon, Francisco Leal, Paulo Chainho, Paul E. Kearney, Jamie Stark, Richard Evans, and Philippe Massonet. Agent oriented analysis using message/UML. In *AOSE*, pages 119–135, 2001. URL citeseer.ist.psu.edu/caire01agent.html. (Cited on pages xvii y 31.)

[23] A.S Camara, R. Ferreira, and P. Castro. Spatial simulation modelling. In H.J. Scholten M. Fisher and D.Unwin, editors, *Spatial Analytical Perspectives on GIS*. London: Taylor & Francis, 1996. (Cited on page 49.)

[24] CERTI. CERTI Web Site. URL http://savannah.nongnu.org/projects/certi/. (Cited on page 164.)

[25] Ven Te Chow, David Maidment, and Larry Mays. *Hidrología Aplicada*. 1994. Traducción de Juan Saldarriaga. (Cited on page 118.)

[26] John Christiansen. A flexible object-oriented software framework for modeling complex systems with interacting natural and societal processes. In D.A. Wolfe, editor, *Proccedings of the 4th International Conference on integrating GIS and Environmental Modeling*, Alberta, Canada, Septembre 2-8 2000. (Cited on page 48.)

[27] K. Clarke and G. Olsen. Refining a cellular automaton model of wildfire propagation and extinction. In B.O. Parks C. Johnstone D. Maidment M. Crane M.F. Goodchild, L.T. Steyaert and H. Glendinning, editors, *GIS and Environmental Modeling:*

Progress and Research Issues, CO: GIS World, pages 333–338. Fort Collins, 1993. (Cited on page 49.)

[28] D. Coleman, P. Arnold, S. Bodoff, H. Gilchrist, F. Hayes, and P. Jeremas. *Object-oriented Development: the Fusion method.* Prentice, Hall Hemel Hampstead (UK), 1994. (Cited on page 24.)

[29] H. Couclelis. From cellular automata to urban models: New principles for model development and implementation. In *Environment and Planning B: Planning and Design*, volume 24, pages 165–174. 1997. URL `http://www.envplan.com/abstract.cgi?id=b240165`. (Cited on page 49.)

[30] CQL. Contextual Query Language. URL `http://www.loc.gov/standards/sru/specs/cql.html`. (Cited on pages 11 y 92.)

[31] Roy Crosbie and John Zenor. *High Level Architecture. Module 1. Basic Concepts*. The Society for Computer Simulation. California State University, 1999. (Cited on pages xviii, 134, 138, 141, 152, 153 y 154.)

[32] C Cushla and A Ochoa. Diseño de actividades de autoprotección ciudadana para municipios. Technical report, Universidad de los Andes, 2002. (Cited on pages 10 y 71.)

[33] Jacinto Dávila. *Agents in Logic Programming*. PhD thesis, Imperial College of Science, Technology and Medicine, London, UK, June 1997. (Cited on pages 8, 18, 19, 43 y 44.)

[34] Jacinto Dávila. *Principles and Practice of Declarative Programming*, volume LNCS 1702, chapter OPENLOG: A Logic Programming Language Based on Abduction, pages 278–293. Springer-Verlag Heidelberg, 1999. (Cited on page 108.)

[35] Jacinto Dávila. *Practical Aspect of Declarative Languages*, volume LNCS 2562, chapter Actilog: An Agent Activation Language. Springer, 2003. (Cited on page 108.)

[36] Jacinto Dávila. Lógica práctica y aprendizaje computacional. *Universidad de los Andes. Centro de Simulación y Modelos. Facultad de Ingeniería*, 2009. URL `http://webdelprofesor.ula.ve/ingenieria/jacinto/libros.html`. (Cited on page 4.)

[37] Jacinto Dávila and Kay Tucci. Towards a logic-based, multi-agent simulation theory. *AMSE Special Issue 2000. Association for the advancement of Modelling & Simulation techniques in Enterprises*, pages 37–51, 2002. Lion, France. (Cited on pages 8, 19, 41 y 45.)

[38] Jacinto Dávila and Mayerlin Uzcátegui. GALATEA: A multi-agent simulation platform. In *Proceedings of MSNN2000: International Conference on Modeling, Simulation and Neural Networks. Mérida. Venezuela.*, 2000. (Cited on pages 4, 8, 19, 41 y 45.)

[39] Jacinto Dávila and Mayerlín Uzcátegui. Galatea: A multi-agent simulation platform. *AMSE Special Issue 2000. Association for the advancement of Modelling & Simulation techniques in Enterprises*, pages 52–67, 2002. Lion, France. (Cited on pages 7, 8, 9, 19, 40, 51 y 57.)

[40] Jacinto Dávila and Mayerlín Uzcátegui. Agents that learn to behave in multi-agent simulations. *The Fifth IASTED Internacional Conference on Modelling, Simulation and Optimization, Oranjestad, Aruba*, pages pp. 51–55, August 2005. (Cited on pages 8, 9, 19 y 40.)

[41] Jacinto Dávila, Kay Tucci, and Mayerlín Uzcátegui. *Simulación multiAgentes con GALATEA. The Gbook*. Universidad de los Andes, Mérida. Venezuela, mayo 2004. (Cited on page 4.)

[42] Jacinto Dávila, E Gómez, K Laffaille, K Tucci, and M Uzcátegui. Multi-agent distributed simulations with galatea. *The 9-th IEEE International Symposium on Distributed Simulation and Real Time Applications*, pages pp. 165–170, 2005. ISBN 0-7695-2462-1. (Cited on pages 4, 8, 9, 19 y 40.)

[43] Jacinto Dávila, Mayerlin Uzcátegui, and Kay Tucci. A multi-agent theory for simulation. In *MSO'2005. Fifth IASTED International Conference on Modelling, Simulation and Optimization*, pages 285–290, Oranjestad, Aruba, August 2005. (Cited on pages 4, 8, 9, 19 y 40.)

[44] Jacinto Dávila, Mayerlin Uzcátegui, and Kay Tucci. From a multi-agent simulation theory to galatea. In *Proceedings of 2007 Summer Computer Simulation Conference (SCSCo'7). The Society For Modelling and Simulation International.*, 2007. (Cited on pages 4, 8, 9, 10, 17, 19, 40, 41, 57, 58 y 63.)

[45] Scott DeLoach. Analysis and design using mase and agent-tool. In *Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference (MAICS 2001)*, 2001. URL `http://people.cis.ksu.edu/~sdeloach/publications/Conference/MaSE-maics20001.pdf`. (Cited on pages 26 y 33.)

[46] Luis Díaz. Integración de sistemas de información geográficos a la plataforma de simulación GALATEA. Master's thesis, Facultad de Ingeniería. Universidad de los Andes. Venezuela, 2002. (Cited on page 5.)

[47] DMSO. Defense modeling and simulation office (dmso) High Level Architecture (HLA) for simulations. *USD(A&T) memorandum, September 1996. DSMO. Alexandria, Virginia, USA*, june 2000. URL `Http://www.dmso.mil/projects/hla/`. (Cited on pages 48, 95 y 134.)

[48] DoD. Dod high level architecture (HLA) for simulation. *U.S. Department of Defense, Under Secretary of Defense for Acquisition and Technology, USD (A&T), memorandum*, 1996. (Cited on pages 12, 95, 134 y 156.)

[49] DoD. *High Level Architecture. Federation Execution Details (FED) File Specifications. RTI 1.3 version 3*. Department of Defense. Defense Modeling and Simulation Office, 31 july 1998. (Cited on page 134.)

[50] DoD. *RTI 1.3-Next Generation Programmerś Guide version 3.2. High Level Architecture. Run-Time Infrastructure*. Department of Defense. Defense Modeling and Simulation Office, September 2000. (Cited on pages xviii, 103, 104, 134, 135, 140, 152, 153 y 154.)

[51] Carlos Domingo, Hernández M., Sananes M. anf Silva J., and Tonella G. GLIDER, lenguaje para simulación de sistemas. Technical report, Universidad de los Andes, 1992. (Cited on page 4.)

[52] A Drogoul, D Vanbergue, and T Meurisse. *Multi-agent based simulation: where are the agents?.*, volume 2581. Springer-Verlag, Berlin, Heidelberg, 2003. ISBN 3-540-00607-9. (Cited on page 4.)

[53] ECLIPSE. The eclipse open source community website. www.eclipse.org, 2004. (Cited on page 85.)

[54] Ramez A. Elmasri and Shamkant B. Navathe. *Fundamentos de Sistemas de Bases de Datos. 3era edición.* Madrid, 2002. (Cited on pages 63 y 64.)

[55] EM. Empirical modelling research group. URL `http://www2.warwick.ac.uk/fac/sci/dcs/research/em/`. (Cited on page 5.)

[56] EpaNet. Environmental protection agency. www.epa.gov/nrmrl/wswrd/dw/epanet.html. URL `www.epa.gov/nrmrl/wswrd/dw/epanet.html`. Fecha de consulta: Marzo 2011. (Cited on page 11.)

[57] EPSG. Ogp geomatics committee. URL `http://www.epsg.org/`. (Cited on page 85.)

[58] Thomas Erl. *Service-Oriented Architecture: Concepts, Technology & Design*. Number ISBN: 0131428985. Prentice Hall Service Technology service from Thomas Erl. (Cited on page 54.)

[59] IEEE 1516 Evolved. IEEE 1516 Evolved, High Level Architecture (HLA). 2010, 2005. (Cited on pages 134 y 156.)

[60] P. M. Fearnside. *Human Carrying Capacity of the Brazilian Rainforest*. Columbia University Press, New York, 1986. (Cited on page 51.)

[61] J Ferber and J-P Muller. Influences and reaction: a model of situated multiagent systems. *Second International Conference on MultiAgent Systems*, pages pp. 72–79, 1996. URL `http://www.aaai.org/Papers/ICMAS/1996/ICMAS96-009.pdf`. (Cited on pages 9, 40 y 41.)

[62] C.W. Fetter. *Applied Hydrogeology*. University of Wisconsin - Oshkosh, 3erd edition, 1994. (Cited on page 118.)

[63] T. Finin and R. Fritzson. Kqml?a language and protocol for knowledge and information exchange. In *In Proceedings of the Thirteenth InternationalWorkshop on Distributed Artificial Intelligence*, Lake Quinalt, WA, 1994. (Cited on page 21.)

[64] D.G. Firesmith and B. Henderson-Sellers. *The OPEN Process Framework: An Introduction*. Harlow-England, 2002. (Cited on page 38.)

[65] David Flanagan. *JavaScript: The Definitive Guide*. O'Reilly Media, 2006. (Cited on pages 11 y 92.)

[66] Robert F. Flanagan. Creating system of systems software: A comparative study of hla and soa. Master's thesis, Texas Tech University, 2006. (Cited on page 134.)

[67] Herve Gallaire, Jack Minker, and Jean-Marie Nicolas. Logic and databases: A deductive approach. *ACM Comput. Surv.*, 16 (2):153–185, 1984. ISSN 0360-0300. doi: http://doi.acm.org/10.1145/356924.356929. (Cited on pages 64 y 65.)

[68] Juan C. Garcia-Ojeda, Scott A. DeLoach, Robby, Walamitien H. Oyenan, and Jorge Valenzuela. O-mase: A customizable approach to developing multiagent development processes. In *Proceddings of the 8th International Workshop on Agent Oriented Software Enginnering*, Honolulu HI, May 2007. (Cited on pages 9, 20, 37, 38 y 39.)

[69] GeoServer. Java-based software Server for Geospatial data. URL `http://geoserver.org/display/GEOS/Welcome`. (Cited on page 92.)

[70] Andreas Geppert and Klaus R. Dittrich. *Advanced Database Technology and Design*, chapter Component Database Systems, pages 403–435. Artech House computing library, 2000. (Cited on page 64.)

[71] Nigel Gilbert and Klaus G. Troitzsch. *Simulación para las Ciencias Sociales*. Mc Graw Hill, 2da. edition, 2006. (Cited on pages 3, 45 y 46.)

[72] Randy Gimblett. Integrating geographic information systems and agent-based technologies for modeling and simulating social and ecological phenomena. In Randy Gimblett, editor, *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity. 2001. (Cited on pages 45 y 46.)

[73] Randy Gimblett, editor. *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity. Oxford University Press, 2002. (Cited on pages 3 y 46.)

[74] JJ Gómez-Sanz and J Pavón. Agent oriented software engineering with ingenias. Vol. 2691 of LNCS:pp. 394–403, 2003. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.111.4611`. (Cited on pages 9, 20 y 32.)

[75] GRASS Development Team. *Geographic Resources Analysis Support System (GRASS GIS) Software*. Open Source Geospatial Foundation, 2011. URL `http://grass.osgeo.org`. (Cited on pages 5 y 166.)

[76] C Guada. Datos estación climatológica mucujún. Technical report, Laboratorio de Geofísica de la Universidad de Los Andes (LAGULA), Se encuentra en: http://www.cecalc.ula.ve/redbc/colecciones, 2000. Fecha de consulta: 15 de octubre 2010. (Cited on page 118.)

[77] gvSIG. Portal gvSIG. http://www.gvsig.org/web/, 2004. Fecha de consulta: 15 octubre 2010. (Cited on pages 11, 84 y 166.)

[78] J Hammer, H Garcia-Molina, R Cho, R Aranha, and A Crespo. Extracting semistructured information from the web. In *Workshop on Management of Semi-structured Data*, pages 18–25, 1997. (Cited on page 163.)

[79] D. Harel. *Statecharts:A visual formalism for complex systems*. 8: 231-247. Sci. Computer Program, 1987. (Cited on page 21.)

[80] IEEE. IEEE Standard for modeling and simulation (M&S) High Level Architecture (HLA) — Object Model Template Specification. 2000. (Cited on pages 145 y 147.)

[81] IEEE1516. IEEE 1516, High Level Architecture (HLA). 2001, 2001. (Cited on pages 134 y 156.)

[82] C Iglesias, M Garijo, J Gonzalez, and J.R. Velasco. A methodological proposal for multiagent systems development extending commonkads. 1996. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.54.4868. (Cited on page 35.)

[83] C Iglesias, M Garijo, J Gonzalez, and J.R. Velasco. Analysis and design of multiagent system using mas-commonkads. In M.P. Singh, A. Rao, and M.J. Wooldridge, editors, *Intelligent Agent IV: Agent Theories, Architectures and Language*, volume Lecture Notes in Computer Science 1365, 1997. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.104.717. (Cited on pages 9, 20 y 31.)

[84] IGVSB. Portal del Instituto Geográfico Venezolano Simón Bolívar. URL http://www.igvsb.gob.ve. (Cited on pages 11 y 86.)

[85] INIA. Red Agrometeorológica del INIA. URL http://agrometeorologia.inia.gob.ve/index.php?option=com_samsitemap&Itemid=34. (Cited on page 118.)

[86] R. Itami. Simulating spatial dynamics:cellular automata theory. *Landscape and Urban Planning*, 1994. (Cited on page 46.)

[87] Robert M. Itami. Mobile agents with spatial intelligence. In *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity. Randy Gimblett, 2001. (Cited on pages 9, 47 y 50.)

[88] I. Jacobson, J. Rumbaugh, and G. Booch. *The Unified Software Development Software Process*. 1999. (Cited on page 32.)

[89] Ivar Jacobson. *Object-Oriented Software Engineering*. Addison Wesley Professional, 1992. ISBN 0-201-54435. (Cited on page 21.)

[90] JAVA. JAVA Web Site. URL http://www.oracle.com/technetwork/java/index.html. (Cited on page 4.)

[91] N.R. Jennings. An agent-based approach for building complex software systems. *Communications of the ACM*, 44(4):35–41, April 2001. (Cited on pages xvii y 22.)

[92] Bin Jiang and H.Randy Gimblett. An agent-based approach to environmental and urban systems within geographic information systems. In *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity. Randy Gimblett, 2001. (Cited on pages 9, 46 y 49.)

[93] Elisabeth A. Kendall, Margaret T. Malkoun, and Chong Jiang. A methodology for developing agent based systems for enterprise integration. In D. Luckose and Zhang C., editors, *Proceedings of the First Australian Workshop on DAI*. Springer-Verlag: Heidelberg,Germany, Lecture Notes on Artificial Intelligence, 1996. URL http://citeseer.ist.psu.edu/old/398817.html. (Cited on page 26.)

[94] Elizabeth Kendall and L Zhao. Capturing and structuring goals. In *Workshop on Use Case Patterns, Object Oriented Programming Systems Languages and Architectures*, 1998. URL www.coldewey.com/europlop98/Program/Papers/Kendall.ps. (Cited on pages 26 y 27.)

[95] D. Kinny. The AGENTIS agent interaction model. In J. P. Muller, M. P. Singh, and A. S. Rao, editors, *Intelligent Agents V — Proceedings of the Fifth International Workshop on Agent Theories, Architectures, and Languages (ATAL-98)*, volume Lecture Notes in Artificial Intelligence. Springer-Verlag, Heidelberg, 1999. URL http://portal.acm.org/citation.cfm?id=749453. (Cited on pages 8, 20 y 21.)

[96] D Kinny, M Georgeff, and A Rao. A methodology and modelling technique for systems of BDI agents. In *LNCS*, volume 1038, pages 56–71, 1996. URL citeseer.ist.psu.edu/kinny96methodology.html. (Cited on pages 8 y 20.)

[97] Donald E. Knuth. Computer Programming as an Art. *Communications of the ACM*, 17(12):667–673, December 1974. (Cited on page ix.)

[98] Robert Kowalski. Database updates in the event calculus. *In Journal of Logic Programming, No. 162.*, 12:121–146., 1992. (Cited on page 5.)

[99] Robert Kowalski. Computational logic in an object-oriented world. *In: Reasoning, Action and Interaction in AI Theories and Systems - Festschrift in Honor of Luigia Carlucci Aiello. (eds. O. Stock, M. Schaerf) Springer Verlag, LNAI*, 2006. (Cited on pages 5, 18 y 66.)

[100] Robert Kowalski. *Computational Logic and Human Thinking: How to be Artificially Intelligent*. Cambridge University Press, 2011. (Cited on pages 5, 18 y 43.)

[101] Robert Kowalski and Fariba Sadri. An agent architecture that unifies rationality with reactivity. *Department of Computing, Imperial College*, 1997. URL http://www.doc.ic.ac.uk/~rak/. (Cited on pages 5, 18 y 43.)

[102] Robert Kowalski and Fariba Sadri. From logic programming towards multi-agent systems ,. *In: Annals of Mathematics and Artificial Intelligence*, 25:391–419, 1999. (Cited on pages 5 y 18.)

[103] Robert Kowalski and Fariba Sadri. Integrating logic programming and production systems in abductive logic programming agents. *In Web Reasoning and Rule Systems (eds. A. Polleres and T. Swift) Springer, LNCS 5837*, 2009. (Cited on pages 5, 18 y 65.)

[104] Robert Kowalski, Fariba Sadri, and P Soper. Integrity checking in deductive databases. *In Proceedings of VLDB, Morgan Kaufmann, Los Altos, Ca.*, pages 61–69, 1987. (Cited on page 5.)

[105] Frederick Kuhl, Richard Weatherly, and Judith Dahmann. *Creating Computer Simulation Systems. An Introduction to the High Level Architecture*. Prentice Hall PTR, 1999. (Cited on pages xviii, 102, 134 y 139.)

[106] A Lavell. Los conceptos, estudios y práctica en torno al tema de los riesgos y desastres en américa latina: Evolución y cambio, 1980-2004: El rol de la red, sus miembros y sus instituciones de apoyo. *Secretaría General, Facultad Latinoamericana de Ciencias Sociales-FLACSO*, 2005. (Cited on pages 10 y 71.)

[107] Kevin Lim, Peter J. Deadman, Emilio Moran, Eduardo Brondizio, and Sthephen McCracken. Agent-based simulations of household decision making and land use change near altamira, brazil. In *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity, pages 277–310. Randy Gimblett, 2001. (Cited on pages 9, 47 y 51.)

[108] L Liu, C Pu, and W Han. Xwrap: an xml-enabled wrapper construction system for web information sources. In *16th International Conference on Data Engineering*, pages 611–621, San Diego, CA, 2000. ISBN= 0-7695-0506-6. (Cited on page 163.)

[109] K. S. Lyytinen and M. Rossi. *METAEDIT+ — A Fully Configurable MultiUser and Multi-Tool CASE and CAME Environment*. Springer-Verlag, 1999. (Cited on page 32.)

[110] Qusay H. Mahmoud. Service-oriented architecture (soa) and web services: The road to enterprise application integration (eai). 2005. in http://www.oracle.com/technetwork/articles/javase/soa-142870.html. (Cited on page 53.)

[111] D R Maidment. Gis and hidrologic modelling. In *First International Conference/Workshop on Integrating Geographic Information Systems and Environmental Modelling*, Boulder, CO, September,

15-19 1992. URL http://www.ce.utexas.edu/prof/maidment/gishydro/meetings/santafe/santafe.htm. (Cited on page 46.)

[112] Z Manna and A Pnueli. *Temporal verification of reactive systems: Safety*. Springer, 1995. (Cited on page 22.)

[113] W. Mark. Multiagent systems engineering: A methodology for analysis and design of multiagent systems. Technical report, MS thesis, AFIT/GCS/ENG/00M-26. School of Engineering, Air Force Institute of Technology (AU), Wright-Patterson AFB, OH., 2000. URL citeseer.ist.psu.edu/mark00multiagent.html. http://macr.cis.ksu.edu/projects/mase.htm. (Cited on pages 8, 20 y 25.)

[114] Z Méndez. Metadatos estación climatológica mucujún. Technical report, Centro de Cálculo Científico de la ULA (CECALCULA), Se encuentra en: http://www.cecalc.ula/redbc, 2000. Fecha de consulta: 15 octubre de 2010. (Cited on page 118.)

[115] N. Minar, C. Burkhard, Langton, and M. Askenazi. The swarm simulation system: A toolkit for building multi-agent simulations. *Overview paper. Santa Fe Institute, Santa Fe, New Mexico*, 1996. (Cited on pages 50 y 51.)

[116] Björn Möller. *The HLA Tutorial. A practical Guide for developing distributed simulations*, 2012. URL WWW.PITCH.SE/HLATUTORIAL. (Cited on page 134.)

[117] Björn Möller and Clarence Dahlin. A first look at the hla evolved web service api. *Euro Simulation Interoperability Workshop, 06E-SIW-061, Simulation Interoperability Standards Organization, June 2006.*, 2005. (Cited on page 157.)

[118] Björn Möller and Staffan Löf. A management overview of the hla evolved web service api. *Fall Simulation Interoperability Workshop, 06F-SIW-024, Simulation Interoperability Standards Organization, September 2006.*, 2006. (Cited on page 157.)

[119] Björn Möller and Björn Löfstrand. Getting started with fom modules,. *Fall Simulation Interoperability Workshop, 09F-SIW-082, Simulation Interoperability Standards Organization, September 2009.*, 2009. (Cited on pages 134 y 142.)

[120] Björn Möller and Lennart Olsson. Practical experiences from hla 1.3 to hla ieee 1516 interoperability. *Fall Simulation Interoperability Workshop, 04F-SIW-045, Simulation Interoperability Standards Organization*, 2004. (Cited on page 157.)

[121] Beniamino Murgante, Giuseppe Borruso, and Alessandra Lapucci. Studies in computational intelligence 176. In *Geocompu-*

tation and Urban Planning, volume 176. Springer, 2009. (Cited on pages vii, viii, 3 y 46.)

[122] H.S. Nwana, D.T. Ndumu, L.C. Lee, and J.C. Collis. Zeus: A toolkit for building distributed multi-agent system. *Applied Artificial Intelligence Journal*, 1(13):129–185, 1999. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.26.277. (Cited on page 33.)

[123] J. Odell, H. Parunak, and B. Bauer. Extending uml for agents. In *Proceedings of the Agent-Oriented Information Systems Workshop at the 17th National conference on Artificial Intelligence*, 2000. URL citeseer.ist.psu.edu/odell00extending.html. (Cited on page 24.)

[124] James J. Odell, H. Van, Dyke Parunak, and Bernhard Bauer. Representing agent interaction protocols in uml. In *In OMG Document ad/99-12-01. Intellicorp Inc*, pages 121–140. Springer-Verlag, 2001. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.103.1632. (Cited on page 24.)

[125] Victor Olaya. The Sextante library. http://www.sextantegis.com/. (Cited on page 166.)

[126] Open HLA. Open HLA Web Site. URL http://sourceforge.net/projects/ohla/. (Cited on page 164.)

[127] L Padgham and M Winikoff. Prometheus: A pragmatic methodology for engineering intelligent agents. In *In Proceedings of the OOPSLA 2002 Workshop on Agent-Oriented Methodologies*, pages pp. 97–108, 2002. URL http://citeseer.ist.psu.edu/article/padgham02prometheus.html. (Cited on pages 8, 20 y 27.)

[128] L. Padgham and M. Winikoff. Prometheus: A methodology for developing intelligent agents. In *Proceedings of the Third International Workshop on AgentOriented Software Engineering, at AAMAS 2002. July, 2002, Bologna, Italy*, 2002. URL citeseer.ist.psu.edu/article/padgham02prometheus.html. (Cited on pages xvii y 28.)

[129] Virginia Padilla and Jacinto Dávila. Simulación multiagentes para gestión de desastres y reducción de riesgo. *Revista Ciencia e Ingeniería.*, Edición Especial Jornada de Modelado y Simulación., 2012. http://erevistas.saber.ula.ve/index.php/cienciaeingenieria/index. (Cited on page 73.)

[130] N. Paton and O. Diaz. *Advanced Database Technology and Design*, chapter Active Database, pages 61–89. Artech House computing library, 2000. (Cited on page 63.)

[131] C Pautasso, O Zimmermann, and F Leymann. Restful web services vs. "big" web services: making the right architectural decision. *International World Wide Web Conference Committee (IW3C2), Beijing, China*, 2008. (Cited on pages 53 y 92.)

[132] PoRTIco. The PoRTIco project. URL http://porticoproject.org/index.php?title=Main_Page. (Cited on pages 12, 95 y 164.)

[133] PostGIS. PostGIS. URL http://postgis.refractions.net/. (Cited on page 87.)

[134] PostGreSQL. The Open Source Object-Relational Database System. URL http://www.postgresql.org/. (Cited on page 87.)

[135] QGIS. QGIS quantum GIS desktop project. http://hub.qgis.org/, 2011. (Cited on page 166.)

[136] M Ramírez. Diagnóstico y reacondicionamiento del acueducto "la ceibita, buena vista". Technical report, Universidad de los Andes, 2005. (Cited on pages xix, 73, 78, 81, 82 y 118.)

[137] Maritza Ramírez. *Hidrología Aplicada*. Universidad de los Andes, 1999. (Cited on page 118.)

[138] Anand S. Rao and Michael P. Georgeff. Modeling rational agents within a bdi-architecture. *Second International Conference on Principles of Knowledge Representation and Reasoning.*, 1991. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.41.3036. (Cited on page 20.)

[139] Anand S. Rao and Michael P. Georgeff. Formal models and decision procedures for multi-agent systems. Technical report, 1995. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.52.7924. (Cited on page 20.)

[140] Anand S. Rao and Michael P. Georgeff. Bdi-agents: from theory to practice. In *Proceedings of the First Intl. Conference on Multiagent Systems*, San Francisco, 1995. URL citeseer.ist.psu.edu/rao95bdi.html. (Cited on page 20.)

[141] REDBC. Red Bioclimática del estado Mérida. URL http://www.cecalc.ula.ve/redbc/. (Cited on page 118.)

[142] David J. Robinson. A component based approach to agent specification. Technical report, MS THESIS, AFIT/ENG/ooM-22. SCHOOL OF ENGINEERING, AIR FORCE INSTITUTE OF TECHNOLOGY (AU), WRIGHT-PATTERSON AIR FORCE BASE, 2000. URL http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.9.6065. (Cited on page 26.)

[143] Lewis A. Rossman. *EPANET*. Water Supply and Water Resources Division. National Risk Management Research Laboratory. Office of Research and Development. U.S. Environmental Protection Agency, Cincinnati, OH, versión 2.0 vE edition, 2010. Traducción: Grupo Multidisciplinar de Modelación de Fluidos Universidad Politécnica de Valencia. (Cited on pages 77, 78 y 84.)

[144] Arnon Rotem-Gal-Oz. What is soa anyway? 2007. in http://www.rgoarchitects.com/Files/SOADefined.pdf. (Cited on page 53.)

[145] E. Rudolph, J. Grabowski, and P. Graubmann. Tutorial on message sequence charts (msc). In *Proceedings of FORTE/PSTV-96 Conference*, October, 1996. URL `http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.29.6230`. (Cited on page 36.)

[146] James Rumbaugh, Michael Blaha, William Premerlani, Frederick Eddy, and William Lorensen. *Object-Oriented Modeling and Design*. Prentice Hall, 1991. ISBN ISBN 0-13-629841-9. (Cited on pages 3, 17, 26 y 27.)

[147] S Russell and P Norvig. *Inteligencia Artificial: Un enfoque moderno. 7ma edición*. Pearson, España, 2004. (Cited on pages 3, 17, 18 y 28.)

[148] Guus Schreiber, Bob J. Wielinga, Robert de Hoog, Hans Akkermans, and Walter Van de Velde. Commonkads: A comprehensive methodology for kbs development. Technical Report 6, 1994. URL `http://www2.computer.org/portal/web/csdl/doi/10.1109/64.363263`. (Cited on page 35.)

[149] Roy Scrudder, William White, Marcus Richardson, and Robert Lutz. Graphical representation of the federation development and execution process. *Simulation Interoperability Worrkshop Proceedings*, Paper Number 98F-SIW-103:584–606, 1998. (Cited on page 95.)

[150] Joaquín Bosque Sendra. *Sistemas de Información Geográfica*. Ediciones Rialp, S.A., Madrid, 1997. (Cited on page 45.)

[151] Y Shoham. AGENTo: A simple agent language and its interpreter. *AAAI-91 Proceedings*, (No. 91):pp. 704–709, 1991. (Cited on page 18.)

[152] Y. Shoham. *Agent Oriented Programming*, volume 60, pages 51–92. Artificial Intelligence, 1993. (Cited on page 32.)

[153] A. Silberschatz, M. Stonebraker, and J. Ullman(eds.). Database research: Achievements and opportunities into the 21st century. *SIGMOD Record*, 25(1):52–63, 1996. (Cited on page 63.)

[154] HA Simon. A behavioral model of rational choice. *Quarterly Journal of Economics*, pages pp. 99–118, 1955. (Cited on page 43.)

[155] SOA. Soa glossary, consulting: may 2012. URL http://www.soaglossary.com/. (Cited on pages 11, 53 y 54.)

[156] SWI-Prolog. SWI-Prolog's home. URL http://www.swi-prolog.org/. (Cited on pages 4 y 110.)

[157] Ernest Teniente. *Advanced Database Technology and Design*, chapter Deductive Database, pages 91–136. Artech House computing library, 2000. (Cited on page 63.)

[158] Paul M. Torrens. Agent-based model and the spatial science. *Geography Compass*, 4/5:428–448, 2010. (Cited on pages 3 y 46.)

[159] HM Valdés. Reducción de desastres como un derecho humanos. *Sistemas de Naciones Unidas*, 1999. Se encuentra en: http://www.radixonline.org/humanrights.htm. Fecha de consulta: 04 de octubre de 2010. (Cited on pages 10, 71 y 72.)

[160] W3C. Web Service Glossary. W3C Working Group Note 11 February 2004. URL http://www.w3.org/TR/ws-gloss/. (Cited on page 53.)

[161] W3C. Linking open data. www.w3.org/ / CommunityProjects/LinkingOpenData, 2011. URL www.w3.org/wiki/SweoIG/TaskForces/CommunityProjects/LinkingOpenData. Fecha de consulta: Octubre 2011. (Cited on page 163.)

[162] Gabriel Wainer. *Discrete-Event Modeling and Simulation*. 2009. (Cited on pages 4, 42 y 63.)

[163] James D. Westervelt. Geographic information systems and agent-based modeling. In *Integrating Geographic Information Systems and Agent-based Modeling Techniques*, volume Santa Fe Institute studies in the Sciences of Complexity. Randy Gimblett, 2001. (Cited on pages xix, 9, 46, 47, 48 y 49.)

[164] Roger White. Cities and cellular automata. *Discrete Dynamics in Nature and Society*, 2:111–125, 1998. (Cited on page 49.)

[165] M. Winikoff, L. Padgham, and J. Harland. Simplifying the development of intelligent agents. In M. Stumptner, D. Corbett, and M. Brooks, editors, *AI2001: Advances in Artificial Intelligence. 14th Australian Joint Conference on Artificial Intelligence*, pages 555–568. Springer, LNAI 2256, 2001. URL http://citeseerx.

ist.psu.edu/viewdoc/summary?doi=10.1.1.23.3955. (Cited on page 28.)

[166] Mark F. Wood and Scott DeLoach. An overview of the multiagent systems engineering methodology. In *Proceedings of the First International Workshop on Agent-Oriented Software Engineering*, pages 207–222, 2000. URL citeseer.ist.psu.edu/wood00overview.html. (Cited on pages xvii y 25.)

[167] Michael Wooldridge, Nicholas R. Jennings, and David Kinny. The gaia methodology for agent-oriented analysis and design. *Autonomous Agents and Multi-Agent Systems*, 3(3):285–312, 2000. URL citeseer.ist.psu.edu/wooldridge00gaia.html. (Cited on page 30.)

[168] E. Yu. *Modelling Strategic Relationships for Process Reengineering*. PhD thesis, University of Toronto, Department of Computer Science, 1995. (Cited on page 33.)

[169] ITU-T. Z100. Ccitt specification and description language (sdl). Technical report, ITU-T, 1994. (Cited on page 37.)

[170] G Zaccak. Wrapster : semi-automatic wrapper generation for semi-structured websiteszaccak2007. Master's thesis, MIT. Dept. of Electrical Engineering and Computer Science, 2007. URL http://hdl.handle.net/1721.1/40537. (Cited on page 163.)

[171] F. Zambonelli, N. Jennings, and M. Wooldridge. Developing multiagent systems: the gaia methodology. In *ACM Transactions on Software Engineering and Methodology, 12(3).*, 2003. URL citeseer.ist.psu.edu/zambonelli03developing.html. (Cited on pages xvii, 8, 20, 22 y 23.)

[172] Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modelling and Simulation*. Second edition, 2000. (Cited on pages 4, 42 y 63.)

COLOFÓN

Este documento fue hecho usando el estilo tipográfico "mira-y-siente" `classicthesis` desarrollado por André Miede. El estilo fue inspirado por el libro seminal de tipografía *"The Elements of Typographic Style"* de Robert Bringhurst. `classicthesis` está disponible para LaTeX y LyX:

`http://code.google.com/p/classicthesis/`

Usualmente los usuarios agradecidos por `classicthesis` envian al autor un mensaje de gratitud, la colección de mensajes pueden verse en:

`http://postcards.miede.de/`

Final Version as of June 6, 2013 (`classicthesis`).