

ANEMONA: UNA METODOLOGÍA MULTI AGENTE PARA SISTEMAS HOLÓNICOS DE FABRICACIÓN

Autor: Adriana Giret Boggino

Director: Dr. D. Vicente Botti Navarro

PARA LA OBTENCIÓN DEL GRADO DE
DOCTOR EN INFORMÁTICA
POR LA
UNIVERSIDAD POLITÉCNICA DE VALENCIA
Valencia, España
JULIO 2005

Fecha: **Julio 2005**

Autor: **Adriana Giret Boggino**

Director: **Dr. D. Vicente Botti Navarro**

Título: **ANEMONA: Una Metodología Multi Agente para Sistemas Holónicos de Fabricación**

Departamento: **Sistemas Informáticos y Computación**

Universidad: **Universidad Politécnica de Valencia**

Grado: **Doctor** Mes: **Julio** Año: **2005**

Firma del Autor

Tabla de Contenidos

Tabla de Contenidos v

Índice de Tablas ix

Índice de Figuras x

Resumen xv

Abstract xviii

Resum xx

1. Introducción 1

1.1. Sistema de Fabricación 3
1.2. Los ordenadores en la Fabricación 5
1.3. La nueva fabricación 8
1.4. Sistemas Holónicos de Fabricación 11
1.4.1. Las raíces de "lo holónico" 12
1.4.2. El proyecto HMS 14
1.4.2.1. El Caso de Estudio HMS 14
1.4.3. Escenario de fabricación holónica 16
1.4.4. Comparación con los enfoques existentes 18
1.4.4.1. Control Jerárquico 18
1.4.4.2. Control Heterárquico 19
1.4.4.3. Sistemas de Fabricación Holónicos versus control jerárquico y heterárquico 20
1.4.4.4. Fabricación Holónica versus paradigmas similares 20
1.5. Motivación y Objetivos 22

1.6. Estructura del Trabajo 23

2. Estado del Arte del HMS **25**
2.1. Desarrollo de Sistemas Holónicos de Control 25
2.1.1. Arquitecturas para control holónico 25
2.1.2. Algoritmos para control holónico 37
2.1.2.1. Planificación 37
2.1.2.2. Scheduling 39
2.1.2.3. Ejecución y Control de Taller 40
2.1.2.4. Control de Máquina y de Dispositivo 40
2.2. Modelado de Sistemas Holónicos de Fabricación 41
2.2.1. Requisitos de modelado 41
2.2.1.1. Requisitos Funcionales 41
2.2.1.2. Requisitos de Ingeniería del Software 43
2.2.2. Métodos HMS 46
2.2.3. Métodos Multi Agente 48
2.2.3.1. Métodos SMA de propósito general 48
2.2.3.2. Métodos SMA para sistemas de fabricación . 58
2.2.4. Modelado de Empresas 60
2.2.5. Resumen Comparativo 62
2.3. Conclusiones 66

3. Holones o Agentes **69**
3.1. Agente 69
3.2. Holón 73
3.3. Comparativa 74
3.3.1. Motivación y origen de cada enfoque 74
3.3.2. Características 75
3.3.2.1. Autonomía 75
3.3.2.2. Reactividad 75
3.3.2.3. Pro-actividad 78
3.3.2.4. Habilidad Social 79
3.3.2.5. Cooperación 80
3.3.2.6. Re-organización 81
3.3.2.7. Racionalidad 82
3.3.2.8. Actitudes Mentales 82
3.3.2.9. Aprendizaje 83
3.3.2.10. Benevolencia 84
3.3.2.11. Movilidad 84

3.3.2.12. Recursión 85
3.3.2.13. Procesamiento Físico y de la Información . . 85
3.3.3. Resumen Comparativo 86
3.4. La recursión en Agentes 86
3.5. Conclusiones 91

4. Agente Abstracto **93**
4.1. Definiciones 94
4.1.1. ¿Por qué el Agente Abstracto? 97
4.2. Comportamiento de un Agente Abstracto 101
4.2.1. Notación 101
4.2.2. Comportamiento de un Sistema Multi Agente 102
4.2.2.1. Reactivo 102
4.2.2.2. Intencional 104
4.2.3. Comportamiento de un Agente Abstracto 118
4.2.3.1. Percepciones 118
4.2.3.2. Acciones 118
4.2.3.3. Objetivos 119
4.2.3.4. Creencias 119
4.3. Conclusiones 119

5. Metodología Multi Agente para Sistemas Holónicos de Fabricación **123**
5.1. Antecedentes 124
5.2. Notación 127
5.2.1. Meta-modelo 127
5.2.2. Meta-modelos del Sistema Multi Agente 129
5.2.2.1. Entidades Básicas 131
5.2.2.2. Meta-modelo de Agente 132
5.2.2.3. Meta-modelo de Tareas y Objetivos 139
5.2.2.4. Meta-modelo de Interacción 144
5.2.2.5. Meta-modelo de Entorno 148
5.2.2.6. Meta-modelo de Organización 151
5.2.3. Notación de los modelos 156
5.3. Proceso de Desarrollo 156
5.3.1. SPEM 157
5.3.2. El método 158
5.3.2.1. Requisitos del Sistema 161
5.3.2.2. Análisis 163

5.3.2.3. Diseño 181
5.3.2.4. Implementación de Holones 189
5.3.2.5. Instalación y Configuración 193
5.3.2.6. Operación y Mantenimiento 193
5.4. Conclusiones 193

6. Caso de Estudio **201**
6.1. Requisitos 201
6.1.1. Organigrama/Departamentos 202
6.1.2. Procesos de Negocio 207
6.1.3. Alcance del Sistema 210
6.1.4. Procesos a Controlar 211
6.1.5. Condiciones de Operación 219
6.1.6. Objetivos 219
6.2. Análisis 222
6.2.1. Iteración 1 222
6.3. Diseño 237
6.4. Conclusiones 241

7. Conclusiones y Trabajos Futuros **243**
7.1. Contribuciones destacadas 243
7.2. Líneas Futuras de Investigación 246
7.3. Publicaciones Relacionadas con la Tesis 248
7.3.1. Artículos en Revistas 248
7.3.2. Artículos en Congresos Internacionales 248
7.3.3. Artículos en Congresos Nacionales 249
7.3.4. Capítulos de Libro 250

Índice de tablas

2.1. Métodos de desarrollo y Requisitos de modelado de Sistemas Holónicos de Fabricación. 64

3.1. Holones Vs. Agentes 87

5.1. Requisitos de modelado para HMS 126
5.2. Elementos de definición de procesos de SPEM 159
5.3. Guías HMS-CU 170
5.4. Guías PROSA - Parte 1 177
5.5. Guías PROSA - Parte 2 178
5.6. Guías PROSA - Parte 3 179
5.7. Guías JADE - Parte 1 190
5.8. Guías JADE - Parte 2 191
5.9. Guías Bloques Funcionales 192

6.1. Condiciones de Operación - Parte 1 220
6.2. Condiciones de Operación - Parte 2 221
6.3. Iteración 1, Objetivos del Sistema 223
6.4. Holones atómicos de KCG - Parte 1 238
6.5. Holones atómicos de KCG - Parte 2 239

Índice de figuras

1.1. Actividades de un Sistema de Fabricación 4
1.2. Esquema de un Sistema CIM 7
1.3. Escenario ideal de un Sistema Holónico de Fabricación 18

2.1. Arquitectura general de un holón 26
2.2. Arquitectura orientada a agentes para holones 27
2.3. PROSA: arquitectura de referencia 29
2.4. Arquitectura basada en agentes y bloques funcionales 30
2.5. Dominio de Cooperación 31
2.6. Holarquías y Sociedad de Holones 33
2.7. La arquitectura HCD 34
2.8. INTERRAP 35
2.9. Jerarquía de control de fabricación típica 38

3.1. Ejemplo de un Sistema de Fabricación simplificado. 77
3.2. Arquitectura General de un Holón. 80

4.1. Agente Recursivo 96
4.2. Compañías Nacionales de la Multinacional AG. 98
4.3. Compañías Regionales de la Multinacional AG. 98
4.4. Tres niveles de abstracción en la Multinacional AG. 99
4.5. Cuatro niveles de abstracción en la Multinacional AG. 100
4.6. Cinco niveles de abstracción en la Multinacional AG. 100

4.7. Congruencia y Coherencia 108

5.1. Estructura de cuatro niveles utilizada en el meta-modelado. . 128
5.2. Expresión BNF para los nombres de las relaciones y de los roles. 130
5.3. Entidades básicas. 131
5.4. Rol, Agente Abstracto y Objetivo. 134
5.5. Rol, Agente Abstracto, Tarea y Estado Mental. 136
5.6. Entidades Mentales. 138
5.7. Estado Mental y Entidades Mentales. 139
5.8. Consulta de Entidad Autónoma. 139
5.9. Objetivos y Tareas. 140
5.10. La tipos de relación entre Tareas y Objetivos. 142
5.11. Descomposición de Objetivos y dependencia entre Objetivos. . 143
5.12. Descripción de Tareas. 144
5.13. Meta-modelo de interacción, Agente Abstracto. 146
5.14. Especificación ModA2 de una Interacción. 147
5.15. Interacción, Organización y Objetivo. 148
5.16. Meta-modelo de Entorno: Recurso. 149
5.17. Meta-modelo de Entorno: Aplicacion. 150
5.18. Tarea Abstracta, Recurso, Aplicación y Entidad Mental. . . . 150
5.19. Organización. Visión Estructural. 151
5.20. Descripción de un Flujo de Trabajo 152
5.21. Elementos de un Flujo de Trabajo y sus relaciones. 153
5.22. Descomposición de Tareas y Flujos de Trabajo. 154
5.23. Asociación de Tarea y Agente Abstracto. 155
5.24. Relaciones Sociales. 155
5.25. Elementos de la notación gráfica de los modelos. 156
5.26. Fases del Método. 160
5.27. Documento de Requisitos. 162
5.28. Niveles de Abstracción en la fase de Análisis. 164

5.29. Fase de Análisis. 165
5.30. Modelos de Análisis. 167
5.31. Definición de la tarea Determinar Casos de Uso. 168
5.32. Definición de la tarea Especificar la Realización de Casos de Uso. 171
5.33. Definición de la tarea Identificar Holones. 174
5.34. Definición de la tarea Especificar Relaciones con el Entorno. . . 180
5.35. Fase de Diseño. 182
5.36. Modelos de Diseño. 183
5.37. Niveles de Recursión en la fase de Diseño. 184
5.38. Definición de la tarea Refinar la Especificación de Holones. . . . 185
5.39. Definición de la tarea Construir la Arquitectura del Sistema. . . 188
5.40. Definición del documento Plantilla de Agente JADE. 197
5.41. Definición del documento Especificación de Interfaces de Bloques Funcionales. 198
5.42. Fase de Implementación de Holones. 199

6.1. Organigrama de KCG. 206
6.2. Procesos de Negocio de KCG. 208
6.3. Proceso de Fabricación de KCG. 214
6.4. Horno Continuo de Rodillo para azulejos cerámicos. 218
6.5. Iteración 1, Modelo de Casos de Uso. 223
6.6. Iteración 1, Modelo de Organización. 224
6.7. Iteración 1, Modelo de Interacción. 225
6.8. Iteración 1, Tareas Abstractas en el Modelo de Organización. . 227
6.9. Iteración 1, Modelo de Tareas y Objetivos. 228
6.10. Iteración 1, Agentes Abstractos y Modelo de Organización. . . 229
6.11. Iteración 1, Modelo de Agente. Holón de Fabricación. 231
6.12. Iteración 1, Modelo de Agente. Holón de Programación. 232
6.13. Iteración 1, Modelo de Agente. Holón orden de Simulación de Pedido. 232

6.14. Iteración 1, Modelo de Interacción. Solicitar Materia Prima. . 233
6.15. Iteración 1, Modelo de Interacción. Simular Programación de Lote. 234
6.16. Iteración 1, Modelo de Interacción. Procesar Orden de Fabricación. 235
6.17. Iteración 1, Modelo de Interacción. Secuencia de mensajes de la interacción Procesar Orden de Fabricación. 236
6.18. Arquitectura del Sistema. Plantilla de Agente JADE del Holón orden de Simulación de Pedido. 240

Resumen

En el área de Sistemas Inteligentes de Fabricación es definitiva la necesidad de contar con metodologías para Sistemas Holónicos de Fabricación (HMS - Holonic Manufacturing Systems), basadas en principios de ingeniería del software, que asistan al ingeniero del software en todas las fases de desarrollo y que provean guías de análisis y diseño claras y no ambiguas. Nosotros estamos convencidos que metodologías del área de Sistemas Multi Agentes (SMA) son buenas candidatas para el modelado de HMS. Algunas razones son: las similitudes entre el enfoque holónico y el enfoque de agentes, la amplia utilización de agentes como la herramienta de implementación de HMS, y la disponibilidad de metodologías SMA completas. No obstante, existen algunas extensiones que debemos incluir a una metodología SMA para que sea capaz de modelar los requisitos HMS de manera adecuada: la estructura recursiva de holones, niveles de abstracción en el sistema, guías de modelado específicas para HMS y un enfoque de análisis y diseño mixto (ascendente y descendente).

En esta tesis proponemos el Agente Abstracto como entidad de modelado para entidades autónomas con estructuras recursivas. La definición de Agente Abstracto extiende la definición tradicional de agente agregando una perspectiva estructural al concepto de agente: "... *un Agente Abstracto puede ser un agente; o puede ser un SMA que a su vez está compuesto de Agentes Abstracto* ...". El Agente Abstracto es un intento de unificación de los conceptos de agente y holón y de simplificación de las etapas de análisis y diseño. De esta manera, será más fácil traducir los productos de modelado obtenidos a partir

de metodologías para HMS en elementos ejecutables para la implementación del sistema holónico.

La contribución principal de esta tesis es una Metodología Multi Agente para Sistemas Holónicos de Fabricación. Esta metodología está basada en la noción de Agente Abstracto y en los requisitos de de modelado de HMS. Nuestra metodología define un proceso de desarrollo mixto, y provee guías específicas para HMS que ayudan al ingeniero del software a identificar e implementar holones. En nuestro enfoque el HMS se especifica dividiéndolo en aspectos más concretos que forman diferentes vistas del sistema: modelo de agente, modelo de organización, modelo de interacción, modelo de entorno y modelo de tareas y objetivos. La forma en la que estas vistas están definidas se basa en la metodología INGENIAS. Las extensiones que hemos hecho a los meta-modelos de INGENIAS tratan con la noción de Agente Abstracto, la redefinición de algunas de relaciones para adaptarse a las entidades de modelados nuevas y la inclusión de características de modelado de tiempo real de la metodología RT-MESSAGE.

El proceso de desarrollo de nuestra metodología ofrece al ingeniero del software guías claras y específicas para HMS, y fases de desarrollo completas para el ciclo de vida del HMS. La primera fase *Análisis de Requisitos del Sistema* y la segunda *Identificación y Especificación de Holones* definen la etapa de análisis de nuestro enfoque. El objetivo de la etapa de análisis es proveer una especificación de alto nivel del HMS a partir de los requisitos del sistema. La etapa de análisis adopta un enfoque descendente recursivo. Una ventaja de un análisis recursivo es que sus resultados, esto es los *Modelos de Análisis*, definen un conjunto de elementos básicos y reglas de composición. La siguiente etapa en el proceso de desarrollo es el *Diseño de Holones* que es un proceso ascendente para producir la *Arquitectura del Sistema* a partir de los *Modelos de Análisis* de la etapa anterior. El objetivo de la etapa de *Implementación de Holones* es producir el *Código Ejecutable* para la etapa de

Instalación y Configuración. Finalmente las actividades de mantenimiento se llevan a cabo en la etapa de *Operación y Mantenimiento*.

Abstract

In the Intelligent Manufacturing field, there is a definitive need, to have methodologies for holonic systems, based on software engineering principles, which assist the system designer in every development steps and provide clear, unambiguous analysis and design guidelines. We believe that methodologies from the Multi Agent Technology are good candidates for modeling HMS. Some reasons are: the similarities between the holonic and the agent approaches, the wide use of agents as the implementation tool for holonic systems, and the availability of complete Multi Agent System Methodologies. Nevertheless, there are some extensions we have to include to a MAS methodology to be able to model the HMS requirements in a proper way: holons recursive structure, systems abstraction levels, HMS specific guidelines and a mixed top-down and bottom-up approach for analysis and design steps.

In this thesis we propose an Abstract Agent notion as a modeling artifact for autonomous entities with recursive structures. The Abstract Agent extents the traditional agent definition adding an structural perspective to the agent concept: "*... an Abstract Agent can be an agent; or it can be a MAS made up of Abstract Agents ...*". The Abstract Agent is an attempt to unify the concepts of holons and agents and to simplify and close the gap between holons and agents in analysis and design steps. In this way, it will be easer to translate modelling products obtained from methodologies for HMS into coding elements for the implementation of the holonic system.

The main contribution of this thesis is a MAS methodology for HMS analysis and design based on the Abstract Agent notion and on the HMS requirements. Our methodology defines a mixed top-down and bottom-up development process, and provides specific HMS guidelines to help the designer in identifying and implementing holons. In our approach the HMS is specified dividing it in more concrete aspects that form different *views* of the system: agent model, organizational model, interaction model, environment model and task/goal model. The way in which the views (models) are defined is inspired by INGENIAS methodology. The extensions we have made to the INGENIAS meta-models deal with the addition of the Abstract Agent notion, the redefinition of some relations to conform with the new modeling entities, the dependencies between them and real time modeling issues from the RT-Message methodology.

The development process of our methodology tries to provide to HMS designer clear and HMS specific modeling guidelines, and complete development phases for the HMS life cycle. The first stage, *System Requirements Analysis* and the second stage *Holons Identification and Specification* define the analysis phase of our approach. The aim of the analysis phase is to provide high-level HMS specifications from the problem *Requirements*, which are specified by the *Client/User* and which can be updated by any development stage. The analysis adopts a top-down recursive approach. One advantage of a recursive analysis is that its results, i.e the *Analysis Models*, provide a set of elementary elements and assembling rules. The next step in the development process is the *Holons Design* stage which is a bottom-up process to produce the *System Architecture* from the *Analysis Models* of the previous stage. The aim of the *Holons Implementation* stage is to produce an *Executable Code* for the *SetUp and Configuration* stage. Finally maintenances functions are executed in the *Operation and Maintenance* stage.

Resum

En l'àrea de Sistemes Intel·ligents de Fabricació és definitiva la necessitat de contar amb metodologies per a Sistemes Holonics de Fabricació (HMS - Holonic Manufacturing Systems), basades en principis d'enginyeria del programari, que assisteixen a l'enginyer del programari en totes les fases de desenvolupament i que proveixen guies d'anàlisis i disseny clares i no ambigües. Nosaltres estem convençuts que metodologies de l'àrea de Sistemes Multi Agents (SMA) són bones candidates per al modelatge de HMS. Algunes raons són: les similituts entre l'enfocament holónic i l'enfocament d'agents, l'àmplia utilització d'agents com l'eina d'implementació de HMS, i la disponibilitat de metodologies SMA completes. No obstant, existeixen algunes extensions que hem d'incloure a una metodologia SMA perquè sigui capaç de modelar els requisits HMS de manera adequada: l'estructura recursiva de holons, nivells d'abstracció en el sistema, guies de modelatge específiques per a HMS i un enfocament d'anàlisi i disseny mixt (ascendent i descendent).

En aquesta tesi proposem l'Agent Abstracte com entitat de modelatge per a entitats autònomes amb estructures recursives. La definició d'Agent Abstracte estén la definició tradicional d'agent agregant una perspectiva estructural al concepte d'agent: "... *un Agent Abstracte pot ser un agent; o pot ser un SMA que al seu torn està compost d'Agents Abstracte* ...". L'Agent Abstracte és un intent de unificació dels conceptes d'agent i holón i de simplificació de les etapes d'anàlisis i disseny. D'aquesta manera, serà més fàcil traduir els productes de modelatge obtinguts a partir de metodologies per a HMS en

elements executables per a la implementació del sistema holonic.

La contribució principal d'aquesta tesi és una Metodologia Multi Agent per a Sistemes Holonics de Fabricació. Aquesta metodologia està basada en la noció d'Agent Abstracte i en els requisits de modelatge de HMS. La nostra metodologia defineix un procés de desenvolupament mixt, i proveix guies específiques per a HMS que ajuden a l'enginyer del programari a identificar i implementar holons. En el nostre enfocament el HMS es especificat dividintlo en aspectes més concrets que formen diferents vistes del sistema: model d'agent model d'organització, model d'interacció, model d'entorn i model de tasques i objectius. La forma en la qual aquestes vistes estàn definidaes es basa en la metodologia INGENIAS. Les extensions que hem fet als meta-models d'INGENIAS tracten amb la noció d'Agent Abstracte, la redefinició d'algunes relacions per a adaptar-se a les entitats de modelatges noves i la inclusió de característiques de modelatge de temps real de la metodologia RT-MESSAGE.

El procés de desenvolupament de la nostra metodologia ofereix a l'enginyer del programari guies clares i específiques per a HMS, i fases de desenvolupament completes per al cicle de vida del HMS. La primera fase *Anàlisi de Requisits del Sistema* i la segona *Identificació i Especificació de Holons* definixen l'etapa d'anàlisi del nostre enfocament. L'objectiu de l'etapa d'anàlisi és proveir una especificació d'alt nivell del HMS a partir dels requisits del sistema. L'etapa d'anàlisi adopta un enfocament descendent recursiu. Un avantatge d'una anàlisi recursiu és que els seus resultats, és a dir, *Models d'Anàlisi*, definixen un conjunt d'elements bàsics i regles de composició. La següent etapa en el procés de desenvolupament és el *Disseny de Holons* que és un procés ascendent per a produir *l'Arquitectura del Sistema* a partir dels *Models d'Anàlisi* de l'etapa anterior. L'objectiu de l'etapa d'*Implementació de Holons* és produir el *Codi Executable* per a l'etapa de *Instalació i Configuració*. Finalment les activitats de manteniment es porten a terme en l'etapa *d'Operació i Manteniment*.

Capítulo 1

Introducción

La manufactura en su sentido más amplio, es el proceso de convertir la materia prima en productos. Incluye: el diseño del producto, la selección de la materia prima, y la secuencia de procesos a través de los cuales será fabricado el producto [Kalpakjian y Schmid, 2002].

En castellano es más común el uso de la palabra fabricar para indicar la acción de producir objetos en serie, generalmente por medios mecánicos [1]. A lo largo de esta tesis utilizaremos indistintamente, para indicar lo mismo, las palabras fabricación y manufactura.

La palabra manufactura se deriva del latín *manu factus*, que significa hecho a mano. La palabra manufactura apareció por primera vez en 1567, y la palabra manufacturar [2] en 1683. En el sentido moderno, la manufactura involucra la fabricación de productos a partir de materias primas mediante varios procesos, maquinarias y operaciones, a través de un plan bien organizado para cada actividad requerida.

La definición de fabricación/manufactura como la acción de hacer/producir productos/artículos revela poco sobre la complejidad del problema. Una definición más específica la da CAM-I (Consortium for Advanced Manufacturing International): "Una serie de actividades y operaciones interrelacionadas que involucran diseño del producto, maquinarias y herramientas, planificación de

[1] Definición del diccionario de la real academia española.

[2] Según el diccionario de la real academia española manufacturar significa fabricar con medios mecánicos

procesos, materiales, compras, manufactura (la producción o fabricación propiamente dicha), servicios de apoyo, marketing, ventas, envío y servicio al cliente".

A pesar de que es difícil ser más preciso, la manufactura existe desde hace aproximadamente 5000-4000 a.C. Es más antigua que la historia registrada, porque los símbolos primitivos y los dibujos en las cuevas o grabados en piedra, se hacían con algún tipo de pincel o de instrumento primitivo utilizando una "pintura" o algún medio de grabar la roca; para estas aplicaciones se tuvieron que hacer herramientas apropiadas. La manufactura de productos para diversos usos, se inició con la producción de artículos hechos de madera, cerámica, piedra y metal. Los materiales y procesos que se utilizaron primero para formar productos mediante la fundición y la forja, han venido desarrollándose gradualmente a través de los siglos, utilizando nuevos materiales y operaciones más complejas, a tasas crecientes de producción y niveles más elevados de calidad.

Hasta la Revolución Industrial, que se inició en Inglaterra en los años 1750, los artículos habían sido producidos en lotes, apoyándose mucho en la mano de obra en todos los aspectos de la producción. Con la revolución industrial el poder mecánico suplantó al poder físico de los trabajadores, con varias máquinas movidas por cintas desde un eje impulsor común. La mecanización moderna se inició en Inglaterra y en Europa con el desarrollo de la maquinaria textil y de las máquinas herramientas para el corte de metales. Esta tecnología fue rápidamente trasladada a Estados Unidos, donde fue desarrollada aun más, incluyendo adelantos importantes en el diseño, manufactura y uso de piezas intercambiables.

Pronto siguieron más desarrollos, dando por resultado numerosos productos. A partir del inicio de la década de 1940, surgieron varios hitos en todos los aspectos de la manufactura. Especialmente importantes los referentes al sistema de manufactura y sus herramientas, como son: Control Automático, Control Numérico (NC), Manufactura Integrada por Computador (CIM), Robots Industriales, Sistemas de Manufactura Flexible, etc. Muchos coinciden que estos adelantos constituyen la segunda revolución industrial. Una característica de esta etapa, además de la posibilidad de reemplazar la mayoría -

o potencialmente todas - las labores físicas, es que también es posible mejorar e incluso algunas veces reemplazar el esfuerzo mental. En la fabricación esto implica la introducción de la "verdadera automatización", con sensores apropiados que proveen el feedback que permite que los dispositivos de control tomen acciones "inteligentes".

1.1. Sistema de Fabricación

La palabra *sistema* [3] se deriva del griego *systema*, que quiere decir combinar. Hoy significa un arreglo de entidades físicas, que se caracteriza por sus parámetros identificables y cuantificables de interacción. La fabricación implica un sistema con una gran cantidad de actividades interdependientes, formadas por distintas entidades (como materiales, herramientas, máquinas, energía y seres humanos).

La fabricación es un sistema complejo, porque está formado de muchos elementos distintos, físicos y humanos, algunos de los cuales son difíciles de predecir y controlar, como el suministro y el coste de materias primas, cambios en el mercado y la conducta y desempeño humanos. Puede ser difícil modelar un sistema tan complejo, por la falta de datos detallados o confiables sobre muchas de las variables que intervienen.

La Figura 1.1 resume las actividades más importantes dentro de un sistema de fabricación. En esta figura se puede observar un conjunto de pasos o etapas de todo un ciclo que en conjunto definen al sistema de fabricación. 1) Una entidad de fabricación (una compañía o una sucursal de una corporación mayor) usualmente posee alguna especialidad, tal como una tecnología específica, conocimiento o equipos. Para explotar esta capacidad se necesita que se identifiquen los mercados apropiados, la magnitud estimada de los mismos y los competidores potenciales. Luego de que se estudia el mercado y se proyecta su desarrollo futuro, se identifican o desarrollan los productos. A partir de entonces el departamento de ventas puede aceptar órdenes de trabajo. 2) Se diseña el producto para cumplir con su función prevista, sea el producto

[3] Conjunto de cosas que relacionadas entre sí ordenadamente contribuyen a determinado objeto (diccionario de la real academia española).

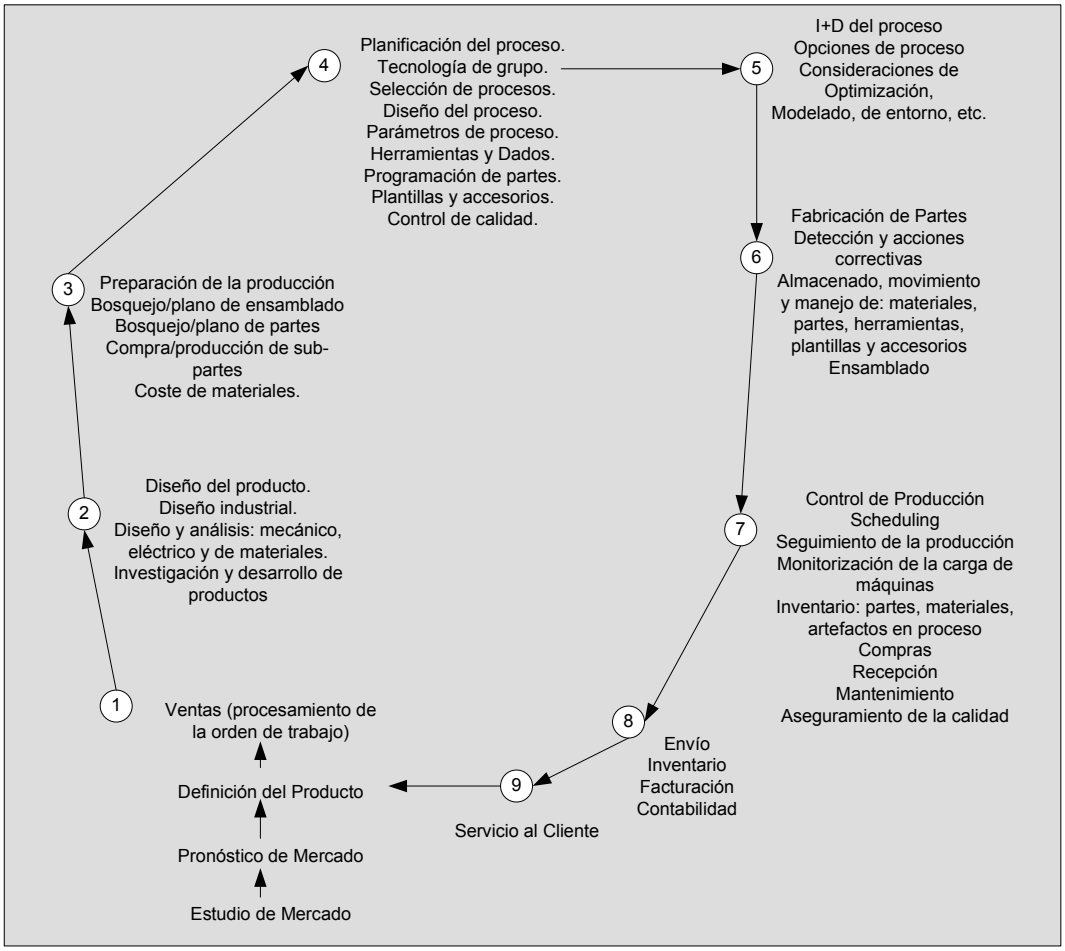


Figura 1.1: Actividades de un Sistema de Fabricación

una máquina herramienta, electrodoméstico, producto de construcción, computador, automóvil, avión, planta de procesamiento químico, central eléctrica, un envase para refrescos, etc. 3) Una vez que el producto esté diseñado, se preparan planos o bosquejos de producción para las partes o ensamblados. En este punto se pueden tomar decisiones sobre las partes que serán compradas a proveedores y aquellas que serán producidas por la misma fábrica. Se prepara un presupuesto de materiales, que en muchos casos es central para el proceso de fabricación. 4) Para los componentes que se producirán en la misma fábrica, se lleva a cabo un diseño del proceso de producción. Se selecciona el mejor proceso y se eligen los parámetros de proceso para optimizar la calidad y las

propiedades del producto final. 5) La elección de la técnica apropiada de fabricación y su optimización son funciones importantes. El desarrollo y pruebas de estas técnicas en la planta de producción pueden ser muy caros. Es por ello que los fundamentos de los procesos son explorados en laboratorio. 6) El proceso de producción propiamente dicho se lleva a cabo en los talleres o instalaciones de trabajo, las cuales se organizan según alguna configuración de planta. Durante la producción, se pueden observar características críticas de los procesos; las dimensiones, calidad, etc., de las partes se verifican de manera sistemática y, cuando se necesitan, se adoptan acciones correctivas. Una de las más importantes funciones auxiliares es el movimiento de materias primas, de partes parcialmente terminadas, herramientas, plantillas y accesorios. Finalmente, las partes fabricadas y compradas se ensamblan en productos los cuales, después de la verificación, están listos para el envío. 7) Las secuencias complejas de producción requieren de buena organización de la fabricación. Los materiales, las partes y las herramientas deben ser encaminados hacia sus destinos y programados para llegar en el momento adecuado. El estado de la producción debe ser conocido. Se debe mantener un inventario actualizado de las partes en proceso, combinado con inventarios de materiales y partes compradas para asegurar que la escasez de productos y materiales no puedan retrasar el proceso de producción. 8) Los productos terminados se envían; se realimenta el proceso de producción con información del control de inventarios sobre el desempeño en las ventas. 9) El servicio al cliente asegura la continuidad de los productos entregados.

1.2. Los ordenadores en la Fabricación

Los ordenadores han sido utilizados en la fabricación desde 1960, como en cualquier otro negocio, para realizar funciones como contabilidad, compras, facturación y control de inventarios. Gradualmente, con el rápido aumento de las velocidades de operación, memorias de mayor capacidad y bajos costes, el uso de los ordenadores se extendió para dar soporte a otras funciones, como por ejemplo: las fases del diseño del producto (CAD - Diseño Asistido por Computador), programación del movimiento de las máquinas-herramientas (NC - Control Numérico y CNC - Control Numérico por Computador), gestión de

inventario (MRP - Planificación de Recursos de Fabricación), planificación de producción, asignación de tareas y recursos, planificación de la capacidad necesaria, funciones necesarias para la ejecución de los planes de producción, aspectos de gestión de fábrica, entre otros. Más recientemente, la fabricación ha sido considerada como un sistema integrado que comprende máquinas y programas software (Figura 1.2) en el cual las interacciones complejas se llevan a cabo con la ayuda del ordenador. El resultado es la fabricación integrada por computador (CIM - Computer Integrated Manufacturing), término amplio que describe la integración computarizada de todos los aspectos de diseño, planificación, fabricación, distribución y administración.

El enfoque CIM pretende ampliar los diversos niveles de automatización de las operaciones de fabricación, incluyendo funciones de procesamiento de información y usando una extensa red de computadores. Los sistemas CIM consisten de subsistemas que se integran en un todo, en la Figura 1.2 se puede ver un esquema de un CIM. Estos subsistemas suelen ser: Planificación y Respaldo Comercial, Diseño del Producto, Planificación del proceso de fabricación, Control del proceso, Sistemas de monitorización del taller, y Automatización del proceso. Los subsistemas se diseñan, desarrollan y aplican de tal manera que la salida de uno sea la entrada de otro.

Un sistema CIM eficiente requiere que todas las acciones se lleven a cabo haciendo referencia a una base de datos común. Esta base de datos almacena información del tipo: datos del producto (forma, dimensiones y especificaciones de pieza), atributos de administración de datos (propietario, nivel de revisión y número de la pieza), datos de producción (procesos de manufactura), datos de operación (scheduling, tamaños de lotes y requerimientos de ensamblado) y datos de recursos (capital, máquinas, equipos, herramientas y personal). Esta base de datos se mantiene actualizada con la información que proviene tanto del personal de la fábrica, como de sensores instalados en las maquinarias y en los equipos que se emplean en la producción.

A finales de la década de 1960 comenzaron a utilizarse los Sistemas Flexibles de Fabricación (FMS - Flexible Manufacturing Systems). Un FMS consiste

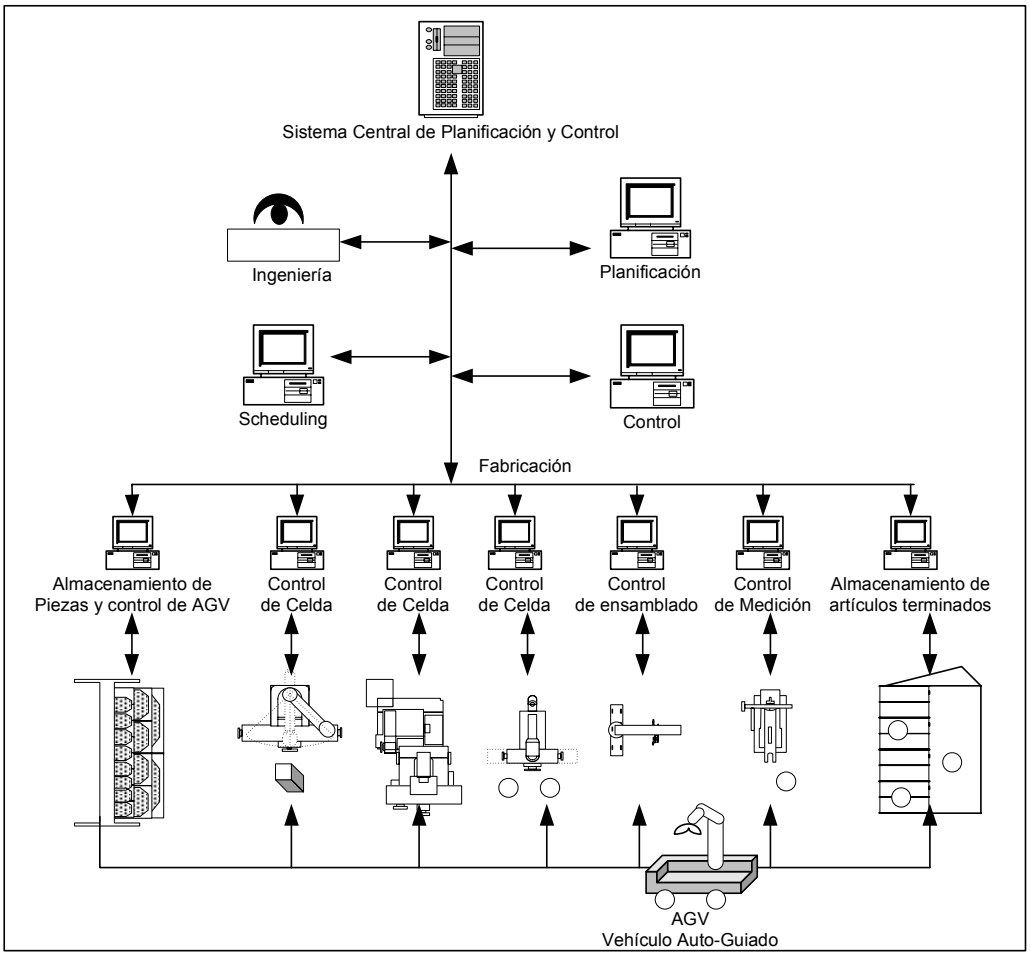


Figura 1.2: Esquema de un Sistema CIM

de varias celdas de fabricación [4], cada una con un robot industrial que da servicio a varias máquinas CNC, y en un sistema automatizado de manejo de materiales (AGV), todos ellos interrelacionados mediante un ordenador central. Las estaciones de trabajo dentro de las celdas se arreglan de tal forma de alcanzar la máxima eficiencia en la producción, con un flujo ordenado de materiales, piezas y productos por el sistema. Se puede considerar que un FMS combina las ventajas de otros dos sistemas: las líneas de transferencia que

[4]Unidad pequeña con una o varias estaciones de trabajo. Una estación de trabajo suele contener una máquina o varias máquinas, cada una de ellas efectúa una operación diferente en la pieza o parte.

son de gran productividad pero de poca flexibilidad; y la fabricación de taller (job shop), que puede manejar una gran diversidad de productos en máquinas independientes, pero que es ineficiente.

Pronto se dieron cuenta que para lograr la flexibilidad y productividad que estos sistemas avanzados prometían, no era suficiente la tecnología existente; también eran necesarias la integración y coordinación efectiva de los sistemas. Guiados por esta promesa de sistemas más flexibles, la tendencia actual es hacia sistemas "inteligentes" que pueden adaptarse rápidamente a cambios, al mismo tiempo que permiten, gracias a un diseño modular y distribuido, construir sistemas extensible o escalables.

1.3. La nueva fabricación

La competitividad global y las necesidades cambiantes de los usuarios están forzando un cambio primordial en los estilos de fabricación y configuración de organizaciones industriales. La planificación de fabricación centralizada y secuencial, la asignación de recursos, y los mecanismos de control tradicionales están siendo considerados poco flexibles para responder a cambios en los estilos de fabricación y a los requisitos de fabricación altamente variables. Los enfoques tradicionales limitan las capacidades de expansión y re-configuración de los sistemas de fabricación. La organización centralizada de tipo jerárquico tradicional puede provocar caídas del sistema por fallos en un sólo punto, o por fragilidad de los planes.

Las empresas de fabricación del siglo 21 se encuentran en un entorno en el cual los mercados son muy cambiantes, nuevas tecnologías emergen constantemente, y los competidores se multiplican globalmente. Las estrategias de fabricación deben cambiar para soportar la competitividad global, la innovación e introducción de nuevos productos, y la respuesta rápida al mercado. Los sistemas de fabricación futuros deberán ser más orientados al tiempo, aunque todavía deberán centrarse en el coste y la calidad de fabricación. Estas consecuencias implican:

- productos más complejos (debido a mayores características y variantes),
- productos que cambian con mayor rapidez (debido a los ciclos de vida

reducidos del producto),

- introducción más rápida de los productos (debido a la reducción del tiempo para salir al mercado), e
- inversiones reducidas (por producto).

El futuro del sector de la fabricación estará determinado por la manera en que éste satisfaga los desafíos de la "nueva fabricación". Tales sistemas de fabricación necesitarán satisfacer los siguiente requisitos fundamentales [HMS, 1994] [Shen y Norrie, 1999]:

- *Integración de la Empresa*: Para lograr la competitividad global y la respuesta rápida al mercado, la empresa de fabricación individual o colectiva deberá integrarse con su sistema de administración (compra, órdenes de trabajo, diseño, fabricación, planificación y asignación de recursos, control, transporte, recursos, personal, materiales, calidad, etc.) y sus socios a través de una red.
- *Organización Distribuida*: Para una integración efectiva de la empresa a través de organizaciones distribuidas, serán necesarios sistemas distribuidos basados en el conocimiento para enlazar la administración de demandas directamente a los recursos, la planificación de capacidades y asignación de recursos.
- *Entornos Heterogéneos*: Tales sistemas de fabricación necesitarán acomodar software y hardware heterogéneos tanto en sus entornos de fabricación como en sus entornos de información.
- *Inter-operabilidad*: Entornos heterogéneos de información pueden utilizar lenguajes de programación diferentes, representar los datos con diferentes lenguajes y modelos de representación, y operar en diferentes plataformas de procesamiento. Los sub-sistemas y componentes de tales entornos heterogéneos deberían inter-operar de una manera eficiente.
- *Cooperación*: Las empresas de fabricación deberán cooperar con sus proveedores, socios, y clientes para el suministro de materiales, fabricación

de partes, comercialización del producto final, etc. Tal cooperación debería ser eficiente y de respuesta rápida.

- *Integración de humanos con el software y hardware*: Las personas y los computadores necesitan estar integrados para trabajar de manera colectiva a varios niveles de desarrollo en el proceso de fabricación y en todo el ciclo de vida del producto, con acceso rápido al conocimiento y la información. Fuentes de información heterogéneas deben ser integradas para soportar estas necesidades y para mejorar las capacidades de decisión del sistema. Se requieren entornos de comunicación bi-direccional para permitir comunicación efectiva y rápida entre los humanos y los computadores para facilitar así su interacción.

- *Agilidad*: Se debe prestar considerable atención en reducir el tiempo del ciclo del producto para ser capaces de responder a los deseos de los clientes de manera más rápida. La fabricación ágil es la habilidad de adaptarse rápidamente a entornos de fabricación de cambio continuo y no anticipado, y, además, es el componente clave en las estrategias de fabricación para la competitividad global. Para lograr la agilidad, las utilidades de fabricación deben ser capaces de re-configuración rápida e interacción con sistemas y socios heterogéneos. Idealmente, los socios se contratan de manera dinámica sólo para el tiempo requerido para completar una tarea específica.

- *Escalable*: Un sistema de fabricación es escalable cuando recursos adicionales pueden ser incorporados dentro de la organización cuando éstos sean necesarios. Esta capacidad debería estar disponible en cualquier nodo de trabajo en el sistema y en cualquier nivel dentro de los nodos. La expansión de recursos debería ser posible sin alterar los enlaces previamente establecidos en la organización.

- *Tolerancia a Fallos*: El sistema debería ser tolerante a fallos tanto a nivel de sistema como a nivel de sub-sistema para detectar y recuperarse de fallos a cualquier nivel, y, así, minimizar el impacto que estos fallos pudieran tener en el entorno de trabajo.

En respuesta a estas tendencias, nuevos paradigmas de fabricación han sido propuestos para lograr alcanzar los desafíos de la fabricación de la "nueva generación".

A principios de 1985, Hatvany [Hatvany, 1985] precisó que la rigidez de las estructuras jerárquicos tradicionales limitan el desempeño dinámico de los sistemas. Sugirió entonces un sistema heterárquico, que se describe como la fragmentación del sistema en unidades pequeñas y completamente autónomas. La arquitectura de control heterárquico se basa en una total autonomía local (control distribuido) resultando en un entorno de control en el cual los componentes autónomos cooperan para alcanzar objetivos globales gracias a la toma de decisiones locales. Estos componentes autónomos son agentes, y la cooperación se estructura a través de protocolos de negociación. Siguiendo estas ideas se ha propuesto la fabricación basada en agentes [Sikora y Shaw, 1997] para mejorar la dinámica de estas organizaciones.

Desde hace unos años, la tecnología agente ha sido considerada como un enfoque importante para el desarrollo de sistemas de fabricación distribuidos [Jennings *et al.*, 1995],[Jennings y Wooldridge, 1998]. En los últimos 10 años, los investigadores han estado aplicando tecnología de agentes a la integración de empresas de fabricación y administración de cadenas de suministro, planificación de fabricación, asignación de recursos y ejecución de control, manipulación de materiales, y desarrollo de nuevos tipos de sistemas de fabricación tales como sistemas holónicos de fabricación.

1.4. Sistemas Holónicos de Fabricación

Los sistemas jerárquicos típicamente tienen una estructura rígida que les impide reaccionar de una manera ágil ante variaciones. Por otra parte, los sistemas heterárquicos tienen un buen desempeño ante cambios, y pueden autoadaptarse continuamente a su entorno. No obstante, el control hetrárquico no provee un sistema predecible y de alto rendimiento, especialmente en entornos heterogéneos complejos, donde los recursos son escasos y las decisiones actuales tienen serias repercusiones en el rendimiento futuro. Por esta razón, el control heterárquico es apenas utilizado en las industrias. El desafío actual radica en el hecho de que los sistemas industriales de la "vida real" necesitan tanto alto

rendimiento como reactividad.

La respuesta a este desafío se busca en las teorías sobre sistemas adaptables complejos. Koestler [Koestler, 1971] hizo la observación que los sistemas complejos sólo pueden surgir si se componen de subsistemas estables, cada uno de ellos capaz de sobrevivir ante disturbios, pero además, capaz de cooperar para formar un sistema estable más complejo. Estos conceptos han dado lugar al término "fabricación holónica". La fabricación holónica (*Holonic Manufacturing* - HM) es una organización altamente distribuida, donde la inteligencia se distribuye sobre las entidades individuales. Se la puede comparar con los sistemas distribuidos de la primera generación, sin embargo, el elemento nuevo en la fabricación holónica es el hecho que las entidades individuales trabajan juntas en jerarquías temporales para obtener un objetivo global.

La Fabricación Holónica es un paradigma desarrollado en el marco del programa *Intelligent Manufacturing Systems* (IMS - Sistemas de Fabricación Inteligente). La Fabricación Holónica se basa en conceptos de los ´´sistemas holónicos", desarrollados por Arthur Koestler. El trabajo realizado en el programa IMS ha trasladado estos conceptos al mundo de la fabricación, considerando al sistema de fabricación como un compuesto de módulos autónomos con control distribuido.

1.4.1. Las raíces de "lo holónico"

Hace más de 30 años, Arthur Koestler propuso la palabra "*holón*" [Koestler, 1971]. Esta palabra es una combinación de la palabra griega *holos* = todo, con el sufijo *on* el cual, al igual que en prot*ón* o neutr*ón*, sugiere una partícula o parte. Dos observaciones impulsaron a Koestler a proponer el concepto holón:

Primero, es más fácil construir sistemas complejos cuando estos están compuestos de formas o subsistemas intermedios estables, que no al contrario. Sistemas complejos, como los organismos biológicos, o las sociedades, están estructurados siempre como una jerarquía de subsistemas estables de múltiples niveles, que se bifurcan en subsitemas de orden inferior, y así sucesivamente. En sus aspectos estructurales no es una agregación de partes elementales, y en sus aspectos funcionales no es una cadena de unidades de comportamiento elementales. Esta característica de los sistemas complejos se ilustra mejor con

la "parábola de los dos relojeros" del ganador del premio Nobel Herbert Simon [Simon, 1990]:

> *Había una vez dos relojeros, llamados Hora y Tempus, que hacían relojes muy finos. Los teléfonos en sus tiendas sonaban frecuentemente; nuevos clientes estaban llamándolos constantemente. Sin embargo, Hora prosperó mientras que Tempus se hizo pobre y más pobre. Al final, Tempus perdió su tienda. ¿Cuál fue la causa de esto?. Los relojes que Tempus hacía estaban diseñados tal que, cuando él tenía que dejar caer una parte ensamblada (para, por ejemplo, coger el teléfono), la pieza inmediatamente caía en pedazos y debía de ser re-ensamblada a partir de los elementos básicos. Hora había diseñado sus relojes de tal manera que él pudiera unir sub-ensamblados estables de aproximadamente diez componentes cada uno. Diez de estos sub-ensamblados podrían ser unidos para hacer un sub-ensamblado mayor. Finalmente, diez de estos sub-ensamblados mayores constituían el reloj completo. Cada sub-ensamblado podía caer sin deshacerse en pedazos.*

Simon concluye que la jerarquía está omnipresente en nuestro mundo. Notar que, en este contexto, la palabra jerarquía o sistema jerárquico debe ser interpretada como un sistema de subsistemas flojamente interrelacionados, cada uno de los últimos siendo a su vez jerárquicos hasta llegar a algún nivel más bajo de subsistemas elementales.

Segundo, Koestler notó, analizando jerarquías y formas intermedias estables en organismos vivos y organizaciones sociales, que - aunque es fácil identificar "sub-todos" y partes - los "todos" y las 'partes' no existen en un sentido absoluto. Esto hizo que Koestler propusiera la palabra holón para enfatizar la naturaleza híbrida de sub-todos/partes en los sistemas de la vida real; los holones son todos auto-contenidos para sus partes subordinadas, y, simultáneamente, partes dependientes cuando son vistos desde niveles superiores.

Koestler utiliza la palabra "holarquía" para denotar organizaciones jerárquicas de holones. Las holarquías son abiertas tanto por arriba como por debajo.

El holón del nivel más alto, el todo global, puede a su vez ser parte de otro todo mayor; el holón del nivel más bajo, la parte más elemental, puede a su vez contener componentes más pequeños dentro.

El término holón puede ser aplicado a cualquier sub-todo estable biológico o social, independientemente del nivel jerárquico. Las células y los órganos son ejemplos de holones biológicos; individuos, familias, tribus, y naciones son ejemplos de holones sociales. Cada uno de los ejemplos anteriores pueden actuar como un todo, mientras que por otra parte, todos pueden ser vistos como una parte componente de un todo mayor.

1.4.2. El proyecto HMS

El HMS - (*Holonic Manufacturing Systems* - Sistemas Holónicos de Fabricación) es una iniciativa de investigación para sistemas de fabricación avanzados inspirada en los conceptos propuestos por Koestler. Gracias a las publicaciones del Caso de Estudio HMS dentro del consorcio IMS, el término HMS es conocido en los laboratorios de todo el mundo, dedicados a la investigación sobre sistemas de fabricación.

El caso de estudio HMS fue un proyecto de viabilidad de un año, para la evaluación de las posibilidades de cooperación global, trasladando los conceptos de Koestler al dominio de fabricación, y definiendo y desarrollando varias demostraciones.

1.4.2.1. El Caso de Estudio HMS

El paradigma holónico de fabricación se ha desarrollando dentro del marco del programa internacional IMS. El objetivo del IMS es la creación de una ciencia de la fabricación que pueda identificar los requisitos de los sistemas de fabricación del siguiente siglo. En un estudio de viabilidad, llevado a cabo en 1994, seis casos de estudio fueron considerados, uno de ellos fue el “Caso de Estudio No. 5-*Holonic Manufacturing Systems*: componentes de sistema de módulos autónomos y control distribuido”, o HMS. El proyecto HMS tenía como objetivo lograr una mejor compresión de los requisitos de los sistemas de fabricación de la siguiente generación y medios para construir sistemas que satisfagan estos requisitos. Desde el principio, el concepto de sistemas holónicos

ha sido considerado como el paradigma que sería capaz de lograr este objetivo.

En la Fase 1 del proyecto HMS durante los años 1995-2000, la investigación fue dirigida hacia tres áreas principales: (1) tecnologías genéricas, (2) benchmarks y test beds en dominios de aplicación específicos, y (3) actividades de gestión de proyectos, transferencia, diseminación y explotación de tecnología. El proyecto estaba compuesto de siete paquetes de trabajo (WP- *Work Package*), que trataban: Gestión de Proyectos (WP1); Requisitos de Usuario del siglo 21 (WP2); Factores críticos de los sistemas de fabricación (WP3); Definiciones de Testbeds (WP4); Testbed benchmarks (WP5); Estrategias del HMS: arquitecturas, metodologías (WP6); y Difusión de los resultados (WP7). Actualmente se está llevando a cabo la Fase 2 [Gruver *et al.*, 2003] del proyecto. Los paquetes de trabajo para esta fase están organizados de la siguiente manera: *Dispositivos de Control Holónico* - nivel de equipos técnicos de fabricación; *Estaciones de Producción Holónicas y Equipamiento Físico* - nivel de celda de trabajo compuesta de un número de dispositivos; *Sistemas Holónicos de Planificación y Ejecución* - scheduling y control de los sistemas holónicos de fabricación a nivel de fábrica y cadenas de suministro.

El objetivo del consorcio HMS es lograr en la fabricación los beneficios que las organizaciones holónicas proporcionan a los organismos vivos o a las sociedades, esto es, estabilidad ante alteraciones, adaptación y flexibilidad ante cambios, y uso eficiente de los recursos disponibles. El nuevo paradigma combina los conceptos naturales de los sistemas jerárquicos y la integración de los elementos autónomos dentro de un sistema distribuido. Cuando por el contrario las arquitecturas de los sistemas de fabricación convencionales están modeladas a lo largo de líneas jerárquicas de relaciones de orden-obediencia, la arquitectura HMS está modelada utilizando relaciones todo-parte.

Con este propósito, el consorcio tradujo los conceptos de Koestler desarrollados para las organizaciones sociales y los organismos vivos en un conjunto de conceptos apropiados para la industria de fabricación. El consocio compuso un glosario con definiciones para un amplio rango de términos que ayudan a entender y a guiar la traducción de los conceptos holónicos a escenarios de fabricación [Seidel, 1994]. Las definiciones principales son las siguientes:

- **Holón**: Una unidad de construcción autónoma y cooperativa de un sistema de fabricación para transformar, transportar, almacenar y/o validar objetos de información y físicos. El holón consiste de una parte de procesamiento de información y muchas veces de una parte de procesamiento físico. Un holón puede ser parte de otro holón.

  Ejemplos de holones de fabricación son herramientas, máquinas, equipos de transporte, piezas de trabajo, trabajadores humanos, información del producto (diseño, plan de desarrollo), máquinas de operaciones, órdenes del cliente, etc. Nuevamente, cada uno de estos puede ser tratado como un todo en sí mismo, o como una parte de una organización mayor.

- **Autonomía**: La capacidad de una entidad de crear y controlar la ejecución de sus propios planes y/o estrategias.

- **Cooperación**: Un proceso por el cual un conjunto de entidades desarrollan planes aceptados mutuamente y ejecutan dichos planes.

- **Holarquía**: Un sistema de holones que pueden cooperar para lograr un objetivo. La holarquía define las reglas básicas para la cooperación de los holones y por tanto limita su autonomía.

Un sistema holónico de fabricación integra el rango completo de actividades de producción, desde la recepción de una orden pasando por el diseño, la fabricación, y el mercadeo para lograr la empresa de fabricación ágil. Está organizado como una holarquía, que define las reglas básicas para la cooperación de los holones. Un sistema holónico de fabricación no está, en cambio, organizado de manera fija, sino se organiza a sí mismo de manera dinámica para lograr sus objetivos, y auto-adaptarse ante cambios en su entornos o en sí mismo.

1.4.3. Escenario de fabricación holónica

Para ilustrar las ideas del enfoque holónico de sistemas de fabricación presentamos a continuación un ejemplo sencillo. Supongamos que hemos identificado tres tipos de holones en un sistema de producción: holón de recurso, holón de producto y holón orden de trabajo (en el capítulo 2 incluimos un resumen de la literatura especializada sobre arquitecturas holónicas). La Figura

1.3 muestra las distintas holarquías que se forman a medida que se produce un determinado producto en este escenario simplificado. Inicialmente, Figura 1.3a, la fábrica consiste únicamente de un conjunto de holones de recursos sin organización (HRs). No existe ninguna relación a-priori entre los distintos recursos ni tampoco ningún código de control pre-definido en anticipación a órdenes de trabajo futuras. De esta manera, cuando llega una orden de trabajo, se crea un holón orden de trabajo (HO), Figura 1.3b, el cual (equipado con los requisitos de la orden del cliente y las especificaciones del producto) empieza a negociar con los holones de recurso para obtener ciertas operaciones de fabricación, Figura 1.3c. Durante el proceso de negociación, el holón orden de trabajo demanda propiedades específicas requeridas por el proceso, tales como alta calidad y entrega rápida, mientras que los holones de recurso tratan de optimizar su utilización - por ejemplo, para asegurar un alto desempeño global de la fábrica. Al final del proceso de negociación, los holones de recurso forman una "línea de producción" y el holón orden de trabajo inicia la creación de los holones producto (HPs). En la Figura 1.3d podemos apreciar que se ha formado una holarquía entre los holones HO1, HR1 y la holarquía definida por HR3 y HR4.

Los holones de producto (HPs) entran en la línea de producción (provenientes, por ejemplo, del almacén de materia prima) e inmediatamente pujan por determinados recursos, según la orden de trabajo, con el objetivo de ser procesados. Cada holón de producto ejecuta este proceso de manera individual centrándose en la siguiente operación. Una vez que estas operaciones hayan sido realizadas, el producto reinicia la negociación con los holones de recurso que ofrecen las siguientes operaciones en su plan de producción, Figura 1.3e. La organización global de la holarquía de recursos (inicial o negociada posteriormente entre los holones de recurso y orden de trabajo) asegura que la carga de producto se distribuya de manera eficiente sobre los recursos disponibles para poder alcanzar los objetivos globales de la holarquía.

En caso de fallos, el holón de recurso afectado se elimina del conjunto de recursos disponibles y se dirige a una zona de reparaciones. Los holones de recurso restantes se re-organizan para superar la pérdida de capacidad. Desde el punto de vista del holón de producto, el proceso continúa normalmente,

sólo con menos holones de recurso con los que negociar. Luego de la reparación, el holón de recurso se une nuevamente al conjunto de holones de recurso disponibles.

Al final del procesamiento de la orden, se elimina al holón orden de trabajo y la holarquía de recursos se disuelve en holones de recurso, que posteriormente tratan de participar en nuevas holarquías, Figura 1.3a.

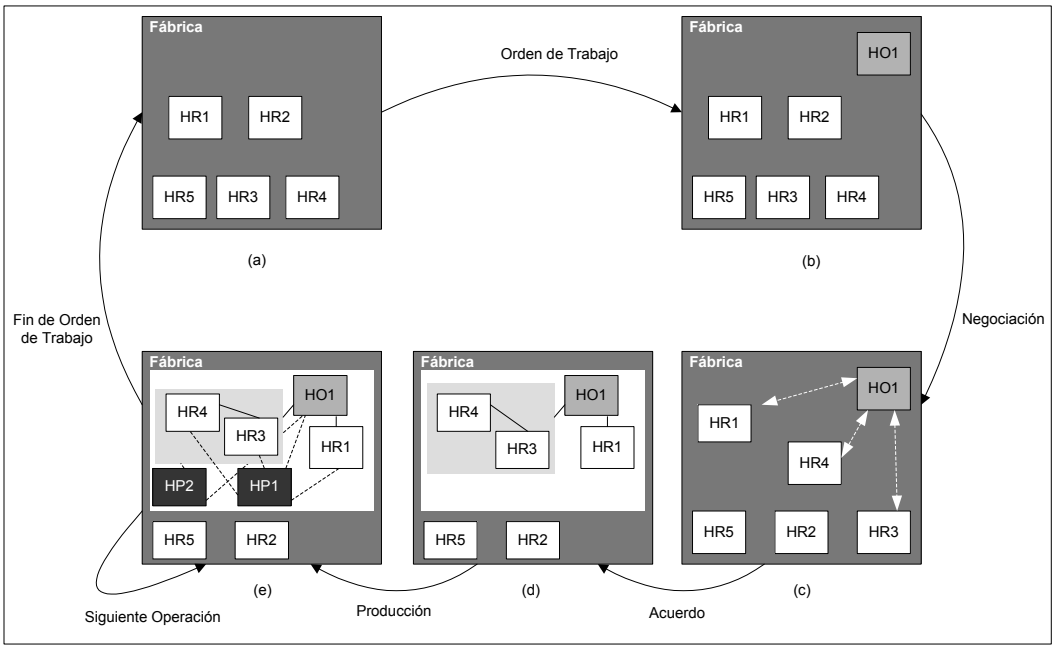


Figura 1.3: Escenario ideal de un Sistema Holónico de Fabricación

1.4.4. Comparación con los enfoques existentes

1.4.4.1. Control Jerárquico

El enfoque tradicional para el diseño de los sistemas CIM es el jerárquico [Jones y McLean, 1996]. Algunas arquitecturas jerárquicas han sido desarrolladas por NIST (AMRF, y más recientemente FCS y MSI [Joshi y Smith, 1994]). El diseño está basado en un enfoque top-down y define estrictamente los módulos del sistema y su funcionalidad. Más aún, la comunicación entre los módulos también se define estrictamente, y se limita, ya que los módulos sólo

pueden comunicarse con sus módulos padres y módulos hijos. En una arquitectura jerárquica, los módulos no pueden tomar la iniciativa. De esta manera el sistema es vulnerable ante perturbaciones y su autonomía y reactividad ante disturbios son débiles. La arquitectura resultante es muy rígida, y por tanto cara de desarrollar y difícil de mantener.

1.4.4.2. Control Heterárquico

Antes de los sistemas de fabricación holónicos, el control heterárquico fue otro enfoque para resolver los problemas de los sistemas jerárquicos [Dilts *et al.*, 1991]. El enfoque heterárquico prohíbe toda tipo de jerarquía con el objeto de dar todo el poder a los módulos básicos, generalmente llamados "agentes". Este enfoque consiste, por ejemplo, sólo de estaciones y órdenes de trabajo. Cada orden negocia con las estaciones de trabajo para obtener el procesamiento de los recursos, para ello utiliza todas las posibles alternativas de procesamiento disponibles para poder afrontar situaciones imprevistas. Este enfoque permite reaccionar de manera adecuada ante modificaciones del entorno (por ejemplo, un nuevo producto que entra al mercado global, tecnologías nuevas o evolucionadas, demandas imprevistas de productos) así como ante cambios en el sistema de fabricación en sí (por ejemplo, defectos, producción variable de estaciones de trabajo, ...). Mientras que este sistema es de hecho muy ágil, y simple de diseñar, de entender y de mantener, con este tipo de sistemas es muy difícil de operar siguiendo un plan pre-definido. La previsibilidad de la producción es muy baja. El control heterárquico típicamente se desempeña bien en entornos simples, por ejemplo en un taller (shop floor) compuesto por estaciones de trabajo idénticas y con capacidad de respaldos. Sin embargo, la mayoría de los sistemas de fabricación son sistemas complejos y heterogéneos, en los cuales las decisiones que se puedan tomar en un momento determinado generalmente influirán en la operación futura del sistema. Para sistemas de fabricación reales, el control heterárquico sólo se puede utilizar si existen recursos abundantes y si el entorno es homogéneo y no demasiado complejo [Van Brussel *et al.*, 1999]. No existe ninguna optimización global y el sistema puede alcanzar, en circunstancias específicas, algún estado "inestable", donde la respuesta a un cambio o perturbación induce a un problema mayor en el

sistema [Valckenaers *et al.*, 1994].

1.4.4.3. Sistemas de Fabricación Holónicos versus control jerárquico y heterárquico

La fabricación Holónica combina las ventajas de los sistemas jerárquicos y de los heterárquicos al mismo tiempo que evita sus desventajas. Para evitar las arquitecturas rígidas de los sistemas jerárquicos, los sistemas holónicos otorgan autonomía (libertad de decisión) a los módulos individuales (holones). Esto otorga al sistema respuesta rápida ante disturbios, y la habilidad para reconfigurarse a sí mismo ante nuevos requerimientos. Además permite la integración de módulos de sistemas en un rango mayor de sistemas de fabricación.

Comparado con los sistemas de control holónicos, los sistemas de control heterárquico pueden ser imprevisibles y además incontrolables. Esto se debe a la inexistencia de jerarquías en los sistemas heterárquicos. Es por ello que los sistemas de fabricación holónicos poseen jerarquías, pero estas jerarquías son flexibles, o "flojas". Esta jerarquía difiere del control jerárquico tradicional en lo siguiente:

- los holones pueden pertenecer a múltiples jerarquías,
- los holones pueden formar jerarquías temporales,
- los holones no dependen de operaciones propias de (ligadas a) cada holón en una jerarquía para lograr sus objetivos.

Para distinguir entre las jerarquías estrictas de los sistemas de control jerárquicos, y las jerarquías flojas de los sistemas de control holónicos, se ha introducido el término holarquía para identificar a las jerarquías flexibles y flojas. Sin embargo, al otorgar reglas y consejos, la holarquía limita la autonomía de los holones individuales, para asegurar desempeños controlables y predecibles a diferencia de los sistemas heterárquicos.

1.4.4.4. Fabricación Holónica versus paradigmas similares

La fabricación holónica es sólo uno de los paradigmas para las "fábricas del futuro", entre otros muchos como pueden ser los sistemas de fabricación

biónicos [Okino, 1993], genéticos [Ueda, 1993], fractales [Warnecke, 1992], aleatorios [Iwata y Onosato, 1994] o virtuales [Kimura, 1993]. Estos conceptos y paradigmas no son necesariamente contradictorios. Muchos de ellos utilizan conceptos de sistemas multi-agente para distribuir conocimiento y toma de decisión. Tienen muchas características comunes y son incluso complementarios. Sin embargo pueden ser distinguidos por la fuente de su origen - matemáticas para la fabricación basada en fractales, naturaleza para los sistemas de fabricación genéticos y biónicos. En la fabricación biónica [Okino, 1993], que ha sido inspirada por metáforas biológicas, el foco principal radica en la naturaleza auto-organizativa de los elementos de un sistema de fabricación, tal que los sistemas biónicos van más allá de los sistemas modulares y aparecen como sistemas sin conexión y vivos. La fabricación genética [Ueda, 1993] extiende las ideas anteriores e imita conceptos del ADN para modelar órdenes de trabajo en la producción. En la fabricación basada en fractales [Warnecke, 1992], los conceptos clave son auto-organización, auto-optimización y la dinámica de las personas en el sistema de fabricación. La fabricación aleatoria [Iwata y Onosato, 1994] es una arquitectura multi-agente basada en cuatro conceptos: la máquina toma decisiones de manera autónoma; el agrupamiento de máquinas es dinámico; las órdenes se comunican a través de una pizarra; y el control de shop floor se ejerce mediante recompensas y penalizaciones. Los sistemas virtuales de fabricación son modelos de computadores integrados que precisamente representan el sistema de fabricación y simulan su operación para predecir y controlarlo [Kimura, 1993].

Los sistemas holónicos se originaron a partir de teorías filosóficas sobre la creación y evolución de sistemas adaptativos complejos (sistemas sociales, teoría de la evolución). Dado que los filósofos no sólo observan los fenómenos sino que también tratan de explicarlos, los sistemas holónicos pueden tener fundamentos más sólidos que muchos de sus competidores. El foco central de los sistemas holónicos es sobre:

- los holones autónomos proveen operaciones robustas y agilidad ante disturbios,

- jerarquías flojas para permitir optimización global,

- holarquías flexibles para permitir reconfiguración ante necesidades cambiantes, y evolución con nuevas tecnologías.

1.5. Motivación y Objetivos

Hoy en día el área de Sistemas Inteligentes de Fabricación y específicamente los Sistemas Holónicos de Fabricación, están atrayendo especial interés de desarrolladores e investigadores tanto del sector industrial como académico. Se han logrado grandes avances en desarrollos de algoritmos, arquitecturas y aplicaciones, sin embargo la mayoría de estos desarrollas se han llevado a cabo de manera más o menos *ad hoc* siguiendo, prácticamente, ninguna metodología de ingeniería del software. Los sistemas de fabricación son, por naturaleza, sistemas de gran escala y complejos, por lo tanto la gestión del ciclo de vida de las aplicaciones que los controlan son muy difíciles de mantener sin una metodología que incorpore los principios de ingeniería del software.

Los requisitos de la nueva fabricación imponen nuevas características a los sistemas de control de procesos industriales y de procesos de negocio de una empresa dedica a la fabricación. Estas nuevas características deben ser consideradas por las metodologías de desarrollo de HMS para poder satisfacer las necesidades de la nueva fabricación.

Aunque uno de los paquetes de trabajo del consorcio HMS en su Fase 1, estaba centrado en el estudio de Arquitectura e Ingeniería de los Sistemas Holónicos de Fabricación. Hemos encontrado muy pocas referencias de trabajos acerca de metodologías de desarrollo. En este trabajo pretendemos profundizar sobre este tema, estudiando metodologías de ingeniería del software para el modelado de HMS.

El objetivo principal de esta tesis consiste en la definición de una metodología para el desarrollo de Sistemas Holónicos de Fabricación. Nuestra hipótesis es que la tecnología Multi Agente es apropiada para el desarrollo de HMS debido a la similitud de ambos enfoques, y además porque la tecnología Multi Agente ha sido utilizada con éxito como herramienta de implementación en aplicaciones del área HMS.

A un nivel más detallado, los objetivos que han motivado la presente investigación son los siguientes:

- Estudiar los requisitos de modelado de los Sistemas Holónicos de Fabricación.
- Estudiar la relación entre los enfoques HMS y SMA.
- Estudiar la aplicabilidad de metodologías SMA existentes al modelado de HMS.
- Definir una entidad de modelado que pueda integrar ambos paradigmas en las etapas de análisis y diseño.
- Definir una notación de modelado para la propuesta metodológica.
- Establecer una secuencia de etapas completas y guías específicas para el desarrollo de sistemas del dominio HMS.
- Ilustrar la utilización de la propuesta en ejemplos reales del sector industrial.

1.6. Estructura del Trabajo

El resto de capítulos de esta tesis se estructuran de la siguiente manera:

- En el Capítulo 2 presentamos un extenso estudio del estado del arte del área HMS. En él resumimos las tres áreas de desarrollo: Arquitecturas para control Holónico, Algoritmos para control Holónico y Modelado de Sistemas Holónicos de Fabricación. Esta última área es central para esta tesis, por este motivo en ella definimos los requisitos de modelado para HMS y desarrollamos un estudio comparativo sobre la adecuación de metodologías de ingeniería del software (del dominio HMS, SMA y Modelado de Empresas) para el modelado de HMS.
- En el Capítulo 3 presentamos un estudio comparativo del enfoque holónico y el enfoque de agentes. El objetivo principal de este capítulo es averiguar si los holones y los agentes son diferentes o no, o si uno es un caso especial del otro.

- En el Capítulo 4 definimos la noción de Agente Abstracto como entidad para el modelado de entidades autónomas con estructuras recursivas.
- En el Capítulo 5 presentamos una Metodología Multi Agente para Sistemas Holónicos de Fabricación. Nuestra propuesta se basa en la noción de Agente Abstracto y en los requisitos de modelado para HMS. Proponemos un proceso de desarrollo mixto (ascendente y descendente), completo, y con guías de modelado claras y específicas para el dominio HMS.
- En el Capítulo 6 ilustramos nuestra propuesta metodológica mediante su aplicación a un caso de estudio real del sector cerámico.
- Finalmente en el Capítulo 7 presentamos las aportaciones principales de esta tesis, las líneas futuras de investigación y las publicaciones relacionadas con la tesis.

Capítulo 2

Estado del Arte del HMS

2.1. Desarrollo de Sistemas Holónicos de Control

Podemos distinguir dos ramas de estudio fundamentales en el área de investigación de control holónico: arquitecturas para el diseño e interconexión de elementos holónicos, y; algoritmos que definen la (inter)operación de holones.

2.1.1. Arquitecturas para control holónico

Un sistema de control de fabricación, para un proceso de producción, se compone de elementos software junto con las diferentes entidades físicas del entorno de fabricación: recursos, productos, pedidos de clientes, operaciones de coordinación, etc. El elemento software y la entidad física, acoplados mediante una red de comunicación apropiada (Figura 2.1), representan a los holones en este proceso. Cada uno de estos holones, una vez creados, se asume que son capaces de cierto grado de razonamiento local, con capacidad de decisión, y con la habilidad para comunicarse de manera interactiva con otros holones.

En la literatura especializada podemos encontrar varios trabajos relacionados con la arquitectura de holones. Christensen en 1994, fue el primero en proponer una arquitectura general de holones, Figura 2.1 [Christensen, 1994]. En ella podemos apreciar los dos componentes principales de un holón: *procesamiento físico* y *procesamiento de la información*. La parte de procesamiento físico es opcional, ejemplos de holones sin parte de procesamiento físico pueden

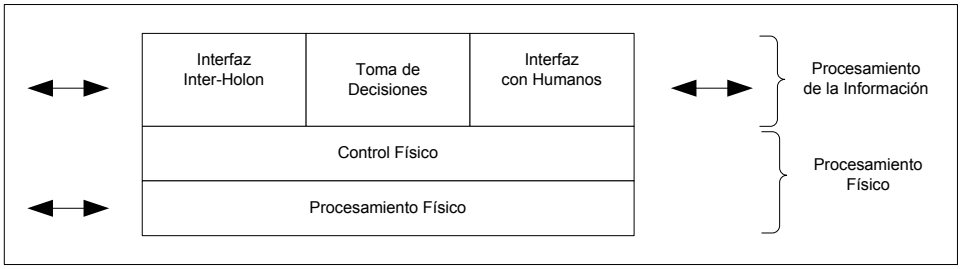


Figura 2.1: Arquitectura general de un holón

ser orden de trabajo, planificador, secuenciador, etc. El procesamiento físico se compone de: el *procesamiento físico* en sí, el hardware que realiza la operación de fabricación; y del *control físico*, controlador (NC, CNC, DNC, PLC) del hardware de operación. El procesamiento de la información se compone de tres módulos: el kernel del holón o *toma de decisiones* encargado de dotar al holón de capacidades de razonamiento y decisión; la *interfaz inter-holón*, para la interacción y comunicación con otros holones, y; la *interfaz con humanos*, para datos de entrada (comandos de operación) y salida (monitorización de estados).

En [Bussmann, 1998], se propone una arquitectura basada en agentes para la parte del procesamiento de la información de la arquitectura general de Christensen. Bussmann basa su propuesta en la visión holónica de entidades autónomas y cooperativas. Esta visión contiene tres aspectos. Primero, los holones son capaces de controlar de manera autónoma el comportamiento de su equipo asociado. Los holones pueden crear y ejecutar sus propios planes y seguir sus propias estrategias dentro de sus capacidades. Este comportamiento autónomo implica la existencia de algún tipo de componente para la toma de decisiones que guíe el control físico del holón. Segundo, dos o mas holones son capaces de cooperar donde y cuando sea necesario. Para hacer esto, estos holones deben ser capaces de identificar cooperación, comprometerse en coordinación o negociación, y finalmente ejecutar la cooperación acordada. Tercero, los holones son capaces de actuar dentro de múltiples organizaciones, llamadas holarquías, las cuales son creadas y modificadas dinámicamente. Crear una holarquía significa (re-)agrupar el proceso de manufactura o el proceso de control para mejorar la productividad. Esto implica la distribución del trabajo

y responsabilidad, y el establecimiento de patrones de interacción. La creación de las holarquías implica consecuentemente que los holones son capaces de identificar la posibilidad para re-organización, de negociar re-organizaciones, y de seguir los patrones acordados.

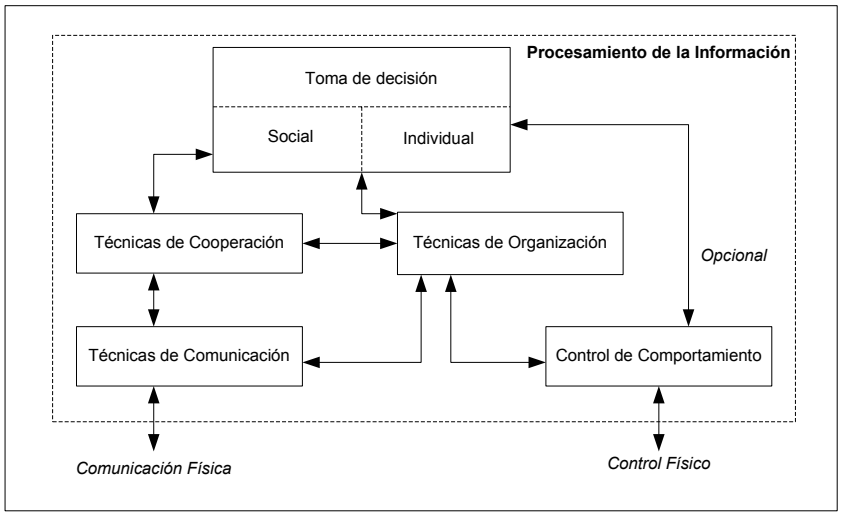


Figura 2.2: Arquitectura orientada a agentes para holones

La incorporación de estos componentes dentro de la arquitectura general de un holón (Figura 2.1) llevó a Bussmann a proponer la arquitectura orientada a agentes que mostramos en la Figura 2.2. Para determinar el comportamiento físico, el holón, dependiendo de la situación actual, elige los planes y estrategias que logren alcanzar de la mejor manera sus objetivos a largo plazo. Estos planes y estrategias se comunican desde el módulo *toma de decisión* al módulo *control de comportamiento* para ser traducidos en operaciones hardware. Por otra parte las posibles cooperaciones son iniciadas por el componente de toma de decisiones y llevadas a cabo por las *técnicas específicas de cooperación* que para ser finalmente ejecutadas necesitan de *técnicas de comunicación* (lenguajes y ontologías del dominio). El holón requiere de técnicas para reorganizar el control de los procesos de fabricación. Estas técnicas monitorizan las acciones de otros componentes y analizan el proceso de control. De esta manera pueden identificar la posibilidad de mejora e iniciar un proceso de negociación para la re-organización (*técnicas de organización*) que es llevada a cabo mediante técnicas de cooperación estándares.

PROSA (Product-Resource-Order-Staff Architecture) [Van Brussel *et al.*, 1998], es la arquitectura holónica de referencia para sistemas holónicos de fabricación que ha sido ampliamente adoptada. Consiste principalmente en una arquitectura inter-holónica, que identifica los tipos de holones, sus responsabilidades, y la estructura en la cual interactúan. La arquitectura básica consiste de tres tipos de holones básicos, Figura 2.3: (i) orden de trabajo, (ii) producto, y (iii) recursos. Estos se estructuran utilizando conceptos orientados a objetos como agregación y especialización. Cada uno de los holones básicos es responsable, respectivamente, de un aspecto del control en la fabricación: (i) logística interna, (ii) plan de fabricación, y (iii) manejo de recursos. Para asistir, con conocimiento experto, a los holones básicos se pueden agregar holones "staff". La estructura del sistema completo de fabricación es una holarquía dual descompuesta en una sub-holarquía de asignación de recursos (órdenes de trabajo, recursos y staff) y una sub-holarquía de control de proceso (producto y la parte de control de proceso de los recursos). Un holón de recurso contiene una parte física (un recurso de producción dentro de un sistema de fabricación) y una parte de procesamiento de la información que controla el recurso. Ofrece capacidad y funcionalidad de producción a los demás holones. Un holón de producto mantiene el conocimiento de proceso y de producto para asegurar la fabricación correcta del producto. Actúa como un servidor de información para los demás holones del HMS. Un holón orden de trabajo representa una tarea en un sistema de fabricación. Es responsable de realizar el trabajo asignado de manera correcta y a tiempo. Gestiona los productos físicos que se están produciendo, el modelo de estado de los productos, y toda la información logística de procesamiento relacionada con la tarea. En [Hadeli *et al.*, 2004], se ilustra una técnica de coordinación y de control de una holarquía PROSA utilizando conceptos inspirados en el comportamiento social de insectos (específicamente hormigas).

El grupo de la Universidad de Keele, liderado por Deen y Fletcher, ha desarrollado una arquitectura para sistemas holónicos basada en agentes y bloques funcionales [Fletcher *et al.*, 2000; Fletcher y Deen, 2001]. Un holón de fabricación por lo general se compone de conocimiento y componentes software, además de un componente (opcional) hardware. Funcionalmente, un holón

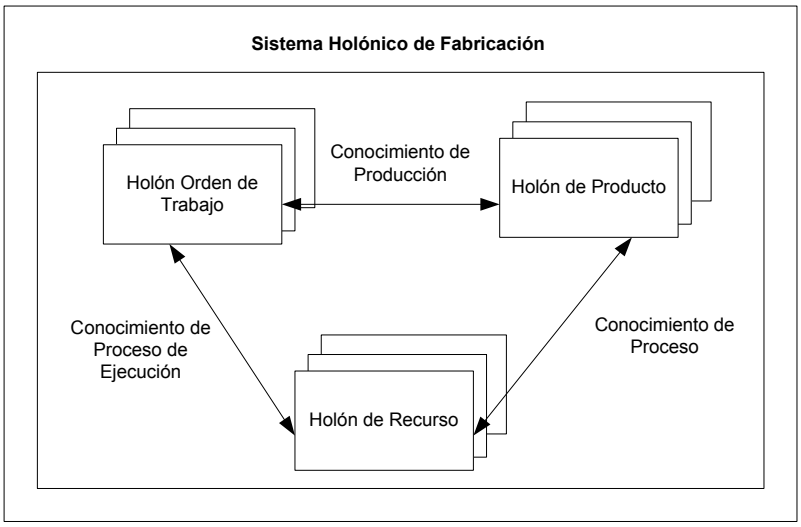


Figura 2.3: PROSA: arquitectura de referencia

puede ser considerado como una composición de un sistema de control inteligente (cabeza) y un sistema de procesamiento (base), Figura 2.4. La cabeza del holón se implementa se basa en una arquitectura de agente compuesta por los módulos definidos en la arquitectura general de holón de [Christensen, 1994]. Los elementos del sistema de control inteligente (ICS) son: PMC (control proceso/máquina) se encarga de ejecutar los planes de control para los procesos que están siendo controlados; PMI (interfaz proceso/máquina) provee la interfaz lógica y física para el sistema de procesamiento a través de una red de comunicación; HI (interfaz humana) provee la interfaz para humanos, tales como operadores, supervisores, personal de mantenimiento, etc; IHI (interfaz inter holónica) se encarga de la comunicación inter-holónica. El sistema de procesamiento consiste de todos los componentes de procesamiento necesarios para realizar las actividades de fabricación. De esta manera el ICS permite a los holones ofrecer las habilidades de fabricación, como subsistemas autónomos en coordinación con el ambiente y los demás holones. El sistema de procesamiento es responsable de las funcionalidades de fabricación de acuerdo a las reglas y estrategias operativas impuestas por el ICS.

En esta arquitectura integrada de agentes y bloques funcionales (IEC 61499

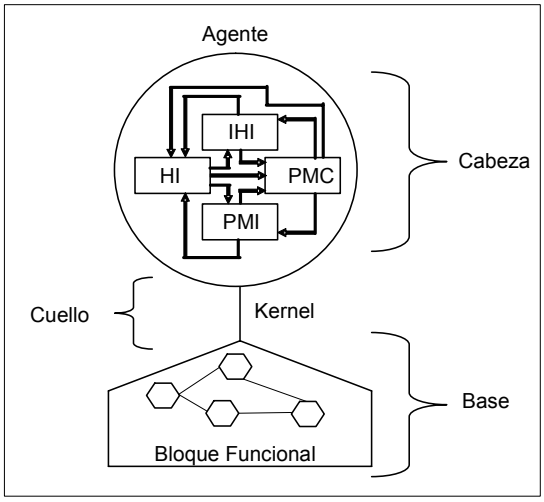


Figura 2.4: Arquitectura basada en agentes y bloques funcionales

[1] [IEC, 2000; 2001]), los agentes son utilizados para manejar las estrategias de planificación de alto nivel (ICS-Cabeza), mientras que los bloques funcionales manejan el control de bajo nivel proceso/máquina en tiempo real (Base). Un kernel holónico se ejecuta por encima de la arquitectura de los bloques funcionales para proveer la interfaz necesaria entre los agentes y el control IEC 61499. Las holarquías y la interacción entre los holones basados en agentes se organizan a través de una estructura llamada dominio de cooperación como se muestra en la figura 2.5. Un dominio de cooperación (holón compuesto) es un espacio lógico a través del cual: (i) los holones se comunican y operan, y (ii) se provee un contexto donde los holones pueden localizar, contactar e interactuar entre ellos. Un dominio de cooperación no puede existir por su propia cuenta debe poseer por lo menos un miembro holón, los dominios son creados dinámicamente por la operación de los componentes funcionales de los holones. Un sistema holónico está compuesto de por lo menos un dominio de cooperación.

[1]IEC, es la Comisión Internacional Electrotécnica, encargada de los estándares internacionales para todos los campos de la electro tecnología. El IEC 61499 corresponde al estándar para los procesos industriales distribuidos. Básicamente consiste en una colección de controladores de dispositivos interconectados y en comunicación por medio de una red que puede estar organizada de manera jerárquica. Se basa en bloques funcionales, los cuales proveen soluciones software a problemas pequeños, por ejemplo el control de un válvula, o el control de una unidad mayor dentro de una fábrica, tal como toda una línea de fabricación.

Un holón puede ser miembro simultáneamente de uno o más dominios de cooperación. Cada dominio está dirigido por un coordinador que es la interfaz del dominio con el exterior (otros dominios). Un holón puede unirse a un dominio de cooperación, consultar los atributos del mismo, intercambiar información entre los holones, y abandonar el dominio cuando haya finalizado su tarea. Para asignar actividades a los holones dentro de un dominio de cooperación se ejecuta un protocolo de cooperación.

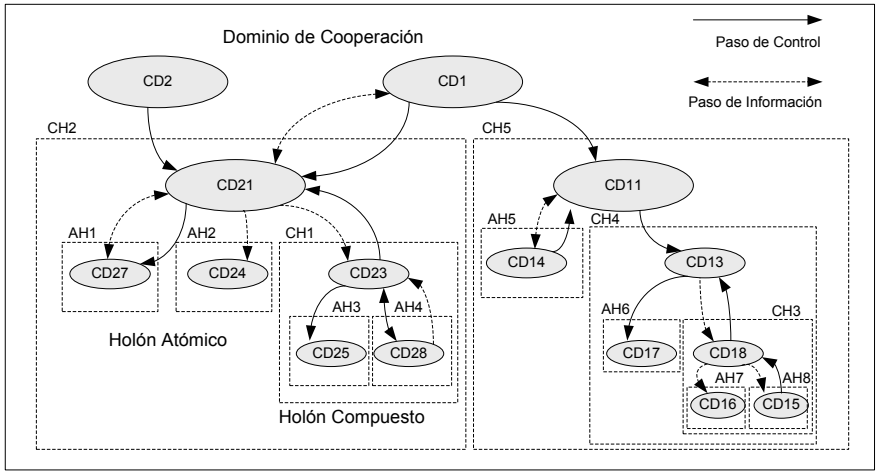


Figura 2.5: Dominio de Cooperación

Deen y Fletcher proponen además, un modelo computacional para redistribuir tareas (re-organización de la holarquía) basados en conceptos de mantenimiento del equilibrio de la temperatura [Deen y Fletcher, 2000]. Durante el procesamiento de una tarea retrasada o con problemas, los holones pueden sufrir sobrecarga, esto se interpreta en el modelo de temperatura como el "calentamiento" del holón. De esta manera cuando un holón determina que su temperatura sobrepasa un límite pre-establecido, éste informa a los demás holones de la situación. Si existe un holón "frío" que pueda realizar la tarea anunciada, entonces se inicia una negociación con el holón "caliente" para transferir la tarea. De esta manera cuando el sistema entero tiende a buscar un equilibrio de temperatura, se logra auto-organización de las holarquías que forman el sistema.

El grupo del Departamento de Mecánica e Ingeniería Industrial de la Universidad de Calgary desarrolla proyectos relacionados con modelos para control inteligente en sistemas de fabricación. Algunos de estos proyectos son: MetaMorph I, ABCDE (Agent Based Concurrent Design Enviromment), DIDE (Distributed Design Environment), FBIICDE (Feature-Based Integrated and Intelligent Concurrent Design System) y MetaMorph II. Este conjunto de proyectos se basan en una arquitectura distribuida para el control inteligente de fábricas, esta arquitectura ha ido evolucionando hasta llegar a la propuesta que presentan en [Brennan *et al.*, 2003; Brennan y Norrie, 2003].

La característica principal de los sistemas basados en MetaMorph [Maturana y Norrie, 1997] es su forma y estructura cambiantes, ya que se adaptan dinámicamente a tareas emergentes y ambientes cambiantes. La arquitectura de MetaMorph utiliza el concepto de dominio de cooperación de Deen y Fletcher pero lo llaman *cluster virtual dinámico*. A diferencia de PROSA en MetaMorph los tipos de holones primarios o básicos son los siguientes: holones de producto, holones de modelo de producto, holones de recurso. Un holón de producto es dual, por una parte consta de un componente físico, el producto en sí desde el inicio hasta el final; y por otra, almacena información acerca del estado del proceso de los componentes del producto durante la fabricación. Un holón de modelo de producto almacena información sobre configuración, diseño, plan de proceso, materiales, calidad, etc., sobre el ciclo de vida del producto. Los holones de recurso son utilizados para representar dispositivos y operaciones de fabricación.

La coordinación y la auto-organización se implementan mediante los clusters virtuales dinámicos (Figura 2.6). Utilizando este mecanismo los holones pueden participar dinámicamente en diferentes clusters (holarquías) y cooperar a través de dominios de cooperación. Los holones primarios cumplen la misma función que los holones coordinadores de Deen y Fletcher, se utilizan como administradores de los clusters para coordinar la interacción entre holones. El cluster existe hasta que la tarea de cooperación desaparezca, cuando la tarea haya sido completada. El ciclo de proceso para un cluster virtual se puede definir de la siguiente manera: (1) El holón primario reúne algunos o todos sus contratos (órdenes de fabricación, órdenes de cooperación con otros holones,

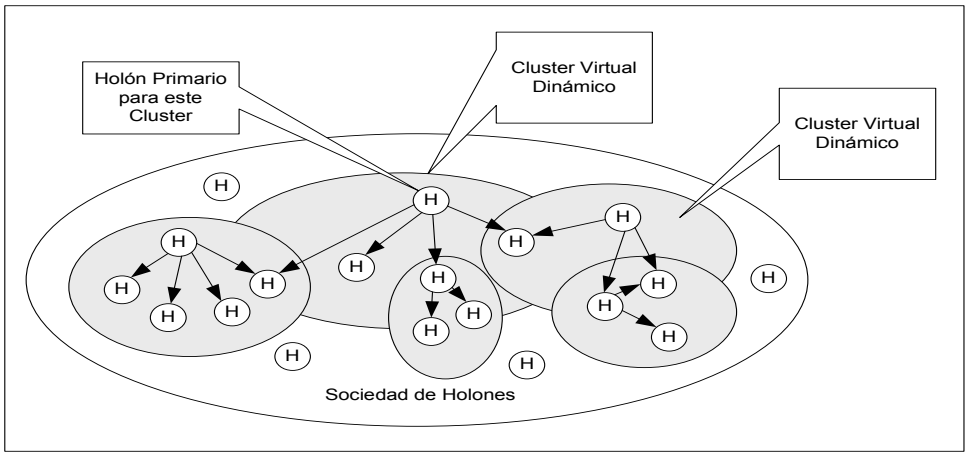


Figura 2.6: Holarquías y Sociedad de Holones

etc.) en una nueva tarea. Luego de un proceso de re-planificación y análisis, lista los requerimientos de cooperación como tareas de cooperación. (2) Se crea un holón mediador para encontrar holones sub-contratados potenciales y para enviar a estos una invitación de puja. (3) Los holones invitados, deciden de manera autónoma cómo participar y envían ofertas para todas las tareas en las que están interesados. (4) Se recogen y evalúan todas las ofertas tanto por el holón primario o por el mediador. Luego de determinar la asignación óptima de la tarea, se establecen contratos directamente entre el holón primario y el sub-contratado. Luego el holón primario y todos los sub-contratados forman un cluster virtual. (5) A medida que la tarea sea resuelta, el cluster asociado, el mediador y los respectivos enlaces desaparecen.

En [Brennan y Norrie, 2003] Brennan y Norrie proponen agentes holónicos utilizando para la capa deliverativa agentes y para la capa de control físico bloques funcionales. En [Brennan *et al.*, 2003] presentan una arquitectura holónica para el control de dispositivos (HCD). En la Figura 2.7 podemos ver los componentes de esta arquitectura. La capa deliverativa tiene dos propósitos: (i) funcionalidad específica del dominio de aplicación, y (ii) funcionalidad genérica. La funcionalidad genérica está definida por los módulos: planificador, modelo de proceso, control de ejecución y diagnóstico. La capa deliverativa se comunica con las otras capas mediante la tabla de datos del dispositivo vía el DTACI (Interfaz Común de Acceso a la Tabla de Datos). Las capas Deliverativas y de Control pueden leer y escribir datos desde o hacia el DTACI. La

capa de Funciones de control es la aplicación definida por el usuario que controla las actividades del hardware o procesos físico del HDC. En este nivel se utilizan bloques funcionales. Finalmente la capa Física representa al hardware (sensores y actuadores) controlado por el HCD. La capa de Simulación es la simulación del hardware.

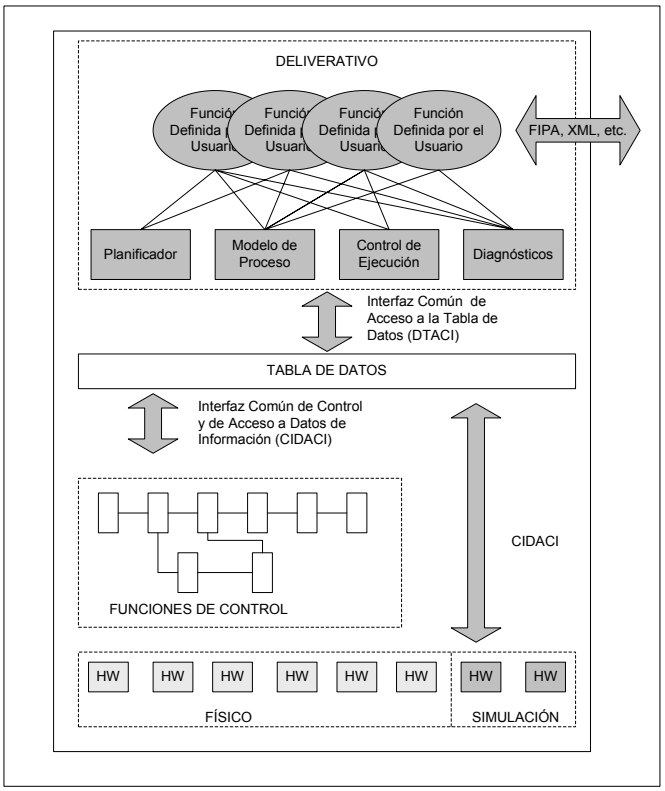


Figura 2.7: La arquitectura HCD

El grupo del Centro Germano de Investigación para la Inteligencia Artificial (DFKI) ha desarrollado, una arquitectura basada en agentes para la implementación de los sistemas holónicos [Fischer, 1998]. El grupo del DFKI se ha basado completamente en la arquitectura de tres niveles concurrentes, INTERRAP de Müller [Müller, 1996] (Figura 2.8). En ella la composición y configuración de las estructuras holónicas de la holarquía se realiza dentro del nivel de Planificación Cooperativa (CPL) que provee las funcionalidades para la comunicación, negociación y administración de las estructuras holónicas.

Han utilizado esta arquitectura en dominios de aplicación tales como: Suply Webs, HMS, Logística en Empresas Virtuales y Fuentes de Información basados en Agentes.

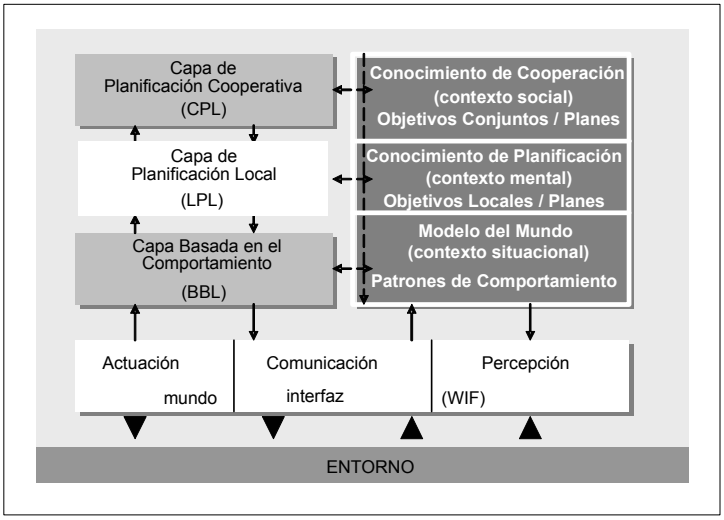


Figura 2.8: INTERRAP

La arquitectura ADACOR (Arquitectura Holónica Adaptativa para el Control de Sistemas de Fabricación) [Leitao y Restivo, 2002] propone un enfoque holónico para introducir la adaptación dinámica y ágil ante fallos en FMSs. La arquitectura se basa en un conjunto de entidades autónomas, inteligentes y cooperativas (holones), para representar los componentes de la fábrica. Estos componentes distribuidos pueden ser tanto recursos físicos (máquinas de control numérico, robots, controladores programables, etc.) como entidades lógicas (productos, órdenes, etc.). ADACOR agrupa los holones de un sistema de fabricación en holón producto, holón tarea, holón operacional, y holón supervisor [Leitao y Restivo, 2003b]. Cada producto es representado por un holón de producto que contiene todo el conocimiento relacionado con el producto y es responsable del proceso de planificación. El holón de producto recibe las órdenes para fabricar productos, para ello mantiene información acerca de la estructura del producto y el plan de procesos necesario para fabricarlo. Cada orden de fabricación se representa por un holón de tarea, que es responsable del control y supervisión de la ejecución de la orden. Incluye la descomposición

de la orden, plan de asignación de recursos y la ejecución de este plan. Los holones operacionales representan a los recursos físicos de fabricación, tales como operarios, robots y máquinas de control numérico. Gestionan el comportamiento de estos recursos de acuerdo a los objetivos, restricciones y habilidades y tratando en todo momento de optimizar su agenda. Los holones de producto, tarea y operacionales son bastante similares a los holones de producto, órden de trabajo y de recurso, presentados en la arquitectura de referencia PROSA [Van Brussel *et al.*, 1998]. El holón supervisor de ADACOR representa al holón staff de PROSA, realiza tareas de coordinación y optimización global, coordinando a varios holones operacionales y supervisores.

La arquitectura HCBA (Arquitectura Holónica Basada en Componentes) [Chirn y McFarlane, 1999] se deriva de los conceptos de CBD (Desarrollo Basado en Componentes) y HMS. Se basa en dos tipos fundamentales de holones: producto y recurso. El holón de recurso es un componente de sistema autocontenido que puede ejecutar operaciones, tales como fabricación, ensamblado, transporte, y verificación. El holón de producto puede también constar de una parte física y de otra de control. La parte física puede incluir materia prima, partes y pallets. La parte de control puede constar de control de trayectoria dentro de la línea de producción, control del proceso, toma de decisiones e información de producción. El sistema de fabricación se construye componiendo estos dos tipos de holones, con lo que se construyen estructuras anidades de productos y recursos.

En [Christensen, 2003], Christensen presenta un resumen de las tendencias actuales y propone como estándar una arquitectura integrada de bloques funcionales y dominios de cooperación (muy similar al trabajo de [Fletcher *et al.*, 2000]). Para el control de bajo nivel propone bloques funcionales, estándar IEC 61499 [IEC, 2000; 2001], y para el control de alto nivel, negociación/coordinación entre los holones de una holarquía, propone agentes, estándar FIPA [FIPA, 2005].

2.1.2. Algoritmos para control holónico

En esta sección presentamos los trabajos relacionados con la definición e implantación de algoritmos para desarrollar soluciones de control de producción basadas en holones. La Figura 2.9 representa una arquitectura jerárquica convencional (MESA) [MESA, 1995] para entornos de planificación y control de la producción, en ella se incluyen: planificación (programación) de órdenes de trabajo, scheduling (secuenciación) del plan, ejecución (lanzamiento) de las órdenes, control de máquinas para la ejecución de la orden, y control del dispositivo para la operación efectiva. Vamos a describir los desarrollos del área de control holónico aplicados a cada una de las actividades de control que aparecen en la Figura 2.9, sin olvidar que el objetivo del área de investigación de control holónico es proveer flexibilidad e inter-operabilidad entre estos niveles. En general, debido a la estructura de los sistemas holónicos, las soluciones de control y planificación holónicas se distribuyen en términos de toma de decisión y la coherencia entre estas decisiones locales se logra mediante interacciones cooperativas entre holones.

2.1.2.1. Planificación

En este apartado resumimos los trabajos relacionados con las siguientes actividades de planificación [2]: (i) la descomposición de una orden en una secuencia de operaciones de producción y (ii) la asignación nominal de las operaciones a tipos de recursos (pero no a recursos o tiempos específicos, ya que esta es una tarea de scheduling más que de planificación).

La planificación holónica de operaciones de producción ha sido estudiada en los siguientes trabajos [Gou *et al.*, 1994; Hasegawa *et al.*, 1994; Deen, 1993; 1994; Biswas *et al.*, 1995; Saad *et al.*, 1995]. Estos enfoques típicamente incluyen los siguientes pasos:

1. Cada holón de producto realiza un descomposición de la especificación del producto en partes constituyentes o sub-ensamblados.

[2] Las soluciones comerciales para la planificación en sistemas de producción, tales como MRP y MRPII, abarcan, en la actualidad, un amplio rango de tareas de planificación (así como de scheduling) además de las funciones centrales de planificación de la producción.

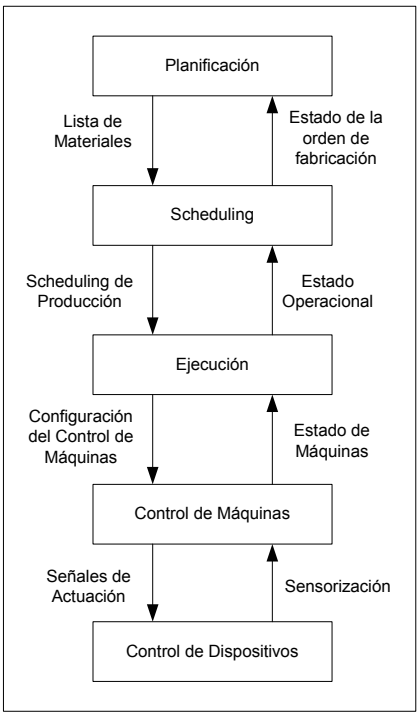


Figura 2.9: Jerarquía de control de fabricación típica

2. Por cada producto, el holón de producto identifica las operaciones de fabricación necesarias.

3. Se seleccionan los tipos de recursos necesarios para proveer las operaciones mediante un enfoque de interacción entre los holones de producto y recursos.

4. Se utiliza un proceso interactivo que involucra a los holones de recurso y de producto para determinar el conjunto apropiado de operaciones.

5. Se concreta una secuencia completa de operaciones (plan de ensamblado) que normalmente la almacena el holón de producto.

Esta secuencia de pasos asume que han sido identificados previamente los productos necesarios para cumplir con una orden de fabricación y además, que tanto el producto o el recurso están coordinando el proceso de planificación.

La ventaja del enfoque holónico de planificación comparado con enfoques

más tradicionales se basa principalmente en la naturaleza distribuida e interactiva del proceso de planificación, permitiendo que se puedan introducir nuevos productos o recursos de producción sin mayores alteraciones al sistema.

2.1.2.2. Scheduling

El scheduling holónico representa una parte significativa de todo el trabajo llevado a cabo en el área de planificación y control holónico de la producción. El scheduling en un entorno de fabricación discreta involucra: (i) la asignación de las operaciones de producción a recursos específicos, y (ii) la especificación de tiempos (inicio, duración y finalización) para estas operaciones. En particular, distinguimos el scheduling del control shop-floor ya que este último está asociado con la ejecución del schedule dentro de la celda o de la línea de producción.

En la literatura especializada podemos encontrar varias aplicaciones, por ejemplo, en el contexto de sistemas flexibles de fabricación [Ramos, 1996; Gou *et al.*, 1994; 1998], líneas de ensamblado [Sugimura *et al.*, 1996], jop shop [Marcus *et al.*, 1996], celdas de ensamblado y maquinado [Heikkila *et al.*, 1995; 1997; Ng *et al.*, 1996], líneas de proceso continuo [Agre *et al.*, 1994; McFarlane, 1995], y mantenimiento de planta [Brown y McCarragher, 1998]. En [Bongaerts *et al.*, 1997; Zhang y Norrie, 1999a; Biswas *et al.*, 1995] se presentaron métodos de scheduling genéricos aplicados al área de fabricación holónica. Una de las razones para el nivel de actividad de esta área ha sido el nivel de desarrollo de las técnicas inteligentes de scheduling [Zweben y Fox, 1994; Prosser y Buchanan, 1994] y de los algoritmos para el control distribuido de fábricas [Dilts *et al.*, 1991; Duffie y Prabhu, 1994; Lin y Solberg, 1994] ya que ambos generalmente son compatibles con un enfoque holónico. Los dos enfoques tratan de lograr una asignación más dinámica de recursos y de tiempos que lo que se puede lograr con métodos de scheduling off-line.

Las características principales de un enfoque de scheduling holónico son:

1. A cada holón se le asocia un módulo de toma de decisión local y capacidad de computación.

2. Se utiliza una estrategia de interacción cooperativa la cual rige la manera

en la que los holones intercambian información y determinan soluciones aceptadas por todos.

3. Un mecanismo o protocolo de intercambio que gestiona el intercambio de los tipos de mensajes necesarios para ejecutar una estrategia cooperativa.

4. Un medio que asegura que los requisitos globales de la fábrica son abordados.

5. Un grado de coordinación central.

Así, como en el caso de la planificación, el enfoque de scheduling holónico se diferencia de los enfoques convencionales principalmente en términos de la distribución de la computación y de las funciones de toma de decisión y de la naturaleza interactiva entre los holones.

2.1.2.3. Ejecución y Control de Taller

La ejecución o el control en el taller involucra el inicio, control, monitorización y terminación de tareas con tiempos y parámetros de producción reales. En un sistema holónico de fabricación, la ejecución se ocupa predominantemente de (i) asegurar que el holón sea capaz de establecer y mantener operaciones autónomas, y (ii) que lleve a cabo tareas compatibles con los requisitos de producción, incluso ante fallos. La ejecución ha sido estudiada en [Gayed *et al.*, 1998; Heikkila *et al.*, 1995; 1997; Valckenaers *et al.*, 1995], en estos trabajos el comportamiento autónomo de los holones de recurso, en cada caso, se gestiona mediante un modelo interno de operaciones. Los elementos nuevos del enfoque holónico para la ejecución son (a) la ejecución se realiza mediante un conjunto de pasos de negociación en lugar de una secuencia predeterminada, y (b) los recursos (máquinas) que ejecutan las operaciones de fabricación son también responsables de las decisiones tomadas acerca del tiempo y la naturaleza de la ejecución.

2.1.2.4. Control de Máquina y de Dispositivo

El control de máquinas trata con el inicio, coordinación y monitorización de las diferentes funciones de máquina o dispositivos que se requieren para la

ejecución de las tareas de producción en una máquina. El control de dispositivos se ocupa de la actuación, sensorización y control de retroalimentación de las operaciones físicas que soporta una máquina o unidad de proceso. En los sistemas holónicos el control ha sido tratado ampliamente como un problema de control convencional asociado a operaciones holónicas de más alto nivel [Bengoa, 1996; Rannanjarvi y Heikkila, 1998; Tanaya *et al.*, 1997; 1998; Zhang y Norrie, 1999b]. El punto central en estos casos ha sido lograr una interfaz efectiva. Sólo en [Overmars y Toncich, 1996] existe la posibilidad de que una máquina que se ejecuta mediante principios holónicos, considere realmente por su cuenta, que la interacción de los dispositivos individuales, que constituyen la máquina, se determine de manera cooperativa.

2.2. Modelado de Sistemas Holónicos de Fabricación

En esta sección presentamos el estado del arte de técnicas y métodos de modelado de sistemas de fabricación. En primer lugar resumimos los requisitos de modelado de los sistemas de fabricación de la nueva generación. Posteriormente presentamos los trabajos reportados en las áreas HMS, SMA y modelado de empresas para terminar con un resumen comparativo de los distintos enfoques.

2.2.1. Requisitos de modelado

Un sistema de fabricación, entendido como un sistema global y complejo que incluye el conjunto completo de actividades dentro de una empresa de fabricación (Figura 1.1), posee un conjunto de características definidas por la "nueva fabricación" (Capítulo 1). A partir de estas características surgen una serie de requisitos, los cuales los podemos agrupar en dos grandes grupos:

2.2.1.1. Requisitos Funcionales

Los sistemas de control de fabricación son sistemas complejos de gran escala, que se diseñan para desempañar una tarea claramente definida en un ambiente estandarizado y bien estructurado. Aunque los procesos de producción experimentan varios cambios y perturbaciones, el grado de incertidumbre

e impredecibilidad no es comparable a aquellos sistemas del espacio, del tráfico, o aplicaciones de servicio.

Los módulos o entidades (holones) que implementan el control en estos sistemas deben cooperar a fin de lograr los objetivos globales de fabricación. Con respecto a estos objetivos, un holón nunca rechaza de manera deliberada la cooperación con otro holón. Sólo rechaza su ejecución, cuando las acciones solicitadas son imposibles o fuertemente desventajosas para el proceso de producción. En este sentido, los holones de producción son semi-autónomos, esta característica está definida implícitamente en la manera en que se organizan los holones para cooperar. Una holorquía es una estructura jerárquica "floja" en la que existe cierto grado de subordinación al holón que la gestiona (ejemplo, el holón staff de la arquitectura PROSA, o el holón primario o coordinador de los dominios de cooperación, ver Sección 2.1.1) y al mismo tiempo autonomía para la actuación de cada holón en interacciones de cooperación. Esto nos lleva al primer requisito:

Requisito 1: Los sistemas de control de fabricación requieren entidades autónomas que se organizan en estructuras que son a la vez jerárquicas y heterárquicas.

El segundo requisito trata con el tipo de comportamiento que la unidad de control a nivel de fábrica debe exhibir. Las unidades de control de producción están continuamente enfrentadas a alto grado de eventos repetidos que son conocidos, pero impredecibles. Este flujo de eventos debe ser manejado de manera efectiva y dentro de limitaciones temporales. El manejo de los eventos puede consecuentemente ser fijado de ante mano con la ayuda de rutinas, mientras que el inicio y la ejecución de las rutinas deben ser realizadas de manera "on-line" (tiempo real). El conjunto de eventos y su patrón de ocurrencia cambia con el tiempo.

Requisito 2: Las unidades de control de fabricación principalmente requieren de un comportamiento basado en rutinas que es al mismo tiempo efectivo y con características de tiempo real. Este comportamiento puede ser tanto configurable o auto-adaptativo.

2.2.1.2. Requisitos de Ingeniería del Software

Aparte de los requisitos funcionales, cualquier sistema de control que se supone será utilizado en un ambiente de fabricación productivo debe satisfacer estándares industriales generales. Estos estándares especifican, entre otras cosas, requisitos para fiabilidad, tolerancia a fallos, diagnóstico, y mantenibilidad. En particular, los sistemas de control deben lograr ciertos grados de fiabilidad que garantizan una operación continua. Esto es igualmente verdadero para el software de control. La confiabilidad del producto, en cambio, se logra únicamente si el proceso de desarrollo del software se realiza siguiendo una metodología de ingeniería, en lugar de desarrollarlo de manera ad-hoc.

A partir de los principios fundamentales de la Ingeniería del Software se derivan los siguientes requisitos:

Requisito 3: Los métodos de programación deben proveer encapsulación de datos y procesos.

Requisito 4: Los programas de control deben tener una semántica clara.

En la literatura especializada, se han seguido, básicamente, se dos enfoques para la descomposición de problemas en el área de la fabricación. (i) La descomposición física (la más obvia): los agentes son utilizados para representar las entidades del mundo físico, tales como trabajadores, máquinas, herramientas, planos, productos, partes, características y operaciones. Este enfoque naturalmente define conjuntos distintos de variables de estado que deben ser gestionados de manera eficiente por los agentes individuales con un número limitado de interacciones; sin embargo, requiere un gran número de agentes relacionados con recursos. Un ejemplo común de este enfoque es el uso de agentes parte y agentes máquina para la planificación y scheduling de fabricación [Duffie y Prabhu, 1994; Lin y Solberg, 1992; Parunak, 1987]. (ii) En la descomposición funcional los agentes son utilizados para encapsular funcionalidades tales como adquisición de órdenes de trabajo, planificación, scheduling, manejo de materiales, gestión del transporte y distribución de productos. En este enfoque los agentes no tienen relación explícita con las entidades físicas. Ejemplos de este enfoque incluyen el uso de agentes para encapsular funcionalidades especiales (por ejemplo, agentes facilitadores [Cutkosky *et al.*, 1993], agentes

broker [Peng *et al.*, 1998], y agentes mediadores [Maturana y Norrie, 1996]) y el uso de agentes para integrar sistemas existentes (por ejemplo, ARCHON [Cpckburn y Jennings, 1999], EXPORT [Monceyron y Barthes, 1992] y CIIMPLEX [Peng *et al.*, 1998]). dos enfoques para la descomposición de problemas en el área de la fabricación. La descomposición física (la más obvia): los agentes son utilizados para representar las entidades del mundo físico, tales como trabajadores, máquinas, herramientas, planos, productos, partes, características y operaciones. Este enfoque naturalmente define conjuntos distintos de variables de estado que deben ser gestionados de manera eficiente por los agentes individuales con un número limitado de interacciones; sin embargo, requiere un gran número de agentes relacionados con recursos. Un ejemplo común de este enfoque es el uso de agentes parte y agentes máquina para la planificación y scheduling de fabricación [Duffie y Prabhu, 1994; Lin y Solberg, 1992; Parunak, 1987]. En la descomposición funcional los agentes son utilizados para encapsular funcionalidades tales como adquisición de órdenes de trabajo, planificación, scheduling, manejo de materiales, gestión del transporte y distribución de productos. En este enfoque los agentes no tienen relación explícita con las entidades físicas. Ejemplos de este enfoque incluyen el uso de agentes para encapsular funcionalidades especiales (por ejemplo, agentes facilitadores [Cutkosky *et al.*, 1993], agentes broker [Peng *et al.*, 1998], y agentes mediadores [Maturana y Norrie, 1996]) y el uso de agentes para integrar sistemas existentes (por ejemplo, ARCHON [Cpckburn y Jennings, 1999], EXPORT [Monceyron y Barthes, 1992] y CIIMPLEX [Peng *et al.*, 1998]). De esto se deriva el siguiente requisito:

Requisito 5: Un método o metodología de desarrollo para sistemas de fabricación holónicos debe proveer una mecanismo de traducción de tareas de control, sobre un recurso de fábrica, o de funcionalidades de la fábrica a entidades con comportamiento autónomo [HMS, 1994].

En el área de sistemas de fabricación se reconoce la necesidad de alguna forma de agregación jerárquica "floja" en sistemas del mundo real, los cuales deben permanecer comprensibles mientras se expanden en un amplio rango de escalas temporales y espaciales. Una fábrica moderna de automóviles, por ejemplo, incorpora cientos de miles de mecanismos individuales en cientos

de máquinas que están agrupadas en docenas o más líneas de fabricación. Los ingenieros pueden diseñar, construir, y operar tales sistemas complejos cambiando el enfoque (nivel de abstracción) desde el mecanismo, a la máquina o la línea de fabricación, dependiendo del problema en cuestión, y reconociendo a los agentes de niveles superiores como agregaciones de agentes de niveles inferiores.

Requisito 6: Un método o metodología de desarrollo para sistemas de fabricación holónicos debe ofrecer al ingeniero estructuras y mecanismos que permitan llevar a cabo procesos de análisis y diseño por capas de abstracción.

Los enfoques tradicionales para el modelado de sistemas de fabricación, tales como CIM, se basan principalmente en un enfoque top-down. Los requisitos del usuario y el diseño conceptual global constituyen el conjunto completo de restricciones de modelado. Con estos enfoques se crean arquitecturas jerárquicas muy rígidas [HMS, 1994]. Por otra parte el modelado de sistemas de fabricación holónicos requiere de métodos de desarrollo mixto, bottom-up y top-down según el nivel que se esté modelando. No se necesita definir el conjunto completo de restricciones al principio. Con esto se permite generar arquitecturas reconfigurables y escalables. Esta característica implica el siguiente requisito:

Requisito 7: Un método o metodología de desarrollo para sistemas de fabricación holónicos debe definir un proceso de desarrollo mixto que integra los enfoques top-down y bottom-up.

Finalmente el siguiente requisito se deriva de las características de la "nueva fabricación" definidas por el consorcio IMS.

Requisito 8: Un método o metodología de desarrollo para sistemas de fabricación holónicos debe integrar el rango completo de actividades de producción, desde la recepción de una orden pasando por el diseño, la fabricación, y el mercadeo para lograr la empresa de fabricación ágil [HMS, 1994].

Teniendo en cuenta estos requisitos a continuación presentamos un análisis de las distintas metodologías reportadas en: (i) el área HMS; (ii) el área SMA, debido a la tendencia generalizada de utilizar la tecnología SMA como

herramienta de implementación de sistemas holónicos (ver Sección 2.1), y finalmente; (iii) modelado de empresas, debido a que un sistema de fabricación forma parte de una empresa de fabricación. El objetivo de este estudio es determinar si estas metodologías son adecuadas para modelar sistemas holónicos de fabricación. En primer lugar presentamos un breve resumen de las distintas metodologías (los detalles se pueden consultar en la literatura especializada). Finalmente presentamos un resumen comparativo en base a los requisitos que enunciamos en esta sección.

2.2.2. Métodos HMS

Es de extrañar que en el área de HMS se hayan reportado (hasta el momento de la escritura de esta Tesis) tan pocos trabajos sobre metodologías. Quizá se deba a que el esfuerzo investigador se ha centrado en el desarrollo de sistemas de control holónicos y en la definición de arquitecturas (Sección 2.1), sin embargo se reconoce la necesidad de disponer de metodologías de diseño que provean de guías de desarrollo claras y no ambiguas [McFarlane y S., 2003].

Leitão y Restivo en [Leitao y Restivo, 2003a], proponen *Un enfoque formal para la especificación de Sistemas Holónicos de Control*. Esta propuesta trata de formalizar la estructura y comportamiento de un sistema holónico de fabricación. Combina la notación UML para especificar la estructura y los aspectos estáticos del sistema y las Redes de Petri para modelar los aspectos de comportamiento. La metodología se basa en la arquitectura ADACOR [Leitao y Restivo, 2002].

La estructura estática del sistema consiste en especificar el conjunto de clases de objetos del sistema, los atributos, métodos y relaciones. Esta estructura se asemeja mucho a los modelos conceptuales (diagrama de clases) de las metodologías orientadas a objetos.

El modelado del sistema se basa en el desarrollo de un modelo de comportamiento para cada uno de estos tipos/clases de holones. Así se tiene:

- por cada producto existe un holón producto que es responsable de toda la planificación del proceso y contiene todo el conocimiento relacionado con el producto. El comportamiento se modelo con un *Modelo de holón*

producto. Este modelo consisten en una Red de Petri que especifica los distintos estados por los que debe pasar y las precondiciones y post-condiciones de cada uno de ellos. También se utiliza para sincronizar distintos holones entre si.

- cada orden de fabricación se representa por un holón tarea, que es responsable del control y supervisión de la ejecución de la orden de fabricación y contiene la información dinámica. El comportamiento de un holón tarea se modelo con un *Modelo de holón tarea*. Esta Red de Petri especifica las funciones de descomposición de la orden, planificación de asignación de recursos y la ejecución de estos planes.

- los holones operacionales representa los recursos físicos de fabricación, tales como operarios, robots y máquinas de control. El comportamiento se modela con el *Modelo de holón operacional*. Éste holón actúa como un servidor reactivo que espera nuevas operaciones (propuestas por el holón supervisor o por los holones tarea).

- el *Modelo de holón supervisor* especifica el comportamiento del holón supervisor, que se encarga de coordinar la actividad de los holones a su mando, introduce coordinación y optimización global y puede coordinar a varios holones operacionales y supervisores.

La propuesta de Leitão y Restivo es muy preliminar, sufre de varias carencias. Por ejemplo no hay manera de especificar relaciones entre los holones del sistema, no se específica cual es la relación que existe entre la estructura estática del sistema y los distintos modelos de comportamiento de los holones. No se establecen etapas de desarrollo. No se especifica cómo se traduce cada modelo de holón en alguna plataforma de implementación. Finalmente lo más crítico, con este enfoque no se modela cooperación, autonomía ni flexibilidad, que son las características holónicas básicas definidas por el consorcio HMS.

Otro propuesta es la de [Fischer *et al.*, 2003]. En ella se propone una organización de agentes para modelar cada holón/holarquía. Las características del holón se especifican como características de la organización de agentes que lo modela. No se rige a ninguna arquitectura de holón particular. Propone distintos tipos de organización que pueden ser utilizados para modelar desde

la total libertad (sistemas heterárquicos) de las entidades autónomas, auto-interesadas, hasta los sistemas jerárquicos de control centralizado, pasando por las holarquías en las que las entidades semi-autónomas son dirigidas por una "cabeza" dentro de la organización. El enfoque de Fisher et. al. también carece de las distintas etapas de desarrollo, se centra sólo en la definición de las holarquías (organizaciones) y sus distintas configuraciones. En un trabajo anterior [Fischer, 1999], propone un diseño basado en la arquitectura INTERRAP (Figura 2.8). Define un modelado por capas de un FMS. Reconoce cinco capas: control y planificación de producción, control de shop floor, control de celda, control de sistemas autónomos, y control de máquina. Se mantiene una estructura jerárquica entre capas y una estructura horizontal cooperativa entre los componentes de la misma capa. Nuevamente falta la definición del proceso de desarrollo, no define una notación para el modelado. No obstante es el único trabajo que considera al sistema de fabricación global integrado en la empresa.

2.2.3. Métodos Multi Agente

En esta sección presentamos un resumen de las metodologías SMA más conocidas en el área. En dos grandes grupos: metodologías SMA de propósito general [3] y metodologías SMA para sistemas de fabricación.

2.2.3.1. Métodos SMA de propósito general

Modelado y diseño de sistemas multi agente en BDI. Este método [Kinny y Georgeff, 1997] extiende técnicas de modelado OO a sistemas de agente basados en la arquitectura BDI [Georgeff y Rao, 1995]. Propone dos niveles de abstracción: externo e interno. En la vista externa, el sistema se modela como una jerarquía de clases de agente (modelo de agente). Las clases de agentes están caracterizadas por su propósito, sus responsabilidades, los servicios que desarrollan, la información acerca del mundo que requieren y mantienen y las interacciones externas (modelo de interacción). Desde el punto de vista interno

[3] Dentro del grupo de metodologías de propósito general se pueden diferenciar a su vez distintos grupos, por ejemplo, orientadas a roles, orientadas al agente, orientadas a la organización, orientadas al sistema, orientadas a la interacción, orientadas al comportamiento, extensiones de metodologías orientadas al conocimiento, etc. Este estudio, sin embargo, queda fuera del alcance de esta tesis.

se emplea un conjunto de modelos (modelo de creencias, modelo de objetivos y modelo de planes) los cuales permiten estructurar el estado de motivación y de información de los agentes así como de las estructuras de control que determinan sus conductas. Se basa, además, en la identificación de roles clave en la aplicación y sus interrelaciones, los cuales sirven de guía para la elaboración de la jerarquía de clases de agente. El análisis de las responsabilidades de cada rol, representado por una clase de agentes, lleva a la identificación de los servicios que provee y usa cada agente, las interacciones externas y los objetivos y eventos a los que se debe responder.

En [Burmeister, 1996], Burmeister presenta un método para el desarrollo de SMAs en el que se especifican tres modelos para el análisis orientado a agentes y se definen algunas guías de construcción. El método se basa en técnicas OO. Divide el análisis en tres submodelos: (i) Modelo de agente, contiene los agentes y su estructura interna (estados o nociones mentales). (ii) Modelo de organización especifica las relaciones, fundamentalmente de tipo jerárquico, entre los agentes y tipos de agentes. (iii) Modelo de cooperación describe la cooperación entre agentes. La conjunción de estos tres modelos define de manera completa el sistema a desarrollar. La fase de diseño consiste en el refinamiento de estos modelos para la implementación posterior del sistema mediante alguna herramienta concreta.

La metodología MAS-CommonKADS [Iglesias *et al.*, 1998] se basa en CommonKADS [Schreiber, 1999], aportando una serie de modelos para desarrollar las fases de análisis y de diseño de SMAs. La principal característica es la incorporación de técnicas orientadas a objetos a CommonKADS, la cual se toma como eje fundamental a lo largo de todo el proceso. En MAS-CommonKADS se proponen diferentes vistas del sistema que se traducen en diferentes fases. En la fase de conceptuación se propone un análisis centrado en el usuario para la captura de requisitos (se emplean casos de uso). El modelo de organización analiza la empresa (organización) en que va a ser introducido el sistema, y sirve para describir las relaciones de la sociedad multi agente entre sí y con su entorno. El modelo de agente describe las propiedades y características de cada agente. Es una de las fases más importantes, consta de una subfase de identificación de agentes (con diferentes opciones) y una subfase de descripción

detallada de cada agente. El modelo de tareas describe las funciones o tareas (cognitivas, comunicación hombre-máquina, comunicación entre agentes) que desarrolla el sistema. En el modelo de experiencia se describe cómo se llevan a cabo tareas cognitivas. El modelo de comunicación se centra en describir cómo se llevan a cabo las tareas de interacción hombre-máquina. El modelo de coordinación desarrolla y describe las interacciones entre los agentes de un sistema multi agente. En el modelo de diseño se desarrolla un diseño de la red, de los agentes (arquitectura de agente) y de la plataforma (sistema operativo y plataforma hardware).

GAIA [Wooldridge *et al.*, 2000] se centra en la idea de que la construcción de sistemas basados en agente es un proceso de diseño organizacional. Una organización en GAIA es una colección de roles, los cuales mantienen ciertas relaciones con otros y toman parte en patrones institucionalizados de interacción con otros roles. Los roles agrupan cuatro aspectos: responsabilidades del agente, los recursos que se le permite utilizar, las tareas asociadas, y las interacciones. GAIA propone trabajar inicialmente con un análisis de alto nivel. En él se usan dos modelos, el modelo de roles para identificar los roles clave en el sistema junto con sus propiedades y el modelo de interacciones que define las interacciones mediante una referencia a un modelo institucionalizado de intercambio de mensajes. Al finalizar esta etapa se pasa al diseño a alto nivel cuyo objetivo es generar tres modelos: el modelo de agentes que define los tipos de agentes que existen, cuántas instancias de cada tipo y qué papeles juega cada agente; el modelo de servicios que identifica los servicios (funciones del agente) asociados a cada rol; y, el modelo de "conocidos" que define los enlaces de comunicaciones que existen entre los agentes. A partir de aquí se aplicarían técnicas clásicas de diseño OO. Sin embargo, esto queda fuera del ámbito de GAIA.

GAIA v.2 [Zambonelli *et al.*, 2003], extiende GAIA basándose en la consideración fundamental que una organización es más que una simple colección de roles, como fue considerado en la primera versión de GAIA, y por tanto introduce nuevas abstracciones organizacionales. Además de los roles y protocolos, el entorno en el cual el SMA está inmerso constituye la abstracción

principal de análisis y diseño. Más aún, se incorporan reglas y estructuras organizacionales como elementos necesarios para la definición de la organización. De esta manera se intenta adaptar la versión original de GAIA para el diseño y construcción de sistemas abiertos.

ROADMAP [Juan *et al.*, 2002] se inició como un intento de extender la versión original de GAIA con: una jerarquía dinámica de roles (una manera de tratar sistemas abiertos), modelos adicionales para describir de manera explícita el entorno del agente (tal como lo hace GAIA v.2). ROADMAP hereda de GAIA, además de los modelos básicos, la vista organizacional del SMA, y las definiciones básicas de roles, protocolos, agentes y servicios. En ROADMAP, un sistema es visto como una organización de agentes, y consiste de una jerarquía de roles y una jerarquía de agentes. La jerarquía de roles es la especificación del sistema, y representa el comportamiento correcto de los agentes. La jerarquía de agentes es la implementación del sistema, y provee su funcionalidad actual. El modelo de entorno y el modelo de conocimiento contienen información de dominio reutilizable. El modelo de casos de uso, el modelo de interacción, el modelo de roles, el modelo de agentes y el modelo de conocidos son específicos de la aplicación. El modelo de protocolo y el modelo de servicios describen los componentes software de bajo nivel potencialmente reutilizables.

El método de desarrollo de sistemas multi agente MASSIVE (Multi-Agent SystemS Iterative View Engineering) desarrollado en el DFKI [Lind, 1999] está constituido por un conjunto de vistas diferentes del sistema a construir donde el desarrollo que se sigue consiste en una visión iterativa del mismo. En él se combinan procesos de reingeniería junto con un método en cascada mejorado que permite realizar refinamientos. Las diferentes vistas que constituyen el método son: Tareas, Entorno, Roles, Interacciones, Sociedad, Arquitectura y Sistema. En la vista de Tarea se analizan los aspectos funcionales del sistema, se genera una jerarquía de tareas, la cual es empleada para determinar las capacidades de resolución de problemas básicos de las entidades del sistema final. La vista de Entorno consiste básicamente en realizar un análisis del entorno desde dos puntos de vista: del desarrollador y del sistema. La vista de Roles empieza a aplicar el paradigma multi agente como solución del problema. Esta

vista consiste en determinar las abstracciones de roles necesarias para cubrir la funcionalidad y restricciones físicas de la especificación del problema. Además, se empieza a considerar cómo asignar roles particulares a agentes concretos. En la vista de Interacciones se modelan las interacciones del sistema. En MASSIVE las interacciones del sistema son vistas como una forma generalizada de resolución de conflictos no limitada a una forma particular, como pueda ser la comunicación. En la vista de Sociedad el objetivo es clasificar la sociedad (conjunto de agentes) que o bien pre-existe o que es deseable desde el punto de vista del desarrollador. La vista de Arquitectura consiste en transformar la especificación del resto de vistas en una arquitectura de sistema donde se definan los atributos estructurales del mismo. La vista de Sistema trata con aspectos del sistema que afectan al resto de vistas o al sistema en su conjunto. Ejemplos de los aspectos tratados en esta vista son el diseño de interfaces de usuario, la estrategia de manejo de errores o la propia implantación del sistema.

En Tropos [Mylopoulos y Castro, 2001] se presenta una metodología de desarrollo de software basado en agentes mediante extensiones de UML y empleando un entorno de modelado denominado i^* [Yu, 1996]. La metodología empieza por la captura de requisitos mediante el modelado de objetivos, tareas, recursos y actores del sistema así como de las dependencias entre los actores. Durante el diseño, el modelo de requisitos se elabora mediante (i) el refinamiento de los actores del sistema, (ii) la identificación de las capacidades necesarias para satisfacer los objetivos del sistema, y (iii) la asignación de estas capacidades a agentes. En el primer paso de diseño, el diagrama de actores del sistema puede ser ampliado con la ayuda de patrones de diseño, aunque realmente no se explica de manera clara cómo se identifican los patrones apropiados, para los demás pasos de diseño la metodología no incluye guías para traducir los elementos identificados en los modelos previos en entidades implementables en alguna plataforma de agente.

MaSE (Multiagent System Engineering) es una metodología realizada en el Air Force Institute of Technology en Ohio por Mark Wood y Scott DeLoach [Wood, 2000]. Su lenguaje de especificación se basa en UML + OCL [Robinson, 2000] y una herramienta de desarrollo denominada AgentTool. Los sistemas

que se pueden diseñar con esta metodología deben ser sistemas cerrados. La escala de los sistemas a construir no debe ser muy elevada, un máximo de diez clases de agente. En la metodología no se incorporan elementos para soportar características de movilidad. No se pueden consider sistemas de tipo dinámico, donde los agentes pueden ser creados y destruidos. Las conversaciones entre agentes se asumen que son uno a uno y no de tipo multicast. Desde un punto de vista general MaSE se divide en dos grandes etapas: análisis y diseño. El análisis se divide en tres fases: captura de objetivos, transformación de roles y aplicación de casos de uso. Por su parte el diseño se divide en cuatro fases: creación de clases de agente, construcción de conversaciones, ensamblado de clases de agente y diseño del sistema. Comienza identificando los objetivos y casos de uso. Posteriormente, los objetivos forman los roles que incorporan las tareas para obtener los objetivos. Los casos de uso se transforman en diagramas de secuencia para tener en cuenta las secuencias de eventos a ser diseñadas. A continuación los roles se integran en clases de agente conectados mediante conversaciones, las cuales se especifican mediante diagramas específicos. Finalmente, se obtienen los agentes a partir de las clases de agente en un diagrama a partir del cual el sistema, según los autores, podría ser generado automáticamente. Sin embargo, actualmente esto no es posible ya que su herramienta de desarrollo no implementa esta posibilidad.

MESSAGE [EURESCOM, 2000; 2001b](Methodology for Engineering Systems of Software Agents) es una metodología orientada a agentes la cual incorpora técnicas de ingeniería del software cubriendo el análisis y diseño de sistemas multi agente. La metodología provee un lenguaje, un método y unas guías de cómo aplicar la metodología, centrándose en las fases de análisis y diseño y lanzando ideas sobre el resto de etapas como implementación, pruebas e implantación. La notación utilizada es UML y el proceso de modelado es el propuesto por Rational (RUP) [Jacobson *et al.*, 1999], que consta fundamentalmente de 4 fases: Inicio, Elaboración, Construcción y Transición. Para desarrollar un SMA, MESSAGE ofrece unas guías para determinar si es apropiado el describir las entidades (o al menos algunas de ellas) del sistema propuesto como agentes [EURESCOM, 2001a].Etapa de Requisitos: las características de los requisitos iniciales para el caso de SMAs no tienen según MESSAGE nada

de especial respecto a otros sistemas. En la etapa de análisis: se provee un modelo de análisis de agentes que introduce la abstracción de agente como uno de los bloques de construcción fundamentales. Otros elementos del análisis son: tareas, objetivos, roles e interacciones. En esta etapa se construyen modelos de: organización, objetivos y tareas, agente, interacción y modelo de dominio [Caire *et al.*, 2001]. En la etapa de diseño: el modelo de diseño provee constructores para describir la comunicación entre agentes y entre un agente y su entorno. También permite la definición de la estructura interna de los agentes. Para la etapa de implementación: se indica que se deben proveer guías para la selección de la plataforma de agente que soporte el sistema a ser desarrollado.

RT-MESSAGE [Julián, 2002], propone una extensión de MESSAGE para el modelado de SMAs de tiempo real. Se basa en la plataforma de SMA de tiempo real SIMBA [Soler *et al.*, 2002]. El principal componente de SIMBA es la arquitectura de agente de tiempo real ARTIS [Botti *et al.*, 1999]. RT-MESSAGE considera las etapas de análisis, diseño e implementación. En la etapa de análisis se emplean los modelos propuestos por MESSAGE ampliados para poder modelar las características propias de sistemas de tiempo real. Por lo que respecta al diseño, toma como entrada todos los artefactos generados en el análisis, centrándose en la transformación de las entidades especificadas en dichos artefactos en entidades computacionales. Para ello se emplea la arquitectura SIMBA, la cual permite diseñar las interacciones, organizaciones y estructura interna de los agentes de tiempo real del sistema. Finalmente, propone la implementación del diseño realizado mediante una herramienta de desarrollo que permite la obtención de un prototipo directamente ejecutable.

INGENIAS [Gómez, 2002] se presenta como una evolución de las ideas de MESSAGE. INGENIAS profundiza en los elementos de MESSAGE para la especificación y el proceso de desarrollo, además de incorporar nuevas herramientas de soporte y ejemplos de desarrollo. INGENIAS, al igual que MESSAGE, define un conjunto de meta-modelos (una descripción de alto nivel de qué elementos tiene un modelo) con los que hay que describir el sistema. Los meta-modelos indican qué hace falta para describir: agentes aislados, organizaciones de agentes, el entorno, interacciones entre agentes o roles, tareas y

objetivos. El proceso de instanciación de los meta-modelos (creación de modelos) no es trivial. Existen muchas entidades y relaciones a identificar, además de dependencias entre distintos modelos. Por ello, INGENIAS define un conjunto de actividades cuya ejecución termina en un conjunto de modelos. Estas actividades a su vez se organizan siguiendo un paradigma de ingeniería del software, el RUP (Proceso Unificado) [Jacobson *et al.*, 1999]. La ejecución de las actividades para producir modelos se basa en la herramienta INGENIAS IDE, una herramienta para modelado visual. Esta herramienta almacena la especificación del sistema utilizando XML. A partir de esta especificación genera código y documentación.

MASB es una metodología propuesta en [Moulin y Brassard, 1996]. Identifica roles de agentes (humanos o artificiales) en una aplicación mediante el análisis de escenarios en los cuales los agentes interactúan con un usuario potencial. En la etapa de análisis, el usuario describe (de manera textual) un escenario típico enfatizando los roles desempeñados por humanos y agentes artificiales, los intercambios típicos de mensajes, los eventos que ocurren durante la ejecución del escenario y las acciones realizadas por los agentes. Un rol se caracteriza mediante un diagrama de comportamiento especificando las actividades, el conocimiento y las interacciones en las que participa el rol particular. La etapa de análisis finaliza con el modelado de los datos locales, la descripción estática y dinámica del mundo, y las interacciones entre el sistema y los usuarios. En la etapa de diseño, se identifican los agentes y se asignan roles a estos agentes. Una vez que se han identificado los agentes, el diseñador especifica las estructuras de conocimiento que caracterizan a los agentes (creencias, decisiones, acciones y razonamiento para desempeñar un rol). Finalmente, se especifican las conversaciones entre agentes sobre la base de los planes que un agente puede ejecutar.

El método Prometheus [Padgham y Winikoff, 2002] comienza con la etapa de análisis en la cual se identifican los objetivos y funcionalidades básicos del sistema a ser desarrollado. Para identificar objetivos, funcionalidades, así como las interacciones entre las funcionalidades, la metodología propone analizar casos de uso típicos. Una vez las funcionalidades hayan sido identificadas, éstas

se agrupan y asignan a agentes de acuerdo al criterio de coherencia y acoplamiento de la ingeniería del software tradicional. En particular provee las siguientes estrategias para agrupar funcionalidades: (i) agrupar, si las funcionalidades usan el mismo dato o requieren la misma información; (ii) agrupar, si esto crea menos enlaces de interacción entre los agentes; (iii) no agrupar, si las funcionalidades no están relacionados o si deberían residir en diferentes plataformas hardware; (iv) no agrupar, si el dato de una funcionalidad no debería estar disponible para otra funcionalidad debido a razones de seguridad o privacidad; y (v) no agrupar, si las funcionalidades cambiarán, o serán modificadas por diferentes personas. Para aplicar estas estrategias, Prometheus provee un diagrama de acoplamiento de datos y un diagrama de "conocidos". De esta manera el proceso de agrupamiento se basa únicamente en el acoplamiento de datos e interacción.

Elammari y Lalonde proponen un método que va desde especificaciones de alto nivel hasta modelos implementables a través de fases de descubrimiento y definición [Elammari y Lalonde, 1999]. Genera cinco modelos: (i) el modelo de alto-nivel identifica agentes y sus comportamientos de alto-nivel; (ii) el modelo interno de agentes describe la estructura interna y el comportamiento de los agentes; (iii) el modelo de relaciones captura las dependencias y las relaciones jurisdiccionales; (iv) el modelo de conversacional describe la coordinación entre los agentes; y finalmente (v) el modelo de contrato define una estructura de acuerdos entre los agentes. La fase de descubrimiento desarrolla el modelo de alto nivel con ayuda de mapas de casos de uso [Buhr, 1998]. Sobre la base de los mapas de casos de uso esta fase identifica a los agentes extrayendo de la descripción del problema sujetos que juegan roles activos en la aplicación. Además identifica los roles y responsabilidades. La fase de definición crea los cuatro modelos restantes.

Miles y otros proponen un método de diseño, llamado "Análisis de Interacción entre Agentes" [Miles *et al.*, 2001], que deriva las interacciones necesarias a partir de un análisis de objetivos y preferencias de los requisitos del sistema. La metodología comienza con la identificación de los objetivos del sistema y elabora estos objetivos en una jerarquía de objetivos. Se asume que cada objetivo se resuelve mediante la interacción de algunos agentes. En un segundo

paso, el diseñador elige un mecanismo de interacción para cada objetivo independiente de la jerarquía de objetivos usando roles para referirse a los agentes que pueden participar en la interacción. Es durante la ejecución del sistema, cuando el objetivo es asumido por un agente "real" que se encarga de buscar a otros agentes para que jueguen los demás roles de la interacción. En caso de que el agente falle en la búsqueda, éste puede utilizar la jerarquía de objetivos para descomponer el objetivo en sub-objetivos que más tarda propone para que se inicien las interacciones asociadas a ellos. Los agentes que se necesitan en tiempo de ejecución se derivan a partir de los roles requeridos en las interacciones. Estos roles se agrupan de acuerdo a las preferencias identificadas en la definición del problema y en base a consideraciones generales, tales como complejidad o funcionalidad limitada de los agentes, u optimización de la carga computacional en un agente individual.

Collinot y otros, proponen Cassiopeia [Collinot *et al.*, 1996] como un método de diseño para derivar el comportamiento individual de agentes a partir de la especificación global de una tarea colectiva. Cassiopeia se centra en la organización del SMA para enlazar el comportamiento individual con el de grupo. Distingue tres niveles de comportamiento: (i) elemental, (ii) relacional, y (iii) organizacional. Cassiopeaia empieza con los comportamientos elementales y construye el sistema definiendo comportamientos más complejos. Cada nivel se diseña en un paso distinto: a) Identificación de los comportamientos elementales. Los comportamientos elementales son aquellos comportamientos que se requieren para lograr ejecutar una tarea colectiva. b) Especificación de las relaciones entre comportamientos. Analizando la estructura organizacional con respecto a las dependencias entre los comportamientos elementales. c) Especificación del comportamiento organizacional. Los comportamientos organizacionales son aquellos que permiten a los agentes gestionar la formación, duración o disolución de los grupos.

El entorno Agent Societies [Dignum *et al.*, 2002] se propone como un método orientado a la organización para el desarrollo de SMAs. Describe: (i) la estructura y las características globales de un dominio a partir de una perspectiva organizacional (modelo de organización); (ii) define la población de

agentes mediante contratos sociales, que regulan la ejecución de roles por parte de agentes individuales (modelo social); y (iii) la interacción entre los agentes mediante contratos de interacción (modelo de interacción).

El método Civil Agent Societies [Dellarocas y Klein, 1999] se basa en sociedades civiles humanas, en las cuales las instituciones sociales establecen e imponen las leyes, y monitorizan el cumplimiento de las mismas para responder ante posibles emergencias. Se basa en: (i) servicios de socialización, en los cuales se establecen contratos sociales entre los agentes y la sociedad, indicando la pertenencia del agente a la sociedad; (ii) servicio de notaría, que verifica que las interacciones entre los agentes son legales para generar un contrato privado apropiado; y (iii) el servicio de manejo de excepciones, que inicializa agentes centinela cuando se generan nuevos contratos. Estos agentes centinela tratan de evitar excepciones y de detectar sus síntomas.

2.2.3.2. Métodos SMA para sistemas de fabricación

En esta sección resumimos trabajos reportados en la literatura especializada sobre métodos SMA para sistemas de fabricación. A excepción del primero de los métodos, los demás son bastante recientes.

Kendall y otros [Kendall *et al.*, 1996] proponen una metodología para el desarrollo de sistemas basados en agentes que se construye en base a metodologías orientadas a objetos, tal como OMT y OOSE [Jacobson, 1992], y el método IDEF [IDEF, 2004] para el modelado de sistemas de fabricación (ver Sección 2.2.4). El método empieza creando un modelo OO y un modelo IDEF (IDEF0) del sistema que se está modelando, y posteriormente identifica los agentes y las interacciones entre estos dos modelos. Los agentes son vistos como los decisores autónomos y son identificados en el modelo OO cuando aparece un actor, y en el modelo IDEF cuando una función crea como salida información de control. Los agentes identificados se modelan como agentes BDI, en particular como sistemas de razonamiento procedural [Rao y Georgeff, 1992]. Las interacciones se identifican cuando dos o más recursos de IDEF aparecen en un intercambio de información. Los patrones de interacción se definen a partir de los casos de uso correspondientes [Jacobson, 1992]. Se completa la definición de los agentes y las interacciones con la ayuda de técnicas de diseño

OO. A pesar de definirse como un método para el desarrollo de sistemas basados en agentes, esta propuesta está limitada por su propia basa (métodos OO y método IDEF0) ya que basa todo el proceso de identificación y definición de agentes en técnicas que no modelan de manera explícita el comportamiento autónomo, ni la toma de decisiones. Además los conceptos orientados a objetos presentan varias limitaciones a la hora de identificar y definir agentes [Wooldridge, 1997].

Ritter y otros [Ritter *et al.*, 2002] proponen un método de "agentificación" para sistemas de fabricación (en otras palabras, un método para la identificación de agentes en un sistema de fabricación). El proceso de agentificación se basa en la agregación física o de dependencia lógica de objetos de fabricación (obtenidos de una lista de inventario). El método, sin embargo, no provee criterios precisos sobre cómo definir las dependencias físicas y lógicas (presentan únicamente un ejemplo). Además, el método no incluye ninguna guía para especificar a qué objetos/entidades de fabricación aplicar el proceso de agregación. El proceso de agentificación por tanto es bastante subjetivo e intuitivo.

Colombo y otros [Colombo *et al.*, 2002] han extendido el enfoque de modelado basado en redes de Petri [Feldmann y Colombo, 1998] para el modelado de sistemas de control de producción basado en agentes. En su metodología, los agentes se identifican sobre la base de un modelo de red de Petri de los procesos de fabricación y las decisiones de control relevantes. A pesar de que este método ofrece un modelo más riguroso para la etapa de agentificación que el método anterior, no presenta ningún criterio definido para la identificación de los agentes en el modelo de la red de Petri (la única guía es la siguiente, "los agentes son motores de decisión y por tanto son responsables del control de decisiones").

Bussmann y otros [Bussmann *et al.*, 2004] proponen la metodología DACS (Metodología para el Diseño de Sistemas de Control de Producción Basados en Agentes). Esta metodología comienza con la especificación del problema de control de producción, que, entre otros aspectos, incluye una especificación de los componentes de producción a ser controlados y su comportamiento físico. La primera fase, *análisis de toma de decisiones*, analiza las decisiones

necesarias para operar el sistema de producción observando las tareas de decisión locales que aparecen en cada componente de producción e identificando cualquier dependencia de decisiones que pueda existir entre las tareas de decisión locales. La segunda fase de la metodología es, la *identificación de los agentes*, se desarrolla en base al modelo de decisión derivado en la primera fase y agrupa las tareas de decisión con el objetivo de asignar cada grupo a un agente. Para el proceso de agrupamiento, el método proporciona algunas reglas que guían el proceso. De esta manera esta fase ofrece un conjunto de operaciones que modifican la red de decisiones, ya sea separando las tareas de decisión o introduciendo nuevas tareas. Estas operaciones permiten al diseñador re-organizar el modelo de decisión y de esta manera mejorar el proceso de agrupamiento.Finalmente, la tercera fase de la metodología, *selección de los protocolos de interacción*, provee un mecanismo para re-utilizar protocolos de interacción existentes con el objetivo de resolver cualquier dependencia de decisiones entre diferentes agentes. Una vez que las dependencias hayan sido clasificadas se deciden los protocolos de interacción que serán utilizados por los agentes. En resumen, la aplicación sucesiva de las tres fases resulta en un diseño basado en agentes para resolver un problema de control de producción. El diseño resultante consiste de una lista de agentes con sus responsabilidades de decisión (en términos de un conjunto tareas de decisión), y de un conjunto de protocolos de interacción para cada dependencia entre diferentes agentes. El método propone como paso de implementación el desarrollo independiente de cada agente, sin embargo no precisa cómo se realiza este paso, ni tampoco que arquitectura ni plataforma de agente pueden ser utilizadas. El método se centra únicamente en el control de procesos de producción sin considerar los demás componentes de un sistema de fabricación.

2.2.4. Modelado de Empresas

En esta sección presentamos trabajos relacionados con notación y métodos para el modelado de empresas. Un Sistema de Fabricación se enmarca dentro de una empresa de fabricación por tanto el modelado del sistema tendrá que integrar el modelado de la empresa.

En el modelado de empresas se diferencian dos niveles [Vernadat, 1996]:

(i) nivel de negocio, secuencias parcialmente ordenadas de actividades de empresas activadas por la ocurrencia de eventos que producen resultados finales identificables; (ii) nivel de actividades de empresa, conjunto de secuencias parcialmente ordenadas de operaciones básicas ejecutadas por recursos activos de la empresa. Los distintos trabajos reportados en la literatura especialidad abordan ambos o alguno de los dos niveles en distintos niveles de detalle.

CIMOSA [ESPRIT, 1993] es el resultado del proyecto AMICE uno de los mayores proyectos de la Comunidad Europea en el dominio de Manufactura e Ingeniería Integrada por Computador (CIME). Algunos de sus resultados han sido utilizados como estándares para el dominio de modelado de empresas. CIMOSA permite la definición de los procesos de negocio en el ciclo de vida del sistema (entorno de ingeniería de la empresa) y en el ciclo de vida del producto (entorno de ingeniería del producto). Clasifica los modelos y elementos de modelado en tres categorías: (i) bloques genéricos de construcción, elementos básicos de modelado para cualquier sistema de fabricación; (ii) modelos parciales, instancias de bloques genéricos o agregados de instancias de bloques genéricos; y (iii) modelos particulares, construcciones totalmente instanciadas que representan un sistema de fabricación o una empresa particulares. Define cuatro vistas para el modelado: (i) vista *funcional*, descripción de funcionalidades y tareas de la empresa; (ii) vista de *información*, modelado de la información utilizada en los procesos de la empresa; (iii) vista de *recurso*, capacidades y recursos necesarios para ejecutar los procesos de la empresa; (iv) vista de *organización*, define la autoridad y las responsabilidades de las entidades de la organización. La desventaja de CIMOSA para el modelado de sistemas de fabricación de la nueva generación, radica, lógicamente, en su orientación CIM. Sin embargo es un buen referente para el modelado de empresas debido a los elementos conceptuales que define, de hecho la mayoría de notaciones se basan o incluyen conceptos definidos en CIMOSA.

UEML (Unified Enterprize Modelling Language) [UEML, 2003] surge a partir de un proyecto financiado por la Comunidad Europea para la unificación de las distintas notaciones de modelado de empresas. Esta notación integra entre otras las siguientes notaciones ampliamente conocidas: IDEF0, IDEF1x e IDEF3 [IDEF, 2004], GRAI net [Doumeingts, 1998], CIMOSA [ESPRIT,

1993], IEM [Spur *et al.*, 1996], ARIS method [ARIS, 2004], etc. UEML 1.0 define el conjunto básico de conceptos y elementos de modelado. Se basa en la identificación de conceptos comunes y no-comunes de las distintas notaciones de modelado de empresas. Incluye los siguientes elementos: actividad, parte de un comportamiento de una empresa que produce salidas a partir de entradas; flujo, representa el flujo de un objeto desde un origen hacia un destino; ancla, es el origen o el destino de un flujo; recurso material o recurso humano, es un tipo especial de objeto necesario para la ejecución de una actividad; objeto de información, un objeto que se puede anexar a un flujo. Además, define los atributos, las asociaciones y restricciones de cada uno de estos elementos. UEML se utiliza en la especificación del nivel de negocio sin abordar la especificación interna de las actividades de la empresa.

2.2.5. Resumen Comparativo

En las secciones anteriores se puede constatar que en el área SMA existe un gran esfuerzo investigador para el desarrollo de metodologías multi agente, prueba de ello es el gran número de trabajos reportados. La mayoría de los enfoques SMA son de propósito general, mientras que sólo un número reducido de métodos SMA se definen como específicos para el dominio de sistemas de fabricación. La tecnología SMA se ha definido como un enfoque prometedor para la "nueva fabricación" debido a sus características de distribución, comportamiento autónomo, capacidad de cooperación, entre otros [Bussmann *et al.*, 2004]. De hecho la mayoría de los desarrollos del área HMS utilizan la tecnología SMA para implementar sus sistemas (Sección 2.1).

Los enfoques SMA específicos para sistemas de fabricación que hemos encontrado en la literatura especializada tienen algunas limitaciones. La propuesta de [Kendall *et al.*, 1996] no utiliza una notación ni una tecnología apropiada para el modelado de agentes. Se basa exclusivamente en conceptos OO y el método IDEF, perdiendo de esta manera expresividad y rigor al modelar agentes. Además IDEF0 es insuficiente para el modelado de todos los aspectos de un sistema de fabricación [Bussmann *et al.*, 2004]. Por otra parte las propuestas de [Ritter *et al.*, 2002] y de [Colombo *et al.*, 2002] se ocupan únicamente

de una pequeña fase dentro del proceso de desarrollo de sistemas de fabricación. Se centran en el control de procesos de producción y dentro de este en la identificación de agentes que controlarán el proceso. Más aun, ambas propuestas son poco rigurosas, ya que se limitan a presentar ejemplos sin proveer una definición general aplicable a un problema de control de producción cualquiera. DACS [Bussmann *et al.*, 2004] es un método específico para el diseño de sistemas basados en agentes para el control de sistemas de producción. Por tanto se ocupa únicamente de este componente del sistema de fabricación. La salida de este método es un diseño basado en agentes, de tal manera que no ofrece ningún soporte guía ni de herramienta para la implementación del sistema basado en agentes que diseña. Sin embargo, es un método riguroso y fácil de utilizar incluso para diseñadores sin conocimientos previos de la tecnología agente.

En el área HMS se reconoce la necesidad de metodologías específicas, robustas y completas para guiar el desarrollo de estos sistemas, y lograr así uniformidad de notación y conceptos [HMS, 1994]. Se han reportado dos trabajos que intentan cumplir con este objetivo, sin embargo ambos son muy preliminares y por tanto carecen de los elementos clave de un método de ingeniería del software: definición del proceso de desarrollo, notación y herramientas.

Finalmente en el área de modelado de empresas se centran principalmente en la definición de los procesos de negocio sin atacar el modelado de las actividades internas de la empresa. Tampoco se ocupan de desarrollar herramientas para contemplar los requisitos de la "nueva fabricación". No obstante es muy importante el esfuerzo de estandarización de notación promovido por UEML [UEML, 2003].

Para terminar con un análisis comparativo centrado en los requisitos de modelado de los sistemas holónicos de fabricación presentamos la Tabla 2.1. En ella incluimos todas las metodologías presentadas en las secciones anteriores, indicamos por cada metodología el grado en el que ésta cumple con los distintos requisitos de modelado de HMS.

¿Cómo podemos explicar el resultado de la Tabla 2.1?:

- Es evidente que el **requisito 1** lo cumple toda metodología basada en

Tabla 2.1: Métodos de desarrollo y Requisitos de modelado de Sistemas Holónicos de Fabricación.

Método	**Requisitos de modelado**							
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Método de Leitão y Restivo	√		√	√	√			
Método de Fischer y otros	√		√	√	√			√
Método de Kinny y Georgeff	√		√	√				
Método de Burmeister	√		√	√				
MAS-CommonKADS	√		√	√				
GAIA	√		√	√				
GAIA v.2	√		√	√				
ROADMAP	√		√	√				
MASSIVE	√		√	√				
Tropos	√		√	√				
MaSE	√		√	√				
MESSAGE	√		√	√	∼			
RT-MESSAGE	√	√	√	√	∼			
INGENIAS	√		√	√	∼			
MASB	√		√	√				
Prometheus	√		√	√				
Método de Elammari y Lalonde	√		√	√				
Método de Miles	√		√	√				
Cassiopeia	√		√	√				
Agent Societies	√		√	√				
Civil Agent Societies	√		√	√				
Método de Kendall y otros	√		√	√	∼			
Método de Ritter y otros	√		√	√	∼			
Método de Colombo y otros	√		√	√	∼			
DACS	√		√	√	∼			
CIMOSA						√		
UEML						√		

la tecnología SMA. Es lógico, en cambio que los enfoques de Modelado de Empresas no lo consideren debido a su orientación de base.

- El **requisito 2** se refiere a características de tiempo real necesarias para el modelado de sistemas de control en fábricas. RT-MESSAGE es el único método SMA que lo considera. Sorprende que los métodos de Kendall, Ritter, Colombo y el método DACS no lo tengan en cuenta siendo que éstos métodos se centran en el control de sistemas de producción donde este es un requisito fundamental.

- El **requisito 3** es un requisito fundamental de cualquier método de ingeniería del software, por tanto todos los métodos que reconocen la etapa de implementación los soportan. CIMOSA y UEML se limitan al modelado de los procesos de negocio de la empresa sin bajar al nivel de modelado e implementación de las tareas dentro de la empresa, es por ello que no lo soportan.

- El razonamiento para el **requisito 4** es similar al del requisito 3.

- El **requisito 5** se refiere a los mecanismos o procedimientos definidos por una metodología para traducir las entidades del dominio de fabricación en entidades con comportamiento autónomo. Los métodos del área HMS soportan correctamente este requisito, ya que, se basan en arquitecturas de holones que tienen una correspondencia directa con entidades del dominio de sistemas de fabricación. Los métodos MESSAGE, RT-MESSAGE e INGENIAS definen unas guías para el diseñador para ayudar en la "agentificación" de entidades del dominio. No cumplen completamente con este requisito ya que las guías son generales (como es lógico debido a su orientación general) y por tanto no abordan de manera específica las características de las entidades y funcionalidades de fabricación. Los métodos SMA para sistemas de fabricación, como era de esperarse, consideran de manera específica a las entidades del dominio de sistemas de control de producción. Sin embargo, no tienen en cuenta los demás componentes del sistema de fabricación.

- Sólo los enfoques de Modelado de empresas consideran el **requisito 6**.

Los métodos del área HMS sorprendentemente no lo tienen en cuenta. En el siguiente capítulo estudiamos en profundidad este requisito y presentamos un resumen de arquitecturas de agentes relacionadas con niveles de abstracción.

- El **requisito 7** se refiere a métodos de desarrollo mixtos (bottom-up y top-down). Todos los métodos analizados siguen uno u otro enfoque sin combinarlos.

- Finalmente el **requisito 8** sólo se considera de manera explícita y específica en el método de Fischer. Es verdad, en cambio, que con algunos métodos SMA se pueden modelar sistemas de gran escala que pueden representar todo un sistema de fabricación, pero las guías definidas en ellos no son específicas y por tanto pueden llevar a análisis y diseños en los que el esfuerzo y la atención del diseñador se desvíen en elementos poco importantes para estos dominios [Bussmann *et al.*, 2004].

2.3. Conclusiones

En este capítulo hemos analizado de manera extensa el estado del arte del área HMS. Hemos tratado de presentar una visión global del área, con el resumen de los trabajos reportados en dos grandes grupos: desarrollos de sistemas holónicos de control y modelado de sistemas holónicos de fabricación.

El mayor esfuerzo investigador del área HMS se ha centrado en desarrollos de sistemas holónicos de control. En este grupo de trabajo encontramos dos sub-grupos: arquitecturas para control holónico y algoritmos para control holónico. Al analizar los trabajos sobre arquitecturas podemos resumir que aunque en un principio distintos grupos definieron sus propias arquitecturas, siempre basadas todas en la arquitectura general de holón de Christinsen [Christensen, 1994], recientemente los trabajos tienden a converger hacia una idea unificada de arquitectura. Esta arquitectura unificada integraría los elementos: dominio de cooperación (o cluster virtual) [Deen, 1993] y [Maturana y Norrie, 1997], bloques funcionales [Fletcher *et al.*, 2000], la arquitectura de referencia PROSA [Van Brussel *et al.*, 1999] y la tecnología agente como herramienta de implementación. Christensen propone [Christensen, 2003]: para el control

de bajo nivel del holón bloques funcionales, estándar IEC 61499 [IEC, 2000; 2001]; y para el control de alto nivel agentes, estándar FIPA [FIPA, 2005]. En cuanto a los algoritmos para control holónico en el área HMS se han reportado trabajos sobre: planificación (programación) de órdenes de trabajo, scheduling (secuenciación) del plan, ejecución (lanzamiento) de las órdenes, control de máquinas para la ejecución de la orden, y control del dispositivo para la operación efectiva. Con el objetivo de proveer flexibilidad e inter-operabilidad entre estos niveles. En general, debido a la estructura de los sistemas holónicos, las soluciones de control y planificación holónicas se distribuyen en términos de toma de decisión y la coherencia entre estas decisiones locales se logra mediante interacciones cooperativas entre holones. La tecnología utilizada para la implementación de la mayoría de estos algoritmos es la tecnología SMA.

El modelado de sistemas holónicos de fabricación constituye el interés fundamental de esta tesis. Es por ello que en este apartado hemos puesto especial interés. La escasez de trabajos en el área HMS es evidente, una prueba de ello son los pocos trabajos específicos para el dominio HMS reportados en la literatura especializada. Más aun, una prueba de la inmadurez de esta área, es el estado preliminar de los dos trabajos sobre metodologías para HMS [Fischer *et al.*, 2003; Fischer, 1998] y [Leitao y Restivo, 2003a]. Estos hechos nos motivaron a estudiar metodologías para el desarrollo de sistemas holónicos de fabricación y a compararlas para obtener una medida cualitativa de la conveniencia de cada una para sistemas HMS. Para realizar esta comparación hemos definido ocho requisitos de modelado de sistemas holónicos de fabricación. Estos requisitos los hemos dividido en dos grupos: requisitos funcionales y requisitos de ingeniería del software. Los primeros se refieren al tipo de programas que se deben implementar como resultado de la aplicación de una metodología. Mientras que los segundos se refieren a las propiedades del método de ingeniería del software. Todos los requisitos son específicos para sistemas holónicos de fabricación. Era evidente que la comparación no la podíamos hacer sólo de los métodos del área HMS, ya que *apriori* conocíamos sus limitaciones. Por tanto debíamos buscar métodos maduros, robustos y con alguna evidencia de aplicación. Los métodos SMA parecían apropiados por varios motivos: existen

varios trabajos reportados, muchos de ellos son de propósito general, un grupo reducido de ellos son específicos del dominio de fabricación, la tecnología agente es la herramienta de implementación por excelencia del área HMS, y en nuestro grupo de investigación contamos con experiencia en el área de metodologías SMA. Finalmente el área de Modelado de Empresas ofrece trabajos interesantes y propuestas de estandarización de notación para el modelado de los procesos de negocio de una empresa. Estos trabajos pueden ser aplicados (y de hecho han sido aplicados) como notación de modelado para empresas de fabricación. Con estos tres grupos: métodos HMS, métodos SMA y Modelado de Empresas, realizamos la comparación basados en los ocho requisitos de modelado. El resultado es la Tabla 2.1. Al analizar esta tabla resulta evidente la necesidad de desarrollar una metodología para sistemas holónicos de fabricación. Este es el objetivo principal de esta tesis, y los resultados los presentamos en el Capítulo 5. La hipótesis de este estudio es que una Metodología SMA es la más apropiada para el desarrollo de sistemas holónicos de fabricación. Para estudiar esta hipótesis en el Capítulo 3 estudiamos la relación entre los dos entidades de modelado y en el Capítulo 4 profundizamos sobre la visión estructural de un agente, resultado del estudio del Capítulo 3.

Capítulo 3

Holones o Agentes

En el capítulo anterior hemos presentado el estado del arte del área de sistemas holónicos de fabricación. De este estudio podemos concluir que como no existen métodos convencionales para implementar sistemas holónicos, la tecnología de Sistemas Multi Agente (SMA) es la más utilizada por la mayoría de los investigadores del HMS. Sin embargo creemos que es necesario realizar un análisis de las características de estos dos enfoques para clarificar la relación que existe entre ellos. En este capítulo, presentamos un estudio comparativo de los sistemas holónicos (HMS) y los SMA. Siguiendo la tendencia generalizada en el área de los sistemas de fabricación inteligente, indicamos en primer término la motivación de ambos enfoques; luego identificamos aquellas propiedades que serán contrastadas para hacer un análisis de cada una, y; presentamos una tabla con un resumen de la comparación. Además, presentamos un análisis de trabajos relacionados con la recursión como característica de agentes. El objetivo principal de este capítulo es averiguar si los holones y los agentes son diferentes o no, o si uno es un caso especial del otro.

3.1. Agente

Un SMA se compone de dos o mas agentes relacionados. Un agente es un sistema computacional autónomo y flexible, que es capaz de actuar en un entorno [Wooldridge y Jennings, 1995]. Flexible significa, que el agente es:

- *Reactivo*, reacciona al entorno en el cual se encuentra.

- *Pro-activo*, es capaz de cumplir su propia agenda (planes u objetivos).
- *Social*, es capaz de comunicarse con otros agentes a través de algún lenguaje.

Algunas propiedades que son atribuidas usualmente a los agentes en mayor o menor grado para resolver problemas particulares son [Nwana, 1996] [Franklin y Graesser, 1996]:

- Autonomía: los agentes pueden operar sin la directa intervención de humanos u otros agentes.
- Habilidad Social: los agentes son capaces de interactuar con otros agentes (humanos o no) a través de un lenguaje de comunicación de agentes.
- Racionalidad: un agente puede razonar acerca de datos percibidos a fin de calcular una solución óptima.
- Reactividad: los agentes son capaces de percibir estímulos del entorno y estos estímulos guían las acciones del agente en su entorno.
- Pro-actividad: los agentes no son sólo entidades que reaccionan a estímulos, sino también tienen un carácter emprendedor y pueden actuar guiados por sus propios objetivos.
- Adaptabilidad: esta característica está relacionada con el aprendizaje que un agente puede lograr y con su capacidad para cambiar su propio comportamiento basado en este aprendizaje.
- Movilidad: es la capacidad de un agente para moverse a través de una red.
- Veracidad: un agente no puede comunicar información falsa de manera deliberada.
- Benevolencia: un agente está dispuesto a ayudar a otros agentes si esto no está en contra de sus propios objetivos.

El estudio de Sistemas Multi Agentes se inició hace cerca de 20 años, en el ámbito de la Inteligencia Artificial Distribuida (Distributed Artificial Intelligence - DAI). La DAI es un sub-campo de investigación de la Inteligencia Artificial (AI). La DAI estudia el comportamiento inteligente de grupo que se deriva a partir de la cooperación de entidades llamadas agentes. Estudia cómo un grupo de módulos cooperan para dividir y compartir el conocimiento del problema y cómo se desarrolla la solución. La DAI se centra en el comportamiento global, con un comportamiento prefijado de los agentes. Se interesa por las técnicas y el conocimiento necesarios para la coordinación y distribución del conocimiento y las acciones en un entorno multiagente.

El SMA estudia la coordinación del comportamiento inteligente entre un grupo de agentes inteligentes autónomos (posiblemente pre-existentes). Se centra en el comportamiento individual a partir del cual se deriva el comportamiento del sistema. Hoy en día los SMAs forman un área de investigación muy activa y se los está empezando a utilizar en aplicaciones comerciales e industriales (para un estudio detallado consultar [Ana MAS, 2004]). Los SMAs se centran en el comportamiento social de entidades inteligentes y se ocupan principalmente de estudiar modelos de comportamiento, estrategias de cooperación y coordinación, optimización del desempeño de tareas, aprendizaje a partir de experiencias propias, formación de coaliciones, etc.

En DAI el problema a ser resuelto se formula de manera centralizada, y luego se distribuye a nodos de computación locales. En SMA, en cambio, el problema se origina y se resuelve en el nodo local y la solución global resultante es emergente.

En resumen, los SMAs son una tecnología software general motivada en cuestiones fundamentales de investigación acerca de autonomía, cooperación, formación de grupo, etc. Se ocupa de responder preguntas tales como: "¿Qué puede ser hecho?", y "¿Cómo puede ser realizado?", y se aplica a una gran variedad de dominios: comercio electrónico, control inteligente de producción, robótica, recuperación de la información, etc.

La aplicación de la tecnología SMA al dominio de fabricación ha ido creciendo paulatinamente a lo largo de los últimos 10 años. Se ha centrado en

todas las áreas de la empresa de fabricación desde el diseño del producto hasta el control de tiempo real. No es un campo de investigación con una visión única. Es mas bien una colección de trabajos que propone mejorar el control existente en la infraestructura de las industrias con la ayuda de técnicas orientadas a agentes. Muchos trabajos, por ejemplo, se centran sobre el control de un sistema de producción flexible, como fue desarrollado dentro de la iniciativa CIM.

Una de las primeras aplicaciones de la tecnología agente en el dominio de fabricación fue el sistema de control de fábricas YAMS (Yet Another Manufacturing System) [Parunak, 1987]. Este sistema utilizaba el protocolo ContratNet [Smith, 1980] en la negociación inter-agente para asignar de manera flexible recursos a tareas de producción en un entorno cambiante. Varias aplicaciones de la tecnología de agente siguieron apareciendo pero siempre centradas en un área específica dentro del sistema de fabricación:

- *Diseño de Productos*: la investigación en esta área se ha centrado en permitir que equipos de diseñadores distribuidos en lugares distantes puedan trabajar juntos para diseñar productos complejos utilizando una variedad de herramientas de análisis y diseño. Ejemplos de estos son: PACT [Cutkosky *et al.*, 1993], RAPPID [Parunak *et al.*, 1997].

- *Integración de empresas y gestión de la cadena de suministro*: el interés de esta área se centra en gestionar una red mundial de proveedores, fábricas, almacenes y distribuidores para asegurar el requisito fundamental de todo sistema de fabricación: adquisición adecuada de materia prima y transporte y distribución eficientes. Algunos trabajos reportados son: ISCM [Fox *et al.*, 1993], AIMS [Park *et al.*, 1993], CIIMPLEX [Peng *et al.*, 1998].

- *Planificación y Scheduling*: se centra en aplicar la tecnología agente a los procesos de selección y secuenciación de las actividades de fabricación (planificación) y a la asignación de recursos y tiempos para estas actividades (scheduling). Ejemplos de aplicación: AARIA [Baker *et al.*, 1999], MASCADA [Bruckner *et al.*, 1998], MetaMorph [Maturana y Norrie, 1996], YAMS [Parunak, 1987].

- *Control de tiempo real*: la investigación en esta área se ha centrado en la aplicación de la tecnología de agente al control de equipos distribuidos (máquinas de control numérico, robots) que deben de reaccionar en tiempo real [Brennan *et al.*, 1997; Wang *et al.*, 2001; Zhang *et al.*, 2000].

Para un estudio detallado de las aplicaciones desarrolladas en el área de fabricación basada en agentes consultar [Shen y Norrie, 1999; Botti y Giret, 2002] y para la aplicación de SMA al diseño y fabricación concurrentes ver [Shen *et al.*, 2001].

Muchos de estos trabajos se han centrado en desarrollar soluciones novedosas para la parte de procesamiento de la información de una infraestructura existente, como sistemas de producción flexibles. Sin embargo, el mayor potencial para conceptos como SMA no se encuentra solo dentro del procesamiento de la información, sino en un enfoque completamente nuevo para la producción en general, tal como se lleva a cabo en HMS (en el Capítulo 5 presentamos mayor detalle).

3.2. Holón

En el Capítulo 1 hemos presentado los conceptos fundamentales del enfoque holónico. A partir de ello podemos resumir que los holones, como paradigma, tienen las siguientes características básicas: autonomía, cooperación y reorganización. Además de estas características, que las podemos llamar "propiedades de comportamiento", los holones tienen "características estructurales". Una de ellas es la "recursividad", la cual permite a los holones estar formados internamente por entidades auto-similares (holones), que a su vez pueden estar formadas por holones, y así sucesivamente (hasta que se llega a un nivel atómico en el cual nuevas sub-divisiones son imposibles o de poca utilidad para el dominio de aplicación). Los holones pueden estar compuestos por holones los cuales 1) pueden ser holarquías, 2) pueden participar en varias holarquías simultáneamente, y 3) pueden llegar, salir, o cambiar - es decir, holones que forman holarquías dinámicas. Otra propiedad estructural importante, como ha sido definida por el consorcio HMS [HMS, 1994], es que los holones usualmente poseen una parte de procesamiento de información con una parte opcional

de procesamiento físico. Los desarrollos del área de Sistemas de Fabricación Holónica se pueden consultar en el Capítulo 2.

3.3. Comparativa

En esta sección presentamos un estudio comparativo de ambos enfoques. Es importante destacar que el estudio lo centramos en el enfoque agente como paradigma general aplicado a distintos dominios además del dominio de fabricación y el enfoque holónico para sistemas de fabricación inteligentes. En consecuencia las características del enfoque holónico que vamos a considerar están ligadas al HMS y por tanto no analizaremos algunas característica que podrían aparecer asociadas con el concepto filosófico más general desarrollado por Koestler [Koestler, 1971]. En primer lugar presentamos la motivación y el origen de cada enfoque, luego analizamos cada propiedad de forma separada.

3.3.1. Motivación y origen de cada enfoque

Como ha sido señalado por Bussmann [Bussmann, 1998], Brennan y Norrie [Brennan y Norrie, 2001] entre otros, ambos enfoques difieren principalmente en la motivación. La investigación del HMS está motivada en las tareas flexibles de fabricación. Por consiguiente, está orientada hacia los estándares de comunicación de bajo nivel y el comportamiento de bajo nivel. Por otra parte, la investigación en el área de los SMA está motivada en la programación de sistemas inteligentes distribuidos. Se centra en el comportamiento social de entidades inteligentes y se ocupa principalmente de la investigación de modelos de comportamiento, estrategias de cooperación y coordinación, optimización del desempeño de tareas, aprendizaje a partir de las propias experiencias, formación de coaliciones, etc. En resumen, a diferencia de SMA, que es un enfoque software amplio que puede ser utilizado además para el control inteligente distribuido, HMS es, por definición, un enfoque específico de fabricación para sistemas de control distribuidos.

3.3.2. Características

Las características que analizamos a continuación se refieren a las propiedades de agente y de holón que hemos resumido en las Secciones 3.1 y 3.2.

3.3.2.1. Autonomía

El comportamiento de las entidades autónomas puede estar basado tanto en sus experiencias propias así como en el conocimiento interno utilizado en la representación de las entidades del entorno particular en el cual actúa. Los agentes han sido utilizados con éxito en dominios en los cuales el grado de incertidumbre e imprevisibilidad requiere unidades de procesamiento capaces de acción autónoma, sin la directa intervención de humanos u otros agentes.

Los sistemas de control de fabricación son sistemas grandes y complejos que están diseñados para llevar a cabo una tarea claramente definida. Sin embargo, en un sistema de fabricación, raramente las cosas ocurren como se esperaban. Se puede necesitar que el sistema realice tareas adicionales que no fueron anticipadas y algunas veces se puede permitir que el sistema omita ciertas tareas. Los recursos disponibles para desempeñar tareas pueden no estar disponibles, y se pueden introducir recursos adicionales. El tiempo de inicio y el tiempo de procesamiento de una tarea también son objetos de variaciones. Una tarea puede tomar más o menos tiempo que el anticipado, y las tareas pueden llegar antes o después de lo previsto a la línea de fabricación. Estas son algunas de las razones por las que los holones son, por definición, entidades autónomas las cuales deben ser capaces de crear, controlar y monitorizar la ejecución de sus propios planes y/o estrategias, y de tomar acciones correctivas en contra de su propio malfuncionamiento [HMS, 1994]. En este sentido, podemos decir que los agentes y los holones comparten esta característica.

3.3.2.2. Reactividad

Un agente responde a eventos que ocurren en su entorno, estos eventos afectan tanto los objetivos del agente o los supuestos que soportan los procedimientos que el agente está ejecutando para lograr sus objetivos. Así, los efectos de los estímulos del entorno pueden ser cambios en los objetivos o supuestos del agente, o en las acciones del agente. De la misma manera, los

holones necesitan reaccionar a cambios en sus entornos. Tales cambios afectan sus objetivos internos o pueden impedir la ejecución de tareas planeadas actuales o futuras.

Supongamos un sistema de fabricación simplificado (ver el área gris en la sección de procesamiento de fabricación en la figura 3.1). Existen 3 tipos de holones: (i) holones orden de trabajo que representan las tareas en el sistema de fabricación; (ii) holones recurso que son recursos de fabricación del sistema, y; (iii) holones producto, los productos en si. Sea HO1 el holón orden de trabajo 1 que está procesando una tarea para producir HP1 (holón producto 1). El proceso de fabricación se lleva a cabo en etapas de procesamiento asignados a holones de recurso HR1, HR2 y HR3, tal que HR1 es la primera etapa de procesamiento, HR2 es la segunda y HR3 la tercera. Por alguna razón, HR1 se toma más tiempo de procesamiento que el tiempo inicial estimado. HR2 y HR3 tienen que darse cuenta de esta situación que impide la ejecución de sus planes actuales y reaccionar de alguna manera, por ejemplo, buscando otros holones orden de trabajo para aprovechar su potencial, o detenerse hasta que HR1 finalice su procesamiento (obviamente, la segunda alternativa es menos productiva para el rendimiento de todo el sistema, pero también es una acción reactiva). A pesar de que este ejemplo es muy simplificado y reducido, en él se refleja la propiedad reactiva de los holones.

Existen algunos problemas asociados con holones que están a su vez constituidos por holones. Uno de estos problemas es: ¿Quién realmente actúa y/o percibe en una holarquía?. Un enfoque es tener por cada holarquía una cabeza que sea responsable de la interacción con el exterior (interacción a su nivel). Y para la interacción interna, permitir a un holón interactuar con los demás holones de su holarquía. Este enfoque ha sido adoptado, entre otros por [Maturana y Norrie, 1997], [Fletcher *et al.*, 2000] y [Van Brussel *et al.*, 1998]. Un segundo enfoque es permitir a todo holón de una holarquía interactuar con cualquier holón, sin importar si los otros holones son de la misma holarquía o de otras holarquías. No existe una interfaz externa única, cada holón de la holarquía es una interfaz. El comportamiento de la holarquía emergerá como la composición del comportamiento de sus holones constituyentes. A pesar de

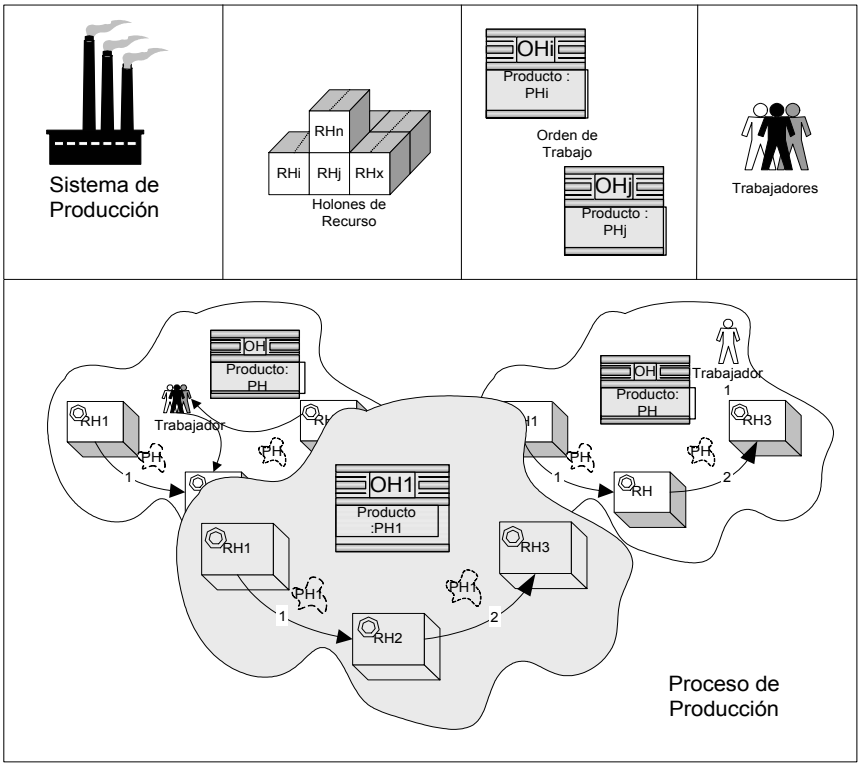


Figura 3.1: Ejemplo de un Sistema de Fabricación simplificado.

que esta es la manera en que los sistemas biológicos están organizados, este es un enfoque muy complicado y difícil de implementar para sistemas de fabricación en los que se requiere cierto grado de certidumbre (de hecho en la literatura especializada no hemos encontrado trabajos que implementen esta estrategia). Surgen algunas cuestiones asociadas con este enfoque: ¿Cuáles son los límites reales de una holarquía/holón?, ¿Cuál es la identidad de una holarquía/holón?, etc.

3.3.2.3. Pro-actividad

Los agentes no actúan simplemente en respuesta a su entorno, son capaces además de exhibir comportamiento dirigido por objetivos tomando la iniciativa [Wooldridge y Jennings, 1995]. No es difícil construir un sistema dirigido por objetivos, como lo han señalado Wooldridge y Jennings. Cuando uno escribe un procedimiento de este tipo, uno lo describe en términos de sus pre y post-condiciones. Los efectos del procedimiento son sus objetivos. Si las precondiciones se satisfacen cuando se invoca al procedimiento, luego se espera que el procedimiento se ejecute correctamente: el objetivo será alcanzado. Pero los sistemas de este tipo asumen que el entorno no cambia mientras se ejecuta el procedimiento. De igual manera, se asume que el objetivo, es decir la razón para ejecutar el procedimiento, permanece válido al menos hasta que el procedimiento haya terminado.

Como lo hemos señalado en capítulos anteriores los entornos donde se ejecutan agentes y holones cambian constantemente, impidiendo que éstos puedan ejecutar un procedimiento a "ciegas" sin preocuparse si aún son válidos los supuestos que soportan la ejecución del mismo. Esto implica que los agentes y holones necesitan un balance entre el comportamiento dirigido por objetivos (pro-activo) y el comportamiento reactivo. Los agentes y holones intentarán alcanzar sus objetivos de manera sistemática, pero además serán capaces de reaccionar a tiempo ante nuevas situaciones para que la reacción sea de alguna utilidad. Supongamos en el ejemplo anterior, que HR1, HR2 y HR3 tienen como objetivo: "maximizar su utilización". Antes que HR1 empieza a ralentizar su procesamiento, HR2 y HR3 tienen huecos de tiempo asignados a HP1, y huecos de tiempo asignados a otros productos para lograr una mayor utilización de su tiempo y potencia de procesamiento, es decir para lograr su objetivo. Cuando HR2 y HR3 descubren que HR1 se está tomando más tiempo del esperado, HR2 y HR3 deben re-asignar el hueco de tiempo asignado a HP1 así como los huecos de tiempo y potencia computacional asignados a los demás productos encontrados en su intento por alcanzar su objetivo. Este ejemplo simple refleja la propiedad pro-activa de los holones.

3.3.2.4. Habilidad Social

Ya que los agentes no actúan usualmente aislados, sino en presencia de otros agentes o humanos, necesitan de habilidades sociales y comportamiento interactivo para comunicarse, cooperar, coordinarse y negociar con ellos. Los agentes son capaces de interactuar y especialmente de comunicarse con otros agentes, gracias a lenguajes de comunicación de agentes (*Agent Communication Language* - ACL). Pero, ¿qué es un ACL?. Un ACL, como ha sido propuesto por Austin [Austin, 1962], "... es un medio a través del cual se comunican las actitudes con respecto al contenido del intercambio entre los agentes; sugiere cuándo el contenido de una comunicación es una aserción, una solicitud, una consulta, etc.". Estándares comunes para ACLs incluyen el Knowledge Query and Manipulation Language - KQML [Finin *et al.*, 1994] (Lenguaje de Manipulación y Consulta del Conocimiento) y una propuesta de FIPA (Foundation for Intelligent Physical Agents), basado en el lenguaje Arcol - ARtimis COmmunication Language [FIPA, 2002]. Estos desarrollos son en parte rivalizados por lenguajes de etiquetado de Internet altamente sofisticados tales como XML [W3C, 2002], el cual también puede ser utilizado por agentes.

La habilidad social de un holón está incluida en su capacidad para cooperar (ver la siguiente característica), ya que los holones necesitan de un medio de comunicación con otros holones para ser capaces de cooperar. En el ejemplo anterior, los tres tipos de holones necesitan de habilidades sociales para intercambiar información acerca del proceso de fabricación, para permitir la ejecución de tareas. Por ejemplo, supongamos que HO1 necesita recursos para producir HP1, entonces HO1 interactúa con los holones de recursos HR1, HR2 y HR3 para obtener funcionalidades de procesamiento y propiedades específicas de la operación, tal como alta calidad o alto rendimiento. Por otra parte, HR1, HR2 y HR3 tratan de maximizar su utilización, mientras que HP1 al mismo tiempo se centra en la siguiente operación para lograr ser procesado por HR1, HR2 y HR3. Como se puede deducir a partir de este simple ejemplo, los holones necesitan interactuar entre sí, y al igual que los agentes, los holones necesitan de habilidad social.

En los sistemas de fabricación, las personas y los computadores necesitan

estar integrados, con acceso al conocimiento e información requeridos, para trabajar de manera colectiva a varios niveles del desarrollo del producto [HMS, 1994]. Estos requisitos llevaron a Christensen a proponer un bloque integrado de Interfaz Humana (ver figura 3.2) en su arquitectura general de holones.

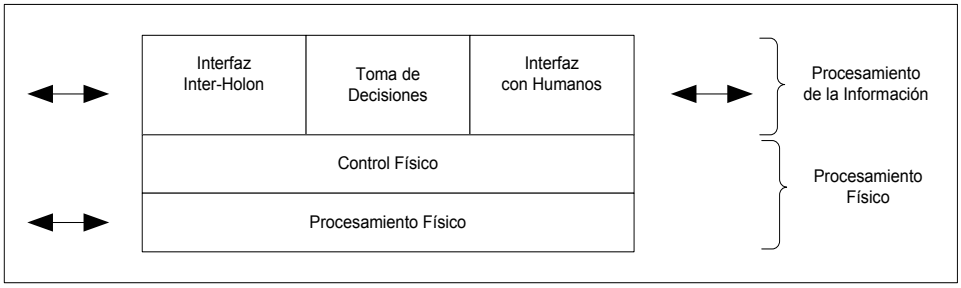


Figura 3.2: Arquitectura General de un Holón.

3.3.2.5. Cooperación

La cooperación es un medio de habilidad social. Las empresas de fabricación deben cooperar con sus proveedores y clientes para conseguir suministros de material, fabricación de partes, comercialización del producto final, etc. Esta cooperación se debería realizar de forma tal que exista respuesta rápida de ambas partes. La cooperación es un requisito imperativo para cualquier modelo funcional para sistemas avanzados de fabricación [HMS, 1994]. Mas aún, todas las unidades de fabricación cooperan para lograr los objetivos globales de fabricación. Con respecto a estos objetivos, un holón nunca rechaza de manera deliberada la cooperación con otro holón. Sólo cuando las acciones solicitadas son imposibles o fuertemente desventajosas para el proceso de fabricación, el holón rechaza sus ejecuciones [Bussmann, 1998]. La coordinación, negociación y otras técnicas de cooperación permiten a los holones interactuar de forma flexible con otros holones en una manera abstracta. Para la cooperación, existe un gran número de enfoques específicos en SMA, ver [OHare y Jennings, 1996] y [Huhns y Singh, 1998] para una apreciación global. Un ejemplo de escenario de cooperación ha sido presentado en la sección anterior.

3.3.2.6. Re-organización

Koestler define una holarquía como una jerarquía de holones la cual funciona (a) como un todo autónomo en supra-ordinación a sus partes, (b) como partes dependientes en sub-ordinación a controles de niveles mayores, (c) en coordinación con su entorno local. Esta definición se deriva en una jerarquía de holones la cual es una mezcla de organización jerárquica y horizontal. El HMS explora ampliamente principios holárquicos para crear colecciones integradas de varios niveles menores de holones. Como lo han señalado Brennan y Norrie [Brennan y Norrie, 2001], la noción de holarquía puede ser implementada utilizando varias enfoques de arquitecturas SMA para federaciones tal como facilitadores, brokers o mediadores.

Los holones son capaces de actuar dentro de múltiples organizaciones (holarquías), las cuales son creadas y modificadas dinámicamente. Los entornos de fabricación reales son altamente dinámicos debido a situaciones diversas y cambiantes [HMS, 1994]: las tasas bancarias cambian por las noches, los materiales no llegan a hora, caída de proveedores de energía, los facilitadores de fabricación fallan, los trabajadores están ausentes, llegan nuevas órdenes de trabajo y se modifican o cancelan órdenes existentes, etc. Tales cambios llevan a desviaciones del plan y asignación de recursos actuales. Es, por tanto, necesario para el sistema en ejecución adaptarse a tales entornos cambiantes. Este es el motivo por el cual las holarquías deben ser organizaciones "abiertas", las cuales deben ser capaces de acomodar la incorporación de nuevos holones, la eliminación de holones existentes, la re-organización de la carga de trabajo, etc. Una holarquía es un sistema abierto en el sentido que los holones pueden entrar o salir de la misma, pero estos holones son siempre "conocidos" (diseñados e implementados para cooperar en la organización y en consecuencia con conocimiento del funcionamiento del sistema). Esta noción de organización abierta no se debe confundir con los sistemas multi agentes abiertos [1] [Esteva *et al.*, 2001; Dignum *et al.*, 2002; Dellarocas y Klein, 1999;

[1] Un sistema multi agente abierto es un SMA que permite la entrada de agentes externos arbitrarios, a diferencia de los sistemas multi agente cerrados en los cuales los tipos de agentes están predefinidos por el diseñador del sistema

Juan *et al.*, 2002]. Aparte de la cooperación, los holones requieren de técnicas para reorganizar el control del los procesos de fabricación. Estas técnicas monitorizan las acciones de otros componentes, es decir, acciones físicas y de comunicación, y analizan el proceso de control. De esta manera, los holones pueden identificar las posibilidades para mejorar e iniciar un proceso de negociación para la organización utilizando técnicas de cooperación estándar. Las implicaciones de la re-organización acordada se distribuyen a los componentes de los demás holones.

3.3.2.7. Racionalidad

Un agente racional es uno que hace lo correcto, es decir una acción que causa que el agente sea el más exitoso [Russell y Norvig, 1995]. Las acciones que un agente ejecuta pueden ser entendidas como sus objetivos. Galliers propone, en [Galliers, 1988], una definición de racionalidad: *...la suposición que un agente actuará para lograr sus objetivos y no actuará de tal manera que sus objetivos no puedan ser alcanzados - al menos en la medida que sus creencias se lo permitan.* Un agente racional actúa de acuerdo a: la secuencia de percepciones, lo que el agente conoce acerca de su entorno, y las acciones que el agente puede realizar. Estos tres elementos determinarán el suceso del agente.

Aunque esta propiedad no aparece explícitamente en la definición de holones, puede ser derivada a partir de una definición más fuerte de autonomía: *un sistema es autónomo en la medida que su comportamiento es determinado por sus propias experiencias* [Russell y Norvig, 1995], y la suposición general que un holón siempre intenta obtener el mejor desempeño global del sistema.

3.3.2.8. Actitudes Mentales

En el área de DAI, ha sido propuesto un número de enfoques para especificar agentes racionales en término de actitudes mentales tales como conocimiento, creencias, deseos, objetivos, acuerdos, e intenciones. Sin embargo, no existe un consenso acerca de precisamente qué combinación de actitudes mentales es más adecuada para caracterizar a los agentes. No obstante, parece ser aceptado, por la mayoría, que las creencias deberían ser tomadas como una de las nociones básicas de la teoría de agentes [Wooldridge y Jennings, 1995].

Como hemos señalado en secciones previas, a pesar de que los procesos de fabricación experimentan varios cambios y perturbaciones, el grado de incertidumbre e impredecibilidad no es comparable a aquella de otros dominios de aplicación de agentes. Como consecuencia, las aplicaciones de fabricación requieren menos deliberación mental y social que las aplicaciones típicas de sistemas multiagente [Bussmann, 1998]. Las unidades de control de fabricación (holones) deben razonar acerca del comportamiento del sistema de fabricación, pero no acerca de sus propias actitudes mentales o de la de otras unidades de control (holones).

3.3.2.9. Aprendizaje

Cuando se diseña un SMA, es generalmente imposible prever todas las situaciones potenciales que un agente puede encontrar y especificar de antemano el comportamiento de un agente de manera óptima. Para superar estos problemas de diseño, los agentes tienen que aprender a partir de y adaptarse a su entorno. Sen y Weiss presentaron, en [Sen y Weiss, 1999], algunas clasificaciones estándar de Aprendizaje Máquina (*Machine Learning* - ML) para los distintos tipos de aprendizaje. Una de ellas es la siguiente:

- De acuerdo con el método o estrategia de aprendizaje: aprendizaje por repetición, aprendizaje a partir de instrucciones o consejos, aprendizaje a partir de ejemplos y por práctica, aprendizaje por analogía, y aprendizaje por descubrimiento.

- De acuerdo con la respuesta que está disponible para la entidad que aprende y que indica el nivel de desempeño alcanzado: aprendizaje supervisado, aprendizaje reforzado, y aprendizaje no supervisado.

Las unidades de control de fabricación (holones) deben ser capaces de adaptarse a cambios del entorno y manejar contextos emergentes. Como lo señalamos en la sección de re-organización, las holarquías pueden reorganizarse para tratar con circunstancias imprevistas. Esta capacidad puede ser mejorada con aprendizaje. Por ejemplo algunos objetos de aprendizaje son [Shen *et al.*, 1998]: combinaciones de recursos de fabricación para un area específica, comportamiento del sistema de fabricación, soporte en favor o en contra de una decisión,

pre-condiciones y post-condiciones para acciones y tareas, tipos de conflictos, heurísticas para resolver conflictos y para negociar, etc.

3.3.2.10. Benevolencia

La propiedad de benevolencia es aquella por la cual el agente coopera con otros agentes cuando y donde sea posible. La benevolencia "ciega" no tiene sentido en el modelado de agentes autónomos para los cuales la cooperación ocurrirá sólo cuando sea considerada ventajosa en término de motivaciones. El agente no puede ocupar todo su tiempo en nuevas cooperaciones con otros agentes, sin tener en cuenta sus acuerdos actuales y motivaciones. Podemos decir que los holones son entidades benevolentes, porque cuando descubren un posible escenario de cooperación ellos cooperan.

3.3.2.11. Movilidad

Los agentes móviles extienden las capacidades de los sistemas distribuidos mediante la movilidad del código. Los agentes móviles son programas que pueden viajar a través de una red de computadoras y se pueden conectar a otros agentes y a plataformas de agentes para desarrollar sus tareas. El paradigma de agentes móviles ofrece un gran número de ventajas. La movilidad y autonomía hacen que las conexiones permanentes sean innecesarias. El uso de agentes es apropiado también en escenarios donde grandes volúmenes de datos deben ser transportados a través de una red mientras que el código de procesamiento es mas bien pequeño. En tales casos, vale la pena considerar mover el código hacia el dato.

Las unidades de control de fabricación están dedicadas al control continuo de componentes de fabricación. La relación $<$ `controlador`, `unidad controlada` $>$ se asigna de antemano y es fija a lo largo de todo el proceso de fabricación. Más aún, toda la información de control necesaria se encuentra en la unidad controlada. Por tanto, los holones raramente necesitarán de movilidad para la ejecución de sus tareas [Bussmann, 1998].

3.3.2.12. Recursión

La condición básica de los sistemas holónicos es que los holones son simultáneamente todo y parte [HMS, 1994]. Esto significa que los holones pueden contener niveles inferiores de holones, que pueden además estar contenidos en otros niveles superiores de holones, resultando en una arquitectura recursiva.

En la definición de agentes nada impide que los agentes tengan una estructura interna compuesta por entidades que a su vez sean agentes. De hecho existen trabajos en el área SMA que utilizan estructuras complejas de agentes [Occello, 2000; Wagner, 2001; Purvis *et al.*, 2002; Ferber y Gutknecht, 1998; Ferber *et al.*, 2004; Parunak y Odell, 2002; Brazier *et al.*, 2002; Peña *et al.*, 2002; 2003; Matsatsinis *et al.*, 2003; Odell *et al.*, 2005; FIPA Modeling TC, 2005] (en la Sección 3.4 presentamos un análisis de estos trabajos).

3.3.2.13. Procesamiento Físico y de la Información

En 1994 Prácticamente todos los investigadores del HMS adoptan la arquitectura general de Christensen (Figura 3.2). En esta arquitectura se puede observar una separación explícita entre la parte de procesamiento físico y la parte de procesamiento de la información. En el área SMA, no existe tal separación explícita. Sin embargo FIPA (Foundation for Intelligent Physical Agents) [FIPA, 2005], aunque reconoció este uso de la tecnología agente desde hace mucho tiempo, no ha sido sino recientemente cuando ha definido la noción de "agente físico" [Ulieru, 2001]. Bussmann, [Bussmann, 1998], propone el uso de SMA como la tecnología que permite el procesamiento de la información en un HMS. Empezando con la visión holónica, los SMAs pueden proveer las técnicas de razonamiento necesarias para desarrollar la arquitectura de procesamiento de la información de un holón y las técnicas de cooperación tal que los holones puedan interactuar con otros holones de la holarquía. Para la parte de procesamiento físico, Fletcher y Deen [Fletcher y Deen, 2001] proponen bloques funcionales para manejar el control de tiempo real para la interacción de bajo nivel proceso/máquina.

3.3.3. Resumen Comparativo

En la Tabla 3.1 presentamos el resumen del análisis comparativo.

3.4. La recursión en Agentes

En esta sección hacemos un análisis de los trabajos reportados en la literatura sobre estructuras complejas de agentes, que estructuralmente pueden estar formados por agentes más simples. El objetivo de este análisis es determinar en qué medida podrían ser utilizadas estas propuestas para representar holones.

Ocello presentó en [Occello, 2000] un enfoque recursivo para construir Sistemas Multi Agentes híbridos. A partir de un conjunto de agentes elementales, Occello propone una definición recursiva de la estructura de un agente y dos funciones recursivas para construir agentes de niveles superiores. Su trabajo se basa en un análisis de las propiedades recursivas en las estructuras de Sistemas Multi Agente, tales como agente y entorno, y dos funciones definidas sobre ellas, interacción y organización. Un agente recursivo es un SMA, esto es, un conjunto de agentes recursivos y objetos recursivos del entorno. La función de interacción permite modelar los actos de comunicación que pueden existir tanto con otros agentes o con el entorno. La función de organización se modela como un conjunto de relaciones entre agentes. Estas relaciones pueden ser de tres tipos: conocimiento, comunicación y subordinación. Sin embargo en el trabajo de Occello no existe ninguna formalización de la función de recurrencia para definir el comportamiento de un nivel de recursión en término de otro nivel.

Wagner, en [Wagner, 2001], presentó un enfoque para el análisis y diseño de sistemas de información en el que una organización es vista como un "agente institucional" que consiste de una serie de "subagentes" que trabajan a nombre de él. Cada subagente tiene ciertos "derechos", acciones que le están permitidas dentro de la organización para el beneficio de ésta; y "obligaciones", acuerdos organizacionales cuyo cumplimiento le ha sido encomendado. La propuesta de Wagner se basa en una arquitectura de agente basada en compromisos, actos de comunicación y eventos. Las relaciones entre las organizaciones se modelan

Tabla 3.1: Holones Vs. Agentes

Propiedad	**Holón**	**Agente**
Autonomía	Si.	Si.
Reactividad	Si.	Si.
Pro-actividad	Si.	Si.
Habilidad Social	Si. La Interfaz Humana es específica de cada holón.	Si. La Interfaz Humana se implementa generalmente por uno o varios agentes especializados.
Cooperación	Si. Los holones nunca rechazan de manera deliberada la cooperación con otro holón.	Si. El agente puede competir y cooperar.
Re-Organización.	Si. Holarquías.	Si. Jerarquías, organización horizontal, heterarquías, etc. Las holarquías se pueden implementar utilizando varias enfoques para federaciones en SMA tales como facilitadores, o mediadores.
Racionalidad	Si.	Si.
Aprendizaje	Si.	Si.
Benevolencia	Si.	Si.
Movilidad	Los holones raramente necesitarán de movilidad para la ejecución de sus tareas.	Si.
Recursión	Si.	No existe una definición formal de agente recursivo, aunque existen trabajos en el área SMA que utilizan estructuras complejas de agentes.
Procesamiento de la Información y Físico	Si. La separación es explícita,aunque la parte de Procesamiento Físico es opcional.	No existe una separación explícita.
Actitudes Mentales	Si. Los holones no necesitan razonar acerca de sus propias actitudes mentales o aquellas de otras unidades de control.	Si.

mediante actos de comunicación. Si bien es cierto que el agente institucional encapsula a la organización, el proceso de desarrollo no explota esta característica ya que no propone un desarrollo guiado por agentes institucionales (desarrollo por capas de abstracción). De esta manera parece que el agente institucional es utilizado simplemente como elemento gráfico para simplificar los diagramas.

En [Purvis *et al.*, 2002] se propone un enfoque de múltiples niveles de abstracción para el desarrollo de sistemas basados en agentes. Los distintos niveles se construyen componiendo agentes más simples de niveles inferiores. Existen dos tipos de agentes: micro-agente y agente. El micro-agente es el agente atómico del nivel más bajo de abstracción. Mientras que, el agente se compone de agentes o de micro-agentes. No obstante el micro-agente no tiene cualidades de agencia: no se comunica utilizando algún lenguaje de comunicación de agente, no mantiene ontologías, no lleva a cabo razonamiento en tiempo de ejecución, se comporta de manera pre-definida y predecible. Un micro-agente es en definitiva un objeto. Por otra parte los agentes no atómicos del nivel inmediatamente superior al micro-agente sólo se componen de micro-agentes, de esta manera con este enfoque se permite construir entidades a las que se llama agente pero que no satisfacen la definición de agencia. Otra deficiencia de este enfoque es que no se orienta a la identificación de conceptos de la tecnología de agente para el desarrollo de sistemas ya que propone un proceso guiado por tareas sin tratar siquiera con la organización del sistema, los objetivos, los roles del dominio, las interacciones, compromisos, etc.

En [Ferber y Gutknecht, 1998; Ferber *et al.*, 2004], Ferber y otros proponen un modelo basado en roles, agentes y grupos (al modelo lo denominaron inicialmente AALAADIN, pero en las últimas publicaciones lo llaman modelo AGR) para el modelado de sistemas multi agentes. Reconocen la necesidad de otorgar características de agencia a grupos de agentes, pero admiten que esto riñe con la red "plana" de grupos que define AALAADIN y por tanto lo simulan con agentes representantes de grupo. Esta aproximación es limitada y muy rígida especialmente para las etapas iniciales de análisis cuando resulta conveniente abstraer las estructuras internas de los grupos que participan en interacciones.

En [Parunak y Odell, 2002], Parunak y Odell proponen conceptos UML y extensiones AUML para soportar grupos anidados de agentes. Presentan un modelo que integra las ideas de AALAADIN con la perspectiva holónica. Parunak y Odell definen estructuras flexibles en relación a la propuesta de Ferber gracias a la adopción de grupos anidados. No obstante, la utilidad del modelo, el trabajo se limita a proponerlo como enfoque interesante dejando sin abordar muchos puntos críticos tales como: la interacción entre grupos; cómo especificar las características de agencia del grupo; cómo se define el comportamiento del grupo en término del comportamiento de sus miembros; cómo se guía un proceso de desarrollo utilizando este modelo; cómo se traduce el agente grupo a entidades ejecutables (agentes en tiempo de ejecución), etc.

DESIRE [Brazier *et al.*, 2002] es un método composicional de desarrollo de Sistemas Multi Agente. En DESIRE se utilizan básicamente dos tipos de abstracciones para guiar el proceso de desarrollo de SMAs, estas son proceso y conocimiento (tipos de información y bases de conocimiento). DESIRE define cinco etapas (descripción del problema, diseño conceptual, diseño detallado y diseño operacional) para el desarrollo del SMA. Estas etapas se describen diseñando diferentes niveles de abstracción de procesos y de conocimiento. Cada uno de estos niveles se definen como *componentes* compuestos de componentes de niveles inferiores adyacentes. La composición de procesos se define especificando intercambio de información entre procesos y conocimiento de control de tareas. En el desarrollo se puede seguir una perspectiva de tarea o una perspectiva de sistema multi agente. La perspectiva de tarea (similar a la propuesta de [Peña *et al.*, 2002; 2003]) establece que la red de procesos (composición y relación) se identifica primero para finalmente delegar estos procesos a agentes. Mientras que la perspectiva multi-agente es a la inversa, primero se identifican los agentes y posteriormente los procesos dentro de cada agente. DESIRE constituye un enfoque interesante de desarrollo de sistemas multi agente por capas, sin embargo no aborda conceptos clave como: organización de grupo de agentes, interacción entre agentes, entidades mentales, etc.

El Grupo Distribuido de la Universidad de Sevilla propone un desarrollo top-down guiado por capas de abstracción [Peña *et al.*, 2002; 2003]. Utilizan

un enfoque basado en mRI (interacción de múltiples roles). El mRI es la unidad de modelado básica de las etapas de análisis iniciales. Las interacciones multi-partitas se encapsulan en mRIs que pueden conectarse a otros mRIs para representar interacciones más complejas y para detectar posibles bloqueos. Estas interacciones complejas se refinan en interacciones más "pequeñas" (simples) que finalmente deberán ser descritas internamente como secuencias de mensajes utilizando AUML. En resumen este enfoque propone definir en primer término el modelo de patrones de interacción de la organización para finalmente instanciarlo en algún modelo de estructura (agentes, grupos, suborganizaciones, etc.). Es similar a la perspectiva de tarea de DESIRE con la diferencia que la red de procesos (composición y relaciones) se reemplaza por mRIs. Los mRIs constituyen una abstracción más compleja y expresiva que simples relaciones entre procesos mediante intercambio de información como es la composición de procesos de DESIRE.

Matsatsinis y otros presentan una arquitectura de agentes complejos aplicada a una herramienta multi agente para el soporte en el diseño y producción de nuevos productos en empresas [Matsatsinis *et al.*, 2003]. La arquitectura del sistema se basa en dos tipos de agentes: agentes elementales y agentes complejos. Un agente elemental se compone de tres módulos: comunicación, planificación y razonamiento. Por otra parte un agente complejo se compone de los mismos módulos pero con la diferencia que la estructura de su módulo de razonamiento está compuesto por una organización de agentes (elementales y/o complejos). De esta manera los agentes de niveles inferiores ejecutan acciones para dar servicio a los módulos de razonamiento de agentes de niveles superiores. Se define una estructura jerárquica que establece relaciones de subordinación entre un agente complejo y el grupo de agentes que implementan su razonamiento. Estas jerarquías son rígidas (no se admite comunicación entre distintos niveles de subordinación) y pre-definidas (se definen en tiempo de diseño y se mantienen en tiempo de ejecución). La limitación fundamental de este enfoque se deriva del tipo de jerarquía que define ya que ésta prohíbe flexibilidad y escalabilidad del sistema.

El Comité Técnico de FIPA dedicado al Modelado (*FIPA Modelling TC*)

ha demostrado recientemente su interés en el estudio de elementos de modelado apropiados para representar estructuras organizacionales complejas. En la actualidad se encuentra trabajando en la definición de un meta-modelo al que denominan Super-estructura del Diagrama de Clases de Agentes [Odell *et al.*, 2005; FIPA Modeling TC, 2005]. En este metamodelo abordan el problema de la abstracción de organizaciones, y admiten la utilidad de considerar organizaciones como agentes. Para ello definen un tipo de entidad denominada "Grupo Agentificado" que permite modelar sistemas de gran escala encapsulando agrupaciones complejas de agentes como agentes. Para el momento de la escritura de esta tesis el metamodelo de FIPA se encontraba aun incompleto y en una versión preliminar y por tanto no hemos podido incluir más datos sobre el mismo.

3.5. Conclusiones

El análisis desarrollado en este trabajo ha demostrado que los holones y los agentes comparten varias propiedades. Entre todas las características analizadas, la recursión es la única que no está presente como tal en la tecnología de agentes, pero, en la literatura especializada de agentes, existen algunos trabajos relacionados, en la Sección 3.4 hemos presentamos una análisis de los mismos. De este análisis podemos concluir que existen varios enfoques unos basados en estructuras compuestas de agentes [Occello, 2000; Wagner, 2001; Purvis *et al.*, 2002; Ferber y Gutknecht, 1998; Ferber *et al.*, 2004; Parunak y Odell, 2002; Matsatsinis *et al.*, 2003; Odell *et al.*, 2005; FIPA Modeling TC, 2005] y otros en modelos de interacción a distintos niveles de abstracción [Brazier *et al.*, 2002; Peña *et al.*, 2002; 2003]. Sin embargo existe la necesidad de una definición formal de agente recursivo que complete e integre las distintas propuestas. Esta estructura compuesta de agente podría ser de utilidad no sólo para modelar sistemas holónicos sino también para el modelado de sistemas multi agente de gran escala. Estamos convencidos de que varios desafíos difíciles para los sistemas automatizados pueden ser atacados otorgando un significado completo al concepto de agente: adoptando una definición de agente recursivo y permitiendo la creación dinámica de agentes (organizaciones de agentes). Uno de los desafíos más difíciles para los sistemas automatizados es la escalabilidad y

adaptación. La potencia de las organizaciones holónicas, o holarquías, reside en que éstos permiten la construcción de sistemas más complejos a partir de sistemas más simples. En consecuencia nos parece adecuado incorporar esta característica al concepto general de agente. En el Capítulo 4 presentamos la definición de Agente Abstracto que proponemos como elemento de modelado de sistemas complejos.

Como conclusión podemos decir que agentes y holones son similares aunque como enfoques han aparecido de manera distinta. ¿Es un holón un agente?, la respuesta es: Si. ¿Es una holarquía un SMA?, la respuesta es: Si. ¿Es un agente un holón?, la respuesta es: Depende de su estructura interna, si se pueden agregar agentes, entonces si (ver Capítulo 4). ¿Es un SMA una holarquía?, la respuesta es: Depende de la estructura organizativa de sus agentes constituyentes, si estamos hablando de estructuras jerárquicas flojas, entonces si (ver Capítulos 1, 4 y 5).

Capítulo 4

Agente Abstracto

¿Puede un agente ser una colección de varios agentes agrupados que interactúan, una jerarquía, o algún tipo de organización?. Gasser en [Gasser, 1992; 2001], indicaba que casi todas las propuestas de arquitecturas de agentes no se han ocupado del problema general de cómo tratar colecciones de agentes como entidades de nivel más alto, es decir, cómo tratar organizaciones como agentes.

En el área de sistemas inteligentes de control de fabricación se reconoce la necesidad de alguna forma de agregación jerárquica "floja" en sistemas del mundo real, los cuales deben permanecer comprensibles mientras se expanden en un amplio rango de escalas temporales y espaciales. Una fábrica moderna de automóviles, por ejemplo, incorpora cientos de miles de mecanismos individuales en cientos de máquinas que están agrupadas en docenas o más líneas de fabricación. Los ingenieros pueden diseñar, construir, y operar tales sistemas complejos cambiando el enfoque (nivel de abstracción) desde el mecanismo, a la máquina o la línea de fabricación, dependiendo del problema en cuestión, y reconociendo a los agentes de niveles superiores como agregaciones de agentes de niveles inferiores. De manera similar, en las aplicaciones de comercio electrónico, una corporación es una entidad legal que es independiente de las personas individuales que constituyen sus empleados y directores.

Nosotros estamos convencidos que varios desafíos difíciles para los sistemas automatizados, no sólo del área de la fabricación, pueden ser atacados dando un significado completo al concepto de agente: adoptando una definición recursiva de agente y permitiendo la creación dinámica de agentes (organización de

agentes). En este capítulo presentamos la definición de Agente Abstracto como entidad de modelado para sistemas de gran escala. La definición de Agente Abstracto se deriva del estudio realizado en el capítulo anterior. La idea es emparejar la noción de agente a la de holón, por tanto incorporamos a la definición comúnmente aceptada de agente una visión estructural que permite modelar agentes constituidos a su vez por agentes.

4.1. Definiciones

Nuestra definición de Agente Abstracto se basa en la definición ampliamente conocida de Wooldridge y Jennings [Wooldridge y Jennings, 1995].

Definición 4.1. Un Agente Abstracto es un sistema informático con entidad única, situado en algún entorno, y que como un todo, percibe el entorno (entradas sensibles de su entorno). A partir de esas percepciones determina y ejecuta acciones de forma autónoma y flexible - reactivo y pro-activo - que le permiten alcanzar sus objetivos y cambiar su entorno. Desde un punto de vista estructural, un Agente Abstracto puede ser un ente atómico; o ser un Sistema Multi Agente (con entidad única) constituido por Agentes Abstractos no necesariamente homogéneos.

Un Agente Abstracto se encuentra en un nivel de abstracción conceptual superior que un agente. Un Agente Abstracto puede ser visto como un SMA, una organización, una federación o una institución con el valor añadido que puede a su vez ser una composición de todos estos modelos de abstracción. Más aun, cuando definimos dos Agentes Abstractos que interactúan, podríamos también modelar organizaciones que interactúan, federaciones, SMAs o instituciones. Un Agente Abstracto existe sólo en las etapas de modelado, al final (en la etapa de implementación) será reemplazado por un grupo de agentes o por un único agente.

La definición 4.1 proporciona una visión funcional y estructural de agente. La visión funcional es la tradicional definición de agente propuesta por [Wooldridge y Jennings, 1995] en la cual un agente es una entidad autónoma, reactiva y pro-activa. Por otro lado, la visión estructural introduce una recursión indirecta al indicar que un Agente Abstracto puede ser un SMA, el

cual está a su vez constituido por Agentes Abstractos, cada uno de los cuales pudiendo a su vez ser un SMA o un agente atómico.

Definición 4.2. Un Sistema Multi Agente está constituido por dos o más Agentes Abstractos que interactúan para resolver problemas que están más allá de las capacidades o conocimiento individual de cada uno.

La definición 4.2 de Sistema Multi Agente permite construir sistemas en los cuales los bloques de construcción son agentes que interaccionan para lograr uno o varios objetivos globales (el objetivo global se corresponde con el objetivo del sistema como un todo). Esta definición extiende la noción tradicional de Sistema Multi Agente al indicar que un SMA está constituido de Agentes Abstractos. Ésta puede ser una propiedad muy útil ya que con ella podríamos tener un SMA compuesto de SMAs que interactúan.

En un Sistema Multi Agente sus agentes constituyentes pueden estar agrupados de distintas maneras, de ahí que podemos hablar de sociedades de agentes [Corkill y Lander, 1998], organizaciones de agentes [Ferber *et al.*, 2004; Dignum *et al.*, 2002; Juan *et al.*, 2002; Dellarocas y Klein, 1999], instituciones electrónicas [Sierra *et al.*, 2004], federaciones [Cutkosky *et al.*, 1993], coaliciones [Shehory y Kraus, 1998; Pechoucek *et al.*, 2000], etc. En esta tesis nos limitamos a hablar de organizaciones de agentes ya que los demás enfoques se pueden ver como casos especiales de este.

La figura 4.1 muestra la representación gráfica en UML de la estructura del Agente Abstracto. Un Agente Abstracto es una generalización de un SMA y de un Agente. Un SMA está compuesto de dos o más Agentes Abstractos que pueden ser Agentes o, a su vez, SMAs. Una Organización es un SMA que agrupa Roles relacionados entre sí y define patrones de actuación y de interacción definiendo Reglas y Normas. En el capítulo 5 presentamos una discusión detallada de las entidades de la figura 4.1.

Existen dos niveles en un Agente Abstracto. El nivel de abstracción y el nivel de recursión. El nivel de abstracción se utiliza en las etapas de análisis y diseño. Cuando empezamos a analizar un SMA A identificamos el grupo de agentes $\{a_1, a_2, ..., a_{n'}\}$, donde n' es el número de agentes del SMA A. Los agentes $\{a_1, a_2, ..., a_{n'}\}$ se dice que están en un nivel de abstracción inferior que A. Sea m el nivel de abstracción de A, luego $\{a_1, a_2, ..., a_{n'}\}$ están en el nivel de

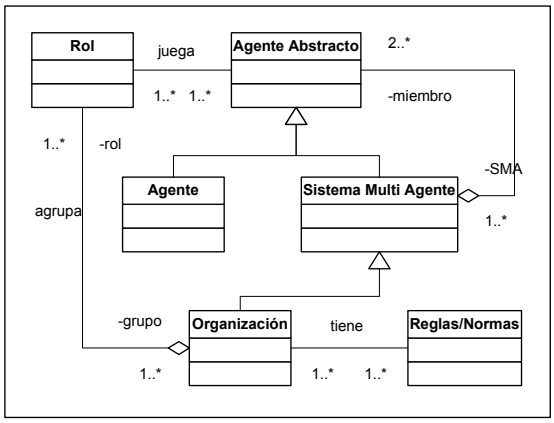


Figura 4.1: Agente Recursivo

abstracción $m-1$. Análisis subsecuentes "abrirán" estos agentes, por ejemplo cuando analicemos a_1, podríamos tener que $a_1 = \{a_{11}, a_{12}, a_{13}\}$. Luego el nivel de abstracción de cada agente en $\{a_{11}, a_{12}, a_{13}\}$ será $m-1$, y así sucesivamente. El nivel de recursión se define como sigue:

Definición 4.3. Sean a un agente y A y A_i Agentes Abstractos. El nivel de recursión de un Agente Abstracto es:

$$LevelR(A) = \begin{cases} 0 & Si\ A\ =\ a, \\ \text{máx}\{LevelR(A_i)\} + 1 & Si\ A\ =\ \{A_1, A_2, ..., A_k\},\ 1 \leq i \leq k. \end{cases}$$

De la definición 4.3 tenemos:

- Un Agente Abstracto de nivel de recursión 0 es un agente.
- Un Agente Abstracto de nivel de recursión 1 es un SMA compuesto de agentes que interactúan.
- Un Agente Abstracto de nivel de recursión $n > 1$ es un SMA compuesto de Agentes Abstractos de nivel de recursión $< n$.

El punto de vista del observador (diseñador) determinará en cada momento la naturaleza de lo que se está observando. Visto desde fuera un sistema puede ser considerado como un agente ya que posee características de agencia. En

cambio visto desde dentro, es decir observando la estructura interna, se lo puede considerar como un conjunto de agentes interrelacionados entre si (con lo cual, desde este punto de vista, deja de ser un agente), o en el caso de que no exista refinamiento, es simplemente un agente. El final de la recursión la define el observador (diseñador) ya que la subdivisión existirá siempre que le sea de utilidad para la definición del problema que está modelando. Finalmente, desde el nivel de abstracción más bajo sólo se "verán" agentes que podemos considerar constituyen el SMA global, pero conforme "subamos" en los niveles de abstracción "veremos" agentes algunos de los cuales serán atómicos y otros serán "refinados" como un SMA.

4.1.1. ¿Por qué el Agente Abstracto?

En esta sección introducimos un ejemplo simplificado con el que podemos ver las distintas perspectivas del observador y su utilidad cuando pretendemos describir un problema complejo con múltiples niveles de abstracción. El detalle de este ejemplo lo presentamos en las siguientes secciones.

Supongamos una compañía Multinacional, llamada AG, con diferentes compañías Nacionales distribuidas en diferentes países. El objetivo es modelar la compañía Multinacional como un SMA.

Cada compañía Nacional puede ser un Agente Abstracto dado que poseen características de agencia. La compañía Nacional es autónoma en su entorno nacional; actúa en su mercado nacional con sus propias reglas de producción y de mercado. Al mismo tiempo, debe ser capaz de interactuar con otras compañías Nacionales para intercambiar materiales, personal, conocimiento, etc, obedeciendo las reglas y normas establecidas por la Multinacional para estas relaciones internacionales.

Las relaciones internacionales entre compañías definen las reglas, normas y políticas de la Multinacional. En la Figura 4.2 podemos ver las áreas geográficas en las cuales las relaciones entre las compañías Nacionales son más estrechas. Además, los acuerdos comerciales entre los distintos países definen nuevas reglas de interrelación entre las compañías nacionales de estas zonas. Por ejemplo, en Europa, los países de la Comunidad Europea están gobernados por ciertos estándares y normas de la comunidad; y en Sudamérica están

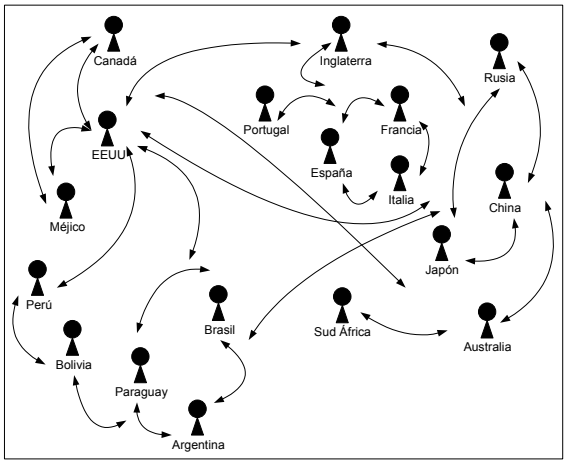


Figura 4.2: Compañías Nacionales de la Multinacional AG.

gobernados por el Mercado Común del Cono Sur - MERCOSUR (Paraguay, Argentina, Chile, Brasil, Uruguay y Bolivia) y por la Comunidad Andina (Bolivia, Colombia, Ecuador, Perú y Venezuela). Las relaciones de los países de estos mercados se gestionan por las reglas locales del mercado. Cada mercado puede ser modelado como un Agente Abstracto (Figura 4.3). Es importante destacar que Bolivia, como compañía Nacional, pertenece a dos compañías Regionales (MERCOSUR y Comunidad Andina).

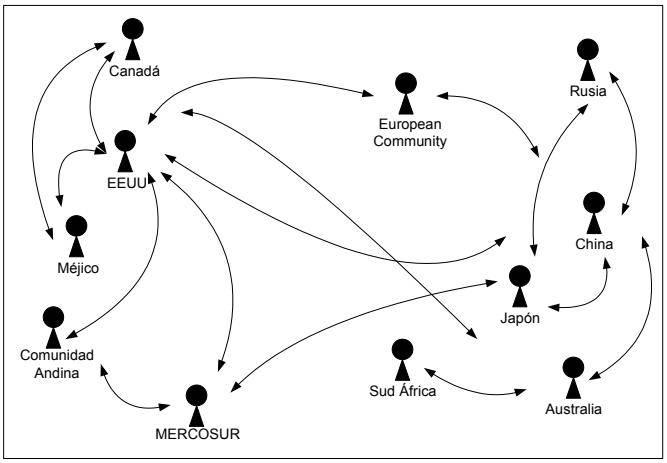


Figura 4.3: Compañías Regionales de la Multinacional AG.

Hasta este punto hemos identificado tres niveles de abstracción (Figura

4.4): la compañía Multinacional, las compañías Regionales, y las compañías Nacionales. Supongamos que hemos modelado la Multinacional como un SMA, que está compuesto por Agentes Abstractos relacionados entre sí mediante ciertos patrones de comportamiento definidos por la Multinacional. Si las compañías Nacionales se componen de agentes simples, podemos concebir a la compañía Nacional como un SMA tradicional (Agente Abstracto de nivel de recursión 1), a las compañías Regionales como Agentes Abstractos de nivel 2 y a la Multinacional como Agente Abstracto de nivel de recursión 3.

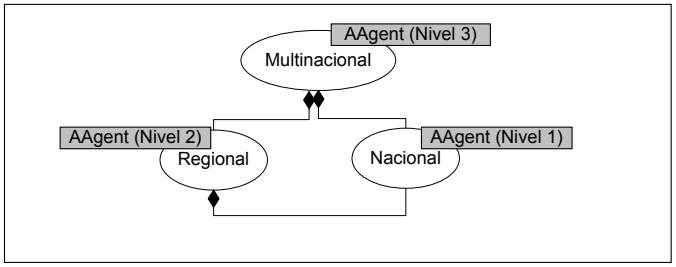


Figura 4.4: Tres niveles de abstracción en la Multinacional AG.

Si el interés del diseñador, además de modelar las relaciones externas, es modelar también la estructura interna de cada compañía Nacional, entonces debería "observar el interior" de la compañía Nacional. Dentro de cada compañía Nacional podrían existir nuevas compañías localizadas en diferentes ciudades o con autonomía para ciertas actividades. Nuevamente, cada compañía Local estaría subordinada a la compañía Nacional y esta última a la Multinacional. De esta manera, tenemos un nuevo nivel de abstracción, la compañía Local como un Agente Abstracto de nivel de recursión 1, la compañía Nacional como Agente Abstracto de nivel 2, la compañía Regional de nivel de recursión 3 y la Multinacional como Agente Abstracto de nivel de recursión 4 (Figura 4.5)

Si la compañía Nacional no se subdivide en compañías locales o compañías autónomas, entonces la compañía Nacional es un SMA tradicional compuesto por agentes específicos del dominio, que llevan a cabo funciones específicas y se encuentra interrelacionados entre sí. Estos agentes implementan los servicios proveídos por la compañía Nacional fuera y dentro de su país. Este mismo análisis se debería hacer para cada compañía Local hasta que llegar a los

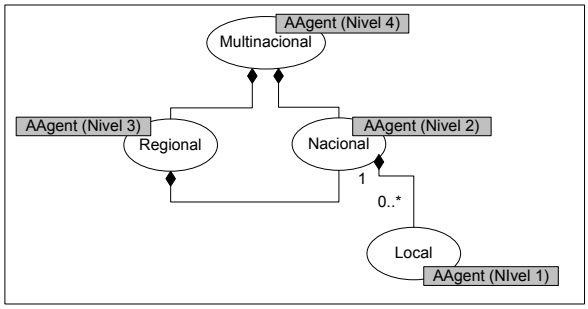


Figura 4.5: Cuatro niveles de abstracción en la Multinacional AG.

agentes, que definen e implementan las actividades de la compañía como un todo. En resumen, el resultado final del análisis debe ser similar al ilustrado en la Figura 4.6. Podemos observar que la compañía Nacional se compone de cero o más compañías Locales, y que cada compañía Local, a su vez, es un Agente Abstracto de nivel de recursión 1. Desde el exterior podemos considerar a la Multinacional como un Agente Abstracto, ya que está situada en un entorno, el mercado mundial; es autónoma; tiene sus propias políticas económicas y de mercado; es social, es decir puede interactuar con otras entidades para comprar, vender, alquilar, reclutar personal, etc.; tiene un comportamiento pro-activo, dado que, por ejemplo, de acuerdo con las tendencias del mercado mundial es capaz de modificar sus políticas de mercado actuales, etc.

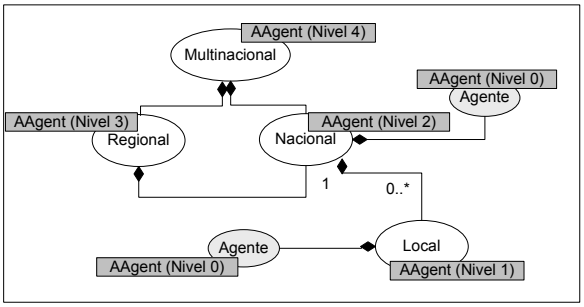


Figura 4.6: Cinco niveles de abstracción en la Multinacional AG.

4.2. Comportamiento de un Agente Abstracto

La perspectiva funcional de la definición 4.1 establece que el Agente Abstracto es una entidad autónoma y actúa de manera reactiva y pro-activa. En esta sección presentamos la formalización de este comportamiento siguiendo el *principio de racionalidad* [Ferber, 1999] y la arquitectura BDI [Kinny y Georgeff, 1997]. El principio de racionalidad establece que el control del agente se basa en el uso de objetivos y creencias para ejecutar tareas. Este principio ha sido ratificado en el modelo BDI de arquitectura de agente. Y es por ello que las definiciones de esta sección se basan en: percepciones, acciones (tareas), creencias y objetivos.

En primer lugar introducimos algunas notaciones que utilizaremos a lo largo de esta sección. Luego presentamos la definición del comportamiento reactivo e intencional de un SMA (Agente Abstracto de nivel de recursión 1) para finalizar con el caso general del Agente Abstracto de nivel de recursión n.

4.2.1. Notación

Sea $A_{m,n}$ un ***SMA***, donde m representa el nivel de abstracción y n representa el nivel de recursión. Cuando no haya posibilidad de confusión utilizaremos simplemente A.

El conjunto de todos los ***SMAs del universo*** se denota por $\overline{A}$.

Sea N el ***número de agentes del universo***.

Los ***agentes*** constituyentes del SMA A, se denotan por a_i, donde i representa el i-ésimo elemento de un conjunto de n' agentes de A (n' es el número total de agentes en A).

Una ***acción primitiva*** que puede ser ejecutada por algún agente $a_i \in A$, se denota por γ_{ij} donde j denota el j-ésimo elemento del conjunto Γ_i, de r_i acciones primitivas del agente a_i.

Una ***acción social*** que puede ser ejecutada por el grupo de agentes $\{a_1, a_2, ..., a_k\}$, $a_i \in A$, se denota por s. Sea S_A el conjunto de acciones sociales de A.

Una ***percepción de un agente*** $a_i \in A$ se denota por p_{ij}, donde j denota al j-ésimo elemento de un conjunto P_i, de h_i percepciones del agente a_i.

Sea o_{ij} un ***objetivo de un agente*** $a_i \in A$, donde j denota el j-ésimo elemento del conjunto O_i, de m_i objetivos del agente a_i.

Sea $\overline{O} = \bigcup_{i=1}^{N} O_i$ el conjunto union de todos los ***objetivos de los agentes del universo***.

4.2.2. Comportamiento de un Sistema Multi Agente

En la mayoría de los casos, los agentes existen en el contexto de Sistemas Multi Agente, cuyo comportamiento global se deriva a partir de las interacciones entre sus agentes constituyentes. En estos casos, los agentes exhiben comportamiento social; interactúan unos con otros; ya sea cooperando para alcanzar un objetivo común o ayudándose unos a otros para lograr sus objetivos individuales.

Un SMA tiene una estructura social interna, que se deriva a partir de las interrelaciones entre los agentes y a partir del propósito del SMA (objetivos compartidos de los agentes que lo componen). Desde fuera un SMA tiene un comportamiento que emerge de esta sociedad interna de agentes. Este comportamiento es:

4.2.2.1. Reactivo

El comportamiento de un agente reactivo responde al principio básico de acción/reacción. Un SMA es reactivo porque cuando un SMA, como un todo, percibe eventos o cambios en su entorno reacciona efectuando cambios en el entorno.

El conjunto de percepciones del SMA A en término de las percepciones de sus agentes componentes se puede definir como:

Definición 4.4. El conjunto de percepciones P_A del SMA A es:

$$P_A = \bigcup_{i=1}^{n'} P_i \ , \quad P_i \text{ es el conjunto de percepciones del agente } a_i \in A$$

De la misma manera, el conjunto de acciones de un SMA A en término de las acciones de sus agentes componentes se puede definir como:

Definición 4.5. El conjunto de acciones Δ_A de un SMA A es:

$$\Delta_A = \bigcup_{i=1}^{n'} \Gamma_i \cup S_A,$$

Γ_i es el conjunto de acciones primitivas del agente a_i,

y S_A es el conjunto de acciones sociales de A.

Volviendo al ejemplo de la Multinacional AG de la Sección 4.1.1. Supongamos que AG es una fábrica de automóviles y que existen dos tipos de compañías en AG: compañías de fabricación que fabrican partes de automóviles, y compañías ensambladoras que ensamblan partes pre-fabricadas. Supongamos compañías Locales de fabricación con un *Agente Director*, unos pocos *Agentes de Gestión* y varios *Agentes Productores*. El *Agente Director* es responsable del siguiente conjunto de acciones o tareas: *aceptar órdenes de fabricación*, *programar órdenes de fabricación* y *asignar órdenes de fabricación a los* Agentes de Gestión. Además, el *Agente Director* percibe el siguiente elemento del entorno: *orden de fabricación*. El *Agente de Gestión* tiene las tareas: *aceptar programación de órdenes*, *programar tareas de fabricación*, *asignar tareas de fabricación a los Agentes Productores*. Las percepciones del *Agente de Gestión* son: *tareas de fabricación* y *órdenes de fabricación*. Los *Agentes Productores* son agentes específicos y cada uno de ellos tiene sus propias habilidades o capacidades (conjunto de tareas que es capaz de realizar). Supongamos el siguiente conjunto de acciones distribuidas entre los *Agentes Productores*: *soldar*, *clavar*, *atornillar*, *transportar*. Supongamos también el siguiente conjunto de elementos del entorno que pueden percibir: *tornillos*, *materiales* y *productos*.

De la Definición 4.5, la compañía Local de Fabricación tiene las siguientes acciones o tareas: *aceptar órdenes de fabricación*, *programar órdenes de fabricación*, *asignar órdenes de fabricación a los* Agentes de Gestión, *aceptar programación de órdenes*, *programar tareas de fabricación*, *asignar tareas de fabricación a los Agentes Productores*, *soldar*, *clavar*, *atornillar* y *transportar*. Lógicamente no todas estas tareas serán ofrecidas al exterior, es por tanto tarea del diseñador decidir (dependiendo de la especificación del problema) cuáles serán las acciones o servicios que la compañía Local de Fabricación ofrezca a

sus clientes (en el Capítulo 5 presentamos guías de selección que ayudan al diseñador). De la definición 4.4 tiene el siguiente conjunto de percepciones: *órdenes de fabricación*, *tareas de fabricación*, *tornillos*, *materiales* y *productos*.

4.2.2.2. Intencional

El comportamiento de un agente intencional está guiado por procesos deliberativos basados en actitudes mentales tales como creencias, conocimientos, deseos, intenciones, compromisos, etc. Siguiendo el *principio de racionalidad* y la arquitectura BDI en esta sección presentamos la formalización de objetivos y creencias del SMA.

Objetivo

Cuando el agente es atómico la definición de sus intenciones puede ser sencilla y clara utilizando cualquiera de las arquitecturas deliberativas [Haddadi y Sundermeyer, 1996; Jennings *et al.*, 1992] o híbridas [Ferguson, 1995; Fischer *et al.*, 1996; Georgeff y Ingrand, 1989] que existen en la literatura especializada de SMA. Pero, ¿cómo definir las intenciones de un SMA en términos de las intenciones de los agentes que lo constituyen ya sean estos atómicos o a su vez SMA?. Esta definición es mucho más compleja y además no existe una arquitectura de SMA que la soporte.

La definición de las intenciones de un SMA es de fundamental importancia para la especificación de su comportamiento y por tanto para la definición de interacciones entre distintos SMAs.

La intención de un SMA puede ser muy compleja de definir, ya que el comportamiento del sistema como un todo no es simplemente la suma o composición de los comportamientos de sus agentes, sino algo más. El comportamiento de un SMA depende no sólo del comportamiento individual de cada agente componente, sino también de las interacciones que existen entre ellos. Un ejemplo muy claro son los sistemas emergentes, en los cuales el comportamiento global resulta de la interacción de los componentes entre sí.

Para definir este comportamiento intencional del sistema, es necesario determinar los posibles tipos de relaciones que pueden existir entre los agentes que componen el SMA. Es decir debemos identificar, ¿Qué es lo que los agentes hacen juntos?.

Parunak y otros [Parunak *et al.*, 2002] describen el comportamiento conjunto de los agentes en términos de:

- **Correlación**: Un conjunto de agentes con información conjunta positiva están Correlacionados. La información conjunta de un sistema se define como la diferencia entre la suma de las entropías de Shannon de cada agente del sistema y la entropía del sistema como un todo [Parunak *et al.*, 2002]. Al nivel más bajo, los agentes están Correlacionados cuando sus acciones son estáticamente dependientes de aquellas de otros agentes. A este nivel no importa si esta Correlación resulta de flujos de información entre agentes, o entre ellos y un controlador central, o si sus raíces son cognitivas o sub-cognitivas. La Correlación se manifestará como un aumento de la información conjunta del sistema, y de hecho se propone esta medida como una medida de la Correlación del sistema. Determinar Correlación en una población de agentes es un proceso empírico que no requiere ningún conocimiento acerca de la estructura interna de los agentes ni de la organización exterior.

- **Coordinación**: La diferencia entre Correlación y Coordinación es que la Correlación describe simplemente el hecho de no-independencia estadística entre comportamientos de agentes, mientras que la Coordinación implica un proceso. Se cree que todos estos procesos involucran comunicación, es decir, flujo de información entre un agente individual y su entorno.

  El interés en los procesos de comunicación enfatizados por la Coordinación requiere que se investiguen características organizacionales entre los agentes, pero sin tener en cuenta la lógica interna de los agentes.

  La relación entre un par de agentes dados, en cualquier momento será uno de dos tipos, dependiendo del estado de los agentes y las reglas del sistema. Cuando los agentes pueden decir No entre ellos dentro de las leyes del sistema, se dice que son agentes pares. Cuando uno de ellos (digamos el agente A) puede decir No al otro (B), pero B no puede decir No a A, se llama a A el agente distinguido y a B el agente subordinado. La relación entre dos agentes puede ser fija (por ejemplo, la relación entre

un programador humano y su agente software). O puede variar con el tiempo (cuando se habla de agentes pares que negocian un plan de trabajo que necesita que uno de ellos supervise al otro, resultando en una relación distinguido-subordinado durante la ejecución).

El entorno tiene un conjunto de variables de estado que pueden ser percibidas por los agentes. Es útil distinguir dos categorías de variables de entorno. Los valores de variables *endógenas* cambian con el tiempo dependiendo de las acciones tomadas por los agentes pares. Los valores de las variables *exógenas* cambian con el tiempo independientemente de las acciones de agentes pares, pueden ser vistas como resultantes de acciones de los agentes distinguidos, y son generalmente indicadas por el diseñador del sistema. En tales entornos, los mecanismos de Correlación pueden ser tanto centralizados o descentralizados.

Mecanismos Centralizados para correlación, todos involucran comunicación entre los agentes distinguidos y sus subordinados. Este flujo puede tomar parte directamente (cuando el agente distinguido Construye o Comanda a los subordinados) o indirectamente (cuando el agente distinguido Coarta a los subordinados manipulando variables de entorno *exógenas* visibles a los subordinados).

Mecanismos Descentralizados para correlación, todos involucran comunicación entre pares. El volumen de investigación sobre negociación se centra en flujos directos de información par-a-par, o Conversación. Flujos indirectos descentralizados ocurren cuando los pares realizan y perciben cambios sobre variables *endógenas* de entorno. Esta clase de coordinación se llama "stigmergy", del griego *stigma* "signo" y *ergos* "trabajo".

- **Cooperación y Disputa**: Para determinar si los agentes Cooperan o Disputan, se debe mirar dentro de estos. Por ejemplo, Los comerciantes en un mercado de artículos exhiben un alto grado de Correlación en sus acciones, resultando de los flujos de información entre ellos (Coordinando). Pero, ¿están ellos Cooperando o Disputando?. No es suficiente observar sus acciones externas. Dos comerciantes ambos pujando por el mismo artículo podrían estar Disputando (cada uno buscando arrebatar

el control del otro), podrían estar Cooperando (elevando el precio para aumentar el valor de sus pertenencias actuales), o simplemente Compitiendo (recursos limitados).

Una condición necesaria para la Cooperación es la existencia entre los agentes que cooperan de intenciones conjuntas. La Disputa sugiere una intención de la parte de un agente de frustrar las intenciones del otro. No se requiere intenciones para la Competición. Así, tanto la Cooperación como la Disputa son Correlaciones dirigidas por las intenciones de los agentes. La Cooperación y la Disputa sólo se puede imputar a poblaciones de agentes cognitivos. Ni la Cooperación ni la Disputa requieren de comunicación descentralizada directa, pero podría resultar a partir de flujos centralizados de información en tiempo de diseño o de interacciones mediadas del entorno.

- **Congruencia y Coherencia**: La idea crucial es que el sistema puede tener objetivos asociados a dos niveles: el sistema, y los agentes individuales. Categorías tales como Disputa y Cooperación toman en cuenta objetivos de agentes individuales, pero no objetivos del sistema. Se propone la Congruencia para caracterizar el grado en el cual patrones de interacción entre agentes (a cualquier nivel desde Correlación pasando por Disputa y Cooperación) satisface (es Congruente con) los objetivos del sistema, mientras que las relaciones entre los agentes en si que derivan la Congruencia es la Coherencia.

En resumen, la forma de relación más básica entre agentes de un sistema es la correlación, que consiste en la dependencia estadística entre las acciones de un agente y las de los demás agentes del sistema. La determinación del nivel de correlación de un sistema no requiere un análisis de la estructura organizacional del sistema, ni de la estructura interna de cada agente. La coordinación en cambio es la correlación que implica proceso, la mayoría de estos procesos involucran comunicación. Para determinar coordinación es necesario analizar la estructura organizacional del sistema, sin tener en cuenta la lógica interna de los agentes. La cooperación y la disputa son correlaciones dirigidas por las intenciones de los agentes. Para determinar si los agentes cooperan o disputan, se debe "mirar" dentro de cada agente. Una condición necesaria para la

cooperación es la existencia de intenciones conjuntas entre los agentes que cooperan. La disputa, en cambio, sugiere una intención de la parte de un agente de frustrar las intenciones del otro. La competición es una coordinación que no requiere de intención, y por tanto se puede imputar a agentes no-cognitivos.

La congruencia y la coherencia son los conceptos que enlazan cualquier tipo de correlación (coordinación, cooperación, disputa y competición) con los objetivos del sistema y los objetivos individuales de los agentes.

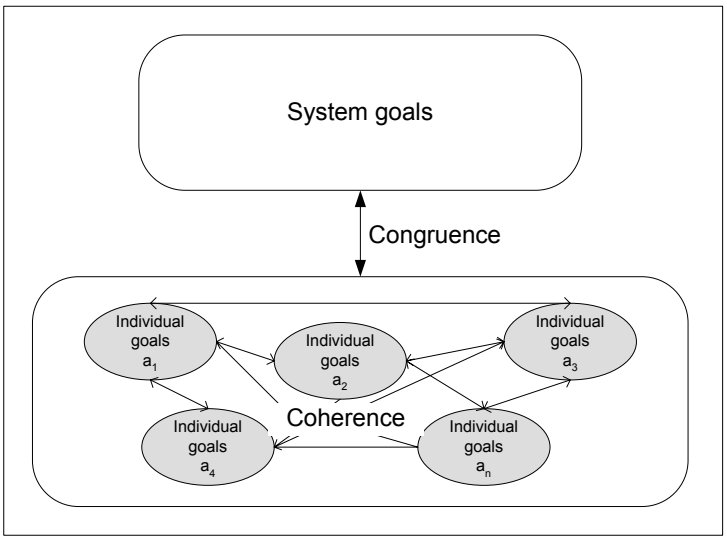


Figura 4.7: Congruencia y Coherencia

En la figura 4.7 se puede apreciar los objetivos asociados a dos niveles de abstracción. Los objetivos a nivel de SMA y los objetivos a nivel de agente individual. La coherencia define los patrones de relación entre agentes ya sean estos de coordinación, cooperación, disputa o competición, que derivan en congruencia. Mientras que la congruencia mide el grado de adecuación del comportamiento del sistema con respecto a los objetivos de sistema como un todo. Por tanto cuanto más congruente sea un sistema más correcto es el sistema (satisface sus objetivos). Para ilustrar estos dos conceptos, consideremos un grupo de agentes en la compañía Local de la Multinacional AG de la Sección 4.1.1. La compañía Local tiene sus reglas y normas de organización y está compuesta por un *Agente Director*, un *Agente Jefe de Producción* y varios *Agentes Trabajadores*. La compañía Local tiene el objetivo *minimizar los*

tiempos de entrega. Para lograr este objetivo la compañía ha establecido las siguientes reglas de comunicación y delegación. El *Agente Director* es responsable de recibir órdenes de producción y de negociar los tiempos de entrega con el cliente. Luego de esto, el *Agente Director* se comunica con el *Agente Jefe de Producción* para programar la orden de producto. El *Agente Jefe de Producción* hace una primera propuesta de tiempo de entrega, tratando de minimizar el tiempo de entrega y tomando en cuenta la disponibilidad de los A*gentes Trabajadores*. El *Agente Director* y el *Agente Jefe de Producción* negocian sobre la primera propuesta. Siguen con esta negociación hasta que llegan a un acuerdo que satisface los objetivos de la compañía. Cuando tienen un acuerdo el plan de producción se programa y los *Agentes Trabajadores* son responsables de ejecutarlo. Un comportamiento congruente es, por ejemplo, cuando el *Agente Director*, el *Agente Jefe de Producción* y los *Agentes Trabajadores* consiguen sus propios objetivos. Un patrón de relación coherente es, por ejemplo, la relación entre el *Agente Jefe de Producción* y el *Agente Director* para negociar una programación que minimiza el tiempo de entrega.

Hasta este punto hemos señalado los posibles tipos de relaciones que pueden existir entre los agentes constituyentes de un SMA. En los siguientes párrafos presentaremos los pasos a seguir para definir los objetivos de un SMA. La idea básica es identificar patrones de relación coherentes, es decir, comportamientos de grupo congruentes con algún objetivo. Para ello, el primer paso es identificar todos los patrones de coordinación, cooperación y competición en el SMA. Una vez hayamos identificado estos patrones, el segundo paso es determinar qué objetivos son satisfechos por estos patrones de relación. Finalmente, los objetivos del SMA serán todos los objetivos identificados en el segundo paso.

Los patrones de interacción (coordinación, cooperación y competición) entre agentes son secuencias ordenadas de acciones que al mismo tiempo pueden ser modeladas como planes para alcanzar los objetivos globales del sistema [Parunak *et al.*, 2002]. A continuación presentamos la definición formal de una secuencia ordenada de acciones (patrones de relación).

Definición 4.6. Una instancia de una acción primitiva se define como $< a_i, \gamma_{ij} >$ tal que el agente $a_i \in A$ participa en la acción primitiva $\gamma_{ij} \in \Gamma_i$.

Definición 4.7. Una instancia de una acción de grupo se define como $<$

$\{a_1, a_2, ..., a_k\}$, $s >$ tal que el grupo de agentes $\{a_1, a_2, ..., a_k\} \subset A$, $1 < k < n'$ participa en la acción $s \in S_A$.

Las acciones primitivas y de grupo se pueden combinar en secuencias finitas (planes) para especificar interacciones más complejas. Una secuencia se compone de al menos una acción y puede contener una mezcla de acciones primitivas/de grupo. Una secuencia se denota por Σ. En los sistemas intencionales, las acciones se llevan a cabo para satisfacer objetivos. Nuestro interés es identificar qué objetivos puede satisfacer Σ.

Una instancia de secuencia de acciones primitivas y de grupo especifica las acciones y los agentes que van a ejecutarlas. Si Σ es una secuencia de acciones, su instancia la denotamos por Σ'. Sea Σ'_A el conjunto de todas las instancias de secuencias de acciones de un SMA A.

Definición 4.8. Sea $\overline{\Sigma} = \bigcup_{A \in \overline{A}} \Sigma'_A$ el conjunto unión de todos los conjuntos de instancias de secuencias de acciones de todos los SMAs en $\overline{A}$.

Debemos definir una relación entre las instancias de secuencias de acciones del universo $\overline{\Sigma}$ y todos los objetivos del universo $\overline{O}$[1]. Esta relación dada una instancia de secuencia de acción obtendrá el conjunto de objetivos que la secuencia satisface. Llamamos a esta relación f. Y la definimos como sigue:

Definición 4.9. Sea f la función $f : \overline{\Sigma} \longrightarrow \overline{O}$. $f(\Sigma')$ es el conjunto de objetivos que Σ' logra satisfacer.

f es una función ya que para toda Σ' existe al menos un objetivo $o \in \overline{O}$ que Σ' puede satisfacer. f es una función no-inyectiva porque un objetivo o un conjunto de objetivos pueden ser satisfechos por una o más instancias de secuencias de acciones. Por ejemplo, consideremos el problema del "mundo de bloques". Un objetivo dado puede tener infinitas instancias de secuencias de acciones que pueden alcanzarlo. f es una función no-sobreyectiva porque puede existir un objetivo en $\overline{O}$ para el cual no existe una instancia de secuencia de acción en $\overline{\Sigma}$ que lo satisface. Por ejemplo, consideremos el objetivo "resolver un problema NP en un tiempo polinomial", no existe ninguna instancia de secuencia de acción (algoritmo) que pueda satisfacerlo.

[1]Se puede probar que $\overline{O}$ existe y que es finito y numerable

La función f define los objetivos de un SMA de manera bottom-up, a partir del comportamiento de los agentes constituyentes define los objetivos del SMA. La manera top-down también es posible, podemos definir el comportamiento de los agentes constituyentes a partir de los objetivos del SMA. Para ello definimos g como sigue:

Definición 4.10. Sea g la relación $g : \overline{O} \longrightarrow \overline{\Sigma}$. $g(o)$ es el conjunto de instancias de secuencias de acciones que logran satisfacer el objetivo o.

Podemos ver a g como el recíproco de f. g es una relación pero no una función ya que podría existir un objetivo $o \in \overline{O}$ para el que no existiera una instancia de secuencia de acción en $\overline{\Sigma}$ que logre satisfacerlo. Sin embargo, podemos definir una función a partir de g de la siguiente manera:

Definición 4.11. Sea $\overline{g}$ la función $\overline{g} : \overline{O} \longrightarrow (\overline{\Sigma} \cup \{\hat{\sigma}\})$, donde $\hat{\sigma}$ es la acción nula que no consigue satisfacer ningún objetivo.

$$\overline{g}(o) = \begin{cases} \Sigma' & Si\ f(\Sigma') = o, \\ \hat{\sigma} & en\ otro\ caso \end{cases}$$

La relación g nos permitirá traducir un objetivo dado de un SMA a un conjunto de comportamientos de los agentes constituyentes del SMA. Es decir, g será una guía para el diseñador del SMA. Más aun, cuando exista un objetivo para el cual no exista ninguna instancia de secuencia de acciones en g, entonces no existirá ningún SMA que pueda satisfacer el objetivo.

Los pasos en la definición operacional del comportamiento de un SMA dependiendo del enfoque son:

- Enfoque bottom-up: Se pretende definir los objetivos del SMA A a partir de las interacciones de sus agentes constituyentes.
  - *Paso 1*: Construir el conjunto Σ'_A con todas las instancias de secuencias de acciones (patrones de coordinación, cooperación y competición) observadas en A.
  - *Paso 2*: Construir OS_A, el conjunto de los objetivos del SMA A, aplicando la función f a cada instancia de secuencia de acción identificada en el paso anterior. Esto es:

Definición 4.12. $OS_A = \bigcup_{i=1}^{|\Sigma'_A|} f(\sigma_i)$, $\sigma_i \in \Sigma'_A$

- Enfoque top-down: el conjunto de objetivos del SMA está definido y se pretende encontrar el conjunto de agentes (con ciertas habilidades) para satisfacer estos objetivos.

  - Construir Σ'_A, el conjunto de instancias de secuencias de acciones del SMA A, aplicando $\overline{g}$ a cada objetivo del conjunto OS_A. Es decir:

    Definición 4.13. $\Sigma'_A = \bigcup_{i=1}^{|OS_A|} \overline{g}(o_i)$, $o_i \in OS_A$.

A modo de ejemplo, supongamos una compañía Local de Fabricación de la Multinacional AG (Sección 4.1.1). Para definir a la compañía Local, con el enfoque bottom-up, debemos analizar los patrones de relación observados en la compañía. Supongamos, entre otros, los siguientes patrones de relación: negociación entre el *Director* y los *Gestores* para programar una orden; comunicación entre el *Gestor* y los *Productores* para la asignación de tareas; comunicación entre un *Gestor* y el *Director* sobre el estado de las órdenes; comunicación entre el *Gestor* y un *Productor* sobre el estado de las tareas; cooperación entre los *Productores* para el transporte de materiales; coordinación entre los *Productores* para construir partes. Supongamos que tenemos definido $\overline{O}$ [2]. Utilizando f sobre los patrones de relación de la compañía Local de Fabricación, podemos obtener los siguientes objetivos: minimizar el tiempo de entrega, fabricar partes, cooperar con socios fabricantes, negociar órdenes de fabricación, recibir órdenes de fabricación, y muchos otros objetivos. Como uno se puede imaginar, la lista de objetivos resultante podría ser muy larga, sin embargo podemos extender la función f con información específica del dominio de modo a eliminar objetivos redundantes o imposibles (en el Capítulo 5 se estudian algunas guías de desarrollo para el diseñador).

Creencia

[2] En lugar de tener $\overline{O}$ podemos tener un subconjunto $\overline{O\prime}$. $\overline{O\prime}$ podría ser el conjunto de objetivos específicos de fabricación. Es decir, podemos definir $\overline{O\prime}$ teniendo en cuenta todos los posibles objetivos en un dominio de fabricación

Las creencias representan el estado de información de un agente BDI, es decir, qué sabe acerca de sí mismo y de su entorno. El diseñador de un SMA debe especificar la estructura de información de los agentes que está modelando, esto es, el conjunto de estructuras con las que se pueden expresar las creencias del agente. En otras palabras, el diseñador debe establecer qué es lo que un agente puede creer (fórmulas válidas de creencia) y además puede especificar las creencias iniciales del agente (instancias de fórmulas de creencias). En esta sección presentamos una formalización de la estructura y estado de información de un SMA en término de la estructura y estado de información de sus agentes constituyentes.

Para formalizar las creencias del SMA utilizamos el modelo de mundos posibles. La semántica se define con estructuras de Kripke [Kripke, 1963]. A continuación presentamos los operadores de creencias y conjuntos de creencias que utilizaremos en esta sección.

Sea $\overline{BS}_{a_i}$ el conjunto de fórmulas de creencias BS_{a_i} del agente a_i.

Sea $\overline{B}_{a_i}(t)$ el conjunto de instancias de fórmulas de creencias del agente a_i en el tiempo t. $B_{a_i}(t)$ es una creencia de a_i en el tiempo t, y $\overline{B}_{a_i}(t)$. T_0 es $t = 0$ el tiempo inicial, en etapa de análisis y diseño, y el tiempo de creación del agente en etapa de ejecución.

La estructura de información de un agente es el conjunto de sus fórmulas de creencias. Mientras que, su estado de información (las creencias del agente en un momento determinado) es el conjunto de instancias de sus fórmulas de creencias.

Definición 4.14. La estructura de información de un SMA es la composición de las estructuras de información de sus agentes constituyentes.

En la etapa de análisis y diseño de un SMA se pueden dar dos situaciones:

- La estructura de información de cada agente (del SMA) está definida y se pretende definir la estructura de información del SMA. Para esta situación aplicamos un enfoque bottom-up:

Definición 4.15. La estructura de información $\overline{BS}_A$ que emerge en un SMA es la conjunción de los conjuntos $\overline{BS}_{a_i}$ de sus agentes constituyentes.
$\overline{BS}_A = \bigcup_{a_i \in A} \overline{BS}_{a_i}$.

- La estructura de información del SMA está definida y se pretende derivar las estructuras de información de los agentes del SMA. Para ello debemos analizar las acciones y percepciones de los agentes del SMA para determinar las creencias que se crean, destruyen o modifican por determinadas acciones y/o percepciones del agente, es decir:

  1. Para cada $BS_A \in \overline{BS_A}$ y cada agente a_i del SMA A, analizar cada percepción $p_{ij} \in P_{a_i}$, del agente a_i para determinar si la creencia BS_A aparece en la percepción p_{ij}. Si es así, anexar BS_A a $\overline{BS}_{a_i}$.
  2. Por cada $BS_A \in \overline{BS_A}$ y cada agente a_i del SMA A, analizar cada acción primitiva $\gamma_{ij} \in \Gamma_{a_i}$ del agente a_i para determinar si BS_A aparece en γ_{ij}. Si es así, anexar BS_A a $\overline{BS}_{a_i}$.

La estructura de información del SMA define las fórmulas de creencias válidas del SMA. Una instancia de una estructura de información se define como el estado de información del SMA. El estado de información del SMA será modificado en los pasos sucesivos de ejecución en respuesta a las acciones y percepciones de sus agentes constituyentes. Podemos decir entonces que el estado de información de un SMA en un momento dado será "algún" conjunto de instancias de fórmulas de creencias de sus agentes constituyentes. Para determinar de qué tipo de conjunto estamos hablando a continuación presentamos una lógica de creencias de grupo.

Halpern y Moses, en [Halpern y Moses, 1986], indicaron que en un SMA existe una jerarquía de estados de creencias de grupo. El estado más débil de creencia es la *creencia implícita* o creencia distribuida, que se corresponde con creencias que están distribuidas entre los miembros del grupo, sin la necesidad de que ningún agente individual la tenga como propia. El estado más fuerte de creencias en la jerarquía es la *creencia común*, se corresponde con la creencia

pública. Nuestra definición de las creencias del SMA está basada en estos estados de creencia.

¿Qué significa decir que un grupo G de agentes cree un hecho φ en el instante de tiempo t?. A continuación presentamos algunas definiciones derivadas del trabajo de Halpern y Moses, [Halpern y Moses, 1986], para los distintos tipos de creencias en un grupo de agentes.

- $D_G\varphi(t)$ (se lee *el grupo* G *tiene la creencia* φ *distribuida en el tiempo* t). Por ejemplo si un miembro del grupo G, en el instante t, cree ψ y otro, en el mismo instante, cree que $\psi \rightarrow \varphi$, se puede decir que el grupo G tiene la creencia distribuida φ en el instante t.

- $I_G\varphi(t)$ (se lee *alguno en* G *cree* φ *en el instante* t). $I_G\varphi(t)$ se cumple ssi algún miembro de G cree φ, en el instante t. Formalmente,

  $$I_G\varphi(t) \equiv \bigvee_{a_i \in G} B_i\varphi(t)$$

  Sea $\overline{I}_G(t)$ el conjunto de todas las creencias individuales del grupo G en el instante t.

- $E_G\varphi(t)$ (se lee *todos en* G *creen* φ *en el instante* t). $E_G\varphi(t)$ se cumple ssi todos los miembros de G creen φ en el instante t. Formalmente,

  $$E_G\varphi(t) \equiv \bigwedge_{a_i \in G} B_i\varphi(t)$$

  Sea $\overline{E}_G(t)$ el conjunto de todas las E-creencias del grupo G en el instante t.

- $E^k_G\varphi(t)$, para $k \geq 1$ (se lee φ *es una* E^k-creencia *in* G *en el instante* t). $E^k_G(t)$ se define como

  $$E^1_G\varphi(t) = E_G\varphi(t)$$
  $$E^{k+1}_G\varphi(t) = E_G(E^k_G\varphi(t))(t), \text{ for } k \geq 1$$

  φ se dice que es una E^k-*creencia* en G en el instante t si *todos en* G *creen que todos en* G *creen que ... que todos en* G *creen que* φ *es verdadero en el instante* t, donde la frase *todos en* G *creen que* aparece en la sentencia k veces.

Sea $\overline{E^k_G}(t)$ el conjunto de todas las E^k-creencias del grupo G en el instante t.

- $C_G\varphi(t)$ (se lee φ *es una creencia común en* G *en el instante* t). La fórmula φ se dice que es una creencia común en G en el instante t si φ es una E^k-*creencia* para todo $k \geq 1$ en el instante t. En otras palabras,

  $$C_G\varphi(t) \equiv E_G\varphi(t) \wedge E^2_G\varphi(t) \wedge ... \wedge E^m_G\varphi(t) \wedge ...$$

  Sea $\overline{C}_G(t)$ el conjunto de todas las creencias comunes del grupo G en el instante t.

Claramente, la noción de creencia de grupo forma una jerarquía
$C\varphi \rightarrow ... \rightarrow E^{k+1}\varphi \rightarrow ... \rightarrow E\varphi \rightarrow I\varphi \rightarrow D\varphi \rightarrow \varphi$

A partir de los distintos estados de información posibles en un grupo, definimos el estado de información de un SMA de la siguiente manera:

Definición 4.16. El estado de información de un SMA A es la unión de los posibles estados de información en un grupo de agentes.
$\overline{B}_A(t) = \overline{D}_A(t) \bigcup \overline{I}_A(t) \bigcup \overline{E^k_A}(t) \bigcup \overline{C}_A(t)$

Es importante destacar que con las creencias distribuidas en un SMA podríamos llegar a situaciones en las cuales tenemos conjunto de creencias incoherentes de agentes, representados por fórmulas como $B_i\varphi \wedge B_j \sim \varphi$ (para a_i y a_j posiblemente diferentes). Esto es correcto cuando queremos representar creencias incoherentes de agentes. Pero, ¿qué debemos de hacer con ellos si queremos representar las creencias a nivel del SMA?.

Meyer y van der Hoek, en [Meyer y van der Hoek, 1993], proponen varios mecanismos para tratar con estas situaciones en el nivel de objeto como un problema de reflexión en el meta-nivel de razonamiento, también llamado *reflexión hacia abajo*. Podríamos utilizar este mecanismo para resolver conflictos en el nivel SMA cuando estuviéramos representado creencias distribuidas o implícitas del SMA. Lo llamamos *reflexión hacia arriba*, ya que empezamos en niveles inferiores (agentes) y reflejamos sus creencias a un nivel superior (SMA). Algunas maneras para resolver estos conflictos podrían ser:

1. *Definir un orden explícito $\prec$ entre los agentes*, $a_1 \prec a_2$ significa que el agente a_1 es 'mejor' en el sentido de preferido o relevante que el agente a_2. El orden $\prec$ se tiene en cuenta para resolver conflictos como: cuando las fórmulas se convierten en inconsistentes al eliminar el operador B_i, sólo las fórmulas pertenecientes al agente preferido se reflejan al nivel del SMA. Por ejemplo, si tenemos a un agente a_1 tal que $M_l, s_l \models B_l(\alpha)$ y un agente a_y tal que $M_y, s_y \models B_y(\sim \alpha)$, y tenemos el orden de prioridad $a_y \prec a_l$, luego tenemos $M', s' \mid\cong D \sim \alpha$. Esto requiere de un orden total entre los agentes del SMA. Sin embargo podemos encontrar fácilmente organizaciones de agentes en las cuales definir un orden total sea imposible o muy complicado.

2. *Definir prioridades dentro de la semántica del lenguaje* (sin tener que cambiar la representación sintáctica). La opción de tratar de resolver los conflictos con la técnica anterior presupone que alguien (el diseñador del SMA) conoce *a priori* la prioridad de los agentes. Esto puede ser no realista en algunas aplicaciones. Por ejemplo consideremos el caso de especificidad en el razonamiento por defecto. En este caso tenemos reglas de actuación (que se traducen en creencias de agentes) pero algunas reglas se aplican en casos más específicos que otras. Ejemplo, una regla es aplicable en el caso que se cumple p para inferir la creencia plausible que se cumple q. Pero, por otra parte, podríamos tener una regla que diga que en la situación $p \wedge q$ es posible que se cumpla $\sim q$. Ahora podemos derivar ambas creencias $B_1 q$ y $B_2 \sim q$ gracias a las reglas respectivas, pero muchos estarán de acuerdo en que, dado $p \wedge r$, $(B_2)\sim q$ debería tener precedencia cuando se refleja hacia arriba.

3. *Utilizar modos para indicar directamente la fuerza/debilidad relativas de creencias posibles.* En este enfoque el conflicto entre una creecnia $B_i\varphi$ y una creencia $B_j\varphi$ puede ser solucionado considerando la fuerza del operador B_i *versus* la del B_j. Una manera de hacer esto es representando cláusulas que involucran a las creencias de los agentes en tal manera que su prioridad relativa esté programada en la representación donde la firmeza mutua de los operadores modales sean utilizados para resolver los conflictos. También se podría utilizar *modos numéricos* tal como *modos*

B_i *calificados* [Van der Hoek y Meyer, 1991], para indicar directamente el grado de confianza en la creencia en cuestión. Este operador calificado B_i puede ser tanto relativo como absoluto.

La reflexión hacia arriba, no es realmente parte de nuestra lógica. Es más un tipo de procedimiento o algoritmo que puede ser aplicado a un conjunto de creencias del SMA que son obtenidas utilizando nuestra lógica.

4.2.3. Comportamiento de un Agente Abstracto

En esta sección presentamos las extensiones para el Agente Abstracto de nivel de recursión n, de las definiciones de la sección anterior.

Las extensiones de las definiciones presentadas en la sección anterior para el SMA a las correspondientes para un Agente Abstracto de nivel de recursión $n > 1$ son directas. Simplemente debemos considerar a todo Agente Abstracto de nivel $n \leq 1$ como un agente simple. Un Agente Abstracto de nivel de recursión n se denota por $A_{m,n}$, donde m denota nivel de abstracción y n nivel de recursión.

4.2.3.1. Percepciones

Un Agente Abstracto de nivel de recursión $n > 0$ tiene el siguiente conjunto de percepciones:

Definición 4.17.

$$P_{A_{m,n}} = \begin{cases} P_A & Si\ n\ =\ 1, \\ \bigcup\limits_{A_{x,j} \in A_{m,n}} P_{A_{x,j}} & Si\ n\ >\ 1. \end{cases}$$

4.2.3.2. Acciones

Las acciones de un Agente Abstracto de nivel de recursión $n > 0$ las definimos como:

Definición 4.18.

$$\Delta_{A_{m,n}} = \begin{cases} \Delta_A & Si\ n\ =\ 1, \\ \bigcup\limits_{A_{x,j} \in A_{m,n}} \Delta_{A_{x,j}} & Si\ n\ >\ 1. \end{cases}$$

4.2.3.3. Objetivos

Los objetivos de un Agente Abstracto de nivel de recursión $n > 0$ se definen de manera operacional como sigue: Si $n = 1$ aplicar la definición 4.12. Si $n > 1$ construir $OS_{A_{m,n}}$ aplicando la función f sobre las instancias de secuencias de acciones obtenidas a partir de observaciones sobre patrones de relación entre los agentes $A_{x,n-1}$ para todo $A_{x,n-1} \in A_{m,n}$.

4.2.3.4. Creencias

- La estructura de información de un Agente Abstracto de nivel de recursión $n > 0$ la definimos de manera recursiva como:

  Definición 4.19.

$$\overline{BS}_{A_{m,n}} = \begin{cases} \overline{BS}_A & Si\ n\ =\ 1, \\ \bigcup\limits_{A_{x,j} \in A_{m,n}} \overline{BS}_{A_{x,j}} & Si\ n\ >\ 1. \end{cases}$$

- El estado de información de un Agente Abstracto de nivel de recursión $n > 0$ es:

  Definición 4.20.

$$\overline{BS}_{A_{m,n}}(t) = \begin{cases} \overline{B}_A(t) & Si\ n\ =\ 1, \\ \overline{D}_{A_{m,n}}(t) \bigcup \overline{I}_{A_{m,n}}(t) \bigcup \overline{E^k}_{A_{m,n}}(t) \bigcup \overline{C}_{A_{m,n}}(t) & Si\ n\ >\ 1. \end{cases}$$

4.3. Conclusiones

En este capítulo hemos presentado la noción de Agente Abstracto. Un Agente Abstracto es una entidad de modelado para Sistemas Multi Agentes de gran escala. La definición de Agente Abstracto ofrece una perspectiva funcional y estructural de agente. La perspectiva funcional se basa en la definición de agentes propuesta por Wooldridge y Jennings por la que un agente es una entidad autónoma, social, reactiva y pro-activa [Wooldridge y Jennings, 1995]. Mientras que la perspectiva estructural establece que un Agente Abstracto puede ser un Sistema Multi Agente o un agente simple. Hemos ampliado, de

esta manera, la noción tradicional de Sistemas Multi Agente, al indicar que un Sistema Multi Agente está formado de dos o más Agentes Abstractos pudiendo ser cada uno de ellos a su vez Sistemas Multi Agente o agentes simples. De esta manera podemos modelar: Sistemas Multi Agente en los que sus entidades constituyentes pueden ser a su vez Sistemas Multi Agentes (especialmente adecuado para la integración de Sistemas Multi Agentes pre-existentes), estructuras organizacionales complejas en las que las entidades participantes pueden ser a su vez organizaciones, dominios de aplicación en los que las entidades del dominio presentan características recursivas o de agregado (sistemas biológicos y sociales, Sistemas Holónicos de Fabricación, etc.).

Un Agente Abstracto que actúa en estructuras organizacionales puede encapsular la complejidad de sub-sistemas (simplificando su representación y diseño) y modularizar sus funcionalidades. De esta manera podemos trabajar en distintos niveles de abstracción, ocultando los detalles de modelado de las entidades constituyentes para poder centrar el análisis en las interacciones que se dan en el nivel determinado. Que un Agente Abstracto participe en interacciones implica que debe exhibir un comportamiento que debe estar modelado de alguna manera. Es por ello que para completar la definición de Agente Abstracto en este capítulo hemos presentado la definición formal de su comportamiento. El comportamiento reactivo del Agente Abstracto lo hemos definido en término de las acciones y las percepciones de sus agentes constituyentes. Mientras que su comportamiento intencional lo hemos definido en término de objetivos y creencias basándonos en el principio de racionalidad y la arquitectura BDI. La definición del comportamiento hacia "afuera" de un SMA, en término de acciones o servicios, entidades del entorno que puede percibir y objetivos que puede satisfacer, constituye la especificación de la interfaz del SMA. Con este enfoque es posible "conectar" SMAs mediante la especificación de sus interfaces y avanzar hacia un desarrollo de SMAs escalable, modular y por capas.

Nuestra propuesta es similar a la de [Occello, 2000] en cuanto que un SMA puede ser visto como un agente, pero se diferencia del trabajo de Occello en varios puntos: (i) el Agente Abstracto es una entidad de modelado y como tal no existe en tiempo de ejecución; (ii) en lugar de utilizar una función

de interacción para definir el comportamiento del Agente Abstracto nosotros lo hacemos mediante el comportamiento reactivo y pro-activo de sus agentes constituyentes; (iii) proponemos definiciones operacionales para definir el comportamiento de un nivel de recursión en término de otros niveles (punto que en el trabajo de Occello no fue abordado). La propuesta del Comité Técnico de FIPA dedicado al Modelado (*FIPA Modelling TC*) [Odell *et al.*, 2005; FIPA Modeling TC, 2005] es muy parecida a nuestro enfoque pero para el momento de la escritura de esta tesis el metamodelo de FIPA se encontraba aun incompleto y en una versión preliminar y por tanto no hemos podido incluir ninguna comparativa formal con nuestro trabajo. Pero nos parece muy importante destacar que uno de los aportes de esta tesis está siendo estudiado de manera muy similar por la organización más importante para la estandarización de la tecnología de agentes.

Es importante destacar, antes de terminar con este capítulo, que el modelado recursivo en el contexto de nuestro trabajo es diferente a los trabajos presentados en [Gmytrasiewics y Durfee, 1995] y en [Tambe, 1995]. Gmytrasiewics y Durfee proponen un Método de Modelado Recursivo como un marco teórico para representar y utilizar el conocimiento que un agente tiene acerca de sus beneficios esperados y el de otros, dada una combinación de acciones elegida por todos los agentes. Por otra parte, Tambe propone un enfoque para los modelos que un agente mantiene acerca de los comportamientos de otros agentes. Esta propuesta consiste en la combinación de características arquitectónicas que permiten a un agente generar comportamientos de otros agentes.

Capítulo 5

Metodología Multi Agente para Sistemas Holónicos de Fabricación

En este capítulo presentamos una metodología Multi Agente para el desarrollo de Sistemas Holónicos de Fabricación. La motivación de este trabajo radica en la necesidad de contar con metodologías específicas para HMS basadas en principios de la ingeniería del software, que asistan al diseñador en todos las fases de desarrollo y que provean guías específicas, claras y no-ambiguas.

El presente Capítulo se desarrolla a partir de los requisitos para metodologías HMS, identificados en el Capítulo 2, que han guiado el análisis de metodologías referenciadas en la literatura especializada en las áreas: HMS, SMA y Modelado de Empresas, presentado en el Capítulo 2 y resumido en la Tabla 2.1 del mismo capítulo. Las similitudes entre los enfoques holónico y agente presentadas en el Capítulo 3, a partir de las cuales definimos la noción de Agente Abstracto (Capítulo 4) como entidad de modelado de entidades autónomas con estructuras recursivas.

Como hemos indicado en capítulos anteriores, un sistema de fabricación es un sistema complejo y de gran escala. Incorpora una serie de niveles conceptuales y de organización ligados con las características propias de la empresa de fabricación que la implementa. En el área de Ingeniería del Software se han identificado tres técnicas fundamentales para tratar con la complejidad de

sistemas de gran escala [Booch, 1994]:

- **Descomposición**: técnica "divide y vencerás".
- **Abstracción**: encapsulamiento y ocultación de los detalles poco importantes mediante la definición de "bloques" de modelado que enfatizan unos pocos detalles importantes y eliminan otros. Cuando el modelo se refina gradualmente, las abstracciones asociadas con los componentes específicos del modelo pueden ser cambiadas.
- **Organización**: proceso de identificación y gestión de las interacciones de los componentes complejos.

Para que la ingeniería del software basada en agentes se pueda aplicar a Sistemas Holónicos de Fabricación, es necesario que exista una infraestructura disponible para que el ingeniero del software pueda emplear los conceptos de la tecnología de agente en el proceso de desarrollo aplicando las técnicas de descomposición, abstracción y organización. Nuestra propuesta aborda la descomposición y abstracción mediante la noción de Agente Abstracto y un proceso de desarrollo guiado por capas de abstracción. Al mismo tiempo ofrece una serie de modelos y guías de desarrollo específicas para el dominio HMS que permiten especificar cómo se estructuran, relacionan e interaccionan estas entidades.

El presente capítulo se organiza de la siguiente manera. En la Sección 5.1 señalamos las ideas que nos motivaron a seleccionar INGENIAS [Gómez, 2002] y RT-MESSAGE [Julián, 2002] como metodologías de partida. En la Sección 5.2 presentamos la notación de nuestra metodología. En la Sección 5.3 definimos el proceso de desarrollo. Finalmente en la Sección 5.4 presentamos las conclusiones de este capítulo.

5.1. Antecedentes

El estudio y trabajos presentados en los capítulos anteriores justifican la aplicación de la tecnología SMA para el desarrollo de Sistemas Holónicos de Fabricación. Mas aún, los desarrollos del área HMS (Capítulo 2), que han utilizado con éxito la tecnología agente como herramienta de implementación,

constituyen evidencias empíricas de que esta tecnología es apropiada para este dominio. Además, las similitudes entre ambos enfoques (Capítulo 3) nos llevan a asegurar que hoy en día la tecnología SMA es la más apropiada para el desarrollo de sistemas holónicos. No obstante, hace falta una infraestructura que proporcione uniformidad de conceptos desde la concepción de la idea hasta su implementación y, sobre todo, que ofrezca un proceso de desarrollo específico del dominio HMS, que guíe al ingeniero del software en todas las etapas del ciclo de vida del sistema. Nuestro propuesta trata de satisfacer estos objetivos.

No pretendemos definir una metodología desde cero, sino aprovechar trabajos previos reportados en el área SMA para adaptarlos al dominio HMS. La metodología que proponemos se basa en metodologías SMA completas que han probado su eficacia en dominios generales. El resumen del análisis comparativo que hemos presentado en la Tabla 2.1 del Capítulo 2 representa el grado de adecuación de distintas metodologías a los requisitos de modelado para HMS (ver Tabla 5.1 para un resumen de los requisitos presentados en el Capítulo 2). De esta tabla y del análisis previo realizado destacamos INGENIAS [Gómez, 2002] y RT-MESSAGE [Julián, 2002]. Ambas metodologías son extensiones de la metodología MESSAGE [EURESCOM, 2000; 2001b] y por tanto tienen la misma base en cuanto a modelos conceptuales para SMA.

Entre las características que nos han motivado a seleccionarlas como metodologías de partida citamos las siguientes:

- La adecuación de INGENIAS y RT-MESSAGE a los requisitos 1, 3 y 4.
- La adecuación de RT-MESSAGE al requisito 2, para la identificación y modelado de características de tiempo real.
- INGENIAS es una metodología completa que soporta todo el ciclo de vida de un SMA. RT-MESSAGE se centra en la fase de diseño detallado para dominios con características de tiempo real.
- INGENIAS y RT-MESSAGE emplean modelos conceptuales para el modelado de SMAs aceptados ampliamente en la literatura especializada.

Tabla 5.1: Requisitos de modelado para HMS

Requisito	**Descripción**
Requisito 1	Los sistemas de control de fabricación requieren entidades autónomas que se organizan en estructuras que son a la vez jerárquicas y heterárquicas.
Requisito 2	Las unidades de control de fabricación principalmente requieren de un comportamiento basado en rutinas que es al mismo tiempo efectivo y con características de tiempo real. Este comportamiento puede ser tanto configurable o auto-adaptativo.
Requisito 3	Los métodos de programación deben proveer encapsulación de datos y procesos.
Requisito 4	Los programas de control deben tener una semántica clara.
Requisito 5	Un método o metodología de desarrollo para sistemas de fabricación holónicos debe proveer una mecanismo de traducción de tareas de control, sobre un recurso de fábrica, o de funcionalidades de la fábrica a entidades con comportamiento autónomo [HMS, 1994].
Requisito 6	Un método o metodología de desarrollo para sistemas de fabricación holónicos debe ofrecer al ingeniero estructuras y mecanismos que permitan llevar a cabo procesos de análisis y diseño por capas de abstracción.
Requisito 7	Un método o metodología de desarrollo para sistemas de fabricación holónicos debe definir un proceso de desarrollo mixto que integra los enfoques top-down y bottom-up.
Requisito 8	Un método o metodología de desarrollo para sistemas de fabricación holónicos debe integrar el rango completo de actividades de producción, desde la recepción de una orden pasando por el diseño, la fabricación, y el mercadeo para lograr la empresa de fabricación ágil [HMS, 1994].

- INGENIAS ofrece una notación definida mediante meta-modelos que facilita su ampliación y la verificación de los modelos generados a partir de ellos.

En las siguientes secciones de este capítulo presentamos nuestra propuesta. La notación de nuestra metodología (Sección 5.2) se basa en los modelos de INGENIAS. Para ello presentamos las extensiones que hemos realizado a los meta-modelos de INGENIAS para incorporar la noción de Agente Abstracto, nuevas entidades para modelar su estructura y comportamiento, características de tiempo real definidas en RT-MESSAGE, y relaciones nuevas y/o modificadas. Finalmente, en la Sección 5.3 presentamos el proceso de desarrollo que se basa en los requisitos de la tabla 5.1.

5.2. Notación

La notación de nuestra metodología se basa fundamentalmente en los modelos de INGENIAS, las características de tiempo real para SMA identificadas en RT-MESSAGE y la noción de Agente Abstracto para el modelado de holones. En esta sección presentamos las ampliaciones que hemos incluido a los modelos de INGENIAS para adaptarlos a la noción de Agentes Abstracto y las características de tiempo real de RT-MESSAGE.

La especificación de los modelos en INGENIAS se realiza mediante meta-modelos, que facilitan su ampliación y verificación. En este trabajo seguimos con esta idea y definimos los modelos de nuestra metodología mediante extensiones a los meta-modelos de INGENIAS. Es importante resaltar, que las extensiones que proponemos son generales (no dependientes de HMS). De esta manera los modelos que proponemos pueden ser útiles además para el modelado de SMA de gran escala en dominios generales.

Para comprender los meta-modelos presentamos a continuación los fundamentos del meta-modelado. Posteriormente definimos los meta-modelos de nuestro enfoque.

5.2.1. Meta-modelo

El meta-modelado es un mecanismo que permite definir formalmente lenguajes de modelado. Así, un meta-modelo de un lenguaje es una definición precisa de sus elementos mediante conceptos y reglas de cierto meta-lenguaje, necesaria para crear modelos en ese lenguaje. Un meta-modelo define las primitivas y las propiedades sintácticas y semánticas de un modelo. Básicamente se trata de usar modelos para describir otros modelos. Por ejemplo, el meta-modelo de UML define los conceptos y reglas que se necesitan para crear modelos UML.

OMG define una arquitectura de cuatro niveles para sus estándares de meta-modelado. En la terminología de OMG estas capas se llaman M0, M1, M2 y M3. La Figura 5.1 ilustra los distintos niveles de la arquitectura definida por OMG tanto para el meta-modelado de modelos, parte derecha de la figura, como para el meta-modelado de procesos de ingeniería del software, parte

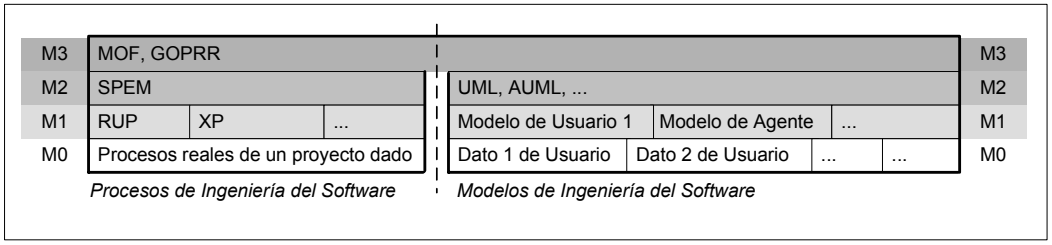


Figura 5.1: Estructura de cuatro niveles utilizada en el meta-modelado.

izquierda (en la sección 5.3.1 se presenta el meta-modelado de procesos). En el nivel M0 están todas las instancias "reales" del sistema, es decir, las entidades físicas que existen en la aplicación. Por ejemplo,

Creencia(puertaA,cerrada).

Por encima de la capa M0 se sitúa la capa M1, que representa el modelo de un sistema software. Los conceptos del nivel M1 representan categorías de las instancias de M0. Por ejemplo,

Clase("Creencia", Atributo("Puerta", String), Valor("Estado","String)).

Análogamente a como ocurría con M0 y M1, los elementos del nivel M1 son a su vez instancias del nivel M2. Esta capa recibe el nombre de capa de meta-modelado. En esta capa aparecerán conceptos como *Clase*, *Atributo*, *Agente*, *Objetivo* o *Creencia* para definir lenguajes como UML, AUML, etc. Por ejemplo,

MetaClase("Clase", MetaAtributo("Nombre", String), MetaAtributo("Atributos", List<"Atributo">))

MetaClase("Atributo",...).

De igual manera, podemos ver los elementos de M2 como instancias de otra capa superior M3, la capa de meta-metamodelado. Para esta capa los lenguajes considerados son MOF [OMG, 2004] y GOPRR [Lyytinen y Rossi, 1999]. OMG

ha definido a la capa M3 de manera reflexiva, es decir, en lugar de definir una capa M4, se estableció que todos los elementos de M3 se pueden definir con instancias de conceptos de M3.

Este tipo de notación facilita el desarrollo de sistemas. Es la base para alinear el meta-modelo de un lenguaje particular con otros lenguajes. Las especificaciones de nivel M0 son lo suficientemente estructuradas para ser procesadas de forma automática, ya sea para verificación de consistencia de las construcciones o para la generación automática de documentación y de código. Además, ofrece el mecanismo estructural para futuras extensiones del meta-modelo. Permite extender de manera sencilla los modelos, incorporando más elementos a los meta-modelos. Esta característica se demuestra en este capítulo al redefinir los modelos de INGENIAS extendiendo sus meta-modelos con el fin de incorporar la noción de Agente Abstracto.

Al igual que en INGENIAS [Gómez, 2002], el lenguaje que utilizamos para la definición de los meta-modelos es la notación UML siguiendo las restricciones indicadas en GOPRR (Graph, Object, Property, Relationship, and Role) [Lyytinen y Rossi, 1999]. Mantenemos la misma notación para facilitar la comprensión y comparación de los meta-modelos extendidos en relación a los originales. Además, este enfoque permite expresar las extensiones y modificaciones de manera simple y completa.

5.2.2. Meta-modelos del Sistema Multi Agente

En esta sección presentamos las extensiones que hemos realizado a los meta-modelos de INGENIAS [Gómez y Pavón, 2002; Pavón y Gómez, 2003; Gómez, 2002]. Las extensiones tienen que ver con la inclusión del Agente Abstracto como entidad de modelado, así como de otras entidades abstractas y sus relaciones para representar la estructura del Agente Abstracto y las consiguientes modificaciones de las relaciones entre estas nuevas entidades y las entidades pre-existentes en INGENIAS. Además, hemos integrado los elementos identificados en [Julián, 2002] para el modelado de características de tiempo real en sistemas multi-agentes.

Antes de presentar los distintos meta-modelos es importante recordar la nomenclatura utilizada en INGENIAS para los distintos diagramas. Repetimos,

por tanto, las reglas nemotécnicas (ver Figura 5.2) definidas en INGENIAS para facilitar la lectura de los meta-modelos:

- El nombre de la relación es precedido por un conjunto de letras que denotan su procedencia. Por ejemplo, se utiliza (**WF**) para denotar flujo de trabajo, la (**A**) representa el meta-modelo de agentes, la (**I**) interacción, una unidad de interacción se representa por (**UI**), el meta-modelo de tareas por (**GT**), se utiliza (**AGO**) para las relaciones sociales, la (**O**) representa organización, la (**E**) al meta-modelo de entorno.

- Por cada asociación entre una entidad *relationship* y un *object* existe una instancia de la primitiva GOPRR *Role*, nombrándose ésta con el mismo nombre que la relación pero antecedido por la letra **R** y terminado en una letra **D** u **O**, para indicar el sentido de la relación (**D** de Destino, **O** de Origen).

Para facilitar la lectura de los distintos diagramas que se presentan en este capítulo, los elementos gráficos que constituyen una extensión o un aporte a los meta-modelos originales de INGENIAS se presentan en color gris. El color gris claro representa redefiniciones de relaciones en la clase hija.

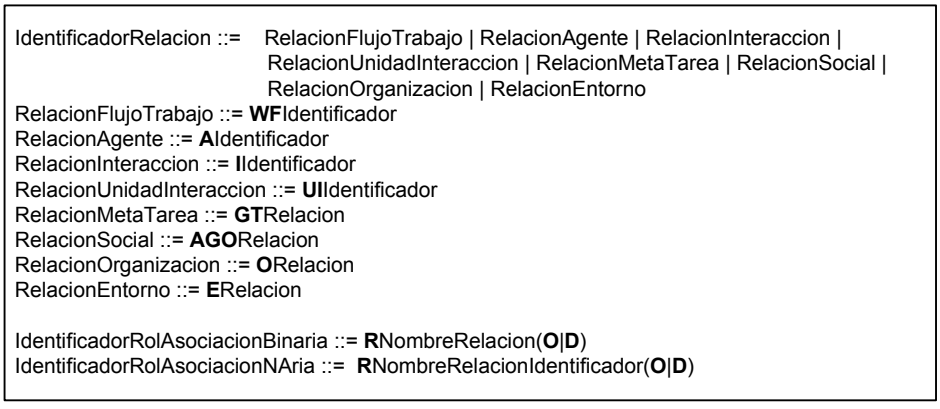


Figura obtenida de [Gómez, 2002]

Figura 5.2: Expresión BNF para los nombres de las relaciones y de los roles.

5.2.2.1. Entidades Básicas

Las entidades básicas de modelado de nuestro enfoque se presentan en la Figura 5.3. Estas entidades se explican en detalle a lo largo de la descripción de los distintos meta-modelos.

Como se puede notar en la Figura 5.3 la Entidad Autónoma de INGENIAS ha sido reemplazada por el A-Agente (Agente Abstracto). En INGENIAS el objeto Entidad Autónoma se utiliza para especificar que tanto el Agente como la Organización tienen identidad y propósitos. En cambio, gracias al concepto de Agente Abstracto una organización puede actuar como un agente y por tanto posee características de agencia. Para incluirlo como entidad básica de modelado teníamos dos posibilidades. La primera, consistía en mantener la Entidad Autónoma y hacer que el A-Agente sea una especialización de ésta y en consecuencia Agente y Organización colgaran de A-Agente. Y la segunda (la adoptada) en la que se sustituye a la Entidad Autónoma por el A-Agente, dado que Agente y Organización poseen características de agencia. De esta manera evitamos mantener una entidad más (lo que implica un nivel más) dentro de la jerarquía de herencia.

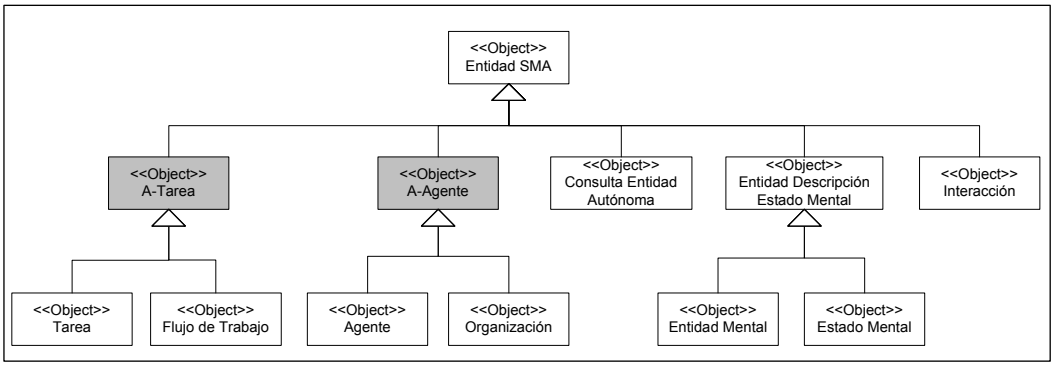


Figura 5.3: Entidades básicas.

En la Figura 5.3 se pueden apreciar algunas de las entidades abstractas que hemos incluido. Por ejemplo: la entidad A-Agente y la entidad A-Tarea que representa una tarea abstracta que puede ser ejecutada por un A-Agente. En

la sección 5.2.2.2 se explicará cómo se representan las nociones mentales de los A-Agentes utilizando Entidad Mental y Estado Mental.

Los meta-modelos extendidos mantienen todas las entidades y relaciones definidas en INGENIAS (excepto la Entidad Autónoma sustituida por A-Agente como ya se ha explicado). Es decir que cualquier modelo desarrollado con INGENIAS es un modelo válido para nuestro enfoque.

5.2.2.2. Meta-modelo de Agente

En INGENIAS el modelo de agente se usa para describir agentes particulares excluyendo las interacciones con otros agentes. Este modelo se centra en la funcionalidad del agente y en el diseño de su control. En este sentido, proporciona información acerca de responsabilidades del agente (asociación de roles, tareas que sabe ejecutar y objetivos que se compromete a alcanzar), y comportamiento del agente (mediante qué mecanismos se va a asegurar la ejecución de las tareas dentro de los parámetros acordados, especificación de estado mental y su evolución).

En nuestro enfoque en el modelo de agente además de poder modelar la autonomía, inteligencia y conceptualización del agente, se podrá hacer lo mismo para el Agente Abstracto, pero con algunas diferencias (los detalles los presentamos en esta sección y en las siguientes). Hemos optado por extender el meta-modelo de agente para el modelado de Agentes Abstractos, en lugar de definir un nuevo modelo para Agentes Abstractos, ya que: (i) el modelo de agente ofrece todos los elementos conceptuales necesarios para modelar al agente, (ii) conceptualmente un Agente Abstracto posee las mismas características de agencia que un agente convencional, y (iii) las extensiones y modificaciones se integran fácilmente en las entidades del meta-modelo de INGENIAS. Además, de esta manera se mantiene uniformidad con los modelos de INGENIAS, y se facilita el trabajo del diseñador, especialmente para las primeras etapas de análisis cuando puede ser difícil (o muchas veces arriesgado) decidir si el Agente Abstracto será simple o compuesto (grupo de agentes).

La definición sencilla y estructurada del meta-modelo de agente de INGENIAS permite que su extensión sea relativamente sencilla. No sólo, en este caso, para incluir la noción de Agente Abstracto y sus correspondientes entidades

y relaciones abstractas sino para cualquier otro tipo de extensión. Una de las ventajas principales del modelo de agente de INGENIAS para la incorporación del Agente Abstracto es la separación de la especificación de la responsabilidad del agente de la especificación de su control. La conceptualización del Agente Abstracto será similar a la de un agente simple, asignación de roles, tareas y objetivos, mientras que la especificación de su control se hará de manera incremental con los distintos modelos de análisis y diseño.

El A-Agente es una estructura abstracta y compleja. En un nivel de abstracción determinado podrá representar a un agente, es decir, para sus usuarios (personas, otros agentes o grupos de agentes) éste será una entidad con comportamiento de agente, independientemente de cual sea su estructura interna. El modelo de agente debe por tanto permitir al analista/diseñador especificar su funcionalidad y comportamiento. Por otra parte cuando el A-Agente es observado desde dentro, el analista/diseñador deberá decidir (según los requisitos del sistema que está modelando) cómo implementar el A-Agente. Si la funcionalidad que define el comportamiento del A-Agente puede ser implementada por un único agente simple, entonces el A-Agente será este agente y todo su comportamiento podrá ser modelado de la manera convencional, como en INGENIAS. En cambio, si el comportamiento del A-Agente debe ser implementado por un grupo de agentes (debido a los requisitos del dominio de aplicación o simplemente porque es decisión del diseñador) entonces el A-Agente en su estructura interna estará compuesto por un grupo de agentes. En este caso, la estructura interna del A-Agente, la estructura organizativa que lo implementa, será especificada en el modelo de organización. Mientras que el comportamiento de ese grupo de agentes, que implementa el comportamiento del A-Agente, será especificado por el conjunto de modelos de interacción, modelo de tareas y objetivos y modelo de entorno.

En la Figura 5.4 se puede observar un fragmento del meta-modelo de agente. En él podemos ver las entidades A-Agente, Agente, Rol, A-Objetivo, Objetivo y Objetivo de Grupo, además de un conjunto de asociaciones que relacionan a las entidades. La relación más importante que aparece en esta figura es la relación Juega que asocia a un A-Agente con el Rol que puede jugar. Esta relación es fundamental en nuestro enfoque y extiende todos los modelos de INGENIAS

de una manera radical. La importancia de esta relación radica en el hecho que un A-Agente puede ser un Agente o una Organización (ver Figura 5.3), y por la relación Juega y el principio de herencia una Organización puede jugar un Rol. Que una Organización juegue un Rol incorpora una serie de elementos nuevos: una organización será responsable de las tareas asignadas a ese rol, una organización tendrá las capacidades definidas en ese rol, una organización perseguirá los objetivos definidos para el rol, etc. Pero fundamentalmente una Organización podrá actuar dentro de otra Organización (mayor) interpretando un papel determinado. Este último punto define una recursión y amplía el meta-modelo de INGENIAS para permitir especificar estructuras organizativas complejas - organizaciones que pueden a su vez estar formadas por organizaciones (ver Sección 5.2.2.6) y no sólo por agentes o grupos de agentes como se permitía antes. Más aun, se podrán especificar interacciones entre entidades de distinto tipo de complejidad (agente - agente, agente - organización, organización - organización) (ver Sección 5.2.2.4).

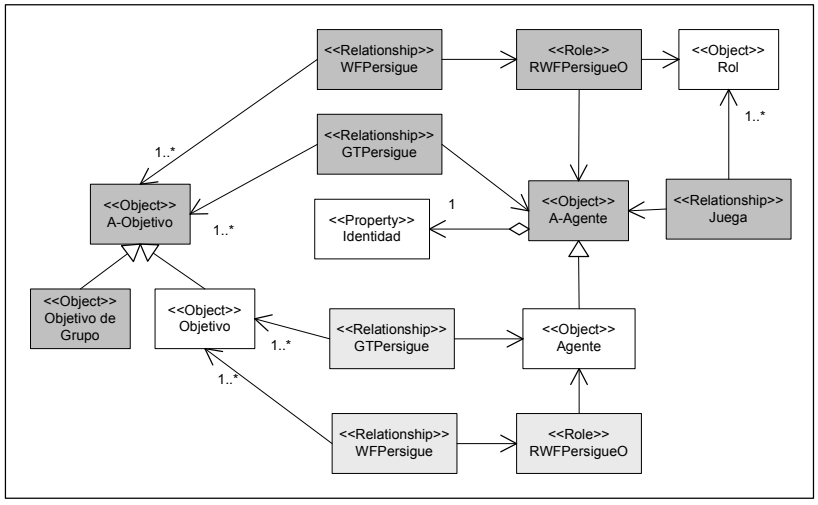


Figura 5.4: Rol, Agente Abstracto y Objetivo.

Como hemos discutido en el capítulo 4 la definición del comportamiento

reactivo y deliberativo de un Agente Abstracto cuando internamente está compuesto por un grupo de agentes es complejo. Por tanto la asignación de tareas de las que es responsable y de entidades mentales que posee debe ser distinta al caso del agente simple. Es por ello que hemos extendido el meta-modelo de INGENIAS con entidades abstractas para tareas y para entidades mentales y las correspondientes relaciones de asociación para permitir modelar las responsabilidades y comportamiento del A-Agente. En la Figura 5.4 podemos ver una de estas entidades abstractas el A-Objetivo. Un A-Objetivo es una generalización de: Objetivo, entidad que representa un objetivo que puede ser perseguido por un Agente, relaciones WFPersigue y GTPersigue; y Objetivo de Grupo, entidad que representa un objetivo que emerge como objetivo del grupo de agentes que forman al A-Agente y por tanto no puede ser asignado a ninguno de los agentes componentes. El objetivo de grupo puede ser: un objetivo totalmente nuevo y en consecuencia diferente a cualquier objetivo de cualquiera de los agentes miembro del grupo; o un objetivo que se construye por conjunción y/o disjunción de los objetivos de los miembros. Como ejemplo de Objetivo de Grupo del primer caso supongamos dos agentes, a_1 con el objetivo o_1 = *producir a una taza de 3 min* y a_2 con el objetivo o_2 = *vender el producto 1 min después de que haya sido fabricado*. El objetivo que emerge como objetivo de grupo es o_{grupo} = *liderar el mercado de productos fabricados*, ni a_1 ni a_2 tienen o_{grupo} como objetivo propio. Un ejemplo para el segundo caso es o_{grupo} = *producir a una tasa de 3 min o vender el producto 1 min después de que* a_1 *lo haya fabricado* ($o_{grupo} = o_1 \vee o_2$).

Las relaciones WFPersigue y GTPersigue asocian al A-Agente con el A-Objetivo que puede perseguir. Pero por herencia estas relaciones implican que un Agente también podría perseguir Objetivos de Grupo. Para evitar este problema, redefinimos estas relaciones en la clase hija y restringimos el destino de las relaciones a la entidad Objetivo.

Definir un objetivo como Objetivo de Grupo sirve para indicar que en pasos posteriores de análisis y diseño éste deberá ser descompuesto o redefinido en términos de otros objetivos más simples (la descomposición de un Objetivo de Grupo se explica en la sección 5.2.2.3). Además, que un A-Agente persiga un Objetivo de Grupo implica que en su estructura interna deberá ser una

organización. Más aún, un Rol que persigue un Objetivo de Grupo no podrá ser asignado a un Agente sólo a una organización.

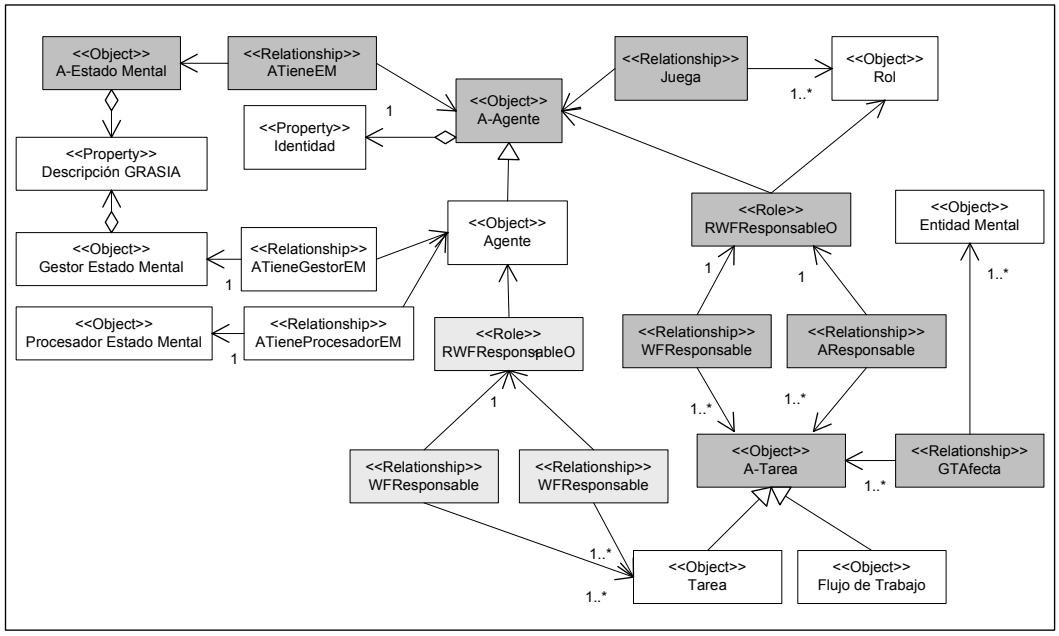


Figura 5.5: Rol, Agente Abstracto, Tarea y Estado Mental.

La capacidad o habilidad de un A-Agente se expresa asociándole las tareas que sabe ejecutar. Hemos incluido la entidad A-Tarea (ver Figura 5.5) para representar tanto las Tareas de las que puede ser responsable un Agente, como las tareas compuestas, Flujo de Trabajo, que se pueden asociar como habilidad a un conjunto de agentes. Un Flujo de Trabajo es un conjunto de tareas que son llevadas a cabo en un determinado orden y en cuya ejecución pueden estar involucrados varios agentes (en la Sección 5.2.2.3 presentamos con mayor detalle los Flujos de Trabajo). La relación WFResponsable permite asociar una A-Tarea tanto a un A-Agente como a un Rol. Para el caso del Agente hemos redefinido WFResponsable, limitando el destino de la relación a Tarea.

El comportamiento del agente se expresa modelando la evolución de su estado mental. El agente actúa sobre su entorno ejecutando tareas, estas tareas

se ejecutan en respuesta a: (i) algún interés propio - sus objetivos, y (ii) el estado del mundo que le rodea - modelado por su estado mental. La ejecución de una tarea puede afectar alguna entidad mental del agente. Visto desde fuera un Agente Abstracto tendrá el comportamiento de un agente. En la Figura 5.4 hemos asociado A-Objetivos, y en el párrafo anterior hemos discutido sobre las A-Tareas. La Entidad Mental, aparece en la Figura 5.5 relacionada con A-Tarea, la relación GTAfecta permite modelar la evolución del estado mental del A-Agente como efecto de la ejecución de una A-Tarea. Hemos modificado la relación GTAfecta de INGENIAS para indicar que A-Tarea puede por ejemplo crear o destruir A-Objetivos. La relación ATieneEM asocia a un A-Agente un A-Estado Mental. Se completa la evolución del estado mental del A-Agente con el modelo de objetivos y tareas, el modelo de interacción, el modelo de entorno y el modelo de organización.

Para el Gestor de Estado Mental y el Procesador de Estado Mental (ver Figura 5.5) mantenemos el enfoque de INGENIAS. Esto se debe a que los Agentes Abstractos son conceptos de modelado, las entidades que realmente se van a ejecutar son sus agentes constituyentes y por tanto sólo estos pueden tener *gestores* y *procesadores*.

En la Figura 5.6 podemos ver las entidades mentales válidas de nuestro enfoque. Hemos añadido como Entidades Mentales de Control el A-Objetivo y el A-Compromiso y como Entidad Mental de Información la A-Creencia. También podemos ver en la figura la entidad Creencia de Grupo que es una especialización de A-Creencia y a su vez puede estar formado por A-Creencias. La Creencia de Grupo se utiliza para representar las creencias de un A-Agente cuando éste es una organización.

La especificación del estado mental es una herramienta útil para modelar representaciones del mundo tal y como el agente lo concibe en distintos momentos. Se define como una agregación de Entidades Mentales. El A-Estado Mental (ver Figura 5.6) lo hemos definido para representar los estados mentales de un A-Agente relación ATieneEM (Figura 5.5). El Estado Mental de Grupo permite modelar organizaciones con estados mentales, que tal y como lo hemos definido en el capítulo 4 se construye a partir de los estados mentales de los agentes que pertenecen a la organización. La relación AContieneE define al

Estado Mental de Grupo como un compuesto de Estado Mental de Grupo o de Estado Mental. La relación AContieneEM se redefine para la clase hija Estado Mental en la Figura 5.7. Las propiedades Tipo Evento y Periodo, de la entidad Evento, las hemos incluido para representar eventos temporales [Julián, 2002].

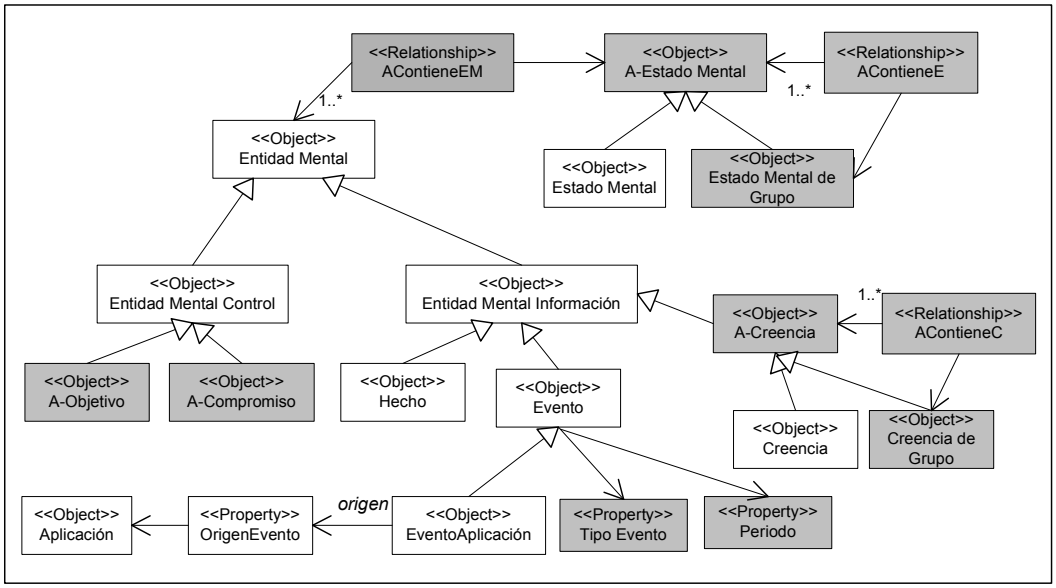


Figura 5.6: Entidades Mentales.

Un agente tiene distintos estados mentales en distintos momentos de su vida. Cuando el agente es creado parte de un estado mental el cual va evolucionando a medida que el agente interacciona con otros agentes o a medida que ejecuta tareas para cumplir determinados objetivos. Para especificar que un A-Agente nace con un determinado A-Estado Mental basta con asociarle ese estado mediante la relación ATieneEM (Figura 5.5). La entidad Consulta Entidad Autónoma se utiliza en INGENIAS para asociar indirectamente un Estado Mental a un Agente. Esta asociación representa momentos concretos de la vida del agente, es decir un estado mental intermedio. Hemos extendido la relación AInstanciaDe (ver Figura 5.8) para permitir que Consulta Entidad Autónoma

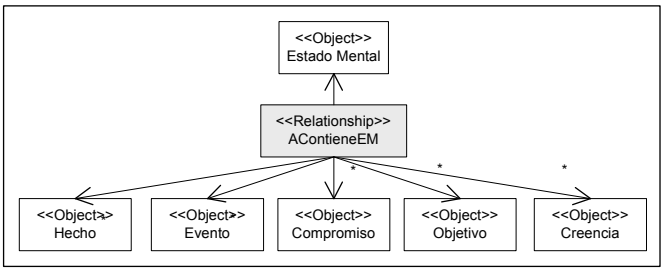


Figura 5.7: Estado Mental y Entidades Mentales.

pueda ser instancia de un A-Agente, de esta manera mediante la relación ATieneEM entre A-Estado Mental y Consulta Entidad Autónoma podemos modelar el estado mental de un A-Agente en distintos momentos de su vida.

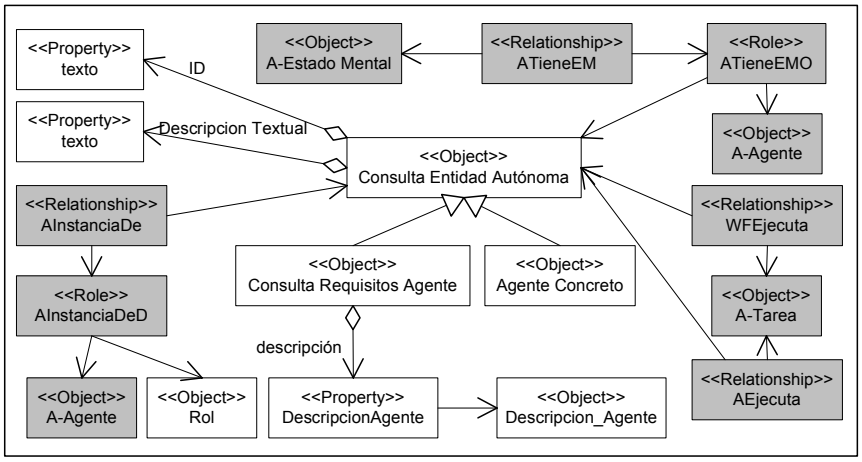


Figura 5.8: Consulta de Entidad Autónoma.

5.2.2.3. Meta-modelo de Tareas y Objetivos

El meta-modelo de Tareas y Objetivos trata con una parte de la especificación del control de alto nivel del agente. En INGENIAS el meta-modelo

de Tareas y Objetivos se construye sobre: tareas, objetivos y relaciones para modelar el control del agente. Como se ha indicado en [Gómez, 2002], la asociación de tareas a objetivos surge del *principio de racionalidad*, las acciones de los agentes se justifican por los objetivos que persigue. Además, la tarea juega un papel fundamental en la evolución del estado mental del agente que es responsable de ella.

En el modelo de agente se pueden asociar, entre otros, A-Objetivos y A-Tareas a A-Agentes para modelar motivaciones y capacidades del A-Agente. Para especificar la parte de control del A-Agente relacionada con A-Tareas y A-Objetivos hemos extendido el meta-modelo de Tareas y Objetivos de INGENIAS para dar respuesta a: cuándo una A-Tarea se puede ejecutar, cuál es la motivación del A-Agente para ejecutar una A-Tarea, cómo afecta una A-Tarea a una Entidad Mental, cómo se puede descomponer un A-Objetivo, cómo un A-Objetivo depende de otro A-Objetivo, etc.

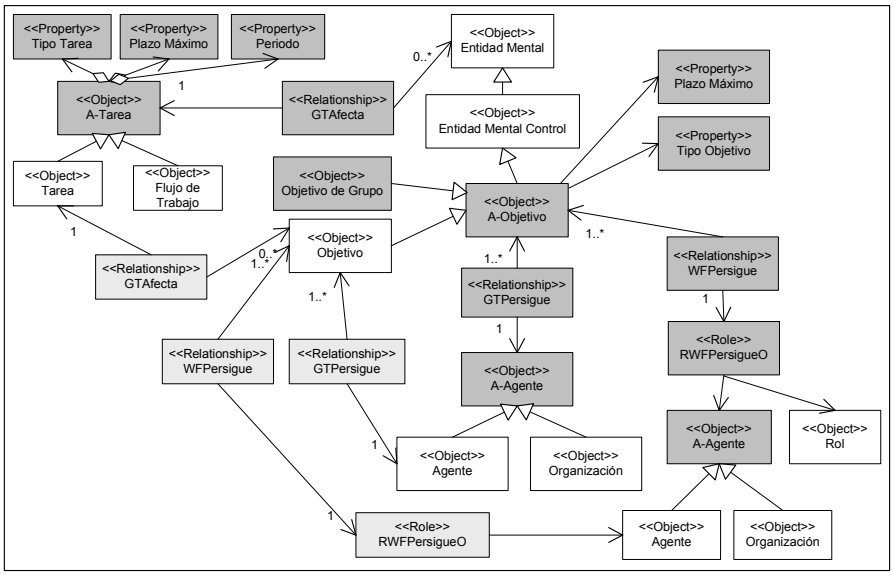


Figura 5.9: Objetivos y Tareas.

En la Figura 5.9 podemos apreciar una parte del meta-modelo de tareas y objetivos de ModA2. Para expresar la motivación de un A-Agente para la

ejecución de una A-Tarea en ModA2 hemos modificado las relaciones GTAfecta, GTPersigue y WFPersigue. Estas modificaciones tienen que ver con las entidades que pueden participar en las relaciones. La relación GTAfecta permite asociar el efecto de una A-Tarea sobre un A-Objetivo. Una A-Tarea puede ser una Tarea o un Flujo de Trabajo(ver Sección 5.2.2.6). De esta manera, por herencia, un Flujo de Trabajo en ModA2, a diferencia que en INGENIAS, puede afectar a un objetivo, así la ejecución de Flujos de Trabajo que aparecen en el contexto de una organización se pueden justificar mediante la asociación de objetivos a la organización, relación GTPersigue. Más aún gracias a las extensiones de ModA2 una Organización se puede asociar por la relación WFPersigue a un objetivo en el marco de un Flujo de Trabajo de, por ejemplo, otra Organización mayor. Para restringir el caso del Agente hemos redefinido las relaciones WFPersigue y GTPersigue en la clase hija de tal manera a prohibir que un Agente pueda perseguir Objetivos de Grupo. De igual manera para restringir la relación GTAfecta para el caso de Tarea hemos redefinido la relación manteniendo la sintaxis de INGENIAS para Tarea y Objetivo. Las propiedades Plazo Máximo, Tipo Tarea y Periodo, asociadas a A-Tarea se utilizan para indicar, si fuera necesario, el máximo tiempo de ejecución para la tarea; si la tarea es crítica, periódica o aperiódica; y para indicar el tiempo entre ocurrencias de la tarea. De igual manera, hemos incluido, las propiedades Plazo Máximo (tiempo máximo para la consecución de un objetivo) y Tipo Objetivo (hard, soft o normal), para permitir modelar características temporales asociadas a tareas y objetivos [Julián, 2002].

Las formas en que una A-Tarea afecta a un A-Objetivo se ilustran en la Figura 5.10. Mantenemos los tipos de relaciones GTModifica, GTDestruye, GTCrea, GTSatisface y GTFalla de INGENIAS pero aplicados a A-Tarea y A-Objetivo (las relaciones se redefinen para las clases hijas Tarea y Objetivo). ¿Qué significa que una A-Tarea afecte a un A-Objetivo?. Para el caso simple, Agentes, Tareas y Objetivos, la interpretación es la de INGENIAS. Cuando trabajamos con el caso complejo, Organización, Flujos de Trabajo y Objetivos de Grupo la interpretación es la siguiente. Cuando una Organización *Org* formada por los agentes {*a1*,*a2*,*a3*}, es responsable de un Flujo de Trabajo *WF1* formado por las tareas {*t1*,*t2*,*t3*}, relación WFDescompone (Figura 5.12), y persigue el Objetivo

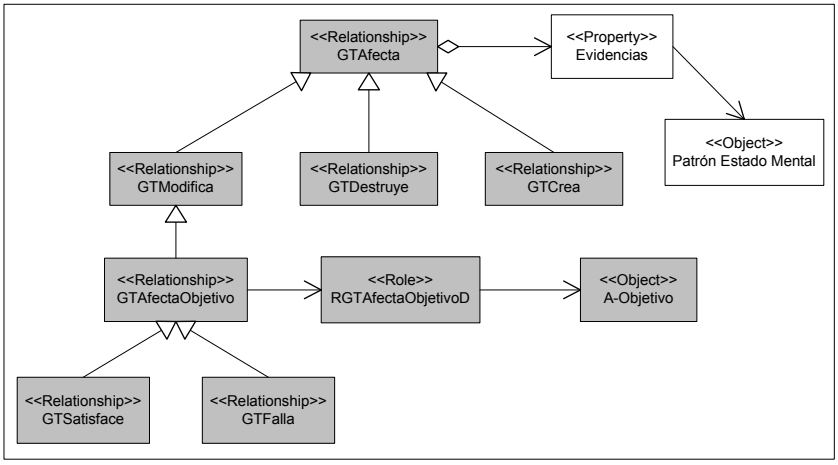


Figura 5.10: La tipos de relación entre Tareas y Objetivos.

de Grupo $O1_{grupo}$ definido como la conjunción de los objetivos $\{o1,o2,o3,o4\}$, relación GTDependeY (Figura 5.11), significa que algunos o todos los agentes de *Org* participarán en la tareas del *WF1* para satisfacer el objetivo $O1_{grupo}$. Esto es, las tareas $\{t1,t2,t3\}$ serán asignadas a agentes de O1, supongamos *a1* y *a3* y habrá una dependencia entre los objetivos $\{o1,o2,o3,o4\}$ y los objetivos de los agentes *a1* y *a3*, supongamos el objetivo $o1_{a1}$ de *a1* y los objetivos $o3_{a3}$ y $o4_{a3}$ del agente *a2*. Además, las Tareas del Flujo de Trabajo afectarán a los objetivos de los agentes participantes, por ejemplo las relaciones *t1* satisface $o1_{a1}$, *t2* satisface $o3_{a3}$ y *t3* satisface $o4_{a3}$.

Un A-Objetivo es una generalización de Objetivo y Objetivo de Grupo (Figura 5.11). El Objetivo de Grupo es la entidad que representa un objetivo que emerge como objetivo del grupo de agentes que forman al A-Agente y por tanto no puede ser asignado a ninguno de los agentes componentes. El objetivo de grupo puede ser: un objetivo totalmente nuevo y en consecuencia diferente a cualquier objetivo de cualquiera de los agentes miembro del grupo; o un objetivo que se construye por conjunción, relación GTDependeY, y/o disjunción, relación GTDependeO, de los objetivos de los miembros. El Objetivo de Grupo se puede descomponer en A-Objetivos, relación GTDescompone, por lo tanto puede estar

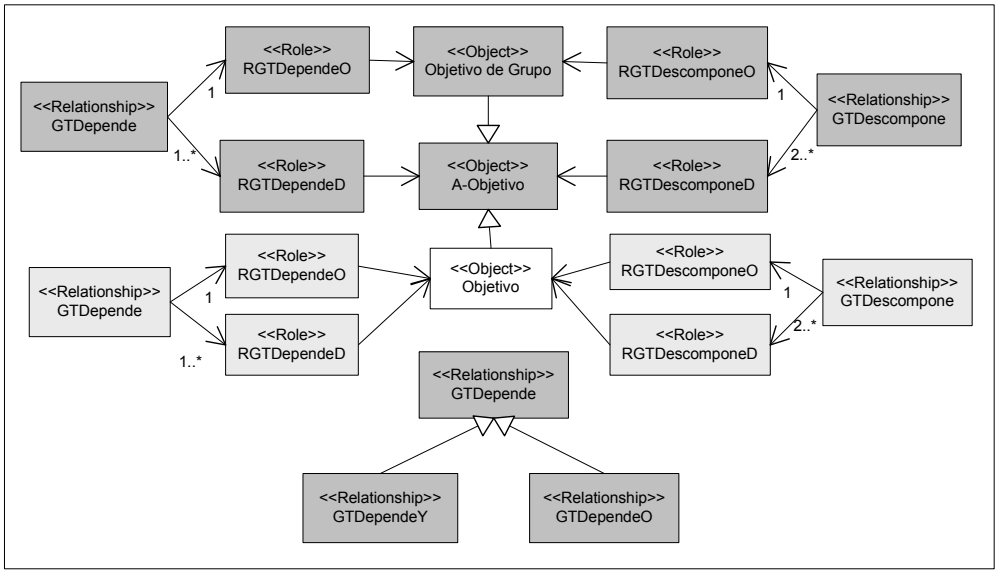


Figura 5.11: Descomposición de Objetivos y dependencia entre Objetivos.

formado por Objetivos y/o Objetivos de Grupo a su vez. Para mantener la consistencia con INGENIAS y restringir el caso de la entidad Objetivo hemos redefinido las relaciones GTDepende y GTDescompone en las clases hijas.

En la Figura 5.12 podemos apreciar que la descripción de una A-Tarea es similar a la descripción de Tarea definida en INGENIAS. Con esta descripción permitimos modelar Flujos de Trabajo que pueden consumir Recursos y Entidades Mentales, relación WFConsume; que pueden usar Aplicaciones, relación WFUsa; o que pueden producir Entidades Mentales, Recursos o Interacciones, relación WFProduce (ver Secciones 5.2.2.6 y 5.2.2.5 para mayor detalle). La descomposición de Flujo de Trabajo en A-Tarea permite anidar Flujos de Trabajo, ya que un Flujo de Trabajo puede descomponerse en Tareas u otros Flujos de Trabajo, relación WFDescompone. La descripción de un Flujo de Trabajo es más compleja que la simple descomposición en Tareas o Flujos de Trabajo, la definición completa la presentamos en la Sección 5.2.2.6.

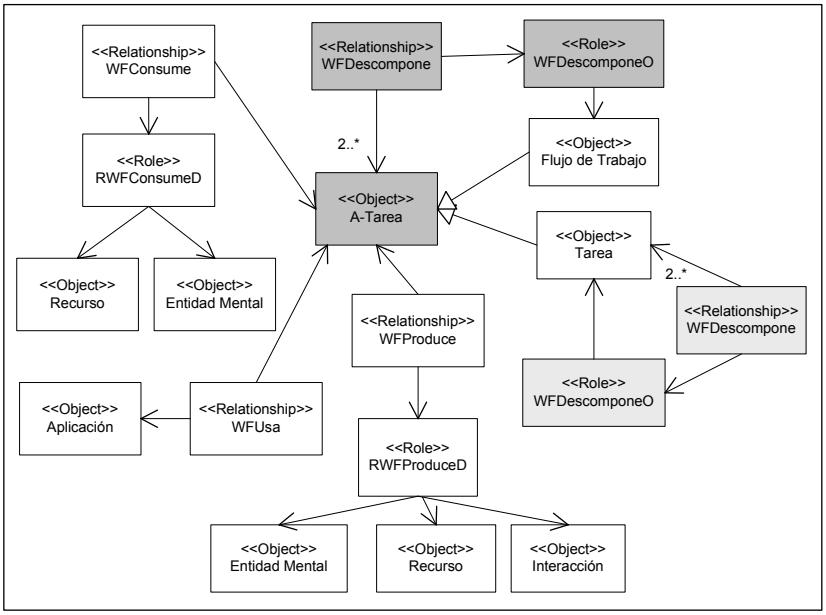


Figura 5.12: Descripción de Tareas.

5.2.2.4. Meta-modelo de Interacción

El meta-modelo de interacción de INGENIAS se construye sobre: agentes, roles, objetivos, tareas, interacciones y unidades de interacción. Los agentes y los roles son los actores de las interacciones. En las interacciones se ejecutan unidades de interacción (pasos de mensaje, lectura y escritura en un espacio de tuplas, etc.) en las que hay un iniciador (emisor) y colaboradores (receptores). Además, se justifica la participación de los actores en la interacción y la existencia de la interacción en sí mediante objetivos. Las interacciones determinan el comportamiento de los agentes. Este comportamiento puede ser: la respuesta del agente ante una determinada acción sobre él mismo (comportamiento reactivo); o una función de los objetivos del agente y las tareas a ejecutar (comportamiento pro-activo).

En ModA2 el modelo de interacción adquiere gran importancia y utilidad.

Primero (y como es común a los enfoques metodológicos tradicionales para sistemas multi-agentes), constituye una herramienta para que el diseñador pueda expresar: la dinámica del sistema que está modelando, la comunicación entre las distintas entidades del sistema, la coordinación y el comportamiento de estas entidades. Segundo (y de gran utilidad para el modelado de sistemas multi-agente de gran escala), el diseñador puede definir escenarios de interacción en los que las entidades participantes, al ser Agentes Abstractos, pueden tener distinto nivel de complejidad (agente, grupo de agentes, organización, sistema multi-agente). Tercero (y de gran utilidad para la integración de sistemas multi-agentes pre-existentes), los escenarios de interacción pueden utilizarse como patrones de integración de los sistemas multi-agentes (organizaciones, federaciones, etc.) encapsulados como Agentes Abstractos en la interacción. Cuarto, el encapsulamiento de la estructura del Agente Abstracto permite abstraer detalles para facilitar y resaltar el contexto, la naturaleza y la ejecución de la interacción según el problema que se está abordando (por ejemplo, en un sistema de fabricación, modelar la interacción entre la gestión de inventario y la gestión de producción para producir un determinado producto, sin tener en cuenta los detalles como las máquinas, la configuración en planta, la disposición de almacenes, etc.). Quinto, los escenarios de interacción asociados a una organización, constituyen la especificación del comportamiento del Agente Abstracto que en un nivel de abstracción mayor (una etapa de análisis previa) representa a esa organización.

La Figura 5.13 presenta alguna de las modificaciones que hemos realizado al meta-modelo de interacción de INGENIAS para poder incluir las entidades abstractas. Estas modificaciones tienen que ver con: los participantes en la interacción, A-Agente; el tipo de participación, relaciones IInicia e IColabora; los objetivos que persigue la interacción, A-Objetivo; y la especificación de la interacción, entidad Especificación. La motivación de la Interacción es el A-Objetivo con el que se relaciona mediante la relación IPersigue. Este A-Objetivo será satisfecho ya sea porque un A-Agente o un Rol que participa en esa Interacción ejecuta alguna A-Tarea que satisface ese A-Objetivo (ver sección 5.2.2.3); o porque alguno o varios de los participantes en la Interacción logran satisfacer sub-objetivos de ese A-Objetivo u otros A-Objetivos de los que depende éste,

relación GTDepende (ver sección 5.2.2.3).

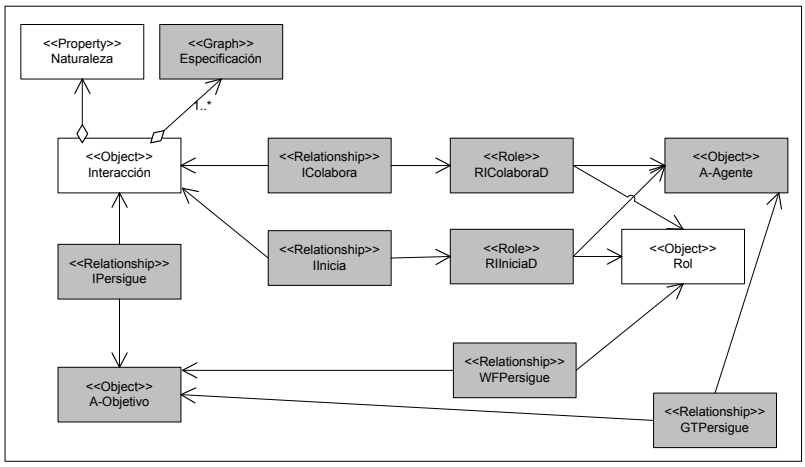


Figura 5.13: Meta-modelo de interacción, Agente Abstracto.

Al igual que INGENIAS, ModA2 agrupa la ejecución y representación de la Interacción en el grafo Especificación. Además de la Especificación GRASIA, la Especificación UML y la Especificación AUML de INGENIAS, agregamos la Especificación ModA2. En la Figura 5.14 podemos ver la especificación de las unidades de interacción de ModA2. Especificación ModA2 es una generalización de Especificación GRASIA ya que hemos modificado esta última para hacerla más general e incluir en ella las entidades abstractas. El A-Agente puede actuar como iniciador de Interacción o de Unidad Interacción, mediante la relación ternaria UIInicia pudiendo ejecutar una A-Tarea. El A-Agente también puede participar como colaborador, relación ternaria UIColabora. Las unidades de interacción pueden ser mensajes, actos del habla o protocolos [Gómez, 2002]. Las relaciones UIColabora y UIInicia se redefinen en la clases hija Agente y Tarea, roles RUIColaboraD, RUIColaboraEjecutarD, RUIIniciaD y RUIIniciaEjecutarD, para restringir el caso simple. Estas redefiniciones no se incluyen en la Figura 5.14 por problemas de espacio. Hemos incluido la propiedad Marca Temporal / Condición asociada a UIColabora y UIInicia para permitir modelos de interacción con características temporales [Julián, 2002], de esta manera

podremos asociar marcas de tiempo a las Unidades de Interacción además de condiciones sobre estas marcas.

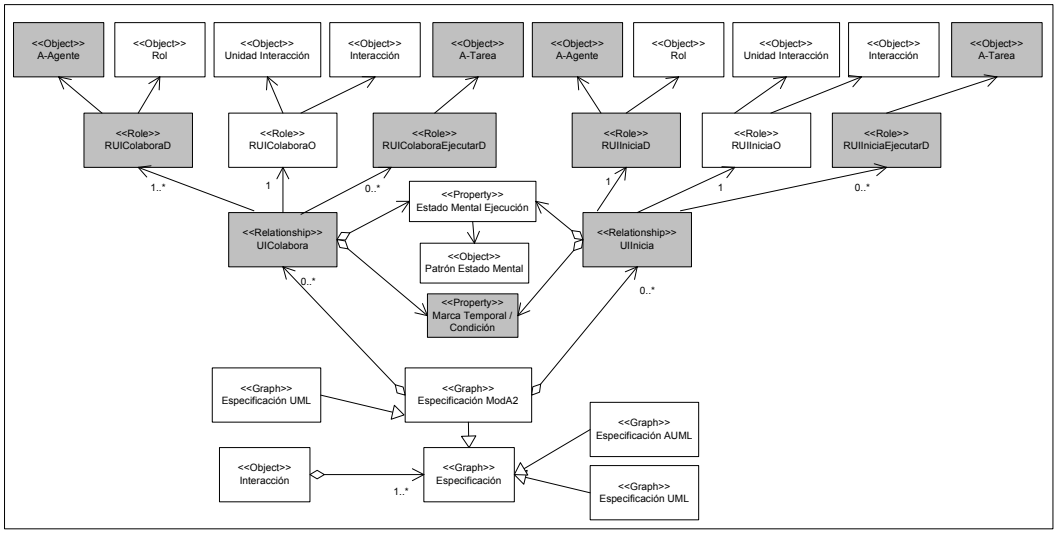


Figura 5.14: Especificación ModA2 de una Interacción.

Los patrones de estado mental (condiciones de comienzo y colaboración) que se pueden asociar a una unidad de interacción se definen de la misma manera que en INGENIAS con el modelo de agente. Esta descripción incluye las entidades que deben existir asociadas al A-Estado Mental del A-Agente y qué condiciones se deben satisfacer (Sección 5.2.2.2).

Las interacciones surgen en el contexto de las organizaciones. En la Figura 5.15 podemos ver la relación entre Interacción, A-Objetivo y Organización. Esta relación es indirecta ya que la Interacción se origina cuando dentro de un Flujo de Trabajo de una Organización el ejecutor de una Tarea activa la Interacción para satisfacer un A-Objetivo que está relacionado con el A-Objetivo que persigue la Interacción.

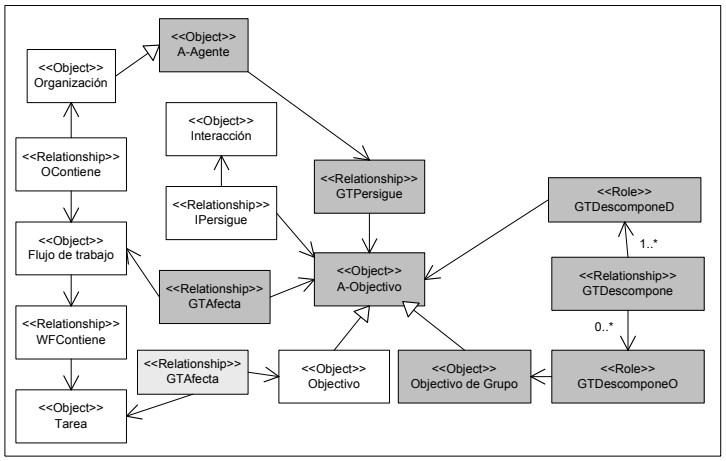


Figura 5.15: Interacción, Organización y Objetivo.

5.2.2.5. Meta-modelo de Entorno

El modelo de entorno de INGENIAS trata con la categorización de las entidades del entorno con las que puede interactuar un agente y la restricción de estas interacciones. Así, el modelo de entorno se construye sobre recursos, aplicaciones y agentes, y se centra en la percepción y actuación de los agentes sobre este entorno.

En la definición de ModA2 hemos realizado muy pocas modificaciones al meta-modelo de entorno de INGENIAS. Esto se debe a que en este meta-modelo el A-Agente es visto desde "fuera" y por tanto sólo actúa como usuario de recursos y aplicaciones o como usuario y/o cliente de otros A-Agentes.

Recurso se relaciona con A-Agente mediante la relación ERecursoPertenece (Figura 5.16), de esta manera hemos extendido la correspondiente relación de INGENIAS para permitir que un recurso pueda pertenecer a una organización.

En la Figura 5.17 podemos ver la relación EPercibe entre A-Agente y Aplicación. Una Aplicación es todo aquel objeto del entorno que proporciona una funcionalidad concreta pero cuyo comportamiento no satisface el principio de racionalidad [Gómez, 2002].

Un agente actúa sobre el entorno ejecutando tareas. De esta manera una

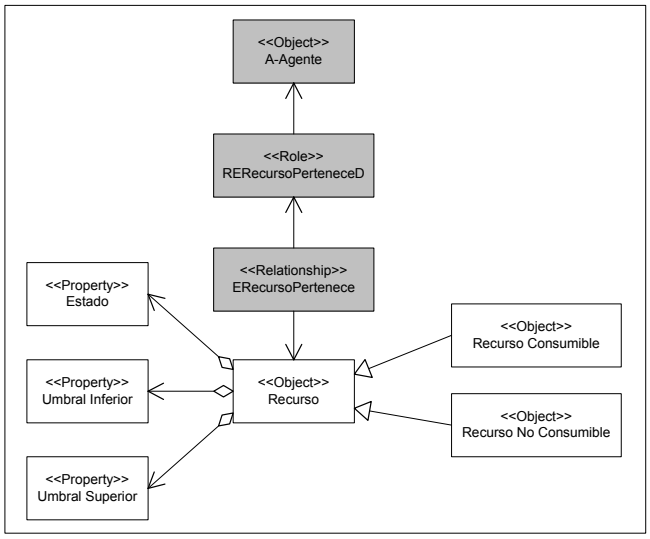


Figura 5.16: Meta-modelo de Entorno: Recurso.

A-Tarea se puede asociar con Entidad Mental, Recurso y Aplicación mediante las relaciones de la Figura 5.18.

Básicamente hemos extendido las relaciones que asocian elementos del entorno con A-Agentes. ¿Qué significado adquieren estas relaciones para el caso de los A-Agentes?. Las instancias de modelos de entorno obtenidas en una etapa de análisis definirán los requisitos y restricciones de los modelos de la siguiente etapa. Cuando las entidades participantes son Agentes simples la instancia de modelo de entorno obtenida se mantiene en la siguiente etapa de análisis (en el caso en que hubiese una) y el significado de las relaciones es el mismo que el que se les da en INGENIAS. Por otra parte, cuando la entidad considerada es un A-Agente compuesto, es decir una Organización las instancias de modelo de entorno se utilizan como requisitos para la siguiente etapa. Las relaciones se traducen de la siguiente manera. Supongamos, el A-Agente *O1* compuesto por los Agentes $\{a1,a2,a3\}$, la Aplicación *Ap1*, el Recurso *R1*, la A-Tarea *T1* compuesta por las Tareas $\{t1,t2\}$ y la Entidad Mental *EM1*. Una relación EPercibe entre *O1* y *Ap1*, significa que al menos uno de los miembros

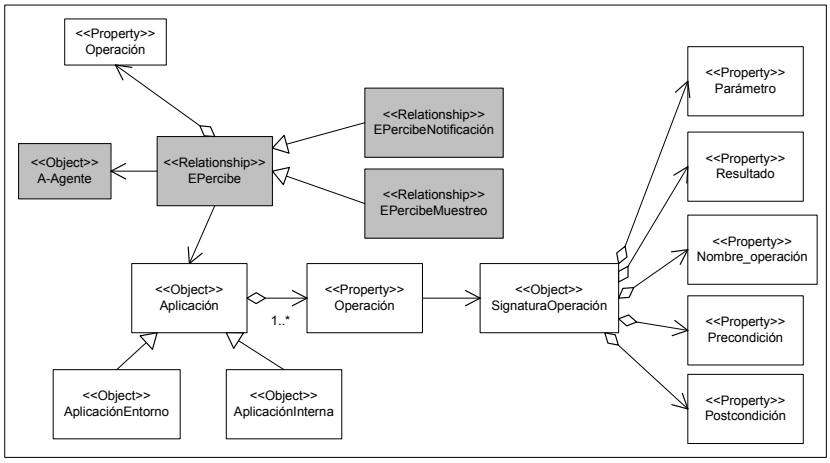


Figura 5.17: Meta-modelo de Entorno: Aplicacion.

de *O1* percibirá *Ap1*, supongamos *a2* EPercibe *Ap1* y *a1* EPercibe *Ap1*. Una relación ERecursoPertenece entre *O1* y *R1*, significa que *R1* pertenecerá a uno de los miembros de *O1*, supongamos *R1* ERecursoPertenece *a1*. Una relación WProduce entre *T1* y *EM1*, o entre *T1* y *R1*, significa que al menos una Tarea de *T1* producirá *EM1* o *R1*, respectivamente. Finalmente una relación WFUsa entre *T1* y *Ap1*, o entre *T1* y *R1*, significa que al menos una Tarea de *T1* utilizará *Ap1* o *R1*, respectivamente.

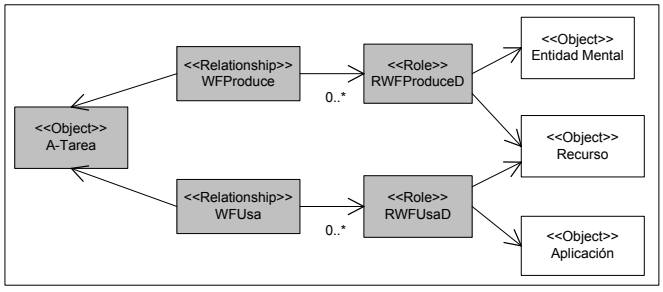


Figura 5.18: Tarea Abstracta, Recurso, Aplicación y Entidad Mental.

5.2.2.6. Meta-modelo de Organización

El modelo de organización constituye uno de los modelos más importantes en el desarrollo de los SMA. Desde hace unos años han surgido varios esfuerzos investigadores que tratan de estudiar y definir métodos de ingeniería del software para SMA utilizando conceptos organizativos como elementos guía y pilares del proceso de desarrollo. El análisis de estas metodologías queda fuera del alcance de esta tesis, sin embargo el lector interesado puede consultar los trabajos reportados en [Argente *et al.*, 2004; Ferber *et al.*, 2004; Dignum *et al.*, 2002; Juan *et al.*, 2002; Esteva *et al.*, 2001; Dellarocas y Klein, 1999].

El modelo de organización de INGENIAS trata con la descripción estructural, funcional y social de la organización del SMA. Las entidades principales sobre las que se basa son: organizaciones, agentes, flujos de trabajo y objetivos que se persiguen globalmente.

En ModA2 el modelo de organización adquiere gran importancia. Las implicaciones conceptuales de las modificaciones introducidas son de gran utilidad y simplifican en gran medida el desarrollo de SMAs de gran escala.

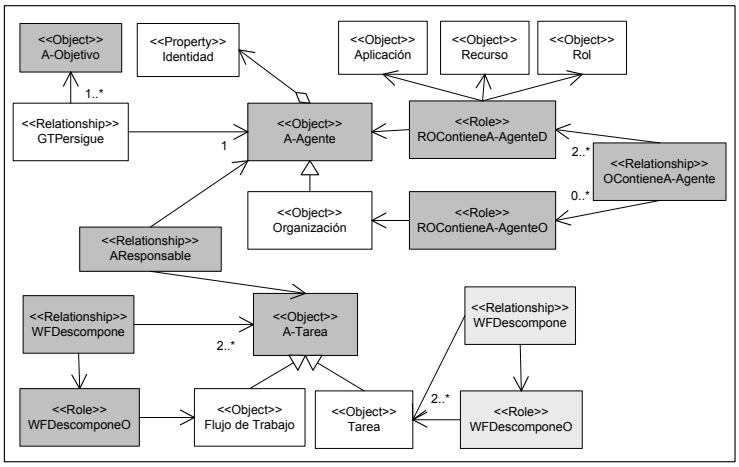


Figura 5.19: Organización. Visión Estructural.

En la Figura 5.19 podemos ver un fragmento del meta-modelo de organización de ModA2. La descripción estructural de Organización se ha generalizado. Hemos añadido una relación de agregación entre organizaciones, relación OContieneA-Agente. La entidad *Grupo* de INGENIAS la hemos reemplazado por Organización (sub-organización), puesto que Organización está compuesto por Agentes y se puede descomponer a su vez en otras Organizaciones. De esta manera hemos eliminado un nivel en la jerarquía de composición y la hemos generalizado para permitir modelar "organizaciones de organizaciones". En ModA2 las organizaciones son temporales, es decir se crean y destruyen en tiempo de ejecución, y definen las reglas básicas para la cooperación de sus agentes componentes.

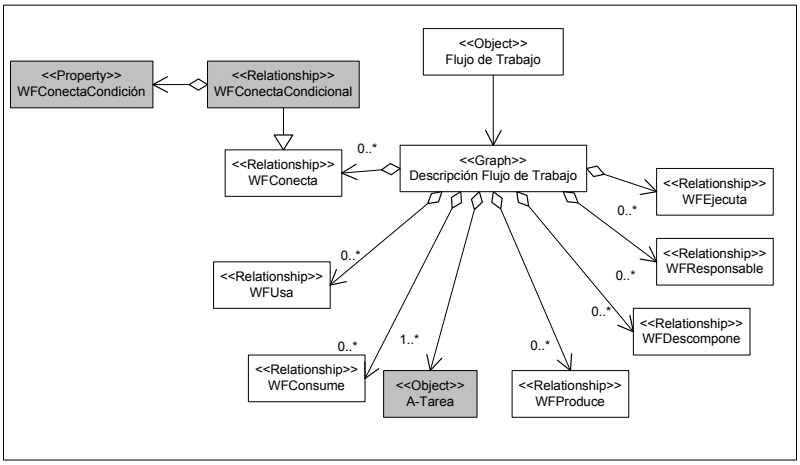


Figura 5.20: Descripción de un Flujo de Trabajo

La relación AResponsable es central en ModA2 y extiende de manera radical la definición funcional "hacia afuera" de una organización. La asociación de A-Tarea con Organizaciones permite definir la interfaz orientada a servicios de una organización. En la literatura especializada no hemos podido encontrar enfoques para definir interfaz de organizaciones. Una interfaz de organización tiene múltiples utilidades: organizaciones reutilizables, desarrollo modular conectando paquetes de organizaciones, definición de organizaciones como

entidades auto-contenidas que pueden participar en dominios de interacción mayores (otras organizaciones, interacción con otro tipo de entidades, etc.), integración de sistemas multi agentes pre-existentes, entre otros.

En el modelo de organización se identifican Flujos de Trabajo entre los A-Agentes componentes de la Organización. En la Figura 5.19 aparece Flujo de Trabajo como elemento asociado a Organización. Flujo de Trabajo es una especialización de A-Tarea y se utiliza cuando el A-Agente representa a una Organización. En la Figura 5.20 podemos ver los elementos que describen a un Flujo de Trabajo. Hemos definido la especialización WFConectaCondicional de la relación WFConecta entre A-Tareas para permitir Flujos de Trabajo con secuencias condicionales de tareas. De esta manera, por ejemplo, podemos especificar tareas anytime, mediante métodos múltiples y anytime progresivo asociando condiciones temporales a la secuencia de tareas [Julián, 2002].

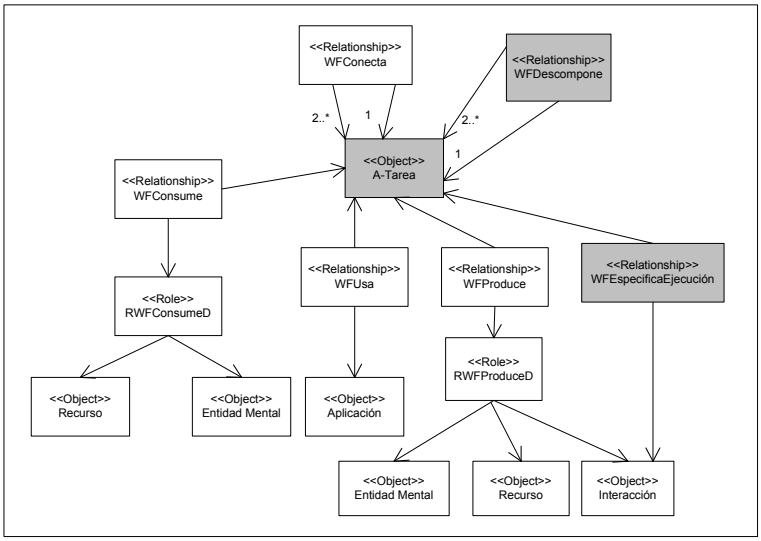


Figura 5.21: Elementos de un Flujo de Trabajo y sus relaciones.

La Organización (A-Agente) ofrecerá hacia el exterior un comportamiento de agente. A vista de sus observadores externos podrá ejecutar tareas y tomar

decisiones. Sin embargo son sus agentes internos (Agentes que existen en tiempo de ejecución) los que realmente se encargarán de actuar ejecutando tareas y de tomar decisiones mediante sus propios procesadores deliberativos (gestor y procesador de estado mental). La Figura 5.23 muestra la asociación de tareas con sus ejecutores. La entidad Flujo de Trabajo constituye el elemento que configura y ordena las tareas y el comportamiento deliberativo de los agentes componentes para definir el comportamiento autónomo de la Organización. En la Figura 5.21 podemos ver la relación entre los distintos elementos de un Flujo de Trabajo. La relación WFEspecificaEjecución implica que en la definición de una A-Tarea se debe interactuar con otros A-Agentes. Supongamos una Organización *Org1* vista como un A-Agente que participa en un Flujo de Trabajo *WF1* ejecutando la A-Tarea *T1* en la que necesita interactuar con el A-Agente *Org2* mediante la Interacción *I1*, esto se modelará con la relación *T1* WFEspecificaEjecución *I1*. Cuando *Org1* sea modelada en su estructura interna, la relación *T1* WFEspecificaEjecución *I1* se traducirá en *t2* WFEspecificaEjecución *I1.1*. Siendo *t2* una Tarea del Flujo de Trabajo *WF2* que implementa la A-Tarea *T1* y *I1.1* una Interacción del Agente de *Org1* (responsable de *t2*) con algún Agente de *Org2*.

En la Figura 5.22 podemos ver la descomposición de Tareas y Flujos de Trabajo.

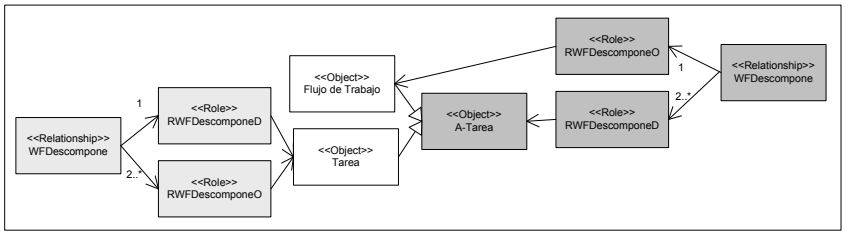


Figura 5.22: Descomposición de Tareas y Flujos de Trabajo.

En INGENIAS los tipos de relaciones sociales se restringen a tres niveles: nivel de organización (relaciones entre organizaciones), nivel de estructura

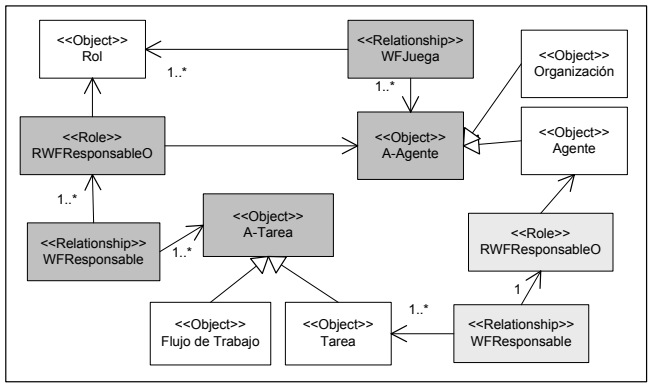


Figura 5.23: Asociación de Tarea y Agente Abstracto.

organizativa (relaciones entre grupos de agentes) y nivel de rol y agente (relaciones de agencia). Sólo se permiten relaciones sociales entre entidades del mismo tipo. En ModA2 están permitidas todas las combinaciones que surgen de los tipos de relaciones definidos en la Figura 5.24. De esta manera, gracias a la posibilidad de definir organizaciones temporales y la libertad de asociación entre entidades de complejidad distinta en ModA2 es posible modelar jerarquías, heterarquías y holarquías.

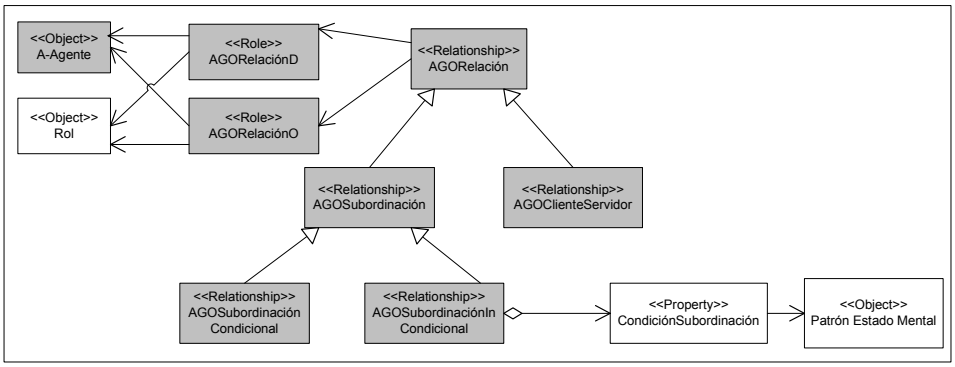


Figura 5.24: Relaciones Sociales.

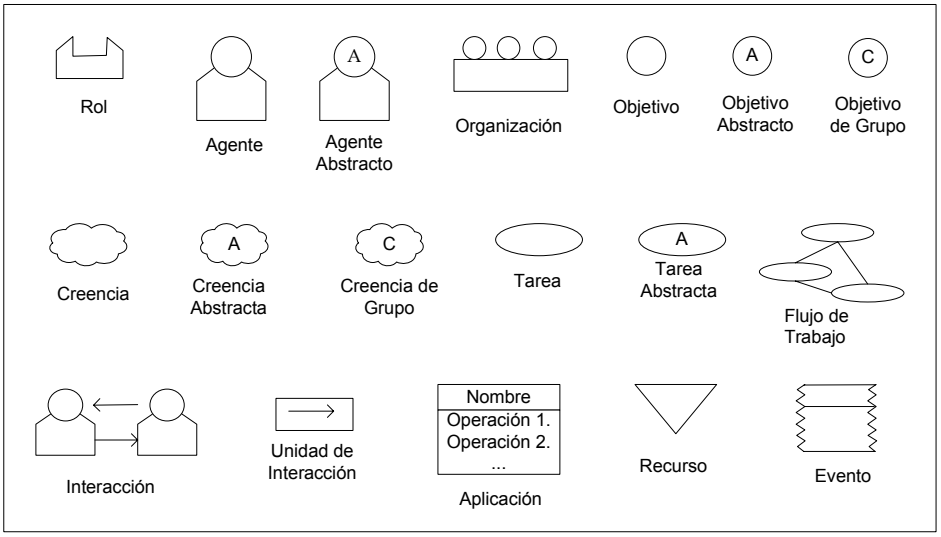


Figura 5.25: Elementos de la notación gráfica de los modelos.

5.2.3. Notación de los modelos

En los apartados anteriores hemos visto la definición de los meta-modelos, nivel M2 de la Figura 5.1. En esta sección presentamos los elementos gráficos que se utilizan en los modelos, nivel M1 de la Figura 5.1. El conjunto de elementos de nuestro enfoque contiene los elementos definidos previamente en INGENIAS y los nuevos elementos para entidades abstractas. El resumen se muestra en la Figura 5.25. La notación se ilustra en el Capítulo 6 con distintos diagramas de un caso de estudio del sector cerámico.

5.3. Proceso de Desarrollo

Todo proceso de desarrollo de una metodología incluye una descripción de los pasos sucesivos, de las actividades, de las guías para el ingeniero del software y de los productos del proceso que ayudan en el desarrollo del sistema en cuestión. En este apartado presentamos el proceso de desarrollo de nuestra metodología en términos de estos elementos. Para ello utilizamos SPEM [OMG,

2002], una notación de procesos y sus componentes.

El proceso de desarrollo que proponemos trata de abordar los requisitos de modelado para HMS (Tabla 5.1). Los requisitos 1, 2, 3 y 4 se atacan mediante la noción de Agente Abstracto, los modelos extendidos basados en INGENIAS y RT-MESSAGE (Sección 5.2) y las guías de desarrollo que presentamos en esta sección. El requisito 5 se ataca mediante las guías de identificación de holones y las guías de diseño que veremos más adelante. El requisito 6 se aborda mediante la noción de Agente Abstracto y una etapa de análisis incremental. El requisito 7 justifica el enfoque de proceso de desarrollo mixto que presentamos. Finalmente, el requisito 8 se aborda mediante las guías de modelado.

Para comprender los diagramas que definen el proceso de desarrollo, presentamos a continuación la notación SPEM. Posteriormente describimos el proceso de desarrollo en sí.

5.3.1. SPEM

SPEM (*Software Process Engineering Metamodel*) es una notación utilizada para definir procesos y sus componentes [OMG, 2002]. Esta notación está basada en el enfoque orientado a objetos para modelar una familia de procesos software relacionados. SPEM proporciona el conjunto mínimo de elementos de modelado de procesos necesarios para describir cualquier tipo de proceso de desarrollo software, sin tener que añadir ningún modelo específico o restricciones para ninguna área o disciplina específica, tal como gestión o análisis de proyectos. Los elementos de proceso de SPEM se describen en términos de conceptos UML. Un subconjunto de conceptos UML ha sido considerado relevante para la definición de SPEM. El conjunto de estos conceptos es el Meta Object Facility (MOF, definido y estandarizado por OMG) [OMG, 2004] que constituye el kernel de la definición. MOF proporciona una meta-definición circular de este lenguaje de modelado [1]. Recordemos los distintos niveles M0, M1, M2, y M3 definidos por OMG como una arquitectura estandarizada de cuatro niveles para el meta-modelado (ver Figura 5.1). En esta sección nos interesa sólo la parte izquierda de la Figura 5.1 que tiene que ver

[1] MOF y GOPRR (Sección 5.2.1) son similares y de hecho se pueden hacer traducciones de uno a otro.

con el modelado de procesos de ingeniería del software. Un proceso de ingeniería en ejecución - esto es, un proceso de producción del mundo real - tal como es aplicado, se encuentra en el nivel M0. La definición del proceso correspondiente se encuentra en el nivel M1. Por ejemplo, el RUP (*Rational Unified Process*) [Kruchten, 2000], XP (*Extreme Programming*) [Beck, 2000], etc. están definidos en el nivel M1. Tanto un proceso genérico, como RUP, y una adaptación especial de este proceso para un proyecto dado están en el nivel M1. El meta-modelo de este proceso está en el nivel M2 y sirve como un patrón para el nivel M1. Con el objetivo de eliminar un número infinito de niveles, OMG ha definido el nivel M3, el meta-meta-modelo, de manera reflexiva, lo que hace que MOF y GOPRR sean capaces de definirse a sí mismos.

SPEM define a un proceso de desarrollo de software como un proceso colaborativo entre entidades abstractas activas llamadas *roles de proceso* que realizan operaciones llamadas *actividades* sobre entidades concretas y tangibles llamadas *productos de trabajo*. Múltiples *roles de proceso* interactúan o colaboran intercambiando *productos de trabajo* y activando la ejecución de ciertas *actividades*. El objetivo global de este proceso es alcanzar un estado "bien-definido" de un conjunto de *productos de trabajo*. El conjunto de *productos de trabajo* así como su estado "bien-definido" y las *actividades* ejecutadas definen una metodología particular.

El conjunto completo de los elementos de modelado de proceso que define SPEM se puede consultar en [OMG, 2002]. En la tabla 5.2 resumimos los principales elementos de modelado de proceso que se utilizan en la literatura especializada y que utilizaremos en esta tesis.

Las relaciones entre los distintos elementos de definición de procesos de SPEM se modelan mediante los diagramas UML [OMG, 2005]: de clase, de paquete, de actividad, de casos de uso, de secuencia y de transición de estado.

5.3.2. El método

El proceso de desarrollo de nuestra metodología es un proceso mixto, ascendente y descendente. Está guiado por la noción de Agente Abstracto para la identificación y modelado de holones. Proporciona guías de modelado claras y específicas para el dominio HMS y cubre todo el ciclo de vida del HMS

Tabla 5.2: Elementos de definición de procesos de SPEM

Elemento	Notación	Descripción
Proceso		Representa a un proceso completo. Es un conjunto de descripciones de proceso internamente consistente que puede ser reutilizado para definir procesos mayores.
Fase		Representa a una fase de un proceso de desarrollo software. Es una especialización de *Definición de Trabajo*.
Definición de Trabajo		Representa un conjunto de tareas. Es un tipo de operación (o tarea compleja) que describe el trabajo desarrollado en el proceso.
Producto de Trabajo		Es cualquier cosa producida, consumida o modificada por un proceso. Puede ser un trozo de información, un *Documento*, un *Modelo*, código fuente, etc.
Encargado de Proceso		Representa el rol primario que ejecuta y es dueño de una *Definición de Trabajo*.
Rol de Proceso		Define las responsabilidades sobre un determinado *Producto de Trabajo* y los roles que ejecutan y ayudan en *Actividades* específicas.
Actividad		Es la sub-clase principal de *Definición de Trabajo*. Describe el conjunto de tareas, operaciones y acciones que son ejecutadas por un *Rol de Proceso*.
Documento		Representa un documento generado en un *Proceso*.
Guía		Los elementos guía pueden asociarse a cualquier elemento SPEM para proveer información más detallada sobre el elemento asociado. Ejemplos: Guías, Técnicas, Métricas, Ejemplos, Perfiles UML, Patrones, etc.
Modelo		Representa los modelos utilizados en el proceso de desarrollo software. Ejemplo: modelo de clases, modelo conceptual, modelo dinámico, modelo de agentes, modelo de interacción, etc.
Paquete de Proceso		Es un contenedor que contiene e importa elementos de descripción de procesos.

mediante fases de desarrollo completas.

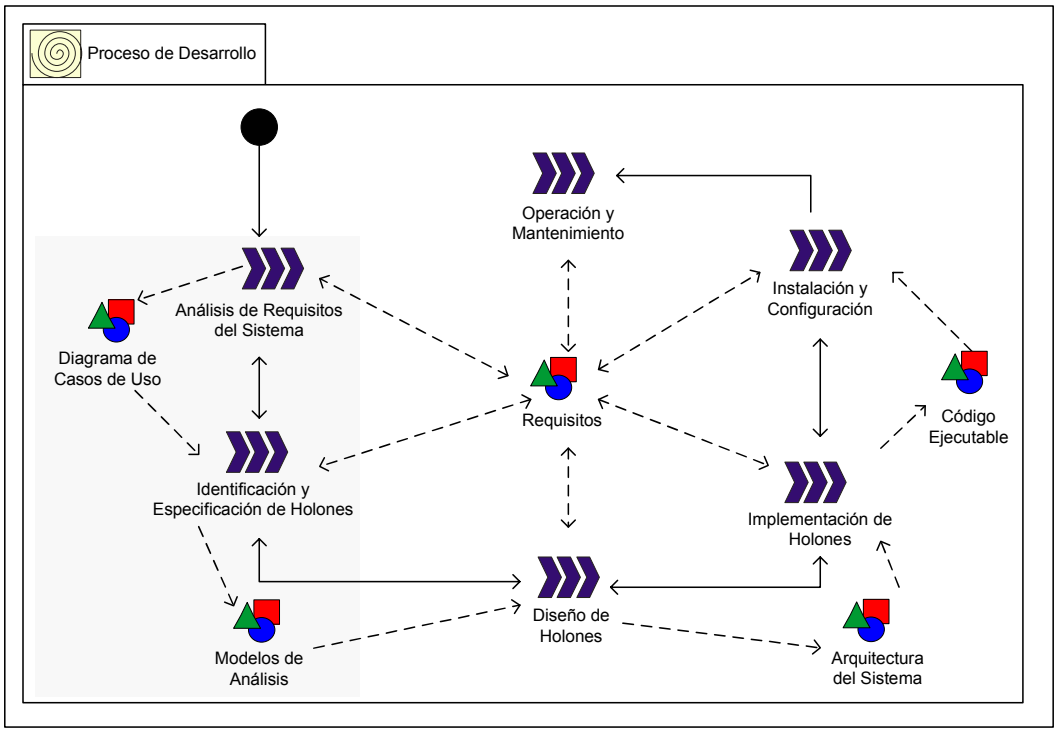


Figura 5.26: Fases del Método.

Los etapas de desarrollo se ilustran en la Figura 5.26. La primera etapa Análisis de Requisitos del Sistema y la segunda etapa Especificación e Identificación de Holones definen la fase de **análisis** de nuestro enfoque. El objetivo del análisis es proveer una especificación de alto nivel del HMS a partir de los Requisitos del problema (documento que se define a partir de las necesidades del Cliente/Usuario y puede ser actualizada por cualquiera de las etapas del proceso de desarrollo). La etapa de análisis sigue un enfoque descendente (top-down) recursivo. Una ventaja de una fase de análisis recursiva es que su resultado, los Modelos de Análisis, proveen un conjunto de componentes elementales y un conjunto de reglas de composición en base a las cuales se puede definir una etapa de **diseño** ascendente. La segunda fase es el Diseño de

Holones que consiste en un proceso ascendente (bottom-up) para producir la Arquitectura del Sistema a partir de los Modelos de Análisis. El objetivo de la fase de Implementación de Holones es producir el Código Ejecutable del sistema para que en la fase de Instalación y Configuración pueda ser implantado. Finalmente en la fase de Operación y Mantenimiento se llevan a cabo las funciones de mantenimiento.

A lo largo de esta sección presentamos cada una de las etapas del proceso de desarrollo. En el Capítulo 6 ilustramos la aplicación de nuestra metodología mediante el desarrollo de un caso de estudio.

5.3.2.1. Requisitos del Sistema

El documento de Requisitos describe la especificación de requisitos del sistema de fabricación y sus problemas de control asociados. Este documento debe contener las siguientes partes (Figura 5.27):

1. una especificación de los departamentos (organigrama) de la empresa de fabricación, incluyendo una descripción de la funcionalidad de cada departamento, las relaciones de poder entre ellos y sus interacciones;

2. una descripción de los procesos de negocio dentro de la empresa de fabricación [UEML, 2003];

3. una especificación del alcance del sistema que se desea desarrollar. Esta especificación debe incluir los procesos actuales que serán reemplazados, nuevos procesos que serán incluidos, y cómo deberán interactuar estos con el resto de componentes del sistema de fabricación.

4. una especificación detallada de los procesos de negocio del sistema de fabricación que deben ser controlados, incluyendo las interfaces de control disponibles [Groover, 1987; Hitomi, 1994];

5. una especificación de las condiciones de operación [Groover, 1987; Hitomi, 1994]; y

6. una especificación de los objetivos y requisitos de producción [Groover, 1987; Hitomi, 1994].

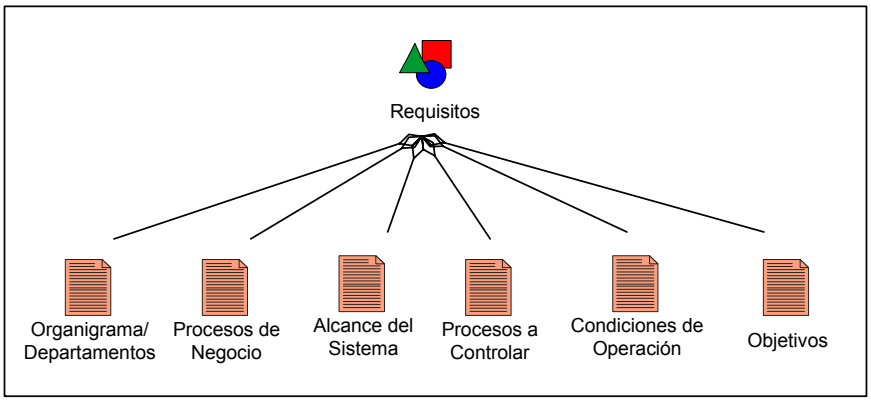


Figura 5.27: Documento de Requisitos.

Los puntos 1 y 2 se pueden desarrollar utilizando diagramas IDEF0 [IDEF, 2004], DFD [Demarco y Plauger, 1979] o diagrama de Actividades de UML [Jacobson *et al.*, 1999]. El punto 3 sirve para delimitar el HMS dentro del sistema de fabricación y su interacción con sistemas pre-definidos. Para ello se modifican los diagramas de los puntos 1 y 2 para destacar los procesos que serán sustituidos o los nuevos procesos que serán incluidos. El punto 4 puede ser descrito utilizando lenguaje natural. Cuando el proceso de negocio que se quiere controlar es el proceso de producción la descripción debe contener: los componentes (físicos) del sistema de producción y su disposición en planta [Castillo y Smith, 2002]. Además, para cada componente se describe su comportamiento físico y su interfaz de control. El comportamiento físico puede ser descrito con ayuda de diagramas IDEF0 o Petri nets [Murata, 1989]. La interfaz de control especifica qué tipo de información acerca del estado del componente de producción se puede disponer y qué tipo de acción (física) puede ser activada en el componente. El punto 5 especifica las entradas y salidas (requeridas) de los procesos a controlar, así como el espectro de posibles cambios y fallos que pueden ocurrir durante la operación del HMS. Las entradas a los procesos se especifican en términos de las órdenes de trabajo o servicios que pueden ser atendidos por el proceso, y la materia prima que es

necesaria para ejecutarlos. La salida se especifica mediante la lista de productos/respuestas esperados. Finalmente en el punto 6, se especifican los criterios de desempeño para el sistema a desarrollar. Algunos ejemplos de objetivos de producción pueden ser: alta productividad, minimizar tiempos de espera, maximizar el uso de la capacidad, flexibilidad con respecto a cambios de recursos y órdenes de trabajo, capacidad de respuesta ante fallos mecánicos o de control, escalabilidad, aseguramiento de la calidad, mantenibilidad, la posibilidad de intervención humana, etc.

5.3.2.2. Análisis

En la fase de análisis, el ingeniero del software debe especificar el HMS en términos de los modelos presentados en la Sección 5.2.2 y de Diagramas de Casos de Uso. Este es un proceso descendente, incremental y recursivo. El objetivo principal de la fase de análisis es identificar los holones que componen el sistema y proveer una especificación inicial de holones. El ingeniero del software debe producir los Modelos de Análisis a partir del documento de Requisitos.

Cada iteración de la fase de análisis identifica y especifica holarquías de diferente nivel de abstracción (Capítulo 4). Esta idea se ilustra en la Figura 5.28. En el nivel más alto (nivel m) podemos considerar a todo el sistema como un único holón (*A-Agentek*) que persigue los objetivos del sistema global definidos en el documento de Requisitos. En la primera iteración de la fase de análisis, el ingeniero del software identifica una holarquía (nivel *m-1*) compuesta por un conjunto de holones que implementan al *A-Agentek*. Así cada holón de esta holarquía representa una entidad autónoma más simple que el *A-Agentek* pero que en cooperación con los demás miembros de la holarquía consiguen implementar la funcionalidad y comportamientos del *A-Agentek*. Los Modelos de Análisis de la primera iteración especifican la holarquía del nivel *m-1*. Al final de la primera iteración, el ingeniero del software debe decidir la conveniencia de descomponer alguno o algunos de los holones del nivel *m-1* en holarquías de nivel de abstracción inferior. Por cada holón que se decide descomponer se inicia, en la siguiente iteración, un proceso de análisis, considerando la especificación del holón de nivel *m-1* como definición del sub-sistema que se quiere modelar. De esta manera, cada iteración nueva tendrá tantos procesos

concurrentes como holones de la iteración previa que se han optado por descomponer/"explotar". En cada iteración se van incrementando los Modelos de Análisis. Este proceso se repite para cada nivel de abstracción hasta que todo holón se define completamente y no existe necesidad de más descomposiciones.

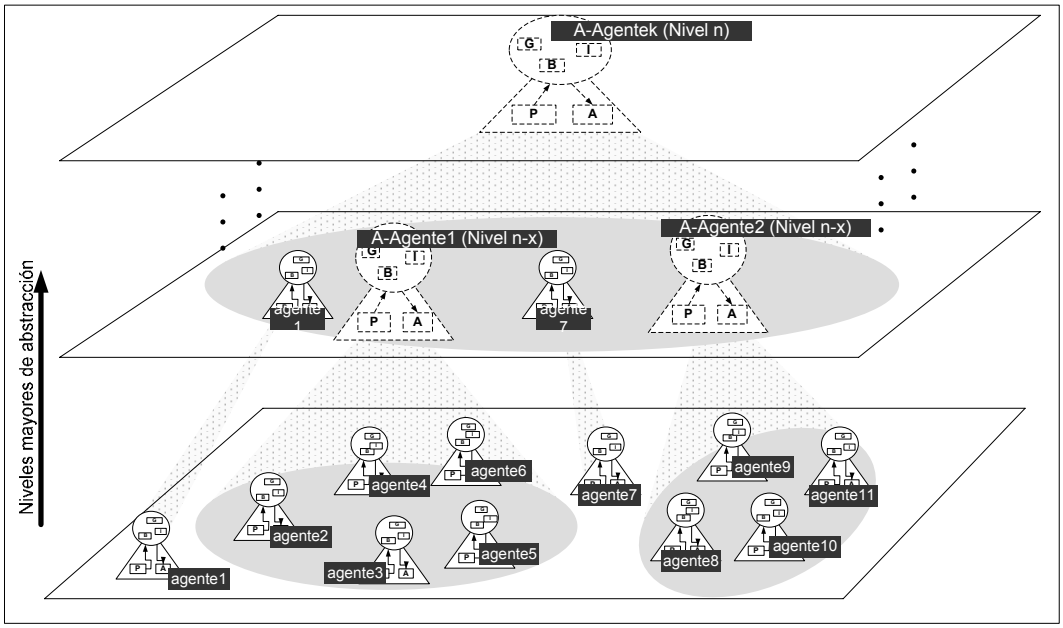


Figura 5.28: Niveles de Abstracción en la fase de Análisis.

El análisis por capas de abstracción para la identificación de holones se fundamenta en los principios de ingeniería del software de abstracción y descomposición, y en la característica recursiva de los HMS [HMS, 1994]. De esta manera en cada nivel de abstracción el ingeniero del software se centra en las peculiaridades de las interacciones y relaciones del nivel sin preocuparse todavía de cómo se implementarán internamente los holones (Agentes Abstractos) que participan en ella.

Vamos a describir a continuación una iteración de la fase de análisis. El diagrama de actividades de la Figura 5.29 ilustra el conjunto de tareas complejas que se desarrollan en una iteración, su secuencia de ejecución y los productos

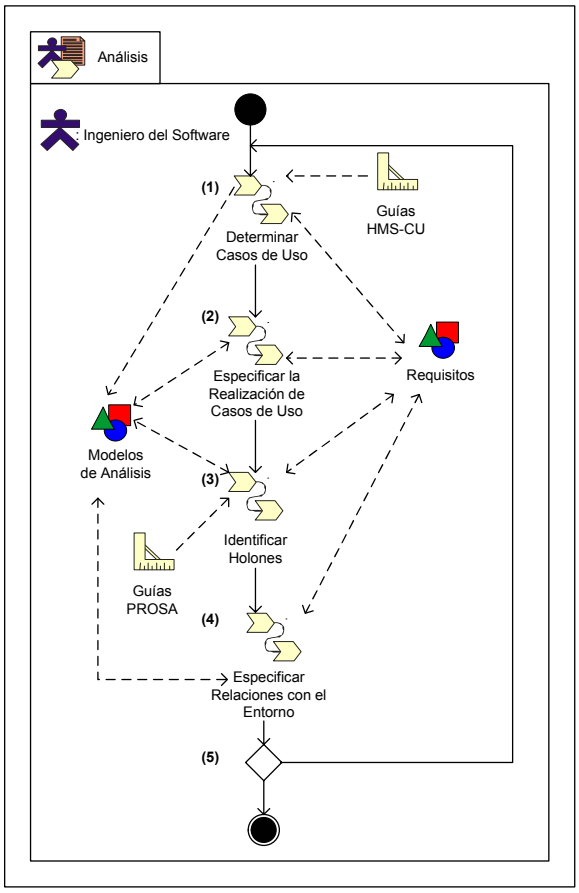


Figura 5.29: Fase de Análisis.

de cada una de ellas. El rol responsable de la etapa de análisis es el Ingeniero del Software. El objetivo en cada iteración es construir un conjunto de modelos en los que se identifican y especifican los holones que implementan una determinada holarquía. Esta holarquía representa, en la primera iteración ($ni = 1$), al sistema global, y en las siguientes iteraciones ($ni > 1$) representa a sub-sistemas del sistema global. En cada iteración se aumentan y refinan los Modelos de Análisis de la iteración anterior. La definición de una iteración es como sigue:

1. Determinar Casos de Uso. Esta tarea compleja tiene como objetivo identificar los objetivos del sistema y los dominios de cooperación [Fletcher *et al.*, 2000] necesarios para satisfacerlos. En este paso el Ingeniero del Software hace uso de las Guías HMS-CU que le ayudan a identificar casos de uso a partir del documento de Requisitos del HMS. El producto de esta tarea es un Modelo de Casos de Uso.

2. Especificar la Realización de Casos de Uso. Esta tarea compleja tiene como objetivo especificar las interacciones entre los casos de uso y definir un primer conjunto de tareas abstractas (Sección 5.2.2.3) necesarias para implementarlas. En este paso el Ingeniero del Software utiliza Roles como responsables de la implementación de los casos de uso (holarquía o dominio de cooperación) y con ellos produce Modelos de Organización, Modelos de Interacción y Modelos de Tareas y Objetivos.

3. Identificar Holones. Esta tarea compleja tiene como objetivo identificar y especificar los holones que componen la holarquía que se está analizando en la iteración actual. En este paso el Ingeniero del Software hace uso de las Guías PROSA que le ayudan a identificar Agentes Abstractos y a asignar Roles, identificados en el paso 2, a Agentes Abstractos. Para ello, se refinan los Modelos de Organización, Modelos de Interacción y Modelos de Tareas y Objetivos del paso 2, y se generan Modelos de Agentes para los holones identificados.

4. Especificar Relaciones con el Entorno. Esta tarea compleja tiene como objetivo identificar y especificar las entidades no-autónomas del entorno con las cuales los holones deben trabajar. El producto de esta tarea es el Modelo de Entorno.

5. Finalmente, por cada holón identificado en la iteración el Ingeniero del Software debe analizar la conveniencia o no de descomponerlo en una nueva holarquía en función de los Requisitos del problema. Si el resultado de este análisis es descomponer, se inicia, en la siguiente iteración, un nuevo proceso de análisis por cada holón a descomponer. En caso contrario se finaliza la etapa de análisis.

Las tareas Determinar Casos de Uso y Especificar la Realización de Casos de Uso definen la etapa Análisis de Requisitos del Sistema, mientras que las tareas Identificar Holones y Especificar Relaciones con el Entorno definen la etapa Identificación y Especificación de Holones.

En la Figura 5.30 podemos ver los distintos modelos que componen el producto de la fase de análisis.

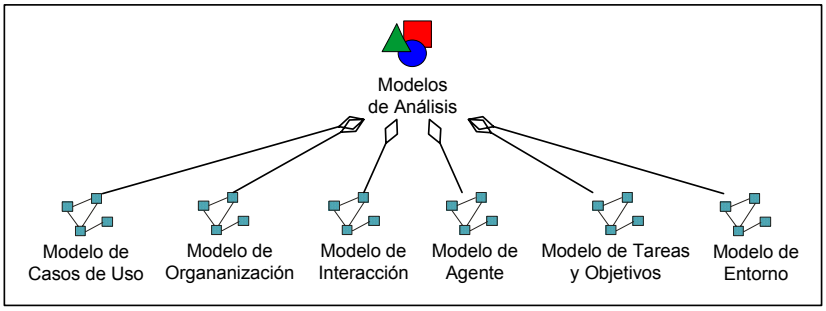


Figura 5.30: Modelos de Análisis.

A continuación presentamos la definición de las tareas complejas de la fase de análisis en términos de las actividades, la secuencia de ejecución, y los documentos y modelos que se producen y/o modifican en cada una de ellas.

Determinar Casos de Uso.

La tarea compleja Determinar Casos de Uso consiste de 4 actividades (Figura 5.31) que se definen de la siguiente manera:

Actividad A1: Estudiar el Dominio. Esta actividad tiene como objetivo que el Ingeniero del Software conozca las características generales de la empresa en la que se va a implantar el HMS, para ello debe estudiar los apartados Organigrama/Departamentos, y Procesos de Negocio del documento de Requisitos. Esta actividad se realiza en la primera iteración, y sólo en el caso en que cambie el dominio (cambios en la empresa, nuevos procesos de negocio) debe ser aplicada en las iteraciones siguientes ($ni > 1$).

Actividad A2: Situar el Problema. Esta actividad tiene como objetivo que el Ingeniero del Software pueda situar dentro de la empresa el HMS que se desea

desarrollar. A partir de los apartados Alcance del Sistema y Procesos a Controlar del documento de Requisitos, debe estudiar la interfaz que se deberá mantener con el resto de procesos de negocio de la empresa. En una iteración $ni > 1$, debe centrar el análisis en la funcionalidad/proceso de negocio que modela el Agente Abstracto que se está explotando.

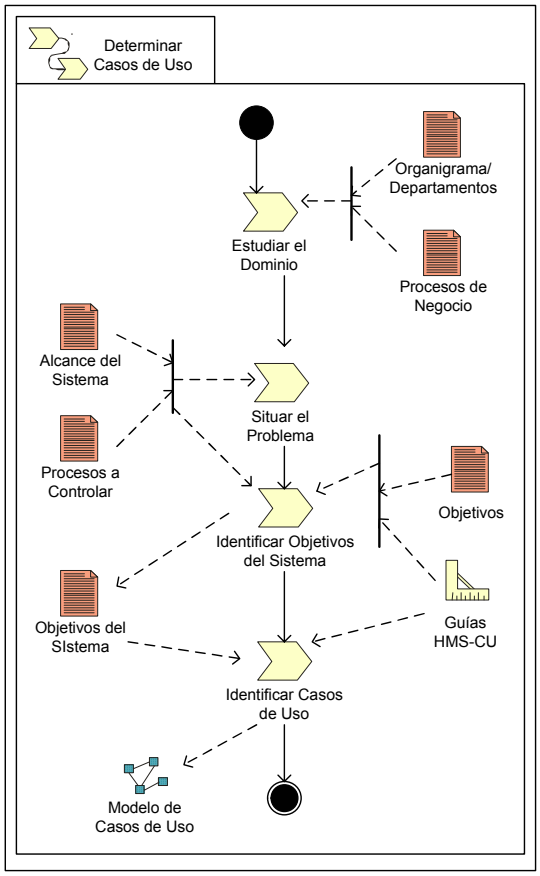


Figura 5.31: Definición de la tarea Determinar Casos de Uso.

Actividad A3: Identificar Objetivos del Sistema. En esta actividad el Ingeniero del Software debe identificar los objetivos que debe satisfacer el HMS. Para ello debe analizar los apartados Objetivos, Alcance del Sistema y Procesos

a Controlar, del documento de Requisitos, para identificar los Objetivos del Sistema. En esta actividad el Ingeniero del Software hace uso de las Guías HMS-CU para derivar los objetivos. Para ello debe responder a las preguntas 1 a 6 de las Guías HMS-CU, ver Tabla 5.3. En una iteración $ni > 1$, debe centrar el análisis en la especificación producida en la iteración anterior del Agente Abstracto que se está explotando.

Actividad A4: Identificar Casos de Uso. En esta actividad el Ingeniero del Software debe identificar Casos de Uso para satisfacer los objetivos identificados en la actividad A3. La identificación de los Casos de Uso se basa en los dominios de cooperación. Para ello debe utilizar las guías 7 a 13 de las Guías HMS-CU, ver Tabla 5.3, y construir un Modelo de Casos de Uso [Jacobson, 1992]. En una iteración $ni > 1$, los Casos de Uso asignados en la iteracción anterior al Agente Abstracto que se está explotando deben ser analizados mediante las Guías HMS-CU para descomponerlos si fuera necesario en Casos de Uso más simples.

Especificar la Realización de Casos de Uso.

La tarea compleja Especificar la Realización de Casos de Uso consiste de 4 actividades (Figura 5.32) que se definen de la siguiente manera:

Actividad A5: Asociar un responsable a cada Caso de Uso. En esta actividad el Ingeniero del Software debe identificar por cada Caso de Uso un proveedor/responsable del mismo. La idea de proveedor/responsable es directa si consideramos que cada Caso de Uso representa un servicio/funcionalidad dentro de nuestro sistema. El responsable del Caso de Uso se modela mediante la entidad Rol, así en esta etapa se especifica que una entidad autónoma es la encargada de proveer el servicio del Caso de Uso sin especificar aun si esta entidad será atómica (un único agente) o un grupo (conjunto de agentes que cooperan entre sí), ni tampoco cómo se implementará el servicio. Estas decisiones serán tomadas más adelante en la fase de análisis. Con la identificación de los distintos responsables se produce una primera versión del Modelo de Organización, que incluye la identificación de un conjunto de Roles que implementan los Casos de Uso del nivel de abstracción (Figura 5.28) que se modela en la iteración actual. Este modelo será refinado y completado en las siguientes actividades de esta tarea compleja y en las siguientes tareas e iteraciones de

Tabla 5.3: Guías HMS-CU

 Guías HMS-CU		
ID	**Guía**	**Comentario**
1	¿Se desea producir algo?. En caso afirmativo, el objetivo será conseguir producirlo.	Cuando el proceso es de fabricación, esta guía se refiere a productos de la fábrica. Cuando se está hablando de otro tipo de proceso dentro de la empresa, el producto se refiere a aquello que se debe generar como resultado del mismo.
2	¿Se desea controlar algún proceso?. En caso afirmativo, el objetivo será controlarlo.	Idéntico al anterior.
3	¿Se deben utilizar recursos externos?. En caso afirmativo, el objetivo será trabajar con ellos.	
4	¿Se puede solicitar al HMS satisfacer una solicitud de trabajo?. En caso afirmativo, el objetivo será satisfacer la solicitud y controlar su proceso de ejecución.	Una solicitud de trabajo u orden de trabajo se refiere a un pedido de fabricación o a un pedido de ejecución de algún proceso de negocio del HMS.
5	¿Se deben controlar o gestionar los objetivos del sistema?. En caso afirmativo, el objetivo será gestionarlos.	Este objetivo se relaciona con la visión global o de gestión del holón staff dentro de la arquitectura PROSA.
6	¿Se debe comunicar el producto del HMS a algún proceso de negocio externo?. En caso afirmativo, el objetivo será comunicar y gestionar la comunicación.	
7	Los objetivos derivados de la pregunta 1 se deben satisfacer mediante dominios de cooperación. Crear un CU por cada objetivo.	
8	Si un objetivo es controlar la ejecución de una máquina, crear un CU para el mismo.	Es probable que este CU se traduzca al control por parte de un holón sobre la máquina. En las primeras iteraciones puede que no aparezcan estos tipos de CU.
9	Si un objetivo es controlar algún proceso, crear un CU para el mismo.	
10	Por cada objetivo de comunicación con recursos externos, crear un CU para la gestión de la comunicación.	
11	Por cada objetivo relacionado con una solicitud de trabajo, crear un CU para implementar el dominio de cooperación encargado de satisfacerlo.	
12	Crear un CU por cada objetivo de staff.	Pregunta 6.
13	Crear un CU para satisfacer cada objetivo de comunicación con procesos de negocio externos.	

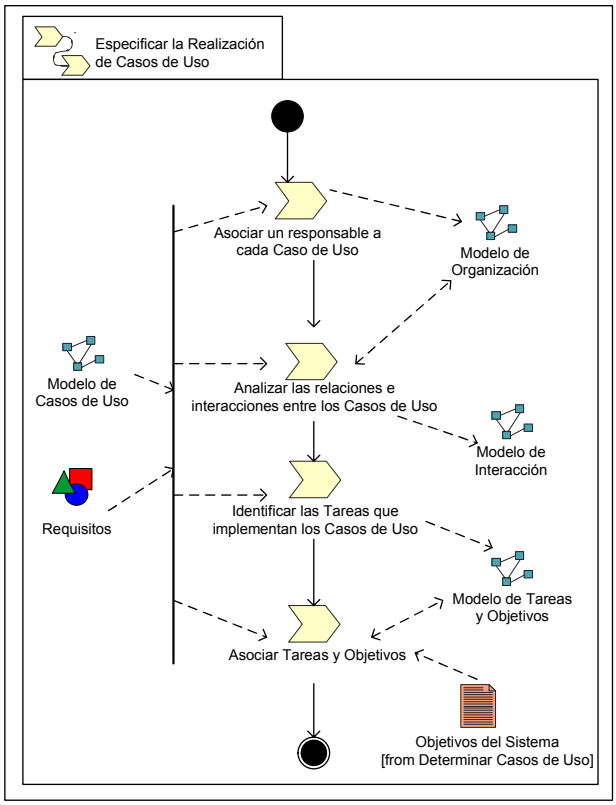


Figura 5.32: Definición de la tarea Especificar la Realización de Casos de Uso.

la fase de análisis.

Actividad A6: Analizar las relaciones e interacciones entre los Casos de Uso. En esta actividad el Ingeniero del Software debe especificar las relaciones e interacciones entre los Casos de Uso. Esto se modela mediante relaciones e interacciones entre los Roles responsables de los distintos Casos de Uso. Para ello se construyen Modelos de Interacción y se refinan Modelos de Organización. La idea de relación e interacción entre los Casos de Uso se fundamenta en el hecho que analizando y especificando de manera aislada un servicio individual no es suficiente para controlar de manera completa el problema que se desea solucionar [Bussmann *et al.*, 2004]. Los servicios, y sus acciones asociadas, no se ejecutan de manera aislada, sino dentro de un sistema en el cual sus

efectos pueden influir en otros servicios del mismo. Con el objetivo de identificar este tipo de dependencias entre Casos de Uso, es necesario analizar y modelar las relaciones e interacciones que pueden existir entre ellos. Para ello el Ingeniero del Software debe centrar el análisis en la identificación de necesidades de comunicación entre los Roles. Estas interacciones pueden ser del tipo coordinación, planificación, negociación y cooperación. Estos requisitos se especifican en el Modelo de Interacción. Además, debe identificar relaciones sociales (Sección 5.2.2.6) entre los Roles, tales como jefe-subordinado, prestación de servicios y cliente de un servicio. Estas relaciones se especifican en el Modelo de Organización.

Actividad A7: Identificar las Tareas que implementan los Casos de Uso. En esta actividad el Ingeniero del Software debe identificar las tareas abstractas (Sección 5.2.2.3) que el Rol encargado de cada Caso de Uso debe ejecutar para implementar el servicio. Estas tareas abstractas serán refinadas en iteraciones sucesivas para transformarlas según sea necesario y de acuerdo a los requisitos del sistema en tareas (atómicas) o en flujos de trabajo. Las tareas abstractas se identifican y se asocian a los Roles refinando los Modelos de Organización y completando Modelos de Tareas y Objetivos. En una iteración $ni > 1$, las tareas abstractas, identificadas en esta actividad, representan la descomposición en tareas más simples de la tarea asignada, en la iteración anterior, al Agente Abstracto que se está explotando.

Actividad A8: Asociar Tareas y Objetivos. En esta actividad el Ingeniero del Software debe asociar las tareas abstractas con los objetivos que estas afectan. Una primera identificación de objetivos se ha obtenido en la actividad A3. El Ingeniero del Software debe trabajar sobre esta lista y las tareas abstractas previamente identificadas para refinar los Modelos de Tareas y Objetivos asociando tareas abstractas y objetivos mediante la relación GTAfecta o sus especializaciones GTSatisface, GTDestruye, GTCrea y GTFalla (Sección 5.2.2.3). Además de los objetivos ya identificados, el Ingeniero del Software puede identificar nuevos objetivos asociados con las interacciones entre los Roles, para ello debe analizar: ¿por qué se activan las interacciones entre los Roles?, ¿se debe satisfacer alguna condición en la ejecución de la interacción?, ¿se debe gestionar de alguna manera las interacciones?. De las respuestas a estas preguntas

pueden surgir nuevos objetivos. Estos objetivos se especifican en los Modelos de Tareas y Objetivos y se asocian a los Roles que deben mantenerlos.

Identificar Holones.

La tarea compleja Identificar Holones es la tarea en la que el Ingeniero del Software debe invertir más tiempo en cada iteración. Se pueden seguir básicamente dos enfoques para la identificación de holones. La descomposición física (la más obvia): los agentes son utilizados para representar las entidades del mundo físico, tales como trabajadores, máquinas, herramientas, planos, productos, partes, características y operaciones. Este enfoque naturalmente define conjuntos distintos de variables de estado que deben ser gestionados de manera eficiente por los agentes individuales con un número limitado de interacciones; sin embargo, requiere un gran número de agentes relacionados con recursos. Un ejemplo común de este enfoque es el uso de agentes parte y agentes máquina para la planificación y scheduling de fabricación [Duffie y Prabhu, 1994; Lin y Solberg, 1992; Parunak, 1987]. En la descomposición funcional los agentes son utilizados para encapsular funcionalidades tales como adquisición de órdenes de trabajo, planificación, scheduling, manejo de materiales, gestión del transporte y distribución de productos. En este enfoque los agentes no tienen relación explícita con las entidades físicas. Ejemplos de este enfoque incluyen el uso de agentes para encapsular funcionalidades especiales (por ejemplo, agentes facilitadores [Cutkosky *et al.*, 1993], agentes broker [Peng *et al.*, 1998], y agentes mediadores [Maturana y Norrie, 1996]) y el uso de agentes para integrar sistemas existentes (por ejemplo, ARCHON [Cpckburn y Jennings, 1999], EXPORT [Monceyron y Barthes, 1992] y CIIMPLEX [Peng *et al.*, 1998]).

Para ayudar en el proceso de identificación y especificación de holones proporcionamos al Ingeniero del Software las Guías PROSA definidas en la Tablas 5.4, 5.5 y 5.6. Estas guías se basan en los tipo de holones PROSA [Van Brussel *et al.*, 1998] ampliamente utilizados en el área HMS. La arquitectura PROSA integra los enfoques de descomposición estructural y funcional.

La tarea compleja Identificar Holones consiste de 5 actividades (Figura 5.33) que se definen de la siguiente manera:

Actividad A9: Determinar tipos de Holones. En esta actividad el Ingeniero del Software debe identificar Agentes Abstractos. Esto es, los Roles identificados

en las etapas anteriores se asignan y especifican mediante holones. Para ello, el Ingeniero del Software debe analizar el documento de Requisitos y los Modelos de Análisis de la etapa anterior, siguiendo las Guías PROSA (Tablas 5.4, 5.5 y 5.6) para refinar el Modelo de Organización asignando Roles a Agentes Abstractos. Así, el resultado de esta actividad es la asignación de roles a tipos de Agentes Abstractos: de recurso, de producto, de orden de trabajo y de staff. En esta actividad se aplican las Guías PROSA: 1 a 13 y 22 a 25 para la identificación de Agentes Abstractos.

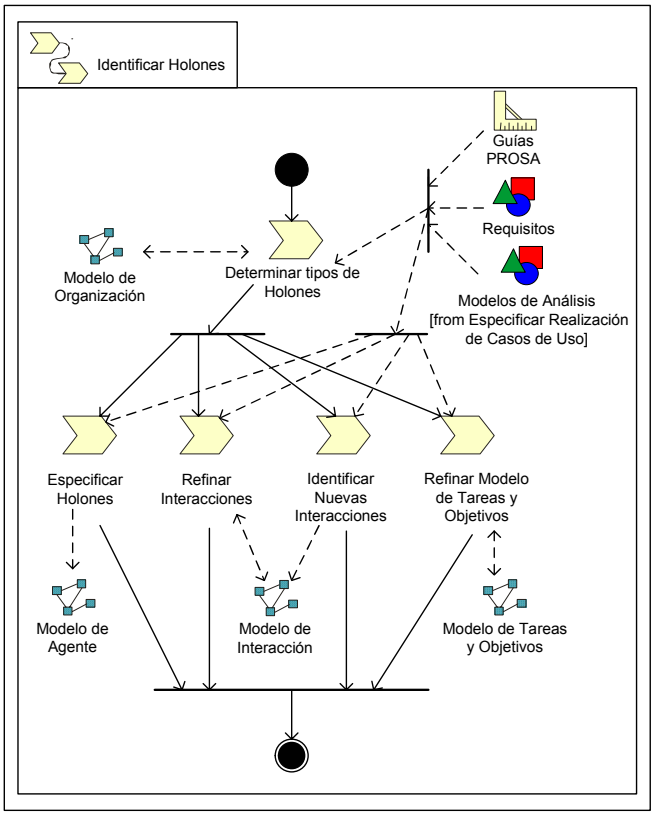


Figura 5.33: Definición de la tarea Identificar Holones.

Actividad A10: Especificar Holones. En esta actividad el Ingeniero del Software debe especificar un Modelo de Agente por cada holón identificado en la

actividad A9. Debe completar la definición del Agente Abstracto a partir de los objetivos y tareas del Rol o Roles que tiene asignados. Puede identificar nuevas tareas o descomposición de tareas. Además, debe identificar aspectos de inteligencia y autonomía que le pueden ayudar en el cumplimiento de sus objetivos. Para completar la identificación de las tareas abstractas de las que es responsable un Agente Abstracto el Ingeniero del Software debe utilizar las Guías PROSA 14 a 17 y la 25. Mientras que para identificar las Entidades Mentales de Información (Sección 5.2.2.2) debe utilizar las Guías PROSA 18 a 21 y la 25. Estas guías definen el conjunto básico y obligatorio que deben incluir los Agentes Abstractos acerca de datos que pueden manejar y funciones o servicios que ofrecen de interfaz. Sobre estos elementos el Ingeniero del Software debe agregar entidades de información y tareas al Agente Abstracto para modelar especializaciones y peculiaridades relacionadas con el sistema que se está modelando. Además, se deben analizar e identificar (si fueran necesarias) restricciones temporales que pueden estar asociadas con entidades mentales y con tareas del Agente Abstracto.

Actividad A11: Refinar Modelo de Interacciones. En esta actividad el Ingeniero del Software debe completar la definición de los Modelos de Interacción de la holarquía que se está analizando. Esto es, indicar la naturaleza de las interacciones, identificar los objetivos que persiguen las interacciones, identificar mensajes intercambiados, identificar emisores y receptores, e identificar (según el documento de Requisitos) restricciones temporales asociadas a las interacciones.

Actividad A12: Identificar Nuevas Interacciones. En esta actividad el Ingeniero del Software debe identificar interacciones (aún no identificadas) relacionadas con las características particulares de la arquitectura PROSA. Para ello debe utilizar las Guías PROSA 26 a 28. Debe modelar también, las interacciones que se refieren a la guía 25. Completar la definición de las nuevas interacciones: identificando los objetivos que persiguen, los mensajes intercambiados, los emisores y receptores, y las restricciones temporales asociadas a las interacciones (según el documento de Requisitos).

Actividad A13: Refinar Modelo de Tareas y Objetivos. En esta actividad el Ingeniero del Software debe refinar los Modelos de Tareas y Objetivos obtenidos

en la etapa anterior. Para ello debe utilizar las Guías PROSA 14 a 17 y la 25. Debe completar los modelos: asociando tareas a objetivos, estableciendo dependencias entre objetivos, descomponiendo objetivos y tareas (si fuera necesario), asociando tareas con interacciones, asociando tareas con entidades mentales, e identificando posibles restricciones temporales asociadas a la ejecución de tareas. Asegurando, en cada iteración, que toda tarea abstracta y objetivo asociados al Agente Abstracto del nivel superior (de cual se deriva la holarquía actual) son asignados a los miembros de la holarquía actual.

Especificar Relaciones con el Entorno.

En la tarea compleja Especificar Relaciones con el Entorno el Ingeniero del Software debe modelar las relaciones de los holones con su entorno. Estas relaciones se derivan a partir del documento de Requisitos y del estudio de los Modelos de Análisis definidos en las actividades previas. Esta tarea consiste de 3 actividades (Figura 5.34) que se definen de la siguiente manera (la definición de las distintas actividades es muy similar a las correspondientes de INGENIAS y RT-MESSAGE):

Actividad A14: Determinar Eventos Externos. En esta actividad el Ingeniero del Software debe determinar la lista de eventos externos que pueden afectar de alguna manera a cada holón y las restricciones temporales que puedan existir sobre la ocurrencia de estos eventos. La identificación de los eventos externos se deriva a partir del documento de Requisitos y/o de los Modelos de Análisis de la iteración anterior. En esta actividad, el Ingeniero del Software debe completar el Modelo de Entorno con eventos y sus características temporales, si las hubiera, tales como: patrón de llegada (periódica o aperiódica) y tiempo mínimo en que se garantiza que no se va a producir una nueva ocurrencia. Si se trata de una iteración $ni > 1$, la lista de eventos asociada previamente al Agente Abstracto que se está explotando, se debe distribuir en eventos que afectan a sus holones componentes. Por cada evento identificado, debe modelar, también, qué debe hacer el holón en respuesta a esos eventos. Para ello completa el Modelo de Agente para indicar el efecto del evento sobre tareas y/o el Estado Mental del Agente Abstracto y el Modelo de Interacciones para indicar la activación de una interacción en respuesta a un evento externo.

Actividad A15: Determinar Aplicaciones. En esta actividad el Ingeniero del

Tabla 5.4: Guías PROSA - Parte 1

 Guías PROSA	
ID	**Guía**
1	En una holarquía pueden existir cuatro tipos de Agentes Abstractos: de recurso, de producto, de orden de trabajo y de staff. Las siguientes guías define las características de cada uno y cómo identificarlos.
2	El Agente Abstracto de producto representa cosas que pueden ser hechas por la holarquía. Si la holarquía representa a toda la fábrica, los Agentes Abstractos de producto se refieren a los items que se encuentran en el catálogo de productos de la compañía.
3	El Agente Abstracto de orden de trabajo representa las tareas solicitadas a la holarquía y se encarga de activar a los Agentes Abstractos de recurso para iniciar la producción. Si la holarquía representa a toda la fábrica, los Agentes Abstractos de orden de trabajo representan a items incluidos en el libro de órdenes.
4	El Agente Abstracto de recurso representa un elemento dentro del sistema que ofrece una capacidad de procesamiento. Si la holarquía representa a toda la fábrica, los Agentes Abstractos de recurso representan los departamentos, los talleres, las máquinas, los empleados, etc.
5	El Agente Abstracto de staff representa a la entidad encargada de gestionar la interacción entre los distintos Agentes Abstractos.
6	Cada medio de producción (fábrica, departamento, taller, máquina, cinta transportadora, horno, herramienta, almacén, personal, etc.) y el procesamiento de la información que lo controla, se modela mediante un Agente Abstracto. Por cada medio de producción, de la holarquía que se está analizando, debe existir un Agente Abstracto que lo modele y se le deben asignar los Roles que se encargan de la gestión y control del medio de producción identificados en las etapas anteriores. Este Agente Abstracto es del tipo recurso.
7	Un Agente Abstracto de recurso ofrece capacidades y servicios de producción a los Agentes Abstractos con los que colabora. Mantiene los métodos para asignar el recurso de producción, el conocimiento y los procedimientos para organizarlo, utiliza y controla el recurso de producción. Todo Rol encargado de alguna de estas actividades se debe asignar a un Agente Abstracto de recurso.
8	Cada definición de producto se modela como un Agente Abstracto. Todo Rol encargado de mantener: información actualizada sobre un producto, sus requerimientos de usuario, su diseño, sus planes de proceso, la lista de materiales para fabricar el producto y los procedimientos de aseguramiento de la calidad, deben ser asignados a un Agente Abstracto de producto. El Agente Abstracto mantiene el "modelo de producto" de un tipo de producto, no el "estado de producción" de una instancia particular del mismo.
9	Un Agente Abstracto de producto mantiene el conocimiento y el proceso de producción para asegurar la fabricación correcta del producto, con la calidad suficiente definida en el documento de Requisitos. Actúa como un servidor de información para los Agentes Abstractos de orden de trabajo de la holarquía. El Agente Abstracto de producto modela funcionalidades relacionadas con el diseño del producto, planificación de proceso, y aseguramiento de la calidad.

Tabla 5.5: Guías PROSA - Parte 2

 Guías PROSA	
ID	**Guía**
10	Cada tarea en un sistema de fabricación (orden de trabajo de un cliente, orden de trabajo para la ejecución de un proceso de negocio, orden de trabajo para almacén, orden de trabajo para mantenimiento y reparación de recursos, etc.) se representa como un Agente Abstracto. Todo Rol encargado de gestionar un proceso de producción o un proceso de negocio, gestionar el estado de producción de los productos solicitados en la orden de trabajo, y mantener toda información logística relacionada con la tarea, debe ser asignado a un Agente Abstracto de orden de trabajo.
11	Un Agente Abstracto de orden de trabajo puede ser considerado como una pieza con cierto comportamiento de control, que se encarga de gestionar la orden de trabajo en su paso por el sistema de fabricación, esto es, negociar con otras entidades y recursos para lograr ser producida.
12	Los objetivos y/o Roles identificados en la actividad A8, relacionados con la gestión de las interacciones entre los distintos Roles de la holarquía, deben ser asignados a Agentes Abstractos de staff.
13	Un Agente Abstracto de staff se encarga de asistir, con conocimiento experto, a los demás tipos de agentes, e implementar, si fuera necesario, un control centralizado en la holarquía.
14	Un Agente Abstracto de recurso es capaz de iniciar tareas de procesamiento sobre productos, esto significa que está autorizado para aceptar o rechazar, según sus objetivos y los de la holarquía, una tarea que le ha sido asignada.
15	Un Agente Abstracto de recurso controla y monitoriza la ejecución de sus procesos (suspende, reanuda, aborta), gestiona sus sub-recursos y planifica sus tareas.
16	Un Agente Abstracto de producto implementa las tareas de diseño de producto, planificación de procesos, y verificación de la calidad del producto.
17	Un Agente Abstracto de orden de trabajo realiza el scheduling (secuenciación), manejo de bloqueos, monitorización de progreso de la orden, y dispara eventos de activación, suspensión, reanudación, aborto, o parada de un proceso dentro de un recurso.
18	Un Agente Abstracto de recurso mantiene creencias sobre sus capacidades (lista de productos que puede procesar), sus tareas en ejecución, sus sub-recursos, y un registro de sus actividades.
19	Un Agente Abstracto de producto mantiene información referente a un plan de proceso, una descripción de producto y los requisitos de calidad del producto.
20	Un Agente Abstracto de orden de trabajo mantiene información sobre el estado del producto que procesa la orden, el progreso de las tareas, y los datos históricos de las tareas.
21	Un Agente Abstracto de staff mantiene información referente a gestión de las interacciones que controla y posiblemente conocimiento experto (planes, heurísticas, etc.) para asistir a los demás Agentes Abstractos. Un Agente Abstracto de staff se puede utilizar para encapsular sistemas como CAD, MRP, etc.

Tabla 5.6: Guías PROSA - Parte 3

 Guías PROSA	
ID	**Guía**
22	Tipos de Agentes Abstractos de orden de trabajo: órdenes para mantener stock; órdenes rápidas, con fechas de entrega más cortas que lo normal en el sistema; órdenes "first-of" (configuración o de cambio de partida), requieren por lo general tiempos mayores de operación, posiblemente repetir procesos fallidos, supervisión e intervención humana; órdenes de clientes; órdenes de mantenimiento.
23	Tipos de Agentes Abstractos de producto: variaciones sobre el mismo producto; una nueva versión de un producto; partes de repuesto; producción en masa; producto de alta calidad, etc.
24	Tipos de Agentes Abstractos de recursos, en este grupo se incluyen los distintos recursos de información, máquina, herramienta, etc. que se pueden encontrar dentro de un sistema de fabricación. El personal y los operarios también se pueden modelar como Agentes Abstractos de recursos.
25	Cuando se está analizando una holarquía en una iteración $n+1$ de la fase de análisis, la interfaz del Agente Abstracto que lo modela en el nivel n (iteración anterior) puede ser implementada de dos maneras: (1) Asignar a un Agente Abstracto de staff la responsabilidad de la gestión de la misma hacia el exterior, esto significa toda comunicación hacia/desde fuera de la holarquía debe pasar por el Agente Abstracto de staff. (2) Distribuir las responsabilidades de gestión de la interacción a los distintos Agentes Abstractos de la holarquía. La última opción aunque correcta puede ser más compleja de implementar y también más difícil de mantener aunque es más flexible que la primera.
26	Para lanzar una nueva orden de trabajo dentro de una holarquía se necesitan los siguientes pasos: (1) el Agente Abstracto de staff, o el encargado de la gestión de la holarquía, determina si la holarquía es capaz de procesar la orden; en caso afirmativo (2) activa a un Agente Abstracto de orden de trabajo responsable de procesarla y confirma la aceptación de la misma al cliente; y finalmente (3) cuando la orden se completa el Agente Abstracto de staff se lo comunica al cliente.
27	Para ejecutar las tareas solicitadas en una orden de trabajo se necesitan los siguientes pasos: (1) el Agente Abstracto responsable de la orden de trabajo solicita un plan de proceso (conjunto de tareas a ejecutar) al o los Agente Abstractos de productos; (2) el Agente Abstracto de orden de trabajo inicia un proceso de negociación con los Agentes Abstractos de recurso para asignar las tareas (del plan de procesos) a recursos; finalmente (3) el Agente Abstracto inicia la ejecución de las tareas solicitando a los Agentes Abstractos de recursos iniciar el proceso.
28	Cuando un nuevo Agente Abstracto de recurso se agrega a una holarquía, éste anuncia su presencia a los demás Agentes Abstractos de la holarquía. En respuesta a este evento, los Agentes Abstractos de producto deben verificar la utilidad del nuevo recurso para sus planes de proceso.

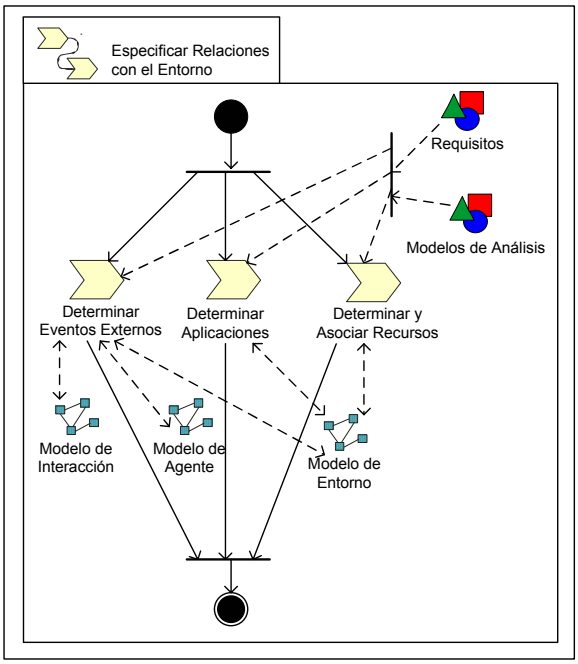


Figura 5.34: Definición de la tarea Especificar Relaciones con el Entorno.

Software debe determinar la lista de aplicaciones no-autónomas con las cuales los holones pueden trabajar. Una aplicación no-autónoma es todo software o hardware con la que el sistema debe interactuar y no puede ser modelada como holón (no satisface las características de autonomía y cooperación [HMS, 1994]). La identificación de las aplicaciones se deriva a partir del documento de Requisitos y/o de los Modelos de Análisis de la iteración anterior. En esta actividad, el Ingeniero del Software debe completar el Modelo de Entorno con las aplicaciones identificadas. Si se trata de una iteración $ni > 1$, la lista de aplicaciones asociadas previamente al Agente Abstracto que se está explotando, se debe distribuir en aplicaciones con las que trabajan sus holones componentes. Cada aplicación identificada debe ser asociada con el Agente Abstracto que hace uso de ella mediante la relación EPercibe.

Actividad A16: Determinar y Asociar Recursos. En esta actividad el Ingeniero del Software debe determinar la lista de recursos con los cuales los holones

pueden trabajar. Un recurso, en el Modelo de Entorno, es todo recurso del sistema de fabricación que no tiene parte de procesamiento de la información [HMS, 1994]. La identificación de los recursos se deriva a partir del documento de Requisitos y/o de los Modelos de Análisis de la iteración anterior. En esta actividad, el Ingeniero del Software debe completar el Modelo de Entorno con los recursos identificados. Si se trata de una iteración $ni > 1$, la lista de recursos asociados previamente al Agente Abstracto que se está explotando, se debe distribuir en recursos con los que trabajan sus holones componentes. Cada recurso identificado debe ser asociado con el Agente Abstracto que hace uso del mismo mediante la relación ERecursoPertenece.

5.3.2.3. Diseño

En la fase de diseño, el ingeniero del software debe construir la Arquitectura del Sistema teniendo en cuenta los detalles de la plataforma destino de implementación. La fase de diseño consiste, básicamente, en traducir los Modelos de Análisis a un conjunto de modelos de diseño que definen los detalles o requisitos de implementación. La fase de diseño de nuestro enfoque se divide en dos etapas (Figura 5.35):

1. Refinar la Especificación de Holones. Esta tarea compleja tiene como objetivo completar los Modelos de Análisis para asegurar que todos los requisitos del sistema están completamente modelados. Esta etapa se define como un proceso ascendente e independiente de la plataforma, en la que el Ingeniero del Software se centra en cada holón atómico con el objetivo de completar su definición. El producto de esta tarea es el Modelo de Diseño definido en la Figura 5.36.

2. Construir la Arquitectura del Sistema. Esta tarea compleja tiene como objetivo construir la Arquitectura del Sistema. Nuestro enfoque se basa en la propuesta de Christensen [Christensen, 2003] para la implementación de HMS. Para el control de alto nivel (procesamiento de la información intra-holón y cooperación inter-holón) utilizamos JADE [Bellifemine *et al.*, 2001] (estándar FIPA). Mientras que para el control de bajo nivel (operaciones físicas) de los holones de recurso utilizamos bloques funcionales (serie de estándares IEC 61499 [IEC, 2000;

2001]). Así, el Ingeniero del Software cuenta con las Guías JADE y las Guías de Bloques Funcionales para derivar las Plantillas de Agentes JADE y la Especificación de Interfaces de Bloques Funcionales respectivamente.

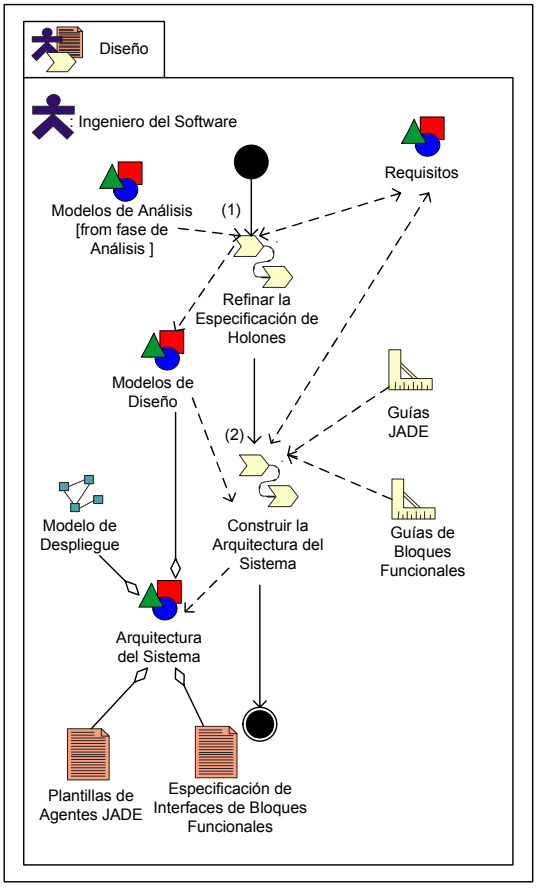


Figura 5.35: Fase de Diseño.

A continuación presentamos la definición de las tareas complejas de la fase de diseño en términos de las actividades, la secuencia de ejecución, y los documentos y modelos que se producen y/o modifican en cada una de ellas.

Refinar la Especificación de Holones.

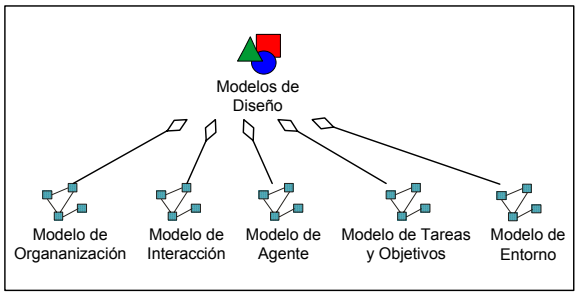


Figura 5.36: Modelos de Diseño.

Esta tarea compleja consiste en un proceso ascendente e iterativo basado en los Modelos de Análisis producidos en la fase de análisis. Los elementos clave de este proceso son los holones, sus niveles de recursión (Capítulo 4) y sus reglas de composición. La Figura 5.37 ilustra la idea. La primera iteración del proceso consiste en el diseño del conjunto de agentes (holones de nivel de recursión $n = 0$) identificados en la fase de análisis. El Ingeniero del Software debe completar detalles de diseño de estos agentes completando sus modelos correspondientes. En la segunda iteración el Ingeniero del Software debe centrar el diseño en los Agentes Abstractos de nivel de recursión $n = 1$, completando sus Modelos de Organización y sus Modelos de Interacción para diseñar la estructura organizativa del Agente Abstracto y las interacciones que se pueden dar entre sus entidades componentes (holones de nivel de recursión $n = 0$). Este proceso ascendente se repite hasta que no exista ningún nivel superior en la holarquía (global) definida por los Modelos de Análisis, o lo que es lo mismo, hasta llegar al Agente Abstracto que modela al HMS global.

Una iteración de esta tarea consiste en la ejecución de 5 actividades (Figura 5.38) que se definen de la siguiente manera:

Actividad D1: Completar Modelo de Agente. En la primera iteración, esta actividad requiere que el Ingeniero del Software se centre en cada holón atómico de la fase de análisis con el objetivo de completar su definición en base al documento de Requisitos y los Modelos de Análisis. Esto significa: detallar los estados intermedios por los que pasa el agente (Estados Mentales, Sección 5.2.2.2) y el

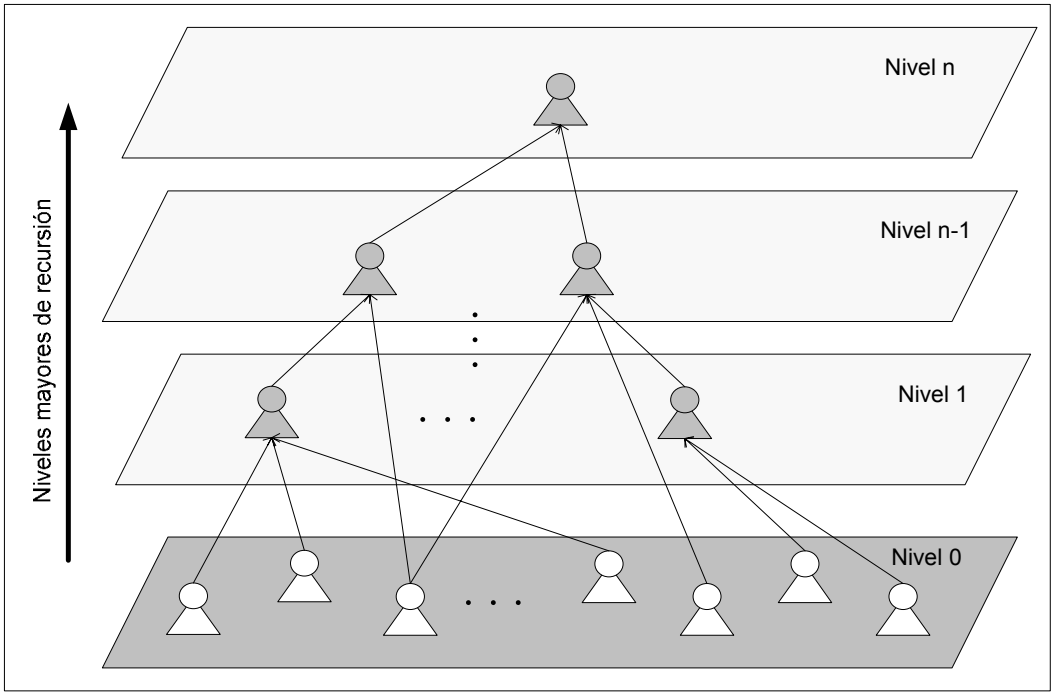


Figura 5.37: Niveles de Recursión en la fase de Diseño.

control del agente basado en sus Estados Mentales, determinar cómo se pasa de un estado a otro y asociar los Estados Mentales con la ejecución de tareas. En las siguientes iteraciones, esta actividad requiere que el Ingeniero del Software compruebe que las responsabilidades y habilidades asociadas a los Agentes Abstractos constituyentes están recogidas en las correspondientes responsabilidades y habilidades asociadas al Agente Abstracto que se está diseñando en la iteración actual. Esto significa: que la composición de las tareas abstractas (del Agente Abstracto) se basa en las tareas abstractas de sus miembros, relación WFDescompone; que la composición de los objetivos es en términos de los objetivos de sus miembros, relación GTDescompone; que la estructura mental se define a partir de la estructura mental de sus miembros, relación AContieneC. En resumen, el Ingeniero del Software debe completar el diseño de la interfaz del Agente Abstracto en términos de sus entidades constituyentes.

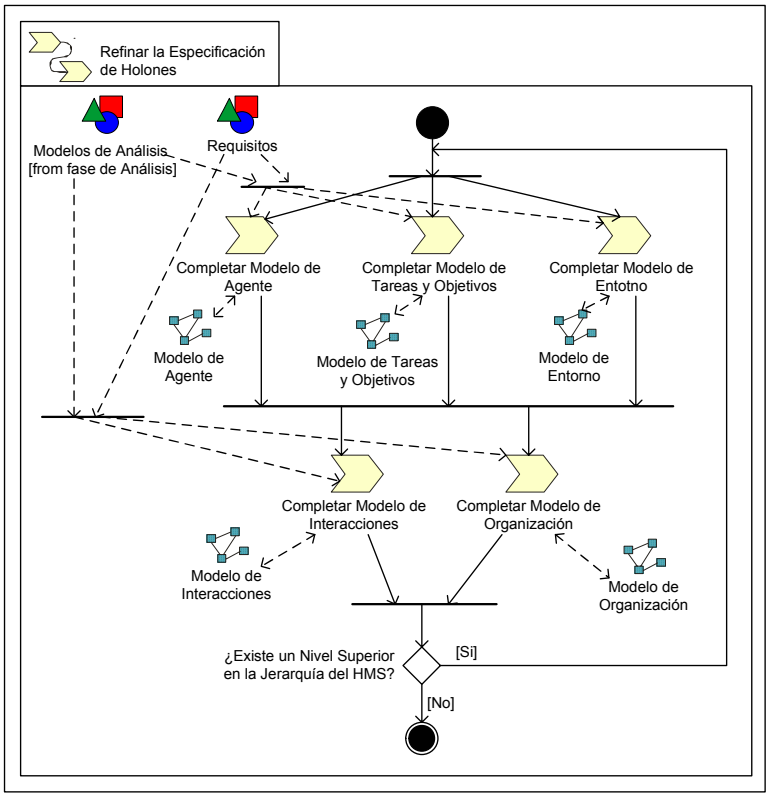


Figura 5.38: Definición de la tarea Refinar la Especificación de Holones.

Actividad D2: Completar Modelo de Tareas y Objetivos. En esta actividad el Ingeniero del Software debe, independientemente de la iteración, asegurar que cada objetivo del Agente Abstracto tiene una tarea correspondiente que logra satisfacerlo, relación GTSatisface. Establecer dependencias entre objetivos, relación GTDepende. Para todas las tareas modeladas se deben identificar Pre y Post condiciones, restricciones temporales (si fueran necesarias). Asociar tareas con interacciones (relación, WFProduce), con entidades mentales (relaciones GTModifica, GTDestruye y GTCrea) y con operaciones de aplicaciones (relación WFUsa).

Actividad D3: Completar Modelo de Entorno. En la primera iteración, esta actividad requiere que el Ingeniero del Software se centre en diseñar la interacción de los holones atómicos con su entorno. Para ello debe, refinar la relación EPercibe de la fase de análisis, por las relaciones EPercibeNotificacion y EPercibeMuestreo para indicar el tipo de percepción. Debe definir, también, los atributos de los recursos del entorno con los que trabaja el agente y asociar recursos, aplicaciones y tareas. Completar la definición de los eventos que afectan a los agentes, teniendo especial interés en refinar las posibles características temporales: patrón de ocurrencia y tiempo mínimo entre ocurrencias. En las iteraciones sucesivas, el Ingeniero del Software debe asegurar que la percepción del Agente Abstracto se define mediante el conjunto de percepciones de sus miembros, esto es, ningún recurso, evento o aplicación de un nivel de recursión dado puede no tener una correspondiente percepción o actuación en el nivel inferior.

Actividad D4: Completar Modelo de Interacciones. En la primera iteración, esta actividad no se aplica. En las siguientes iteraciones, esta actividad tiene como objetivo diseñar el conjunto de interacciones que se llevan a cabo entre los miembros (de niveles de recursión menor) de un Agente Abstracto para conseguir satisfacer los objetivos de este último. Para ello, el Ingeniero del Software debe centrarse en los miembros del Agente Abstracto y en las interacciones de estos, para en base a ello: traducir mensajes a unidades de interacción, establecer un orden de ejecución entre estas unidades, establecer condiciones mentales para la ejecución de las unidades de interacción y establecer restricciones temporales (si fueran necesarias) sobre la ejecución de las unidades de interacción.

Actividad D5: Completar Modelo de Organización. En la primera iteración, esta actividad no se aplica. En las siguientes iteraciones, esta actividad tiene como objetivo diseñar las dependencias sociales entre los miembros del Agente Abstracto que se está diseñando. Para ello, el Ingeniero del Software debe refinar las dependencias entre los miembros mediante relaciones de subordinación y relaciones cliente-servidor.

Construir la Arquitectura del Sistema.

Esta tarea compleja tiene como objetivo completar la Arquitectura del Sistema para que pueda ser implementada en la fase de implementación del proceso de desarrollo. Para ello, el Ingeniero del Software debe incluir detalles de la plataforma de implementación. En nuestro enfoque estos detalles se recogen mediante la especificación de Plantillas de Agentes JADE y la Especificación de Interfaces de Bloques Funcionales. En esta etapa, el Ingeniero del Software debe centrar su atención sólo en los holones atómicos (agentes) identificados y especificados en los Modelos de Diseño. Esto se debe a que los Agentes Abstractos son entidades de modelado (no se implementan, ver Capítulo 4) y cuya funcionalidad "teórica" se implementa mediante los agentes del nivel de recursión $n = 0$ (implementables).

Esta tarea consiste de 4 actividades (Figura 5.39) que se definen de la siguiente manera:

Actividad D6: Determinar Distribución. En esta actividad el Ingeniero del Software debe determinar el número de plataformas de agentes necesarias para distribuir el sistema. Dependiendo de las características del HMS, el Ingeniero del Software debe determinar si el sistema final será desplegado en una única plataforma de agentes o en en varias. Para determinar este número debe hacer uso de las Guías JADE 1 a 4 (Tabla 5.7). El producto de esta actividad es una lista de Plataformas Identificadas, donde se indica por cada plataforma un nombre, un número y la lista de agentes que se implementarán en ella.

Actividad D7: Definir Plantillas de Agentes. En esta actividad el Ingeniero del Software debe especificar por cada agente del Modelo de Diseño una Plantilla de Agente JADE. La Plantilla de Agente JADE contiene características específicas de la plataforma JADE [Bellifemine *et al.*, 2001; JADE, 2005] tales como identificadores de agentes, comportamientos, servicios, comunicación, ontologías, etc. Para construir estas plantillas el Ingeniero del Software debe hacer uso de las Guías JADE 5 a 12 (Tablas 5.7 y 5.8).

Actividad D8: Definir Interfaz de Bloques Funcionales. En esta actividad el Ingeniero del Software debe completar, por cada agente con parte de procesamiento físico, la Especificación de la Interfaz de Bloques Funcionales de las tareas del agente destinadas al control del recurso (equipo, máquina, máquina

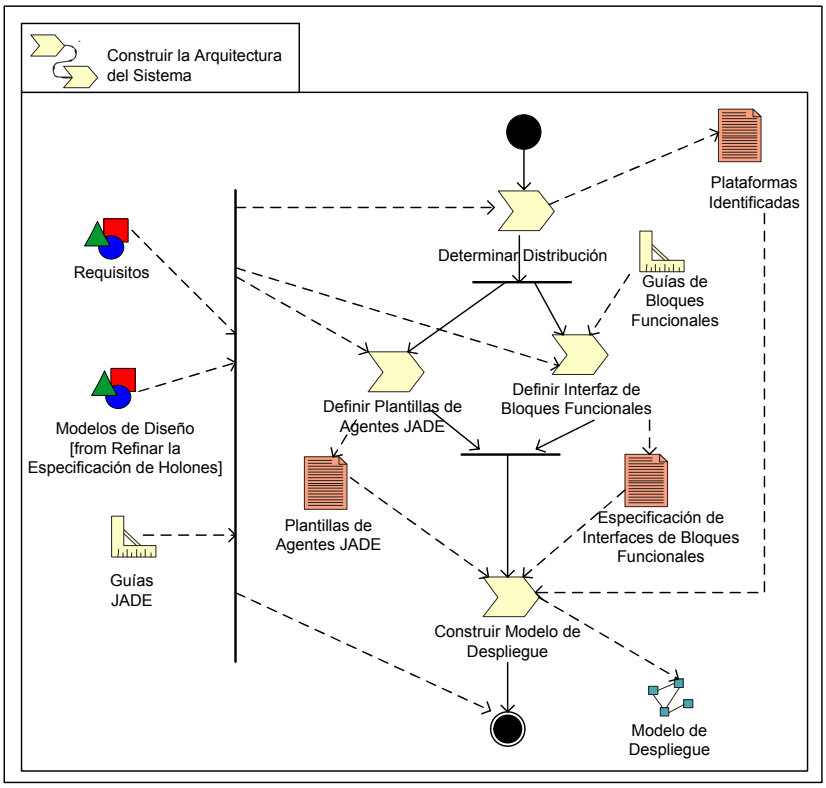


Figura 5.39: Definición de la tarea Construir la Arquitectura del Sistema.

herramienta, etc.). La Especificación de Interfaz de Bloques Funcionales contiene una tabla en la que el Ingeniero del Software debe indicar las secuencias, deseadas de operaciones; las secuencias anormales que pueden ocurrir; el comportamiento del dispositivo en respuesta a comandos enviados a actuadores que producen una respuesta, dependiente del tiempo, percibida mediante sensores. Además, se pueden incluir Diagramas de Transición de Estados para representar el comportamiento del dispositivo. Para completar estas plantillas el Ingeniero del Software debe hacer uso de las Guías de Bloques Funcionales 3 a 7 (Tabla 5.9).

Actividad D9: Construir Modelo de Despliegue. En esta actividad el Ingeniero del Software debe construir un Modelo de Despliegue UML [OMG, 2005]

para mostrar la disposición física de los distintos nodos (recurso de ejecución tal como computador, dispositivo o memoria) que componen al sistema y el reparto de las plataformas y contenedores sobre dichos nodos. Para ello debe hacer uso del documento de Requisitos, los Modelos de Diseño, y el documento de Plataformas Identificadas.

5.3.2.4. Implementación de Holones

En la fase de implementación de holones, el Programador debe implementar el HMS siguiendo la Arquitectura del Sistema definida en la fase de desarrollo anterior. Para ello el desarrollador debe ejecutar las siguientes actividades:

Actividad I1: En esta actividad el Programador debe definir e implementar las distintas plataformas de agentes identificadas en la Arquitectura del Sistema. Para ello debe seguir las guías del programador de JADE [JADE, 2005].

Actividad I2: En esta actividad el Programador debe implementar los distintos agentes identificados en la Arquitectura del Sistema, para ello puede utilizar las guías del programador y de administración de JADE [JADE, 2005]. Además, debe realizar pruebas de integración del agente en la plataforma.

Actividad I3: En esta actividad el Programador debe implementar nuevos bloques funcionales y/o reutilizar bloques funcionales previamente definidos, utilizando el lenguaje de especificación IEC 61131-3 Structured Text [IEC, 2000]. Además, debe realizar pruebas de integración de los bloques funcionales en los dispositivos, máquinas o equipos a controlar. En esta actividad debe seguir las guías de programación definidas en la familia de estándares IEC 61499 [IEC, 2000; 2001].

Actividad I4: En esta actividad el Programador debe integrar, por cada holón, la parte de procesamiento de la información implementada como agente JADE con la parte de procesamiento físico implementada mediante bloques funcionales. Además debe realizar pruebas de integración local (intra-holón) e integración global (inter-holón). En esta actividad puede utilizar técnicas de prueba tradicionales de la Ingeniería del Software.

Tabla 5.7: Guías JADE - Parte 1

 Guías JADE	
ID	**Guía**
1	Si el HMS representa a un sistema de fabricación donde se observa distribución física, esto es departamentos, plantas de fabricación físicamente distribuidos y/o distintas sucursales, el control de cada uno de estos componentes puede ser gestionado en una plataforma de agente distinta. Las interacciones entre los agentes que "habitan" en distintas plataformas se gestionará mediante comunicación inter-plataforma.
2	Si el HMS representa a un sistema de fabricación sin distribución física, el Ingeniero del Software debe analizar los Agentes Abstractos identificados en los Modelos de Diseño. Si se trata de un sistema con dominios de cooperación muy relacionados, compuesto por Agentes Abstractos que participan en varios dominios de cooperación y donde el número de agentes no es muy elevado, entonces el Ingeniero del Software puede considerar la posibilidad de implantar todo el HMS en una sola plataforma de agentes.
3	Si el HMS representa a un sistema de fabricación sin distribución física, el Ingeniero del Software debe analizar los Agentes Abstractos identificados en los Modelos de Diseño. Si el sistema está compuesto por un número muy elevado de agentes, el Ingeniero del Software debe considerar la posibilidad de distribuir el HMS en un número de plataformas de agentes distintas. Para determinar ese número, debe analizar las relaciones y dependencias entre los dominios de cooperación, requisitos de seguridad y encapsulamiento de información, funcionalidad, interacciones internas, optimización de interacciones, etc. , y agrupar los dominios de cooperación de acuerdo a estos criterios. El número de grupos debe ser considerado como tentativo para definir el número de plataformas.
4	En las guías 1, 2 y 3 considerar también los principios de la Ingeniería del Software sobre alta cohesión y bajo acoplamiento, aplicar el estudio a las plataformas identificadas.
5	Para la parte de procesamiento de la información de cada agente completar la plantilla de la Figura 5.40, siguiendo las guías 6 a 12.
6	En el punto 1 de la plantilla de agente JADE (Figura 5.40), completar con una referencia al nombre del agente en los Modelos de Diseño.
7	En el punto 2 de la plantilla de agente JADE (Figura 5.40), completar con el nombre de la plataforma a la que pertenece el agente según el documento Plataformas Identificadas.
8	Para el apartado 3 de la plantilla de agente JADE (Figura 5.40), revisar los Modelos de Agente y los Modelos de Tareas y Objetivos para completar los apartados 3.1 y 3.2 con todas las tareas que el agente ofrece como servicios. En el apartado 3.2 indicar si el servicio se ofrece dentro de la plataforma del agente o, también, a agentes de otras plataformas, analizar para ello los Modelos de Interacción en los que aparece el agente.

Tabla 5.8: Guías JADE - Parte 2

 Guías JADE	
ID	**Guía**
9	En el apartado 4 de la plantilla de agente JADE (Figura 5.40), identificar comportamientos que implementarán los servicios identificados en el punto 3, y las tareas internas del agente (que no se ofrecen como servicios) revisar para ello los Modelos de Agente y los Modelos de Tareas y Objetivos. En el caso en que el comportamiento identificado implemente un servicio, indicar en el punto 4.3 el nombre del servicio que implementa. Por cada comportamiento indicar, en el punto 4.2, su tipo (Simple, Cíclico, OneShot, Compuesto, Secuencial, Paralelo, FSM).
10	En el apartado 5 de la plantilla de agente JADE (Figura 5.40), indicar los elementos que puede utilizar el agente dentro del contenido de mensajes (vocabulario). Para ello analizar el Modelo de Agente y los Modelos de Interacción para recoger las entidades mentales de tipo información y los contenidos de los mensajes de las interacciones. En el punto 5.1 se indica el nombre, en el 5.2 el nombre de una o más ontologías que se está extendiendo, y en el punto 5.3 el conjunto de elementos que definen la estructura del concepto.
11	En el apartado 6 de la plantilla de agente JADE (Figura 5.40), indicar el conjunto de mensajes que puede intercambiar el agente. Para ello analizar los Modelos de Interacción, específicamente las Unidades de Interacción que el agente puede intercambiar. Completar el punto 6.1 con el nombre de mensaje, el 6.2 con el tipo de la Unidad de Interacción, el 6.3 con una referencia al nombre de la interacción de los Modelos de Diseño, y en el 6.4 el tipo de participación (iniciador, colaborador).
12	En el apartado 7 de la plantilla de agente JADE (Figura 5.40), indicar si el agente tiene parte de procesamiento físico (es decir, es un holón de recurso que controla una máquina), en caso afirmativo completar con la referencia a los documentos de la Especificación de Bloques Funcionales que modelan las tareas de procesamiento físico del agente.

Tabla 5.9: Guías Bloques Funcionales

 Guías de Bloques Funcionales	
ID	**Guía**
1	Un bloque funcional puede ser visto como una unidad de procesamiento. Los bloques funcionales son utilizados como bloques de construcción para definir una aplicación de control o monitorización de dispositivos. Algunos bloques funcionales proveen comportamiento de control y otros proveen comportamiento de entrada/salida.
2	Algunos dispositivos incluyen bloques funcionales pre-instalados, que no pueden ser borrados y algunas veces tampoco permiten que nuevos bloques funcionales sean añadidos. Otros dispositivos en cambio permiten que se puedan añadir y eliminar bloques funcionales como sean necesarios. Por lo tanto, es necesario estudiar la especificación del dispositivo que se quiere controlar para determinar la definición de la interfaz, la posibilidad de crear nuevos bloques funcionales, modificar o eliminar bloques existentes.
3	Completar una plantilla de Especificación de Interfaces de Bloques Funcionales (Figura 5.41) por cada tarea (parte de procesamiento físico) de cada agente (holón de recurso) definido en los Modelos de Diseño, siguiendo las guías 4 a 7.
4	Completar el punto 1 con el nombre del agente que se está diseñando, el punto 2 con el nombre de la plataforma a la que pertenece y el punto 4 con el nombre de la tarea de procesamiento físico que se está diseñando.
5	Asignar en el punto 3 un identificador único a la plantilla de especificación.
4	Estudiar la especificación del fabricante sobre el dispositivo que se desea controlar y la tarea definida en el Modelo de Agente y en el Modelo de Tareas y Objetivos para determinar la secuencia de operaciones que se desea controlar, así como posibles secuencias anormales. Completar los puntos 5 y 6 con esta información.
6	Estudiar la especificación del fabricante sobre el dispositivo para determinar su comportamiento, y el documento de Requisitos para determinar las condiciones de operación. Completar el apartado 7 con una entrada por cada combinación: comando de entrada, dirección de actuador al que se envía el comando, dirección del sensor que obtiene la respuesta, respuesta esperada, tiempo para obtener una respuesta.
7	Completar el punto 8 con un diagrama de transición de estados para modelar el comportamiento del dispositivo.

5.3.2.5. Instalación y Configuración

En la fase de instalación y configuración, el Ingeniero del Software debe guiar el proceso de implantación del HMS en el entorno de ejecución destino (fábrica, máquina, taller, celda, etc.). En esta fase se deben realizar los ajustes de configuración del sistema para satisfacer los objetivos de desempeño especificados en el documento de Requisitos. En esta actividad se pueden aplicar las técnicas tradicionales de Ingeniería del Software, y las definidas en el estándar IEC 61499 [IEC, 2000; 2001].

5.3.2.6. Operación y Mantenimiento

En esta fase de deben utilizar las técnicas tradicionales de la Ingeniería del Software, teniendo en cuenta que cualquier actividad de mantenimiento que involucre nuevas funcionalidades, adaptación a cambios en el entorno y corrección de posibles defectos de implementación o diseño, implica la ejecución de un nuevo proceso de desarrollo para adaptar los documentos de análisis, diseño e implementación a las nuevas características.

5.4. Conclusiones

En este capítulo hemos presentado una metodología Multi Agente para el desarrollo de Sistemas Holónicos de Fabricación. Nuestra propuesta se basa en los 8 requisitos de modelado para Sistemas Holónicos de Fabricación que hemos presentado en el Capítulo 2 y en la noción de Agente Abstracto para el modelado de entidades autónomas con estructuras recursivas (Capítulo 4).

La notación de nuestra metodología se define a partir de INGENIAS [Gómez, 2002] y RT-MESSAGE [Julián, 2002] y se fundamenta en los requisitos 1, 2, 3 y 4 para el modelado de sistemas holónicos. De INGENIAS hemos adoptado la división del sistema en cinco vistas y la definición de la notación mediante meta-modelos. De RT-MESSAGE hemos adoptado la identificación de entidades para el modelado de las características de tiempo real de sistemas multi agente. En nuestro enfoque la entidad de modelado central es el Agente Abstracto. Hemos extendido los meta-modelos de INGENIAS para incluir la noción de Agente Abstracto, nuevas entidades para modelar su estructura y

comportamiento, características de tiempo real definidas en RT-MESSAGE y relaciones nuevas y/o modificadas. Es importante resaltar que la notación que proponemos es general (no dependiente de HMS), y por lo tanto puede ser utilizada además para el modelado de SMAs de gran escala en dominios generales.

El proceso de desarrollo de nuestra metodología es específico para el dominio HMS y se fundamenta en los 8 requisitos de modelado de Sistemas Holónicos de Fabricación. Es un proceso mixto (ascendente y descendente), está guiado por la noción de Agente Abstracto para la identificación y modelado de holones, proporciona guías de modelado claras y específicas para el dominio HMS, y cubre todo el ciclo de vida del HMS mediante fases de desarrollo completas.

En la definición del proceso de desarrollo hemos utilizado SPEM (*Software Process Engineering Metamodel*) [OMG, 2002]. De esta manera: (i) ofrecemos al ingeniero del software una definición fácil de interpretar, y; (ii) facilitamos también, siguiendo el enfoque de [FIPA Methodology TC, 2005], la definición de fragmentos metodológicos (fases o modelos) para su posible reutilización en conexión con fragmentos de otras metodologías.

Nuestra metodología se basa en un proceso de desarrollo de 5 fases.

- **Análisis**. El objetivo de la fase de análisis es identificar los holones que componen el sistema y proveer una especificación de alto nivel del HMS. Esta fase sigue un enfoque descendente (top-down), recursivo, e incremental. El ingeniero del software desarrolla un análisis por capas de abstracción guiado por la noción de Agente Abstracto para identificar los holones y los dominios de cooperación (escenarios) en los cuales éstos pueden participar. En esta fase el ingeniero del software especifica el HMS en términos de: modelos de agente, modelos de interacción, modelos de organización, modelos de entorno y modelos de tareas y objetivos. Para completar estos modelos nuestra metodología ofrece guías específicas del dominio HMS, estas guías se basan en la arquitectura PROSA [Van Brussel *et al.*, 1998] y los dominios de cooperación de [Fletcher *et al.*, 2000].

  Una ventaja de una fase de análisis recursiva es que su resultado provee

un conjunto de componentes elementales y un conjunto de reglas de composición en base a las cuales se puede definir una etapa de diseño ascendente.

- **Diseño de Holones**. El objetivo de la fase de diseño es definir la arquitectura del sistema a partir de los modelos obtenidos en la fase de análisis. Esta fase sigue un enfoque ascendente guiado por los Agentes Abstractos identificados en la fase anterior. En esta fase el ingeniero del software diseña el sistema teniendo en cuenta las características de la plataforma destino de implementación. Nuestra metodología ofrece guías específicas para el diseño del HMS siguiendo el enfoque de [Christensen, 2003]. Para el control de alto nivel del holón (procesamiento de la información) ofrecemos guías de diseño para la plataforma JADE [JADE, 2005], estándar FIPA [FIPA, 2005], mientras que para el control de bajo nivel del holón (procesamiento físico) ofrecemos guías de diseño para bloques funcionales, estándar IEC 61499 [IEC, 2000; 2001].

- **Implementación de Holones**. El objetivo de esta fase es la implementación del HMS diseñado en la fase anterior. En esta fase el programador hace uso de las guías de programación y administración de JADE [JADE, 2005] y de las guías de programación definidas en el estándar IEC 61499 [IEC, 2000; 2001] para el lenguaje de especificación IEC 61131-3 Structured Text.

- **Instalación y Configuración**. El objetivo de esta fase es la instalación y configuración del HMS en el entorno de ejecución destino (fábrica, máquina, taller, celda, etc.). En esta actividad se aplican las técnicas tradicionales de Ingeniería del Software, y las definidas en el estándar IEC 61499 [IEC, 2000; 2001].

- **Operación y Mantenimiento**. El objetivo de esta fase es mantener operativo el HMS en su entorno de ejecución y realizar las operaciones de mantenimiento necesarias, para ello se utilizan las técnicas tradicionales de la Ingeniería del Software.

En el siguiente capítulo ilustramos la aplicación de nuestra metodología en el desarrollo de un caso de estudio de una empresa cerámica.

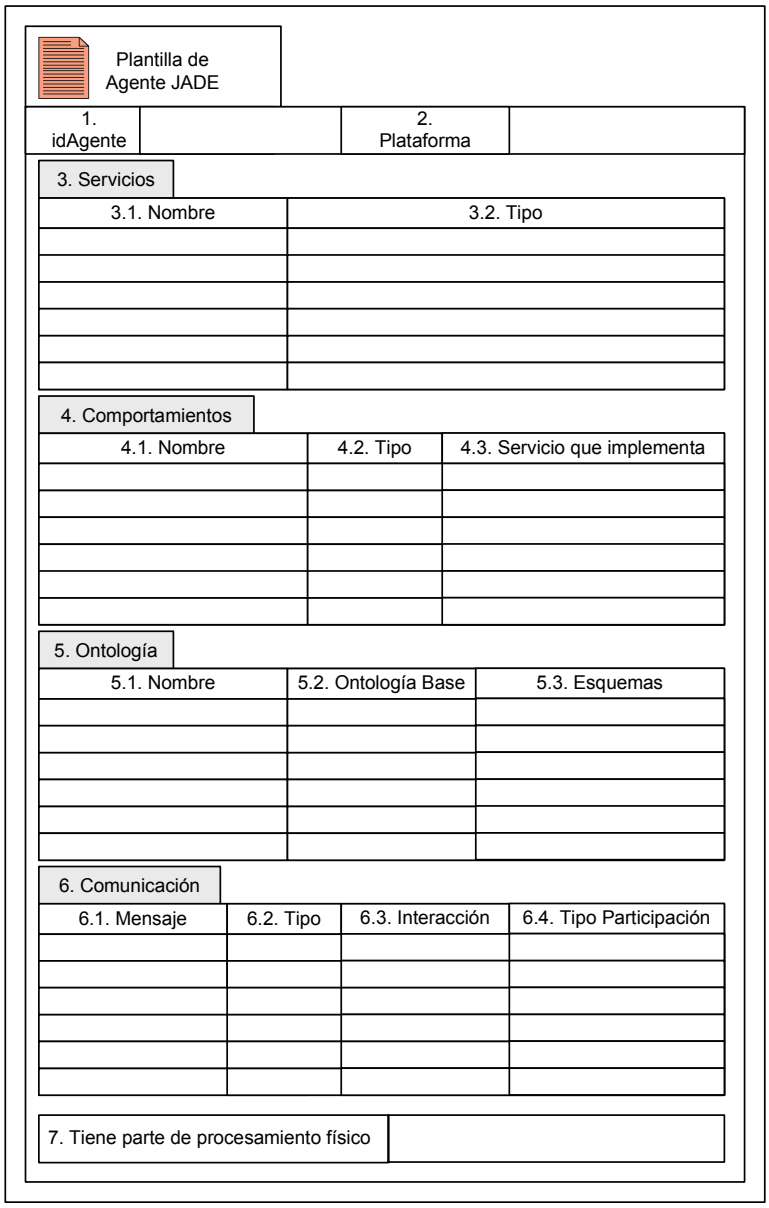


Figura 5.40: Definición del documento Plantilla de Agente JADE.

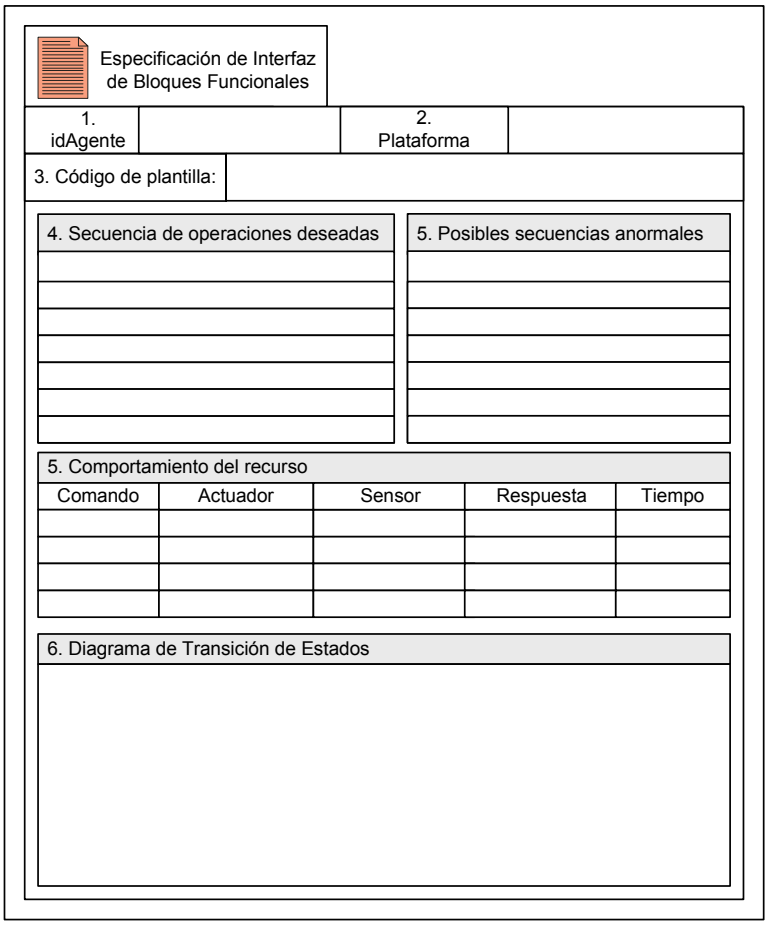


Figura 5.41: Definición del documento Especificación de Interfaces de Bloques Funcionales.

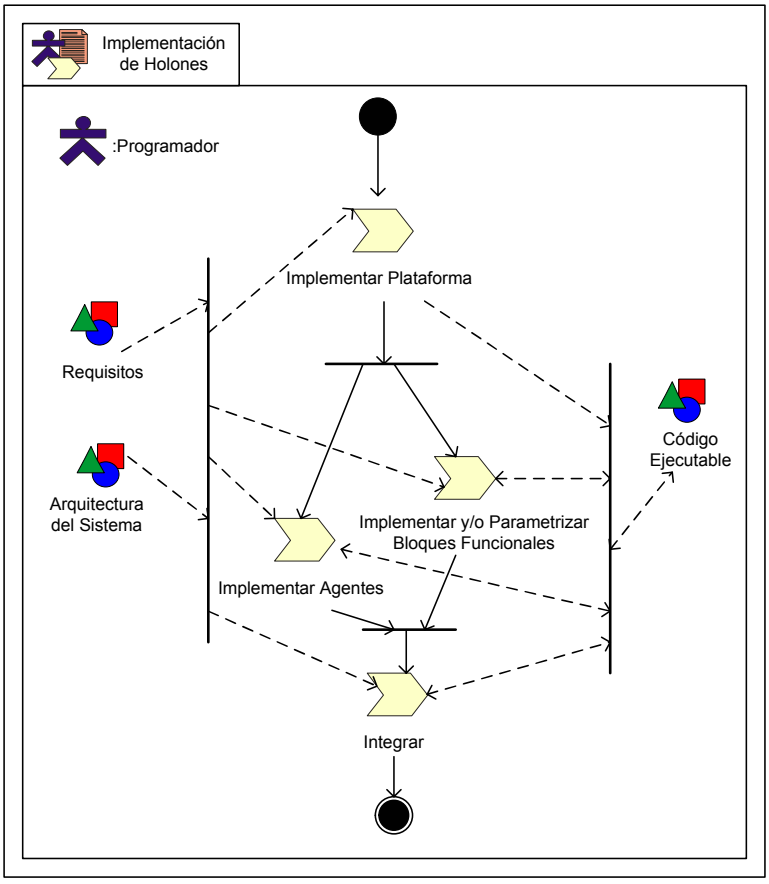


Figura 5.42: Fase de Implementación de Holones.

Capítulo 6

Caso de Estudio

En este capítulo presentamos y desarrollamos un caso de estudio de una empresa del sector cerámico. El trabajo que presentamos en este capítulo se enmarca dentro del proyecto interdisciplinar "*Implementación y Aplicación de Sistemas Multi Agente en Entornos Industriales con Restricciones Temporales*" financiado por el Vicerrectorado de Investigación, Desarrollo e Innovación de la Universidad Politécnica de Valencia. Este proyecto es una colaboración del Grupo de Tecnología Informática e Inteligencia Artificial, GTI-IA (al cual pertenezco) y el Centro de Investigación Gestión e Ingeniería de la Producción, CIGIP, de esta Universidad.

Este capítulo se estructura siguiendo las distintas fases de nuestra metodología. En primer lugar presentamos el documento de Requisitos del Sistema (Sección 6.1), en la Sección 6.2 ilustramos los distintos productos de las actividades de la fase de análisis. En la Sección 6.3 presentamos la Arquitectura del Sistema resultado de la aplicación de las actividades y guías de la fase de diseño. Finalmente, en la Sección 6.4 presentamos las conclusiones del presente capítulo.

6.1. Requisitos

KCG es una empresa dedicada al diseño, fabricación y comercialización de pavimentos y revestimientos cerámicos. Sus productos se venden en más de 150 países. El Grupo KCG está formado por:

- **KT**: diseño, fabricación y comercialización de productos cerámicos.
- **MT**: diseño, fabricación y comercialización de productos cerámicos.
- **WD**: diseño y fabricación de esmaltes cerámicos.
- **KR Tiendas**: comercialización de productos cerámicos, hidromasaje, sanitarios, muebles de baño, grifería, complementos de baño y adhesivos para la construcción.
- **Delegaciones**: en diferentes países.

Los productos cerámicos de KCG abarcan: pavimentos y revistimientos tradicionales de pasta roja y blanca, revistemientos pulidos y rectificados, porcelánicos técnicos y esmaltados. Los clientes de KCG son: empresas constructoras (con las que se trabaja generalmente bajo pedido), almacenistas y clientes al menudeo (con los que se trabaja según el stock de almacén de productos terminados). Sus proveedores se dividen en tres grupos, según el material que proveen: (i) proveedores de arcilla; (ii) proveedores de "fritas" (productos sólidos que mezclados con elementos líquidos producen esmaltes) y esmaltes; y (iii) proveedor de gas.

En los siguientes apartados describimos las distintas partes del documento de Requisitos del sistema a desarrollar.

6.1.1. Organigrama/Departamentos

La Figura 6.1 muestra el organigrama actual de KCG. Para simplificar el diagrama, mostramos sólo los jefes/directores de área o departamento, sin incluir el personal asignado a cada departamento. La estructura organizativa se define de la siguiente manera:

- Presidente Grupo KCG: es la cabeza de KCG. Se encarga de las funciones de gestión global, para ello mantiene una relación de jefe-subordinado con: el Jefe de Delegaciones, el Jefe de Diseño, el Jefe de Marketing, el Jefe de Tiendas, el Jefe de Empresas Cerámicas y el Jefe de Transporte.

- Jefe de Delegaciones: es la interfaz de las delegaciones en el grupo KCG. Se encarga de la gestión de las distintas delegaciones de KCG en los diferentes países, para ello mantiene una relación de jefe-subordinado con los distintos jefes de cada delegación. Para las funciones de cooperación (solicitud de fabricación, información de marketing, exportación de productos, etc.) con los demás departamentos del grupo KCG interactúa con: el Jefe de Diseño, el Jefe de Marketing, el Jefe de Tiendas, el Jefe de Transporte y el Jefe de Empresas Cerámicas.

- Jefe de Diseño: es el encargado de la gestión y definición de los diseños de los productos del grupo KCG. Para ello coopera con el Jefe de Empresas Cerámicas para comunicar diseños de productos y aprobar diseños definidos por KT, WD y MT. Comunica y recibe información sobre histórico de ventas del Jefe de Marketing, del Jefe de tiendas y del Jefe de Delegaciones.

- Jefe de Marketing: es el encargado de la gestión de ventas globales y campañas de publicidad del grupo KCG. Para ello mantiene una relación de jefe-subordinado con el Jefe de Ventas y el Jefe de Publicidad. Además, coopera con el Jefe de Delegaciones, el Jefe de Diseño, el Jefe de Tiendas, el Jefe de Transporte y el Jefe de Empresas Cerámicas, para recoger información de ventas y comunicar políticas de marketing del grupo.

- Jefe de Tiendas: es el encargado de la gestión global de las distintas tiendas del grupo KCG. Para ello mantiene una relación de jefe-subordinado con los distintos Jefes de Tienda KR. Además, coopera con el Jefe de Delegaciones, el Jefe de Diseño, el Jefe de Marketing, el Jefe de Transporte y el Jefe de Empresas Cerámicas, para comunicar pedidos de fabricación de clientes, información sobre diseños más vendidos en tiendas, orden de transporte de productos de fábrica a tienda.

- Jefe de Empresas Cerámicas: es el encargado de la gestión de colaboración de las distintas empresas cerámicas del grupo KCG. Para ello mantiene una relación de jefe-subordinado con el Director de KT, el

Director de WD y el Director de MT. Además, coopera con el Jefe de Delegaciones, el Jefe de Diseño, el Jefe de Tiendas, el Jefe de Transporte y el Jefe de Marketing, para comunicar información sobre ventas directas de almacén, comunicar y recibir información sobre diseño de productos, recibir órdenes de transporte de productos y recibir órdenes de fabricación.

- Jefe de Transporte: es el encargado de la gestión del transporte de productos de una empresa cerámica a otra empresa cerámica, de una empresa cerámica a una tienda, de una tienda a otra tienda, etc. Para ello coopera con el Jefe de Tienda y el Jefe de Empresas Cerámicas.

- Jefe de Delegación i: es el encargado de la gestión de una delegación de KCG en un país determinado, se encuentra bajo la dirección del Jefe de Delegaciones para las políticas de empresa del grupo KCG.

- Jefe de Ventas: es el encargado de gestionar las ventas globales del grupo KCG. Se encuentra bajo la dirección del Jefe de Marketing para registrar las ventas de tiendas, ventas directas de almacén de KT, WD y MT, y las ventas de las distintas delegaciones.

- Jefe de Publicidad: se encarga de definir y gestionar las políticas de publicidad del grupo KCG.

- Jefe de KR Tienda i: se encarga de la gestión de una tienda KR. Comunica al Jefe de Tiendas las órdenes de compra y/u órdenes de fabricación, recibe catálogo y muestras de productos.

- Director KT: se encarga de la dirección de la empresa cerámica KT según la política del grupo KCG. Bajo su mando se encuentran los distintos departamentos de KT: Compras KT, Diseño KT, Producción KT, RRHH KT, Almacén KT, Ventas KT.

- Jefe de Compras KT: se encarga de la gestión de proveedores de materia prima de la empresa KT. Controla a los Jefes de Almacén de materia prima de las dos fábricas de KT.

- Jefe de Diseño KT: se encarga de definir y mantener los diseños de productos a fabricar por KT. Algunos de estos diseños pueden ser definidos por el departamento de Diseño del grupo KCG.
- Jefe de Producción KT: se encarga de gestionar y controlar los procesos de planificación, secuenciación, lanzamiento y control de producción de KT. Coordina la labor de los jefes de Programación y de Fábricas de KT.
- Jefe de RRHH KT: se encarga de las labores de gestión de los empleados de KT.
- Jefe de Almacén KT: se encarga de gestionar el almacén de productos terminados de KT.
- Jefe de Ventas KT: se encarga de gestionar las ventas directas de almacén de productos terminados de KT.
- Jefe de Programación KT: se encarga de la definición del plan y el calendario de producción de KT.
- Jefe de Fábricas KT: se encarga de controlar y coordinar al Jefe de Fábrica 1 KT y al Jefe de Fábrica 2 KT.
- Jefe de Fábrica i KT: se encarga de la gestión de lanzamientos de producción y el control de producción en la fábrica i de KT.
- Jefe de Prensa y Esmaltado Fábrica i KT: se encarga de la gestión de las distintas líneas de prensa y esmaltado de la fábrica i de KT.
- Jefe de Hornos Fábrica i KT: se encarga de la gestión de los distintos hornos de la fábrica i de KT.
- Jefe de Clasificación Fábrica i KT: se encarga de la gestión de las distintas líneas de clasificación y embalaje de la fábrica i de KT.
- La definición de los departamentos de las empresas WD y MT son similares a KT.

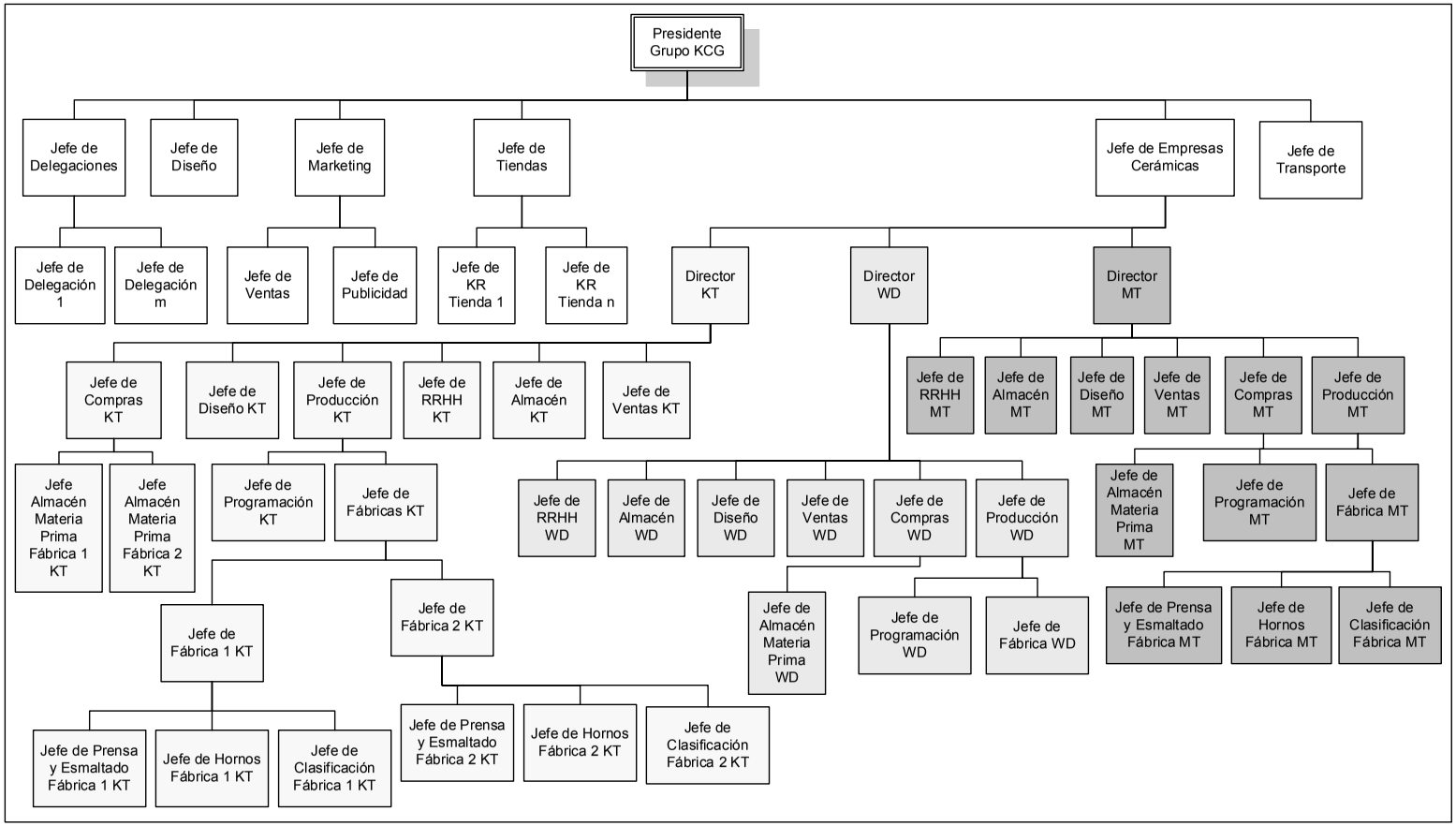


Figura 6.1: Organigrama de KCG.

6.1.2. Procesos de Negocio

Los procesos de negocio se ilustran (en notación DFD) en la Figura 6.2. Podemos observar que existe un proceso para la definición de productos (proceso 1). Un conjunto de procesos (procesos 2, 3, 4, 5 y 6) para la producción de productos cerámicos, para atender pedidos de empresas constructoras, o para mantener un stock de almacén de productos terminados para servir a almacenistas y clientes al menudeo. Un proceso de compras (proceso 8) para comprar materia prima a proveedores. Y finalmente, un proceso de ventas (proceso 7) que se alimenta de los productos terminados y atiende a clientes de la empresa.

En el DFD de la Figura 6.2 podemos ver que el proceso 1 - Definición de los productos cerámicos a producir - se activa a partir de Muestras Propias de productos cerámicos (definidos por el Departamento de Diseño de KCG) y de Muestrarios de Proveedores de Fritas y Esmaltes, el resultado del proceso 1 es la definición del listado con los Tipos de productos (Catálogo de Productos) a producir para una temporada. Este listado contiene la especificación de cada tipo de producto que incluye el material a utilizar, el diseño del producto, tipo de matriz y fondo a utilizar en el prensado, configuración de máquinas de esmaltado, configuración del horno de cocción del producto cerámico y expectativa de venta por producto.

El proceso global de producción es básicamente la secuencia de los procesos 2, 3, 4, 5 y 6. El proceso 2 - Análisis de Previsión de Demandas - utiliza como datos de entrada los Tipos de productos y el Histórico de Ventas, para a partir de estos, calcular la Previsión de Demandas para un periodo. La Previsión de Demandas se utiliza para mantener unos niveles de Almacén de Productos Terminados. En la actualidad este proceso lo gestiona en su totalidad el Jefe de Producción quien con la ayuda de una herramienta informática define una previsión inicial que posteriormente la comunica al Departamento de Marketing donde se valida y se ajusta si fuera necesario. Este proceso es muy importante para la posterior definición del plan maestro. La previsión de demandas se basa en el histórico de ventas a empresas constructoras y clientes al menudeo.

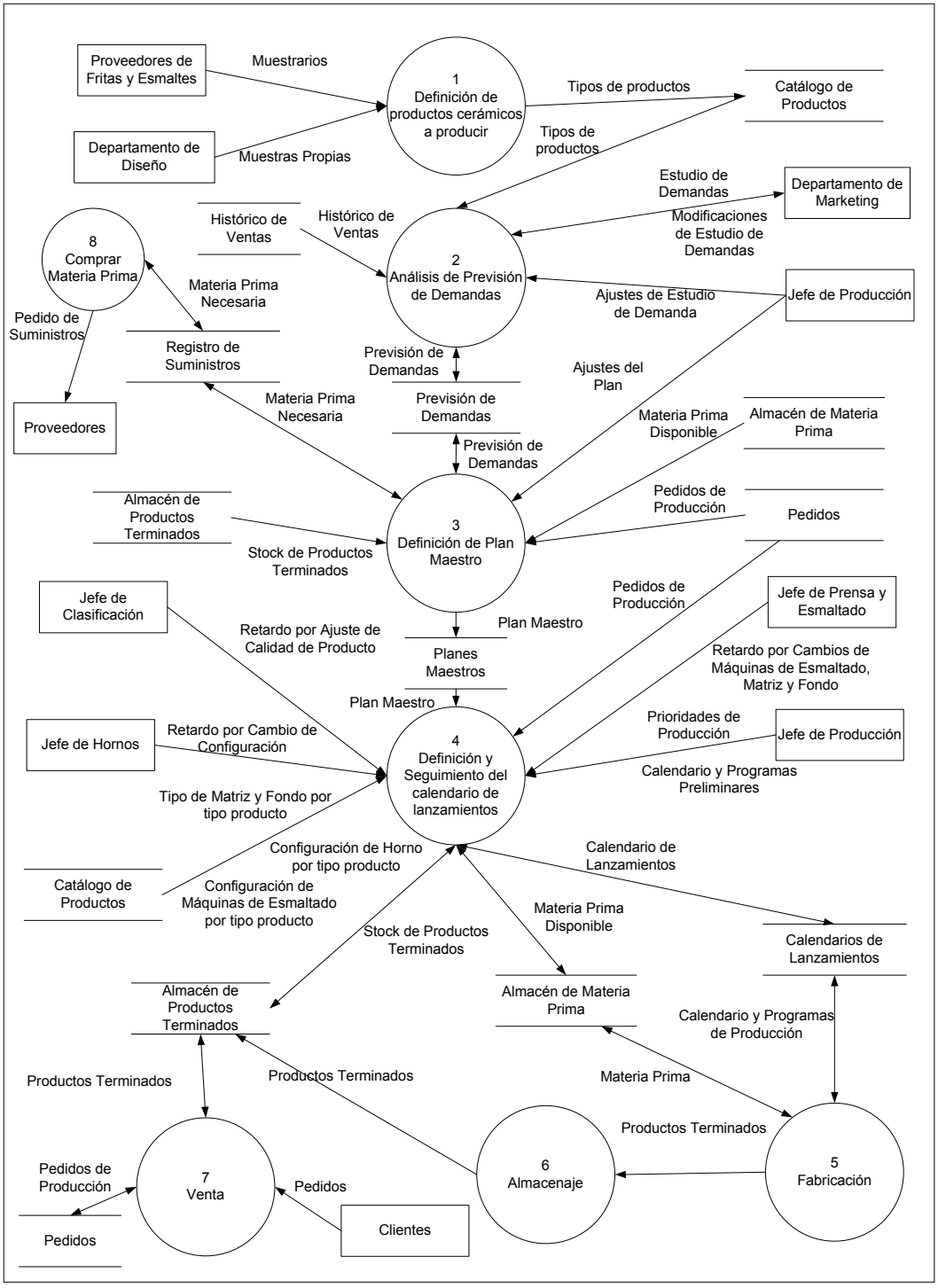


Figura 6.2: Procesos de Negocio de KCG.

Es en el proceso 3 - Definición de Plan Maestro - donde el Jefe de Producción define el Plan Maestro [1] (con un alcance de tres o cuatro meses) y donde se confeccionan pedidos de suministros a Proveedores (si fuera necesario) a partir de los Pedidos de Producción, la Previsión de Demandas, la Materia Prima Disponible en Almacén y el Stock de Productos Terminados. El Plan Maestro se compone de órdenes de fabricación (donde se especifica modelos, tamaño de lote a fabricar y prioridad, esta prioridad está marcada basándose en criterios fundamentalmente comerciales, por ejemplo importancia del cliente cuyo pedido ha originado la orden).

El jefe de Producción interactúa con los distintos Jefes de áreas de fábricas (esmaltado, prensa, horno y clasificación) para concretar la lista de órdenes de fabricación a ejecutar en los próximos cinco o seis días (entre las más prioritarias del Plan Maestro), sobre la base de los informes de incidencias de los responsables de las diferentes secciones, proceso 4. En el proceso 4 - Definición y seguimiento del calendario de lanzamientos - se realizan las tareas de carga, secuenciación y temporización. Este proceso es el más importante para la productividad de la fábrica. Utiliza como datos de entrada el Plan Maestro, informes de los distintos responsables de secciones, disponibilidad de materia prima, stock de almacenes de productos terminados, revisa los calendarios definidos en reuniones previas y añade nuevas órdenes de fabricación si fuera necesario. Los objetivos de este proceso son: lograr el cumplimiento de las prioridades prefijadas por el proceso 3, maximizar la utilización de los recursos y minimizar el inventario en curso (manteniendo unos niveles de seguridad por producto). Con todos estos datos se obtiene un Calendario de Lanzamientos que se utiliza en el proceso 5 - Fabricación - para producir los productos cerámicos. Finalmente los Productos Terminados se almacenan en el Almacén de Productos Terminados mediante el proceso 6 - Almacenaje.

Todos los procesos de negocio de la Figura 6.2 no están integrados, no están automatizados, son muy dependientes del personal humano encargado del mismo y no son flexibles. Todo esto implica que: un error de cálculo en uno de ellos acarrea errores en los demás procesos que no pueden ser detectados a

[1] Entendido como un programa anticipado de producción, que representa lo que la compañía planifica que va a producir expresado en configuraciones, cantidades y fechas específicas.

tiempo, los cambios en los requisitos del cliente son difíciles de adaptar a los procesos de negocio, la productividad de la empresa es muy dependiente de las previsiones de demanda, la velocidad de respuesta al pedido de un cliente es muy dependiente del stock de productos terminados y la coordinación entre las distintas empresas cerámicas es ineficiente.

6.1.3. Alcance del Sistema

El Sistema Holónico de Fabricación deberá:

- Automatizar la gestión de los almacenes de materia prima y ofrecer un sistema de ayuda a la gestión del departamento de compras. Esto implica, modificación del proceso 8 de la Figura 6.2.
- Ofrecer un sistema de ayuda a la Definición del Plan Maestro, proceso 3 (Figura 6.2).
- Automatizar el proceso de Definición y Seguimiento del Calendario de lanzamientos, proceso 4 (Figura 6.2).
- Automatizar el proceso de lanzamiento y control de Fabricación, proceso 5 (Figura 6.2).
- Integrar los procesos 4 y 5 de la Figura 6.2.
- Minimizar el trabajo sobre previsiones de demanda.
- Automatizar el proceso de gestión de almacenes de productos terminados, proceso 6 (Figura 6.2).
- Ofrecer un sistema de ayuda a la gestión de ventas en tiendas y en almacén, proceso 7 (Figura 6.2), mediante la coordinación con los procesos 4 y 5 de la Figura 6.2.
- Ofrecer un sistema de ayuda a la coordinación entre las distintas empresas cerámicas.

La interfaz del HMS con el resto de procesos de negocios no modificados de KCG deberá mantenerse inalterada.

6.1.4. Procesos a Controlar

En esta sección describimos los distintos procesos a controlar.

El proceso de **Gestión de Almacén de Materia Prima - (P1)** se encarga de mantener el registro de materia prima disponible y su localización dentro de los almacenes, fecha prevista de materia agotada, fecha de entrada de materia prima al almacén, y fecha, cantidad y destino de materia prima que sale del almacén. El sub-sistema de **Ayuda a la Gestión de Compras - (P2)** debe interactuar con el proceso de Gestión del Almacén de Materia Prima y el sub-sistema de Producción (procesos 4 y 5 de la Figura 6.2) para determinar y recomendar al departamento de compras, el pedido de materia prima a proveedores, según: la capacidad del almacén, fecha prevista de utilización de materia prima, tiempo estimado de servicio por parte de los proveedores y coste de transportes según fechas.

El proceso de **Gestión de Almacén de Productos Terminados - (P3)** se encarga de mantener el registro de los productos terminados que entran y salen del almacén, la localización de los mismos en los distintos almacenes, el stock existente de cada tipo de producto y el espacio disponible para alojar productos. Este proceso debe interactuar con el sub-sistema de Producción para permitir la entrada de productos terminados desde fábrica y compartir información sobre disponibilidad de espacio en almacén. Además, debe interactuar con el sub-sistema de Ayuda a la Gestión de Ventas para ofrecer información sobre disponibilidad de productos y recibir solicitudes de retirada de productos de almacén.

El sub-sistema de **Ayuda a la Gestión de Ventas - (P4)** se encarga de la gestión de las ventas en tiendas y de las ventas directas desde almacén de productos terminados. Interactúa con el proceso Gestión de Almacén de Productos Terminados para determinar el stock del producto solicitado por el cliente. En caso de que no haya stock en almacén, inicia un proceso de negociación con el sub-sistema de Producción para determinar el coste y la fecha en la que el pedido del cliente estará disponible. De esta manera el comercial en tienda o en almacén puede dar una respuesta al cliente sobre la disponibilidad de su pedido en un tiempo relativamente corto (online, no debe superar los 10 min).

El sub-sistema de **Ayuda a la Definición del Plan Maestro - (P5)** debe definir y recomendar al Jefe de Producción los programas anticipados de producción. Estos programas incluyen configuraciones de productos y cantidades a producir (lotes de fabricación) y fechas previstas de producción. Para cumplir con su objetivo el sub-sistema de Ayuda a la Definición del Plan Maestro se basa en las previsiones de demandas (definidas por el Departamento de Marketing), los pedidos de producción (gestionado por el Departamento de Ventas), la información obtenida del proceso de Gestión de Almacén de Productos Terminados y del proceso de Gestión de Almacén de Materia Prima, y de la interacción con el sub-sistema de Producción (procesos 4 y 5 de la Figura 6.2).

El sub-sistema de **Producción - (P6)** integra los procesos 4 y 5 de la Figura 6.2. Se encarga de la definición y seguimiento de los calendarios de lanzamientos y del control del proceso de fabricación propiamente dicho. La definición de los calendarios (**P6.1**) prioriza los pedidos de clientes, y sólo en caso de holgura en los programas hace uso del plan maestro definido por el sub-sistema de Ayuda a la Definición del Plan Maestro.

El proceso de **Fabricación - (P6.2)** (proceso 5, Figura 6.2), forma parte del sub-sistema de Producción, y es el más complejo de todos los procesos a controlar. En el proceso de fabricación de gres [2] se puede hablar de dos tipos de procesos denominados de monococción y bicocción. En cada tipo de proceso productivo se pueden distinguir varios tipos de formatos (tamaño). A su vez, cada formato posee diferentes modelos que serán función de la forma y de las aplicaciones que reciben (dibujo, esmaltes, etc,...). Actualmente en la empresa cerámica sobre la que se disponen los datos, se fabrica mediante procesos de monococción porosa y bicocción rápida. En ambos grupos de productos continuamente se van eliminando modelos que no tienen suficiente demanda e incorporando otros nuevos (procesos 1 y 2 de la Figura 6.2), por lo que el número de modelos fabricados sufre continuas modificaciones, tendiendo a aumentar el número de un año a otro.

La fabricación del revestimiento cerámico (Figura 6.3) se realiza a partir de

[2]La definición del proceso de fabricación ha sido obtenida del trabajo presentado en la Tesis Doctoral [Andrés, 2001] desarrollada en el CIGIP.

arcillas, que se someten a un tratamiento de molturación vía húmeda y, posteriormente, de atomización. La arcilla atomizada se prensa, formando unas piezas, que en el caso de bicocción se cuecen antes de ser esmaltadas (bizcocho), luego se esmalta y sufre una segunda cocción (fino), en monococción pasan directamente a esmaltarse (la Figura 6.3 presenta la línea de producción de productos cerámicos de monococción). Por último sea cual sea el proceso, el producto esmaltado pasa al horno. Entre las líneas de esmaltado y los hornos existen almacenes intermedios debido al distinto ritmo de producción que hay en cada sección. Una vez cocido el producto se transporta a una zona de almacenamiento a la espera de ser clasificado en diferentes calidades por medio de máquinas. Al mismo tiempo, un operario analiza los defectos de superficie. Se confeccionan las cajas de cartón en las que se envasa el producto y varios AGVs (vehículos auto-guiados) se encargan de recoger estas cajas y de almacenarlas en los palets (proceso 6 de la Figura 6.2) que se transportan al almacén de producto terminado. El producto queda ya listo para su expedición.

En el diagrama de procesos de la Figura 6.3 el Almacén de Materia prima representa a la superficie (era) dedicada al almacenamiento de arcillas. Por su parte el proceso 5.1 - Transporte de Arcilla - representa la secuencia Silos de Arcilla Mezclada, Molino, Atomización, Silos de Arcilla Atomizada, transporte entre cada uno de estos y finalmente transporte entre los silos de arcilla atomizada y las líneas de prensado. Con la mezcla previamente atomizada, se alimenta el carro de la prensa, donde se configura el producto final. Con el Prensado (proceso 5.2) se da forma a la pieza y se le dota de una resistencia mecánica que permita que la pieza conformada pueda ser transportada en las siguientes etapas del proceso. El siguiente proceso al que es sometida la pieza es el Secado (proceso 5.3), cuya misión es eliminar agua de las mismas para, posteriormente, conseguir que la adherencia de las aplicaciones de esmalte no provoque fallos de tipo superficial. A través de cintas transportadoras se conducen las piezas al Secadero. Durante este trasporte se someten a un proceso de cepillado, tanto de la cara superior como de la cara inferior, con el fin de eliminar el polvo que queda adherido a la pieza después del proceso de prensado. El tiempo de permanencia de la pieza en el secadero, así como la

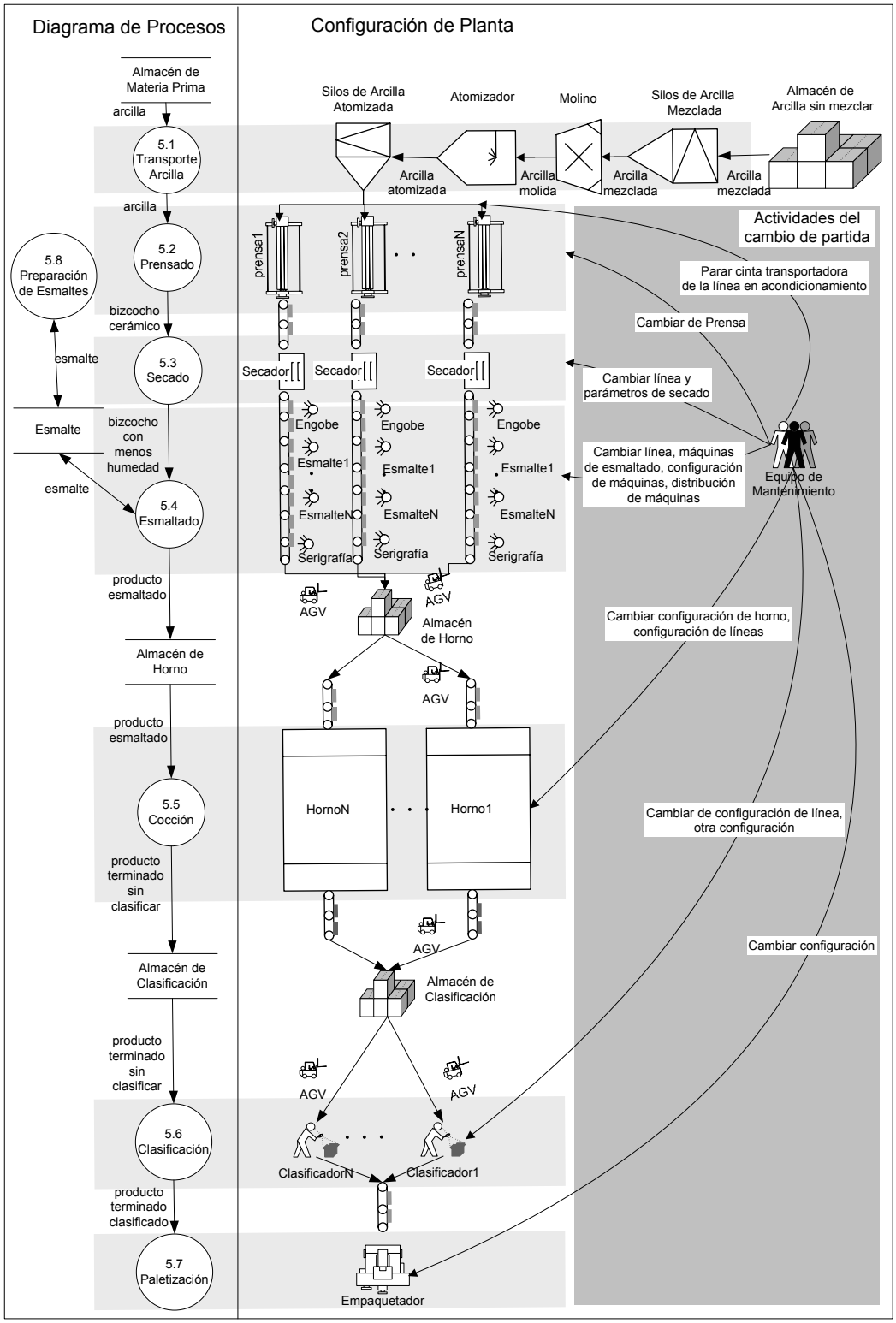


Figura 6.3: Proceso de Fabricación de KCG.

temperatura utilizada, depende fundamentalmente del espesor. La temperatura y humedad se controlan periódicamente. Las piezas salen del secadero a una temperatura óptima para el proceso de Esmaltado (proceso 5.4). En la primera parte de la línea, cada unidad recibe dos aplicaciones por medio de una serie de campanas denominadas engobe y base. La aplicación de la base se debe aplicar justo cuando el engobe empieza a secar (esto se consigue regulando la velocidad de la línea). Estas capas se aplican para eliminar el color rojizo de la pasta, eliminar o corregir los defectos superficiales del soporte y mejorar la adherencia. A continuación, hay un largo tramo de línea para decorar las piezas mediante distintas aplicaciones de esmaltes. Las máquinas de esmaltado son diseñadas específicamente para los distintos tipos de esmalte y acabado. Estas instalaciones se pueden montar y desmontar sobre la cinta transportadora, para variar la configuración de la línea en función del diseño del producto a fabricar. El producto esmaltado pasa a los hornos donde se cuece el bizcocho esmaltado (proceso 5.5) para dar el producto final. La alimentación de los hornos es mediante gas natural. En su paso por el horno el bizcocho atraviesa varias secciones denominadas prehorno, precalentamiento, cocción, enfriamiento natural y enfriamiento forzado. El control del horno para la regulación de la temperatura se realiza mediante un sistema de control de microprocesadores. Del horno sale el producto a una temperatura elevada, pero ya con las características finales de resistencia y dureza adecuadas. Las piezas una vez hayan salido del horno pasan al almacén donde se almacenan para su posterior clasificación (proceso 5.6). En esta sección se somete a los azulejos, uno a uno a varias pruebas para clasificarlos por calidades y calibres (resistencia mecánica, clasificación visual, planaridad, calibres y tono). En la clasificación de las piezas se distinguen tres calidades (primera, segunda y tercera). Los azulejos con defectos más graves como despuntados o desconchados se clasifican como tiestos, las piezas así clasificadas son desviadas y caen a un depósito. Posteriormente serán molidas para su reincorporación a las nuevas arcillas de las eras. Una vez clasificado el material, se procede a empaquetar el mismo en cajas de cartón que contienen un número de piezas dependiente del formato del azulejo. Entre líneas de esmaltado y hornos, y hornos y líneas de clasificación las unidades se depositan mediante sendos manipuladores en unas

estructuras compuestas por diferentes repisas denominadas vagonetas. Para el transporte de estas vagonetas se utilizan AGVs. El proceso de Preparación de Esmaltes (proceso 5.8) es auxiliar y se realiza previamente al inicio de la producción. Los esmaltes suelen ser de fabricación propia. Para fabricar los esmaltes se utilizan fritas, colores, aditivos y agua. Todas estas materias primas se introducen en unos molinos especiales, en unas cantidades adecuadas y se muelen hasta que la granulorometría y viscosidad sean las prescritas. Se almacenan entonces en cubas constantemente homogeneizadas mediante unos batidores hasta el momento de su uso.

El diagrama de procesos de la Figura 6.3 corresponde a la fabricación de un lote de un determinado producto. Debido a la enorme competencia del sector cerámico, las empresas deben hacer frente a demandas con productos muy personalizados, dentro de una gama muy variada de los mismos. Así se llega a tener extensos catálogos que complican la capacidad de programación de producción de la empresa para fabricar tal variedad de modelos. Las líneas de fabricación se deben compartir entre distintos modelos de productos, lo que trae asociado un conjunto de problemas relacionados con los cambios de partida. Los cambios de partida, como se puede ver en la configuración de planta de la Figura 6.3, implican una serie de actividades de configuración de la línea para producir un determinado tipo de producto (por parte del personal del grupo de mantenimiento). Además del tiempo que se pueda tardar en configurar la línea, existe otro retardo asociado con calibrar las distintas máquinas para conseguir cierta calidad del producto final. Esto se debe a que como consecuencia de características no controlables de los procesos, el producto final que se obtiene sufre pequeñas desviaciones con respecto a la dimensión nominal (calibre) a la vez que aparece una variabilidad en los colores. Esto produce dos problemas importantes a considerar:

- Pese a todos los esfuerzo realizados en mejorar el control del proceso es imposible asegurar una calidad del 100 % y, por lo tanto, se debe realizar al final del proceso una clasificación por calidades. Esto implica que se fabriquen unos lotes mínimos que permitan que el producto llegue al final del proceso para ser clasificado y que se realicen los ajustes necesarios en las variables controlables del proceso para corregir las anomalías que

puedan aparecer.

- Debido a la dificultad de asegurar un producto final uniforme de primera calidad, se tiende a aumentar el tamaño de lotes, para evitar pérdidas debidas a defectos de calidad. Estos problemas propician que se tienda a sobredimensionar el tamaño de los lotes a fabricar con lo que los inventarios de producto acabado aumentan.

Al inicio de cada nueva orden de fabricación aparecen unos tiempos de ajuste, necesarios para calibrar las máquinas de manera que se obtenga el producto deseado. A estos tiempos de ajuste se les deben de añadir los tiempos invertidos en modificar las máquinas y su posición en la línea, denominados tiempos de preparación, con lo que la suma de ambos origina lo que se denomina tiempo de cambio de partida que pueden llegar a ser muy elevados (el tiempo que se tarda en cambiar prensas y ajustar línea de esmaltado es del orden de 7 horas, el de horno y de clasificación es de 1 hora)[3].

En la actualidad los procesos de control y configuración de máquinas de la línea de fabricación se realizan mediante la intervención de operarios. Esto hace que: los coste de personal sean elevados, ya que en todo momento se debe mantener una cuadrilla de varios operarios dedicados a estas tareas; la productividad de la línea depende de la habilidad y pericia de los operarios de mantenimiento; muchas de las tareas de configuración son mecánicas y por tanto automatizables; la propagación de las tareas de ajustes de configuración a lo largo de la línea se llevan a cabo de manera poco eficiente ya que los ajustes a cada máquina se realizan manualmente; la retro-alimentación a los procesos de Definición y Seguimiento de los calendarios de lanzamiento no es en tiempo real y es poco flexible ya que se sigue una política de comunicación por jefes (esmalte, prensa, horno, clasificación). En consecuencia para flexibilizar y automatizar determinados procesos dentro de las líneas de producción, la empresa pretende modificar los sistemas de control y configuración de: líneas, máquinas de esmaltado, engobe y serigrafiado, control del horno, máquinas de secado, clasificación y paletización. Se seguirá manteniendo en cambio: el proceso manual de cambio y configuración de prensas debido a las

[3]En [Andrés, 2001] se puede encontrar un estudio detallado de los tiempos de cambio de partida.

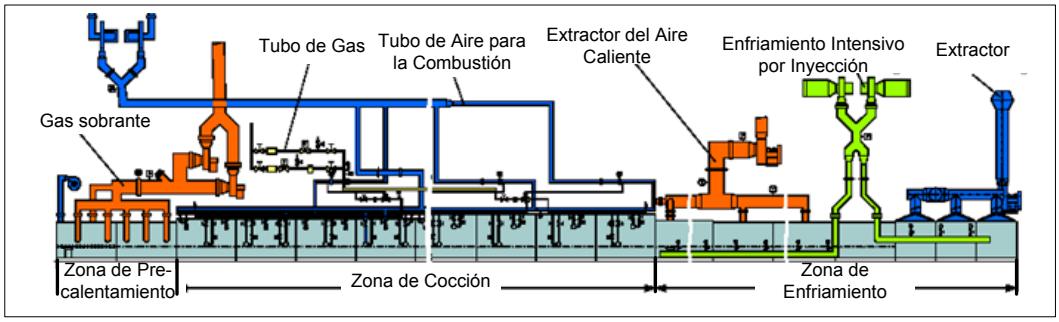


Figura 6.4: Horno Continuo de Rodillo para azulejos cerámicos.

características de este recurso, y; la clasificación manual mediante operarios expertos. Además, se debe permitir que operarios humanos puedan corregir algunas configuraciones de ajuste en la línea de producción.

Entre los componentes físicos a controlar se encuentran: las distintas cintas transportadoras, las máquinas de esmaltado serigrafiado y engobe, las máquinas de secado, los AGVs, los hornos, la máquina de clasificación, etc. Por problemas de extensión de la presente tesis presentamos sólo la descripción del horno.

En la Figura 6.4, se ve el esquema de un horno continuo de rodillo para azulejos cerámicos. El horno está compuesto por un conjunto de rodillos que transportan los azulejos en el interior del horno y el horno en sí, que a su vez se divide en 3 zonas (pre-calentamiento, cocción, y enfriamiento). El horno está diseñado de tal manera que: el aire calentado en la zona de enfriamiento se aprovecha en la zona de cocción y la combustión de gases de la zona de cocción se re-conduce a la zona de pre-calentamiento. La regulación de temperatura se realiza de forma independiente en cada una de las 3 zonas, aunque el horno en sí está formado por un único volumen. La temperatura del interior del horno alcanza los 1500ºC. La longitud total del túnel es de 80 m y el medio de combustión es el gas. Cada zona tiene un termopar para medir la temperatura, un termopar de seguridad y varios quemadores donde se realiza el proceso de combustión. También tiene chimeneas y extractores para la evacuación de los

humos generados en la combustión. El otro sub-sistema del horno es mecánico, el conjunto de rodillos, que llevan las piezas a lo largo del horno. Regulando la velocidad de los rodillos se consigue ajustar el tiempo de permanencia de la carga del horno, en función del tratamiento término requerido. El proceso de control del horno tiene que tener en cuenta todos los diferentes estados de operación que puede tener el horno: entrando carga, lleno de carga, saliendo carga, cambio de temperatura en alguna de las zonas por tipo de producto, error en algún componente, velocidad de los rodillos, sincronización con los sistemas de carga y descarga del horno.

6.1.5. Condiciones de Operación

En esta sección describimos por cada proceso a controlar las entradas y salidas requeridas, así como el espectro de posibles cambios y fallos que pueden ocurrir durante la operación. La información se presenta en las Tablas 6.1 y 6.2.

6.1.6. Objetivos

Los objetivos del HMS son:

- Coordinar la actividad de las empresas cerámicas de KCG.
- Integrar los procesos de negocio 3, 4, 5, 6, 7 y 8 de la Figura 6.2.
- Minimizar la programación basada en planes maestros y priorizar la producción bajo demanda.
- Flexibilizar el sistema de producción de KCG con respecto a los cambios de recursos y órdenes de trabajo.
- Automatizar el control de fabricación.
- Capacidad de respuesta ante fallos mecánicos o de control en planta.
- Asegurar la calidad de producción.

Tabla 6.1: Condiciones de Operación - Parte 1

Proceso	**Entrada**	**Salida**	**Cambio/Fallo**
P1	Información de entrada de materia prima (tipo, cantidad, fecha, proveedor). Información de salida de materia prima (tipo, cantidad, fecha, destino). Solicitud de información de disponibilidad de materia prima (tipo, cantidad, fecha).	Información de disponibilidad de materia prima.	Retraso en la entrada y/o salida de materia prima al almacén. Nuevo tipo de materia prima, con características de conservación nuevas.
P2	Capacidad actual del almacén. Disponibilidad de materia prima (tipo materia prima, fecha prevista de utilización). Flujo de materia prima que sale del almacén (tipo materia prima, cantidad, periodo de tiempo)	Recomendación de pedido de compra a proveedor (tipo materia prima, cantidad, fecha, proveedor)	Nuevo proveedor y/o materia prima.
P3	Información de entrada de producto terminado (tipo, cantidad, fecha, fábrica de la que procede). Información de salida de producto terminado (tipo, cantidad, fecha, destino). Solicitud de información de disponibilidad de producto terminado (tipo, cantidad, fecha). Solicitud de información sobre espacio disponible en almacén (fecha, tipo de producto).	Información de disponibilidad de producto terminado. Información de espacio disponible en almacén.	Retraso en la entrada y/o salida de producto terminado al almacén. Nuevo tipo de producto terminado, con características de conservación y dimensiones nuevas.
P4	Pedido de Cliente (tipo producto, cantidad, fecha).	Disponibilidad del pedido (fecha, precio).	
P5	Información sobre previsión de demandas, pedidos de producción, disponibilidad de materia prima, stock de productos terminados. Solicitud de definición de Plan Maestro. Solicitud de modificación de Plan Maestro.	Plan Maestro tentativo (configuración de producto, cantidad a producir, fecha prevista de producción).	Nueva configuración de producto.

Tabla 6.2: Condiciones de Operación - Parte 2

Proceso	**Entrada**	**Salida**	**Cambio/Fallo**
P6	Solicitud de programación de pedido de cliente. Solicitud de Producción de lote.	Simulación de programación de pedido de cliente. Lote producido.	Pedido no se puede programar. Lote no se puede producir. Nueva configuración de planta. Nueva definición de producto.
P6.1	Solicitud de definición de programa. Solicitud de modificación de programa.	Programa definido. Programa Modificado.	Programa no se puede definir. Programa no se puede modificar.
P6.2	Solicitud de fabricación de lote.	Lote fabricado.	Error en planta (máquina, recurso, producto). Nueva máquina, recurso, o producto.
Horno	Solicitud de cambio de temperatura de horno según zona. Solicitud de cambio de velocidad de rodillos.	Temperatura modificada. Velocidad de rodillo modificada.	Error en componente de horno. Problema de sincronización con el subsistema de carga y descarga del horno.

- Permitir la intervención humana en los procesos de ayuda y control.
- Optimizar la velocidad de respuesta de KCG a sus clientes.

6.2. Análisis

En esta sección presentamos los distintos modelos de análisis resultado de la aplicación de las actividades de análisis al caso de estudio.

El objetivo de la fase de análisis es identificar los holones que componen el sistema y proveer una especificación inicial de holones. Esta fase es descendente, iterativa e incremental.

6.2.1. Iteración 1

En la primera iteración descomponemos el sistema de fabricación en un HMS, en el que sus entidades constituyentes cooperan en dominios de cooperación para cumplir con los objetivos del sistema. Para la identificación de estos dominios/escenarios de cooperación aplicamos las actividades A1, A2, A3 y A4 y las Guías HMS-CU (Tabla 5.3) de la actividad compleja Determinar Casos de Uso (Figura 5.31). Los resultados se muestran en la Tabla 6.3 y en la Figura 6.5.

La Tabla 6.3 resume los objetivos identificados mediante las Guías HMS-CU 1 a 6 del Capítulo 5. A partir de estos objetivos y las Guías HMS-CU 7 a 13, especificamos los dominios de cooperación (casos de uso) que los satisfacen, Figura 6.5. Hemos identificado 10 casos de uso [4] para satisfacer los objetivos del sistema, en las siguientes actividades de la fase de análisis se asocian los objetivos del sistema con los responsables de los casos de uso (Figura 6.6). La descomposición o especialización de los casos de uso se lleva a cabo en las siguientes iteraciones.

El siguiente paso en la fase de análisis es Especificar la Realización de Casos de Uso (Figura 5.32). La asignación de Roles responsables de casos de uso

[4]Los casos de uso tienen un relleno amarillo para poder diferenciarlos de la notación de las tareas.

Tabla 6.3: Iteración 1, Objetivos del Sistema

ID	Objetivo
1	Coordinar las actividades de producción de las empresas cerámicas.
2	Automatizar el proceso de gestión de almacenes de materia prima.
3	Automatizar el proceso de gestión de almacenes de productos terminados.
4	Ofrecer una recomendación "online" al Comercial de tienda/almacén sobre la fecha de entrega de un pedido de cliente.
5	Recomendar al Departamento de Compras pedidos de compra de materia prima.
6	Optimizar la velocidad de respuesta de KCG a sus clientes.
7	Recomendar al Jefe de Producción un Plan Maestro.
8	Minimizar la programación basada en planes maestros y priorizar la producción bajo demanda.
9	Automatizar la programación y re-programación de procesos de producción.
10	Automatizar el control de fabricación.
11	Integrar la programación y re-programación con el control de fabricación.
12	Asegurar la calidad de producción.
13	Permitir la intervención humana en los procesos de ayuda y control.

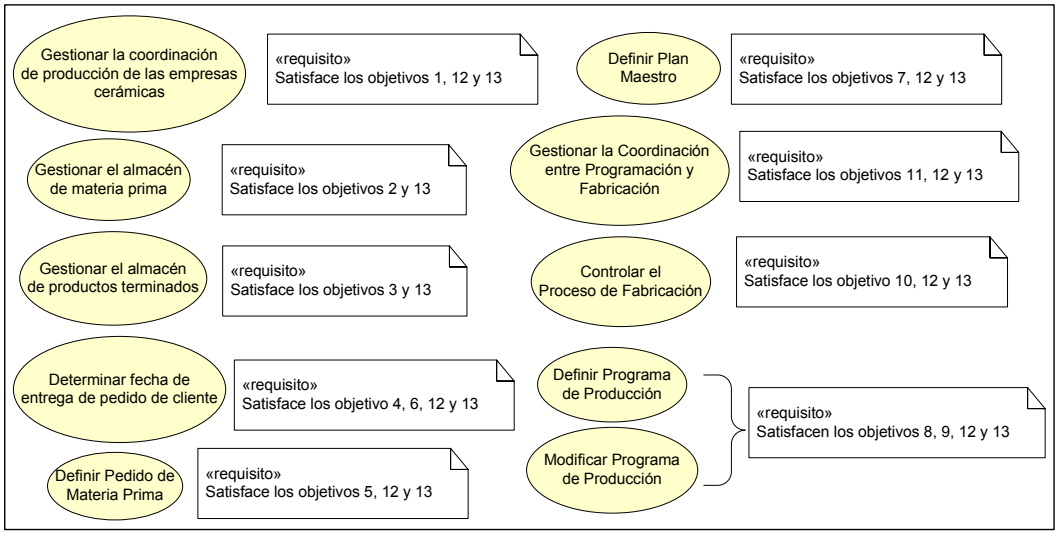


Figura 6.5: Iteración 1, Modelo de Casos de Uso.

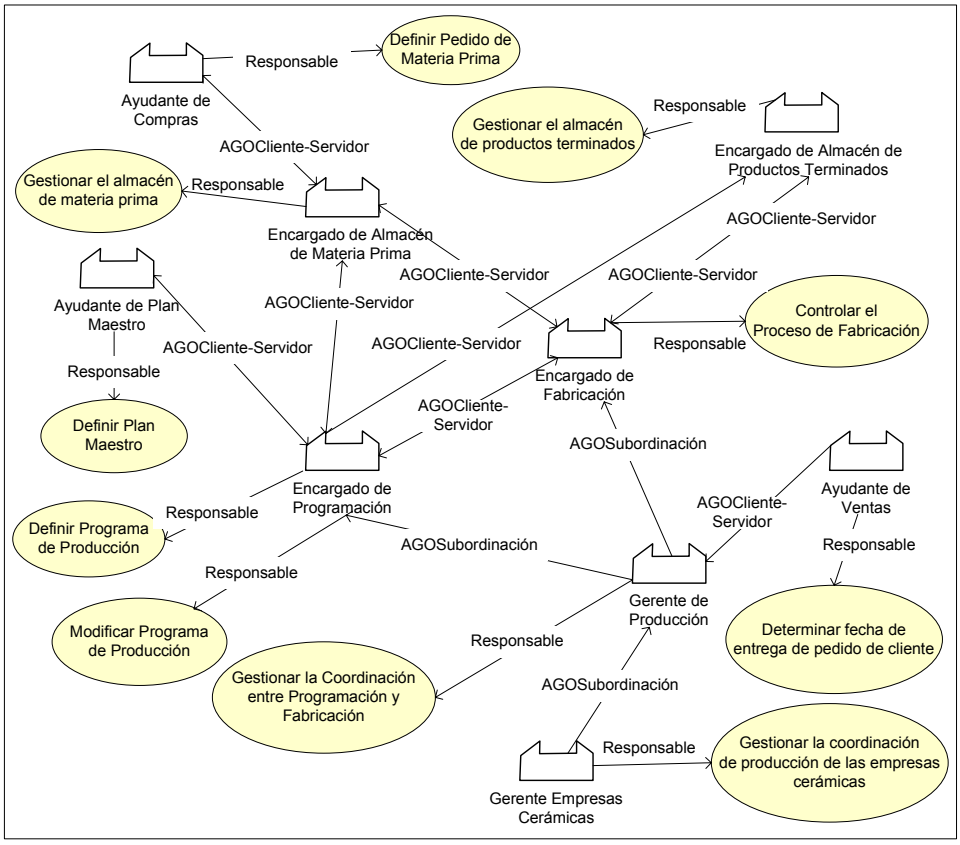


Figura 6.6: Iteración 1, Modelo de Organización.

(Actividad A5) se puede ver en el Modelo de Organización de la Figura 6.6. En esta figura se modelan también las relaciones sociales entre los Roles de la organización (Actividad A6). Estas relaciones (se derivan del documento de Requisitos) son del tipo AGOCliente-Servidor cuando entre dos Roles se necesita comunicación del tipo solicitud de información o de servicios (ejemplo, Ayudante de Ventas y Gerente de Producción), y del tipo AGOSubordinación cuando entre dos Roles existe una relación de jefe-subordinado (ejemplo, Gerente de Producción y Encargado de Programación).

Además del Modelo de Organización, la Actividad A6 produce un Modelo de Interacción (Figura 6.7) donde se especifican las necesidades de comunicación

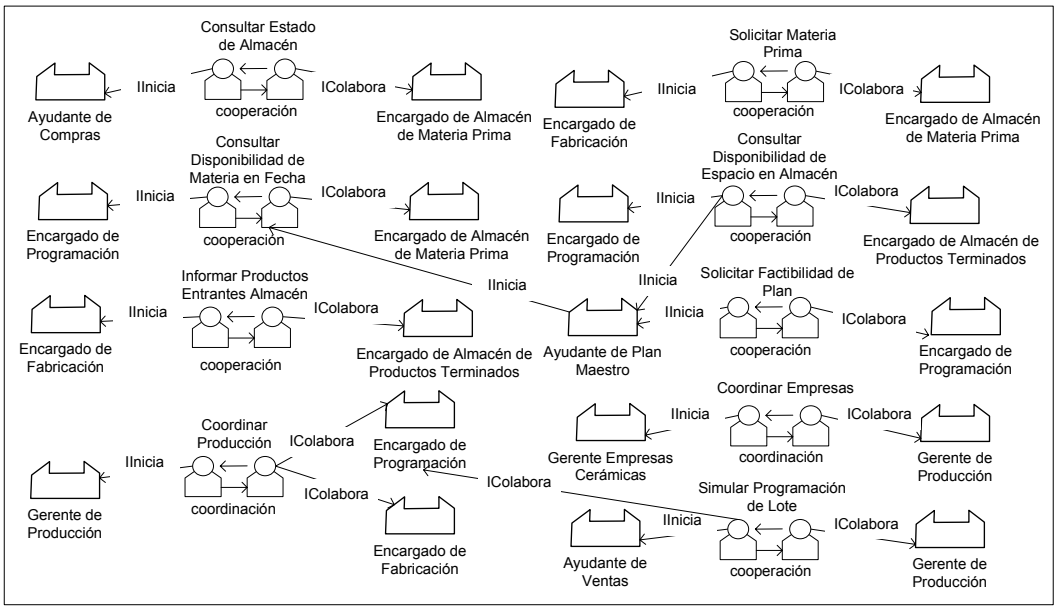


Figura 6.7: Iteración 1, Modelo de Interacción.

entre los Roles del HMS.

Las interacciones que hemos identificado para el HMS de KCG son:

- Consultar Estado de Almacén. El Ayudante de Compras necesita información sobre el stock de materia prima disponible, para ello inicia un proceso de comunicación con el Encargado de Almacén de Materia Prima. Existe un encargado por cada almacén así que esta interacción puede ser iniciada con cualquiera de ellos.

- Solicitar Materia Prima. El Encargado de Fabricación debe solicitar materia prima para el proceso de fabricación al Encargado de Almacén de Materia Prima.

- Consultar Disponibilidad de Materia en Fecha. En los casos de uso Definir Programa de Producción y Modificar Programa de Producción, el Encargado de Programación necesita información sobre la materia prima necesaria para producir un lote en una fecha determinada. Para ello, consulta esta

disponibilidad al Encargado de Almacén de Materia Prima. El Ayudante de Plan Maestro puede iniciar también esta interacción con el objetivo de obtener información para la definición de un Plan Maestro.

- Informar Productos Entrantes Almacén. En esta interacción el Encargado de Fabricación informa al Encargado de Almacén de Productos Terminados que entran al almacén una cantidad determinada de productos. En actividades e iteraciones sucesivas de análisis, se especifican más detalles de la interacción.

- Coordinar Producción. El Gerente de Producción inicia una interacción con el Encargado de Programación y el Encargado de Fabricación para coordinar las tareas de ambos en el proceso de producción.

- Solicitar Materia Prima. En el proceso de fabricación es el Encargado de Fabricación el que solicita materia prima a los almacenes. Para ello interactúa con el/los Encargados de Almacén de Materia Prima.

- Consultar Disponibilidad de Espacio en Almacén. Con el objetivo de definir o modificar un calendario de fabricación el Encargado de Programación debe conocer la disponibilidad de espacio en almacén para albergar productos terminados desde el proceso de producción. El Encargado de Almacén de Productos Terminados es el que ofrece esta información. El Ayudante de Plan Maestro puede iniciar también esta interacción con el objetivo de obtener información para la definición de un Plan Maestro.

- Solicitar Factibilidad de Plan. El Ayudante de Plan Maestro antes de comunicar el Plan Maestro al Jefe de Producción, consulta con el Encargado de Programación sobre la factibilidad del plan que ha definido. El Encargado de Programación es el que tiene conocimiento actualizado sobre la capacidad de la/las plantas, y por tanto es capaz de responder a esta consulta.

- Coordinar Empresas. Esta interacción se lleva a cabo para satisfacer el objetivo *Coordinar las actividades de producción de las empresas cerámicas*. El encargado de coordinar la interacción es el Gerente de Empresas

Cerámicas que inicia un proceso de coordinación entre los distintos Gerentes de Producción de las empresas cerámicas de KCG.

- Simular Programación de Lote. El Ayudante de Ventas coopera con el Gerente de Producción para obtener una respuesta sobre la fecha posible de entrega de un pedido a un cliente.

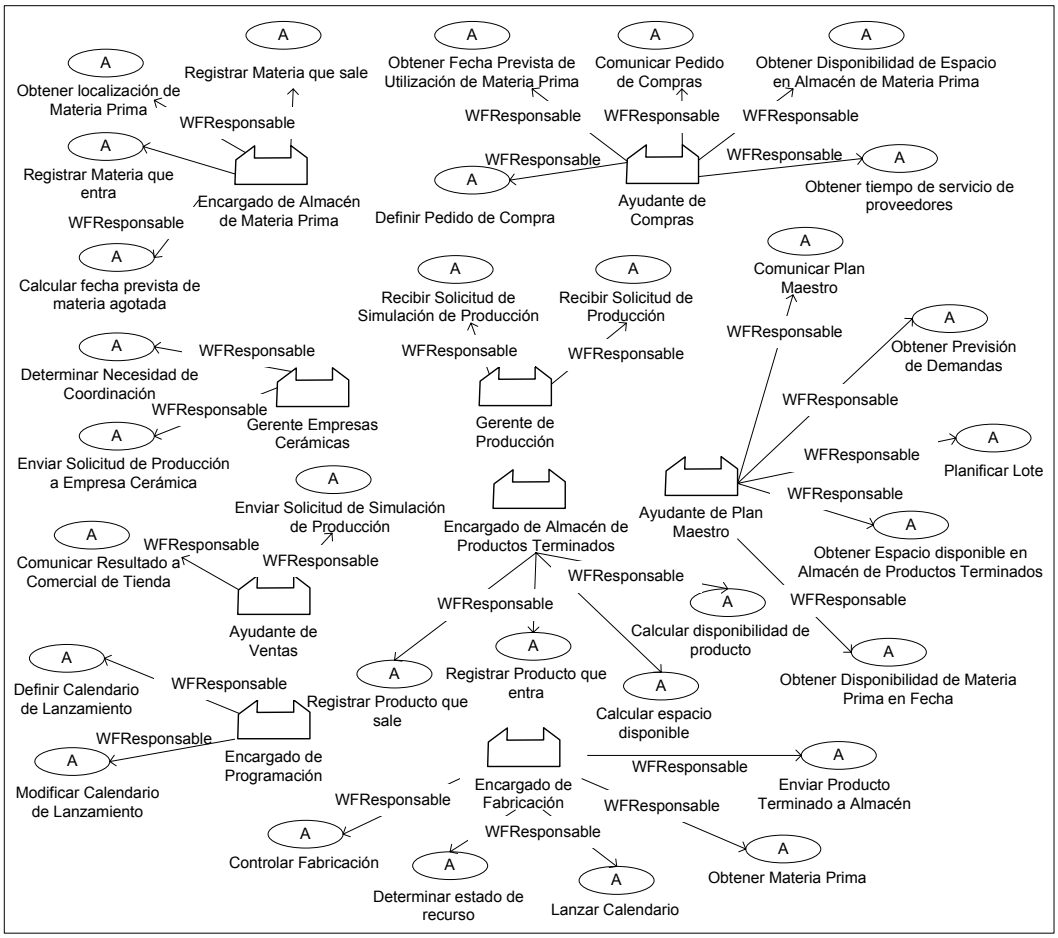


Figura 6.8: Iteración 1, Tareas Abstractas en el Modelo de Organización.

La Actividad A7 produce el Modelo de Organización de la Figura 6.8. En él, se modelan las Tareas Abstractas que los Roles necesitan implementar para

cumplir con los casos de uso de los que son responsables. En la Figura 6.9 podemos ver el Modelo de Tareas y Objetivos producto de la Actividad A8. En este diagrama se refleja la asociación de Tareas Abstractas con los objetivos del sistema que éstas afectan, y también dependencia entre objetivos.

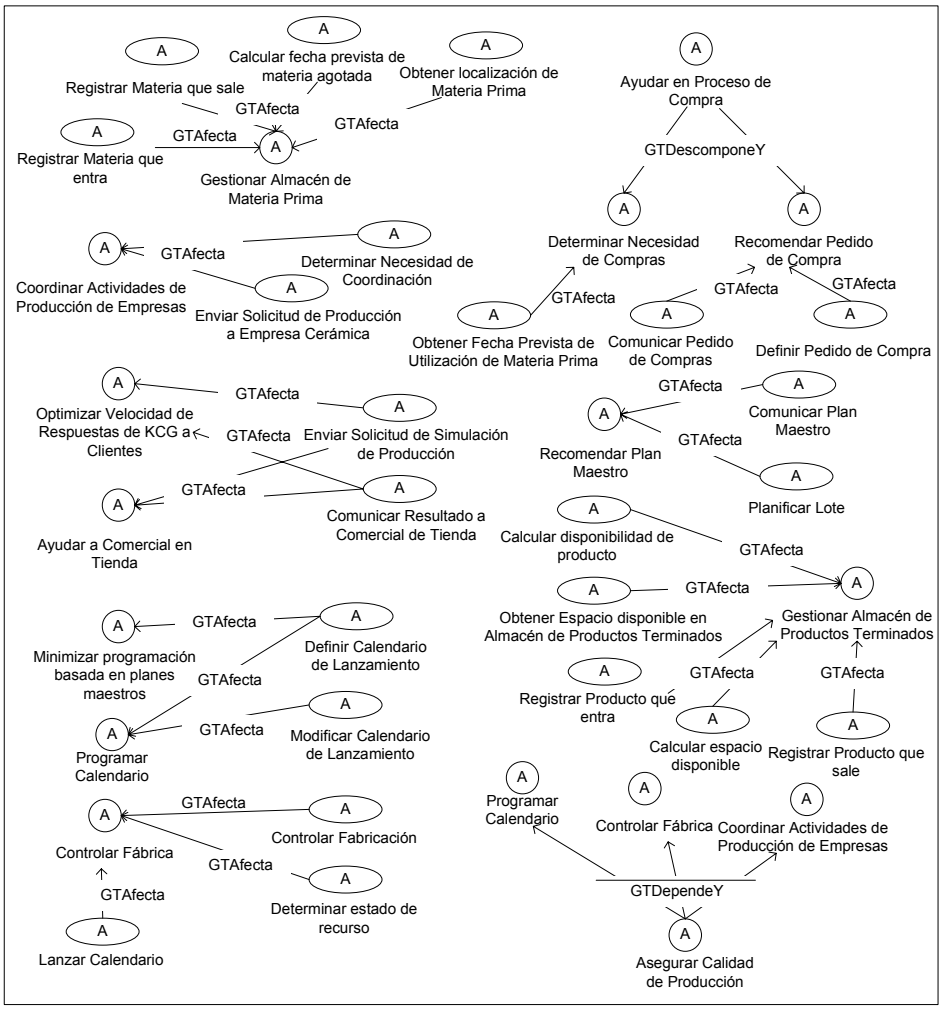


Figura 6.9: Iteración 1, Modelo de Tareas y Objetivos.

El siguiente paso en la fase de análisis es la tarea compleja Identificar Holones (Figura 5.33). En este paso se identifican los holones del sistema, se asocian

Roles a holones y se produce una primera especificación de holones. La Figura 6.10 muestra el Modelo de Objetivos ampliado con los holones (Agentes Abstractos) identificados mediante la Actividad A9 y las Guías PROSA (Tablas 5.4, 5.5 y 5.6).

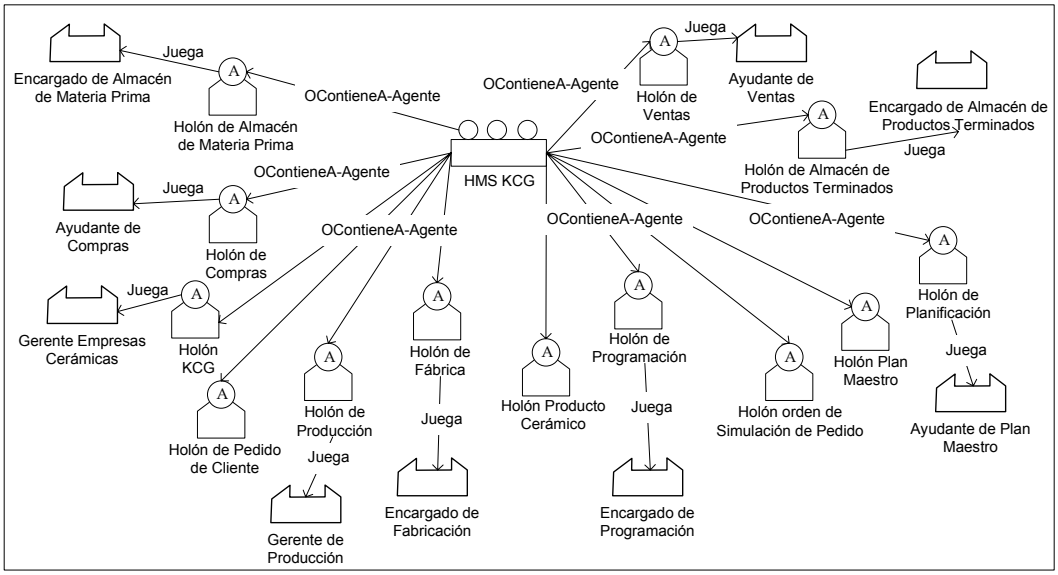


Figura 6.10: Iteración 1, Agentes Abstractos y Modelo de Organización.

Los holones que hemos identificado en el HMS de KCG son:

- Holón de Compras. Es un Agente Abstracto de recurso que ofrece capacidad de procesamiento para definir un pedido de compra de materia prima. En la holarquía de la Figura 6.6 juega el rol de Ayudante de Compras.

- Holón de Almacén de Materia Prima. Es un Agente Abstracto de recurso que ofrece capacidad de procesamiento para gestionar el almacén de materia prima. En la holarquía de la Figura 6.6 juega el rol de Encargado de Almacén de Materia Prima.

- Holón de Almacén de Productos Terminados. Es un Agente Abstracto de recurso que ofrece capacidad de procesamiento para gestionar el almacén

de productos terminados. Juega el papel de Encargado de Almacén de Productos Terminados.

- Holón de Fábrica. Es un Agente Abstracto de recurso que ofrece capacidad de producción. Juega el rol Encargado de Fábrica.
- Holón de Programación. Es un Agente Abstracto de recurso que ofrece capacidad de programación de lotes de fabricación. Juega el rol Encargado de Programación.
- Holón de Planificación. Es un Agente Abstracto de recurso que ofrece capacidad de cómputo para definir un plan maestro. Desempeña el rol Ayudante de Plan Maestro.
- Holón de Ventas. Es un Agente Abstracto de recurso que ofrece capacidad de procesamiento para gestionar las ventas en tienda y las ventas directas en almacén. También se encarga de producir una respuesta sobre la fecha de servicio de un pedido de un cliente. Desempeña el rol Ayudante de Ventas.
- Holón de Producción. Es un Agente Abstracto de staff que se encarga de gestionar la interacción entre el Holón de Programación y el Holón de Fábrica. Desempeña el rol Gerente de Producción. Existe un Holón de Producción por cada empresa cerámica.
- Holón KCG. Es un Agente Abstracto de staff encargado de la gestión de toda la holarquía KCG. Se encarga de gestionar la interacción entre todos los holones de la holarquía y desempeña el rol Gerente de Empresas Cerámicas para la coordinación de las actividades de las empresas KT, WD y MT.
- Holón de Pedido de Cliente. Es un Agente Abstracto de orden de trabajo que se genera en el proceso de ventas y es procesado mediante el Encargado de Producción utilizando recursos de fabricación de KCG.
- Holón orden de Simulación de Pedido. Es un Agente Abstracto de orden de trabajo que se genera en el proceso de ventas cuando no hay stock

del producto/s solicitado/s en almacén. Esta orden de trabajo se procesa mediante el Encargado de Programación.

- Holón Producto Cerámico. Es un Agente Abstracto de producto, mantiene la información de diseño y especificación de un producto cerámico.
- Holón Plan Maestro. Es un Agente Abstracto de producto, mantiene información de definición de plan maestro según la empresa cerámica a la que pertenece.

Una vez identificados los holones de la holarquía actual debemos completar una especificación inicial de estos holones, Actividad A10. La Figura 6.11 muestra el Modelo de Agente del Holón de Fabricación. Hemos asignado al Holón de Fabricación los objetivos y tareas asociados con el rol Encargado de Fabricación. Además, siguiendo las Guías PROSA, hemos identificado nuevos objetivos y tareas, y definido su estructura de información en términos de Creencias Abstractas.

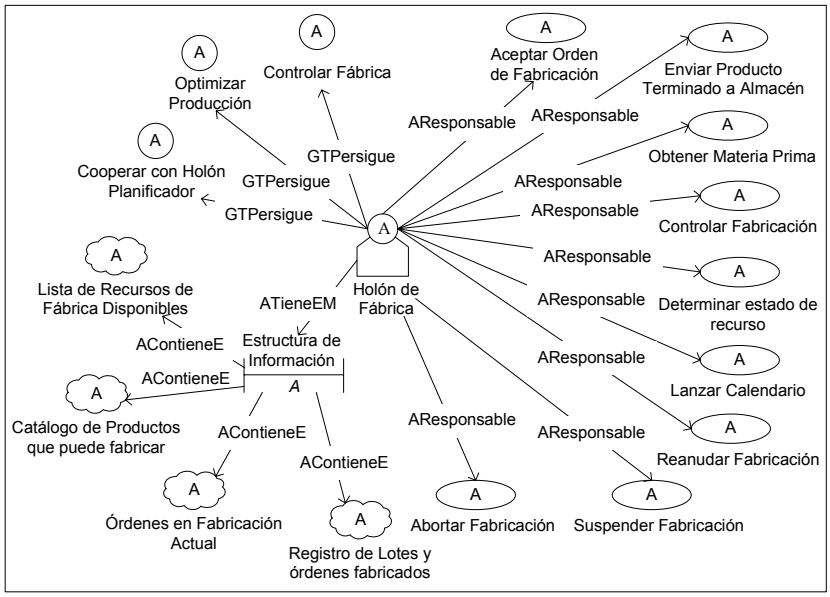


Figura 6.11: Iteración 1, Modelo de Agente. Holón de Fabricación.

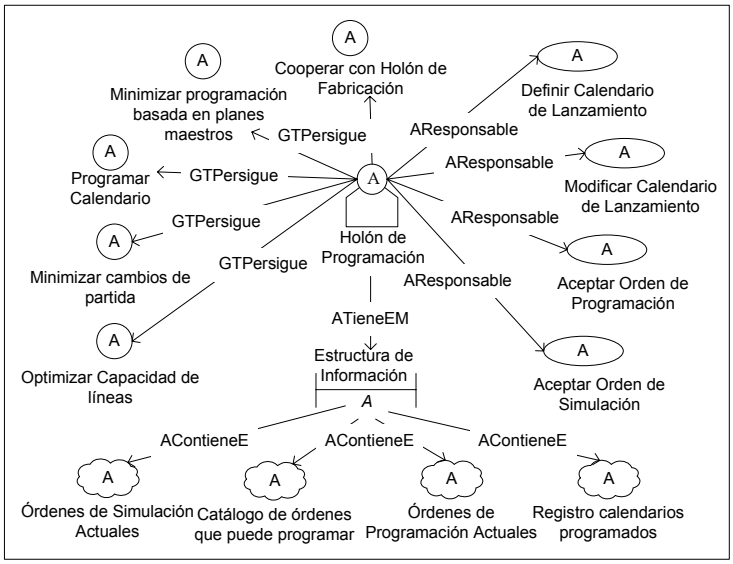


Figura 6.12: Iteración 1, Modelo de Agente. Holón de Programación.

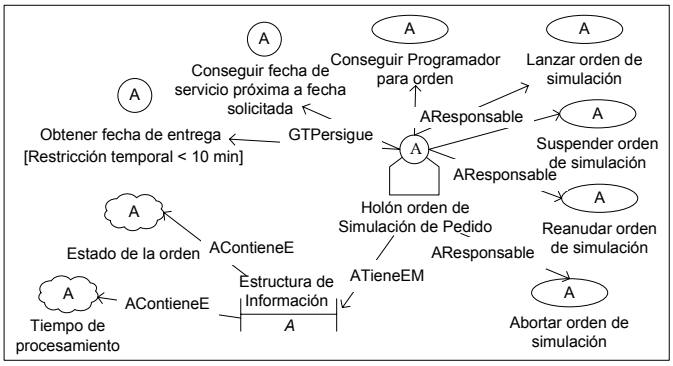


Figura 6.13: Iteración 1, Modelo de Agente. Holón orden de Simulación de Pedido.

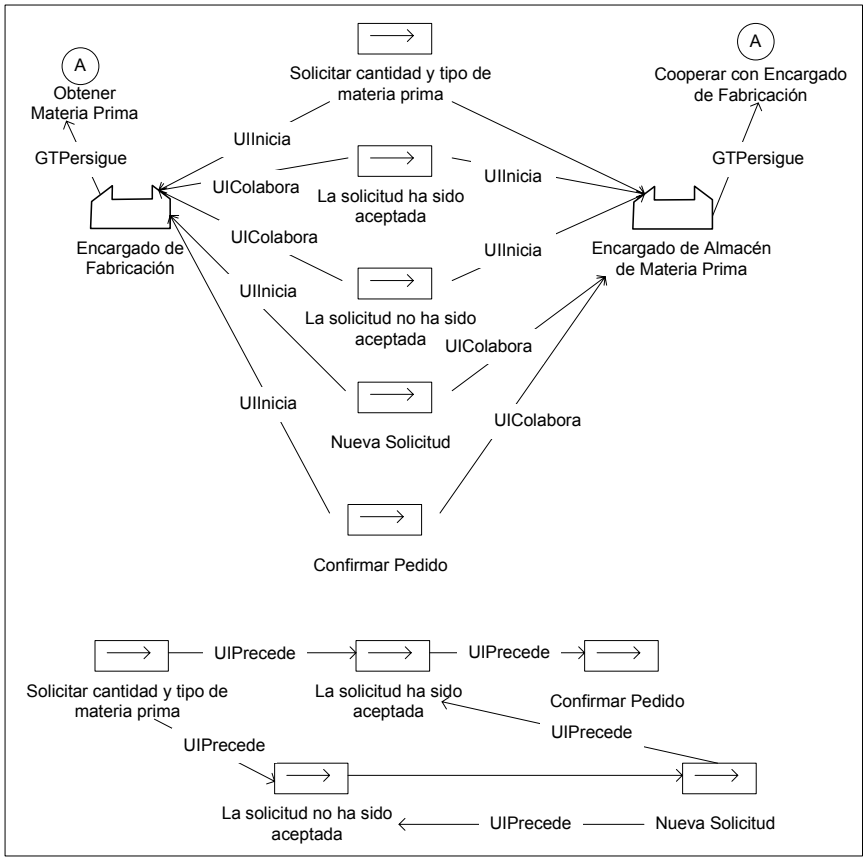


Figura 6.14: Iteración 1, Modelo de Interacción. Solicitar Materia Prima.

La Figura 6.12 muestra el Modelo de Agente del Holón de Programación. Mientras que la Figura 6.13 muestra el Modelo de Agente del Holón Orden de Simulación de Pedido.

La Actividad A11 tiene por objetivo especificar las interacciones identificadas en la Actividad A6. La Figura 6.14 muestra la especificación de la interacción Solicitar Materia Prima entre el Encargado de Fabricación y el Encargado de Almacén de Materia Prima. En esta figura podemos ver los objetivos que persiguen los distintos roles que participan en la interacción, los mensajes intercambiados, y la secuencia de mensajes.

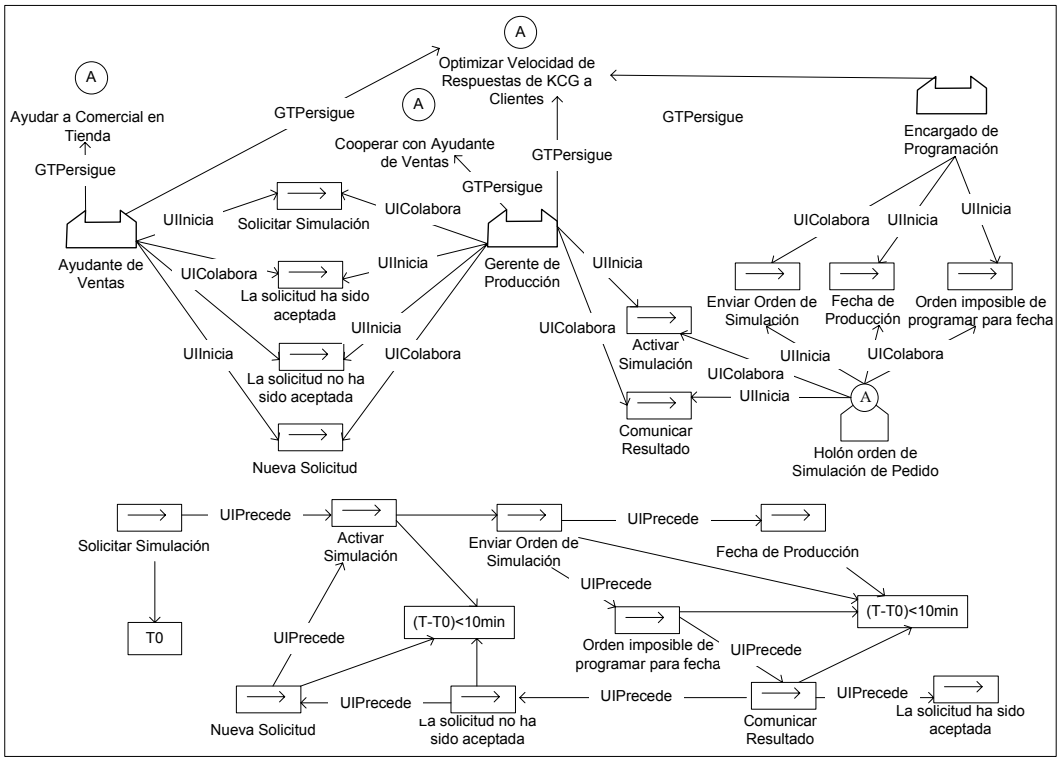


Figura 6.15: Iteración 1, Modelo de Interacción. Simular Programación de Lote.

En la Figura 6.15 se puede ver la interacción Simular Programación de Lote que tiene una restricción temporal sobre la duración máxima de la interacción. El tiempo máximo en que el Ayudante de Ventas puede obtener una repuesta del Gerente de Producción sobre la fecha de producción del pedido es de 10 minutos. Esta restricción se traduce en condiciones temporales asociadas a los mensajes en la interacción. Además, se puede ver al Holón orden de Simulación de Pedido que es el encargado de gestionar con el Encargado de Programación para obtener una fecha factible de entrega del pedido al cliente. La estructura de esta interacción se deriva de la Guía PROSA 26, Tabla 5.5.

Las nuevas interacciones identificadas en la Actividad A12 tienen que ver con las Guías PROSA 26 a 28. Estas guías ayudan a identificar interacciones relacionadas con las particularidades de la arquitectura PROSA que no han

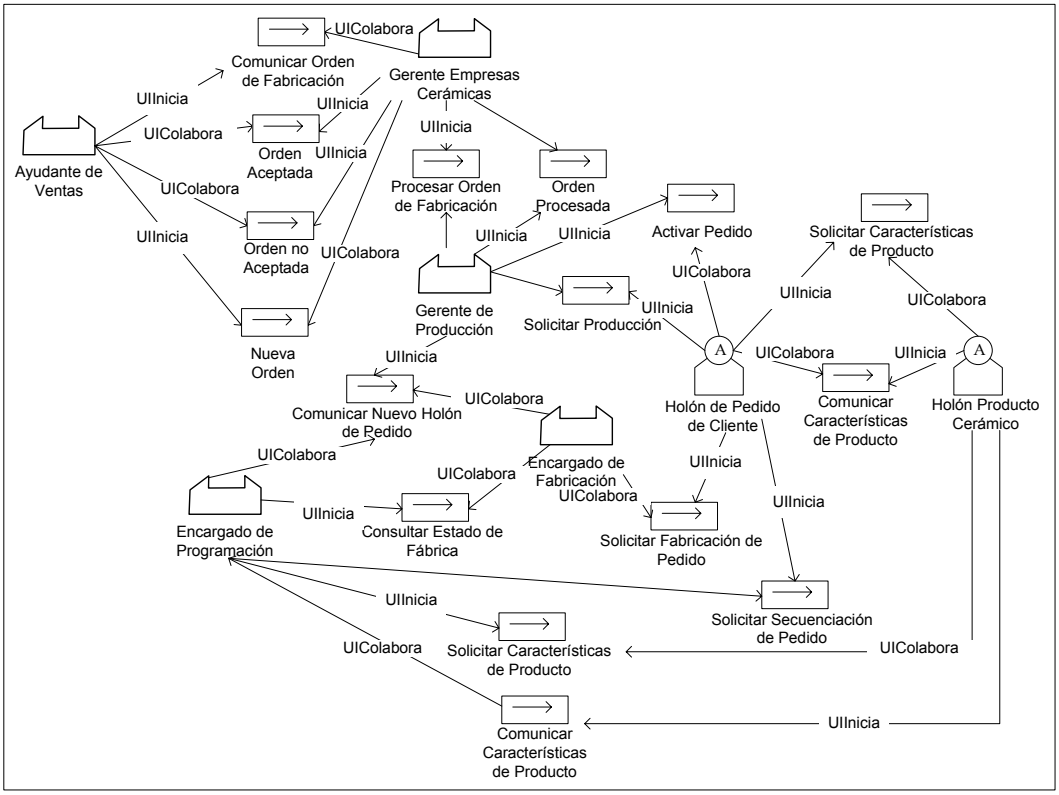


Figura 6.16: Iteración 1, Modelo de Interacción. Procesar Orden de Fabricación.

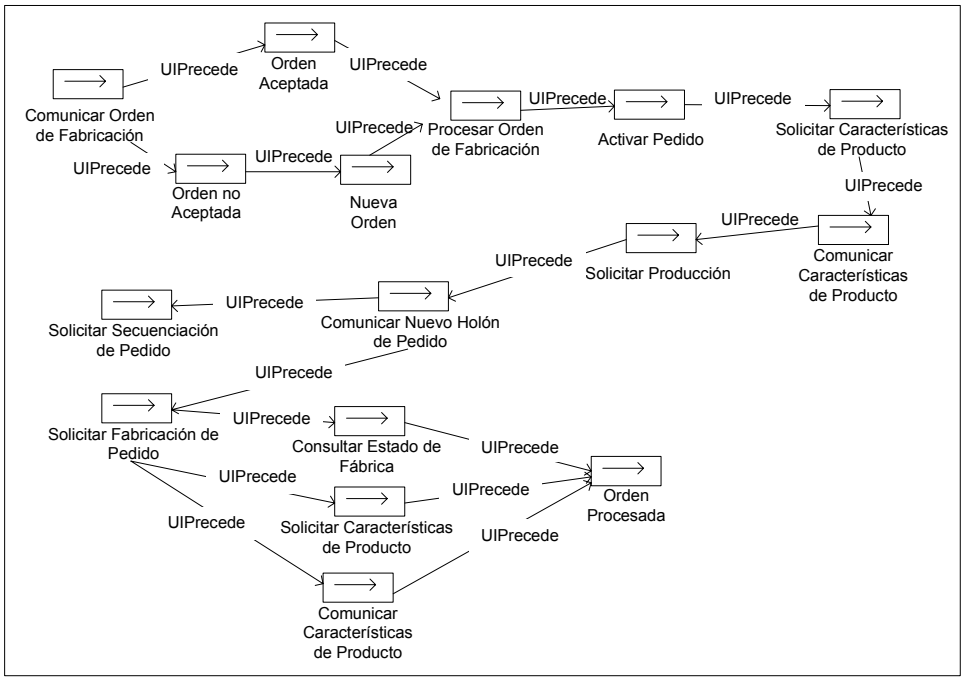


Figura 6.17: Iteración 1, Modelo de Interacción. Secuencia de mensajes de la interacción Procesar Orden de Fabricación.

sido aún identificadas. En la Figura 6.16 se puede ver la interacción para procesar una orden de fabricación (una orden de fabricación es un pedido en firme). La orden de fabricación del cliente se recibe mediante el Ayudante de Ventas que se la comunica al Gerente de Empresas Cerámica, para que este último determine en qué empresa/s cerámica/s se va a procesar el pedido. El Gerente de Empresas Cerámicas se comunica con el Encargado de Producción para indicarle la nueva orden de fabricación que entra al sistema. El Encargado de Producción activa a un Holón de Pedido de Cliente para que sea el encargado de la gestión del proceso de producción de la orden particular. Una vez que el Encargado de Producción recibe el mensaje Solicitar Producción del Holón de Pedido de Cliente, comunica al Encargado de Programación y al Encargada de Fabricación que el Holón de Pedido de Cliente iniciará con ellos un proceso de colaboración para procesar un pedido. La Figura 6.17 muestra el orden de ejecución de los distintos mensajes de esta interacción.

6.3. Diseño

En esta sección presentamos los distintos modelos de diseño resultado de la aplicación de las actividades de diseño de holones al caso de estudio.

El objetivo de la fase de diseño de holones es definir la arquitectura del sistema. La fase de diseño tiene dos etapas principales. La primera en la que se completan los modelos de la fase de análisis y la segunda en la que se especifican características dependientes de la plataforma de implementación. La primera etapa sigue un proceso ascendente.

El conjunto de holones de nivel de recursión 0 identificados en los Modelos de Análisis son los que se muestran en las Tablas 6.4 y 6.5. La primera tarea de la fase de diseño consiste en completar la especificación del sistema, refinando los modelos identificados en la fase de análisis.

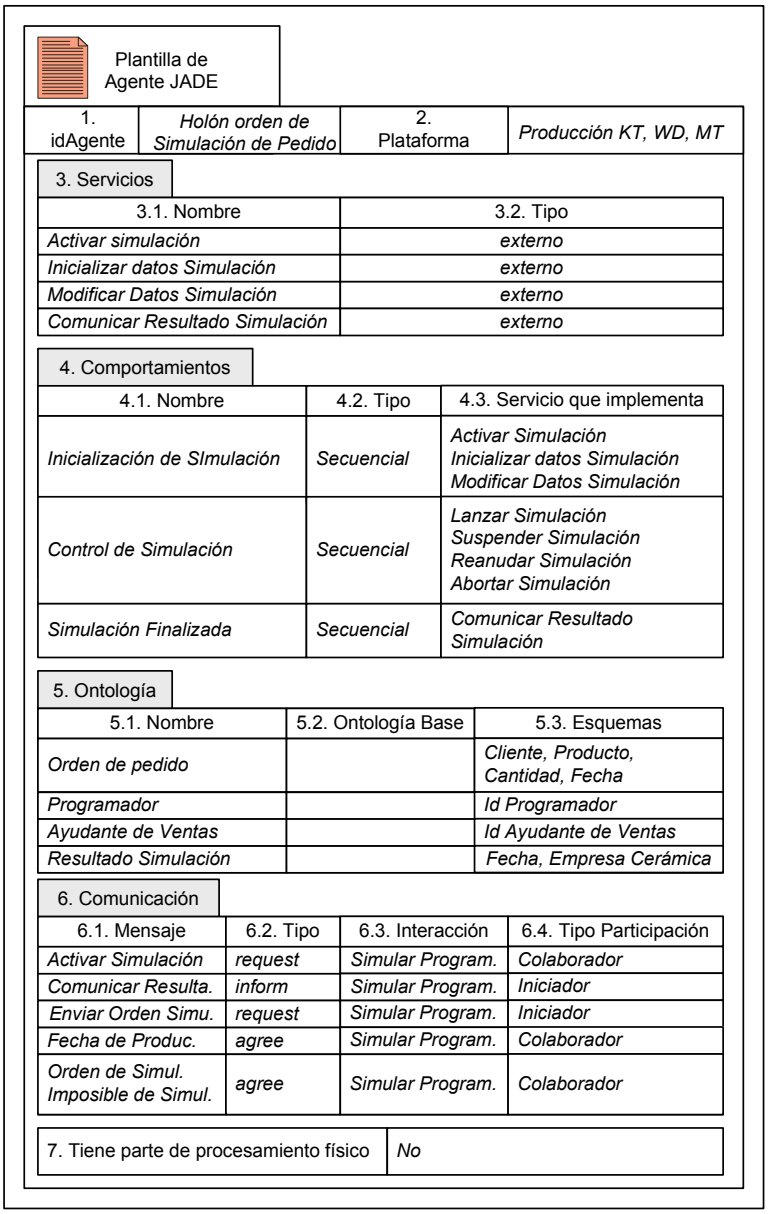
La Figura 6.18 muestra la plantilla de agente JADE del Holón Orden de Simulación de Pedido.

Tabla 6.4: Holones atómicos de KCG - Parte 1

Holón	**Holarquía a la que pertenece**
Holón Gestor de BD de Materia Prima	Gestión de Almacén de Materia Prima.
Holón orden de Salida de materia	Gestión de Almacén de Materia Prima.
Holón orden de Entrada de materia	Gestión de Almacén de Materia Prima.
Holón Gestor de Almacén de Materia Prima	Gestión de Almacén de Materia Prima.
Holón Silo de Arcilla	Gestión de Almacén de Materia Prima.
Holón Era	Gestión de Almacén de Materia Prima.
Holón Atomizador	Gestión de Almacén de Materia Prima.
Holón Gestor de Información de Compras	Ayuda de Compras.
Holón Pedido de Materia Prima	Ayuda de Compras.
Holón KCG	Gestión de Coordinación de empresas cerámicas.
Holón parte de Pedido de Cliente	Procesamiento de Orden de Fabricación y Planificación Maestra.
Holón de Producción	Gestión de la Coordinación de Producción, Gestión de Coordinación de empresas cerámicas, Simulación de Pedido de Producción y Procesamiento de Orden de Fabricación.
Holón Gestor de Fabricación	Gestión de la Coordinación de Producción, Gestión de Fabricación, Simulación de Pedido de Producción.
Holón Prensa i	Gestión de Fabricación.
Holón Secadora i	Gestión de Fabricación.
Holón Engobe i	Gestión de Fabricación.
Holón Esmalte i	Gestión de Fabricación.
Holón AGV i	Gestión de Fabricación.
Holón Cinta i	Gestión de Fabricación.
Holón Horno i	Gestión de Fabricación.
Holón Almacén de Horno i	Gestión de Fabricación.
Holón Clasificador i	Gestión de Fabricación.
Holón Almacén de Clasificación i	Gestión de Fabricación.
Holón Empaquetador i	Gestión de Fabricación.
Holón Jefe Prensado y Esmaltado i	Gestión de Fabricación.
Holón Jefe Horno i	Gestión de Fabricación.
Holón Jefe Clasificación i	Gestión de Fabricación.
Holón orden de mantenimiento recurso i	Gestión de Fabricación.
Holón orden de Producción	Gestión de Programación y Gestión de Fabricación.
Holón de producto i	Gestión de Programación y Gestión de Fabricación.

Tabla 6.5: Holones atómicos de KCG - Parte 2

Holón	**Holarquía a la que pertenece**
Holón orden de definición de Calendario	Gestión de Programación.
Holón Gestor de Programación	Gestión de Programación.
Holón orden de modificación de Calendario	Gestión de Programación.
Holón Scheduler i	Gestión de Programación.
Holón orden de Simulación de Pedido	Simulación de Pedido de Producción y Ayuda de Ventas.
Holón Plan Maestro	Planificación Maestra.
Holón Gestor de Planificación	Planificación Maestra.
Holón Gestor de BD de Productos Terminados	Gestión de Almacén de Productos Terminados.
Holón orden de Salida de producto terminado	Gestión de Almacén de Productos Terminados.
Holón orden de Entrada de producto terminado	Gestión de Almacén de Productos Terminados.
Holón Gestor de Almacén de Productos Terminados	Gestión de Almacén de Productos Terminados.
Holón unidad de Transporte i	Gestión de Almacén de Productos Terminados.
Holón Gestor de BD de Ventas	Ayuda de Ventas.
Holón orden de Venta	Ayuda de Ventas.



Plantilla de Agente JADE

1. idAgente	*Holón orden de Simulación de Pedido*	2. Plataforma	*Producción KT, WD, MT*

3. Servicios

3.1. Nombre	3.2. Tipo
Activar simulación	*externo*
Inicializar datos Simulación	*externo*
Modificar Datos Simulación	*externo*
Comunicar Resultado Simulación	*externo*

4. Comportamientos

4.1. Nombre	4.2. Tipo	4.3. Servicio que implementa
Inicialización de SImulación	*Secuencial*	*Activar Simulación* *Inicializar datos Simulación* *Modificar Datos Simulación*
Control de Simulación	*Secuencial*	*Lanzar Simulación* *Suspender Simulación* *Reanudar Simulación* *Abortar Simulación*
Simulación Finalizada	*Secuencial*	*Comunicar Resultado Simulación*

5. Ontología

5.1. Nombre	5.2. Ontología Base	5.3. Esquemas
Orden de pedido		*Cliente, Producto, Cantidad, Fecha*
Programador		*Id Programador*
Ayudante de Ventas		*Id Ayudante de Ventas*
Resultado Simulación		*Fecha, Empresa Cerámica*

6. Comunicación

6.1. Mensaje	6.2. Tipo	6.3. Interacción	6.4. Tipo Participación
Activar Simulación	*request*	*Simular Program.*	*Colaborador*
Comunicar Resulta.	*inform*	*Simular Program.*	*Iniciador*
Enviar Orden Simu.	*request*	*Simular Program.*	*Iniciador*
Fecha de Produc.	*agree*	*Simular Program.*	*Colaborador*
Orden de Simul. Imposible de Simul.	*agree*	*Simular Program.*	*Colaborador*

7. Tiene parte de procesamiento físico	*No*

Figura 6.18: Arquitectura del Sistema. Plantilla de Agente JADE del Holón orden de Simulación de Pedido.

6.4. Conclusiones

A lo largo de este ejemplo hemos demostrado la aplicabilidad de nuestra propuesta metodológica a un caso real del sector industrial. Hemos planteado el análisis y diseño de un HMS para la gestión y ayuda a la decisión de varios procesos de negocio y de producción de un grupo de empresas cerámicas. Con este ejemplo se puede constatar que el desarrollo por capas de abstracción, orientado por la noción de Agente Abstracto y las guías de desarrollo de nuestro enfoque, permite que el ingeniero del software se centre en cada momento en el análisis y diseño de las características y requisitos del problema en cuestión (las entidades del nivel actual y sus interacciones) aislando la complejidad de los demás niveles. Así el desarrollo del HMS se traduce en un proceso gradual de identificación y especificación de holones.

Capítulo 7

Conclusiones y Trabajos Futuros

Finalmente en este capítulo resaltamos las contribuciones destacadas del trabajo, las líneas futuras de investigación y las publicaciones relacionadas que hemos producido como resultado del trabajo de investigación desarrollado en la tesis.

7.1. Contribuciones destacadas

Esta tesis consta de dos puntos centrales y complementarios:

- La definición de Agente Abstracto, y
- La definición de una Metodología Multi Agente para Sistemas Holónicos de Fabricación.

Estos dos puntos se basan fundamentalmente en un análisis del enfoque holónico para los sistemas de fabricación y en un extenso estudio del estado del arte del área de Sistemas Holónicos de Fabricación. Este estudio ha demostrado la necesidad de contar con metodologías de ingeniería del software para el desarrollo de Sistemas Holónicos de Fabricación que guíen al ingeniero del software en todas las etapas del ciclo de vida del sistema, que ofrezcan guías claras y no ambiguas, y que proporcionen una infraestructura conceptual adecuada para el modelado e implementación de holones. Este hecho nos motivó a medir el grado de adecuación al área HMS de metodologías de ingeniería del software existentes, con el objetivo de determinar la o las metodologías que

podían adaptarse mejor al modelado de HMS. Como criterios de análisis hemos definido una lista de 8 requisitos de modelado para Sistemas Holónicos de Fabricación, que ha guiado el estudio comparativo de las metodologías. De este estudio hemos concluido que las metodologías del área SMA eran las más apropiadas por varias razones: (i) La tecnología agente es la herramienta de implementación por excelencia del área HMS. (ii) Las similitudes entre las nociones de agente y holón. (iii) La existencia de metodologías multi agente completas que han demostrado buen desempeño en el desarrollo de sistemas complejos. (iv) El grado de satisfacción a los requisitos de modelado de HMS. No obstante, es necesario adaptar las metodologías del área SMA para que puedan satisfacer todos los requisitos de modelado para HMS. Hace falta una infraestructura que proporcione uniformidad de conceptos desde la concepción de la idea hasta su implementación y, sobre todo, que ofrezca un proceso de desarrollo específico del dominio HMS, que guíe al ingeniero del software en todas las etapas del ciclo de vida del sistema. Este ha sido el objetivo principal de esta tesis.

Para profundizar sobre las similitudes entre las nociones de agente y holón, desarrollamos un estudio comparativo de ambos enfoques. La conclusión de este estudio es que los holones y los agentes comparten varias propiedades, y que entre todas las características analizadas, la recursión es la única que no está presente como tal en la tecnología de agentes. Para unificar las nociones de modelado de agente y de holón y para cumplir con uno de los requisitos de modelado de HMS, definimos el concepto de Agente Abstracto como entidad de modelado de entidades autónomas con estructuras recursivas. La definición de Agente Abstracto se basa en una visión funcional de agente y en una visión estructural. La visión funcional es la definición de agente como entidad autónoma y flexible, aceptada comúnmente en la literatura especializada. Mientras que la visión estructural establece que un Agente Abstracto puede ser un Sistema Multi Agente o un agente simple. Hemos ampliado, de esta manera, la noción tradicional de Sistemas Multi Agente, al indicar que un Sistema Multi Agente está formado de dos o más Agentes Abstractos pudiendo ser cada uno de ellos a su vez Sistemas Multi Agente o agentes simples.

Para que la ingeniería del software basada en agentes se pueda aplicar a Sistemas Holónicos de Fabricación, es necesario que exista una infraestructura disponible para que el ingeniero del software pueda emplear los conceptos de la tecnología de agente en el proceso de desarrollo aplicando las técnicas de descomposición, abstracción y organización. Nuestra propuesta metodológica aborda la descomposición y abstracción mediante la noción de Agente Abstracto y un proceso de desarrollo guiado por capas de abstracción. Al mismo tiempo ofrece una serie de modelos y guías de desarrollo específicas para el dominio HMS que permiten especificar cómo se estructuran, relacionan e interaccionan estas entidades. Nuestra propuesta aprovecha trabajos previos reportados en el área SMA para adaptarlos al dominio HMS. Se basa en INGENIAS [Gómez, 2002] y RT-MESSAGE [Julián, 2002], ya que a partir del estudio de metodologías, estas dos han demostrado alto grado de adecuación a los requisitos de modelado para HMS.

La notación de nuestra metodología se define a partir de INGENIAS [Gómez, 2002] y RT-MESSAGE [Julián, 2002] y se fundamenta en los requisitos para el modelado de HMS. De INGENIAS hemos adoptado la división del sistema en cinco vistas y la definición de la notación mediante meta-modelos. De RT-MESSAGE hemos adoptado la identificación de entidades para el modelado de las características de tiempo real de sistemas multi agente. En nuestro enfoque la entidad de modelado central es el Agente Abstracto. Hemos extendido los meta-modelos de INGENIAS para incluir la noción de Agente Abstracto, nuevas entidades para modelar su estructura y comportamiento, características de tiempo real definidas en RT-MESSAGE y relaciones nuevas y/o modificadas. Es importante resaltar que la notación que proponemos es general (no dependiente de HMS), y por lo tanto puede ser utilizada además para el modelado de SMAs de gran escala en dominios generales.

El proceso de desarrollo de nuestra metodología es específico para el dominio HMS y se fundamenta en los 8 requisitos de modelado de HMS. Es un proceso mixto (ascendente y descendente), está guiado por la noción de Agente Abstracto para la identificación y modelado de holones, proporciona guías de modelado claras y específicas para el dominio HMS, y cubre todo el ciclo de vida del HMS mediante fases de desarrollo completas.

En la definición del proceso de desarrollo hemos utilizado SPEM (*Software Process Engineering Metamodel*) [OMG, 2002]. De esta manera: (i) ofrecemos al ingeniero del software una definición fácil de interpretar, y; (ii) facilitamos también, siguiendo el enfoque de [FIPA Methodology TC, 2005], la definición de fragmentos metodológicos (fases o modelos) para su posible reutilización en conexión con fragmentos de otras metodologías.

Por lo tanto, podemos resumir las aportaciones de esta tesis en las siguientes

- Definición de una lista de requisitos de modelado para Sistemas Holónicos de Fabricación.
- Estudio comparativo del grado de adecuación de metodologías de desarrollo de tres áreas: HMS, Modelado de Empresas y Metodologías SMA.
- Estudio comparativo del enfoque holónico y del enfoque agente.
- Definición de Agente Abstracto como entidad de modelado de entidades autónomas con estructuras recursivas.
- Una metodología Multi Agente para Sistemas Holónicos de Fabricación basada en los requisitos de modelado de HMS y en la noción de Agente Abstracto.
- Extensión de los meta-modelos de INGENIAS para incorporar: la noción de Agente Abstracto, entidades recursivas asociadas, relaciones, y características de tiempo real definidas en RT-MESSAGE.
- Un proceso de desarrollo mixto (ascendente y descendente), completo, y con guías de modelado claras y específicas para el dominio HMS.
- Un ejemplo de utilización aplicado a un caso de estudio del mundo real.

7.2. Líneas Futuras de Investigación

Como líneas futuras de investigación podemos destacar tres. Desarrollo de una herramienta de soporte para el modelado por capas de abstracción,

extender la metodología mediante más experiencias de aplicaciones a casos reales del sector industrial y finalmente el estudio y extensión de la metodología para que pueda ser aplicada a dominios generales.

Respecto al desarrollo de una herramienta, podemos destacar que la herramienta INGENIAS IDE [Grasia, 2005] puede ser un buen punto de partida ya que nuestra propuesta metodológica adopta muchas características de INGENIAS. La herramienta deberá gestionar el proceso de descomposición y composición de manera semi-automática, para este proceso se puede estudiar el trabajo desarrollado en [Peña *et al.*, 2002; 2003; MacMAS, 2005]. Para la etapa de implementación se puede adoptar la misma política definida en INGENIAS e incluir un módulo para la generación de código de Bloques Funcionales [IEC, 2000; 2001]. Puede ser interesante también, incluir un módulo de patrones de diseño y reutilización de holarquías parametrizables, para aprovechar desarrollos anteriores y mantener una biblioteca de holarquías.

En cuanto a la aplicación de la metodología a más casos reales del sector industrial, es importante destacar que en la actualidad mantenemos una relación de cooperación con el Centro de Gestión e Ingeniería de la Producción (CIGIP) de esta Universidad. Mediante esta cooperación están surgiendo nuevos dominios de aplicación. Seguir evaluando la metodología en estos dominios, puede ser muy productivo no sólo para medir las características de la metodología sino también para incorporar nuevas guías de ayuda para el ingeniero del software.

La noción de Agente Abstracto puede ser útil no sólo para el modelado de Sistemas Holónicos de Fabricación sino también para el modelado de sistemas complejos de gran escala, en los que se hace necesario trabajar en distintos niveles de abstracción. Estudiar las modificaciones necesarias para adaptar nuestro enfoque metodológico a estos entornos constituye un trabajo interesante. Analizar el modelado de normas, reglas, sistemas abiertos, etc., y estudiar cómo se propagan estas características a los distintos niveles del sistema en desarrollo, son temas abiertos de investigación. Estudiar la definición de fragmentos metodológicos de nuestro enfoque [FIPA Methodology TC, 2005] para conectar fases o modelos de nuestra propuesta con fragmentos de otras metodologías representa también una línea de trabajo interesante.

7.3. Publicaciones Relacionadas con la Tesis

En esta sección presentamos las publicaciones que están relacionadas con esta tesis. Están clasificadas en: artículos en revistas, artículos en congresos internacionales, artículos en congresos nacionales y capítulos de libro.

7.3.1. Artículos en Revistas

- A. Giret and V. Botti. Holons and Agents. *Journal of Intelligent Manufacturing*. Kluwer Academic Publishers. ISSN: 0956-5515. Vol. 15 No. 5 pp. 645-659. (2004)

- A. Giret and V. Botti. Towards an Abstract Recursive Agent. *Integrated Computer-Aided Engineering*. IOS Press. ISSN: 1069-2509. Vol. 11 No. 2 . pp. 165-177. (2004)

7.3.2. Artículos en Congresos Internacionales

- A. Giret. *A Multi Agent Methodology for Holonic Manufacturing Systems*. Proceedings of AAMAS 2005. Doctoral Mentoring Program. To appear. (2005)

- A. Giret, E. Argente, S. Valero, P. Gómez, and V. Julián. *Applying Multi Agent System Modelling to the Scheduling Problem in a Ceramic Tile Factory*. Proceedings of IMCM'05: International Mass Customization Meeting 2005. To appear. (2005)

- A. Giret y V. Botti. *Metodología Multi Agente para Procesos Industriales*. Actas del CAIP' 2005: 7º Congreso Interamericano de Computación Aplicada a la Industria de Procesos. En proceso de publicación. (2005)

- A. Giret and V. Botti. *On the definition of meta-models for analysis of large-scale MAS*. Multiagent System Technologies. Proceedings of MATES 2004. Springer-Verlag, LNAI - 3187 . ISSN: 0302-9743. pp. 273-286. (2004)

- A. Giret and V. Botti. *Towards a Recursive Agent Oriented Methodology for Large-Sacale MAS*. P. Giorgini, J. Müller, and J. Odell (Eds.), Agent-Oriented Software Engineering IV. Springer-Verlag, LNCS 2935 . ISSN: 0302-9743. pp. 25-35. (2004)

- A. Giret and V. Botti. *MAS Beliefs*. Proceedings of IADIS International Conference Applied Computing 2004. ISBN: 972-98947-3-6. pp. 75-78. (2004)

- A. Giret and V. Botti. *Multi Agent System Goals*. Proceedings of IADIS International Conference Applied Computing 2004. ISBN: 972-98947-3-6. pp. 79-82. (2004)

- A. Giret and V. Botti. *Towards a Recursive Agent Model for an Agent-Oriented Methodology*. Proceedings of the Fourth International Workshop on Agent-Oriented Software Engineering, pp. 125-135. (2003)

- A. Giret and V. Botti. *Multi Agent Systems of Multi Agent Systems*. Proceedings of the 2nd International Workshop on Software Engineering for Large-Scale Multi-Agent Systems. (2003)

- A. Giret and V. Botti. *Holons and Agents. Do they Differ?*. Research and Development in Intelligent Systems XIX. Proceedings of ES2002, the Twenty-second SGAI International Conference on Knowledge Based Systems and Applied Artificial Intelligence. Springer. BCS Conference Series. ISBN: 1852336749. pp. 309-322. (2002)

7.3.3. Artículos en Congresos Nacionales

- A. Giret and V. Botti. *Towards a Recursive Agent Oriented Methodology*. Actas del Workshop Agentes Inteligentes en el Tercer Milenio . (2003)

- A. Giret and V. Botti. *Recursive Agent*. Proceedings of the Iberoamerican Workshop on Distributed Artificial Intelligence and Multi-Agent Systems, pp. 1-11. (2002)

7.3.4. Capítulos de Libro

- A. Giret, V. Julián y V. Botti. *Aplicaciones Industriales de los Sistemas Multiagente*. Agentes Software y Sistemas Multi-Agente. Conceptos, Arquitecturas y Aplicaciones. Pearson Prentice Hall. ISBN: 84-205-4367-5. pp. 186-203. (2005)

- V. Botti y A. Giret. *Aplicaciones de los Sistemas Multiagente (SMA) en Industria*. Agentes inteligentes: Sistemas Multiagentes y aplicaciones. Editorial Club Universitario. ISBN: 84-8454-182-7. pp. 19-82. (2002)

Bibliografía

[Agre *et al.*, 1994] Agre, J.; Elsley, G.; McFarlane, D.; Cheng, J.; y Gunn, B. 1994. Holonic Control of Cooling Control System. In *Proceedings of Rensslaers Manufacturing Conference*.

[Ana MAS, 2004] Ana MAS. 2004. *Agentes software y sistemas multiagente. Conceptos, arquitecturas y aplicaciones*. Pearson.

[Andrés, 2001] Andrés, C. 2001. *Programación de la producción en talleres de flujo híbridos con tiempos de cambio de partida dependientes de la secuencia : modelos, métodos y algoritmos de resolución : aplicación a empresas del sector cerámico*. Tesis Doctoral, Universidad Politécnica de Valencia.

[Argente *et al.*, 2004] Argente, E.; Giret, A.; Valero, S.; Julian, V.; y Botti, V. 2004. Survey of MAS Methods and Platforms focusing on organizational concepts. In Vitria, J., Radeva, p. and Aguilo, I., ed., *Recent Advances in Artificial Intelligence Research and Development*, Frontiers in Artificial Intelligence and Applications, 309–316.

[ARIS, 2004] ARIS. 2004. ARIS. http://www.aris.com/.

[Austin, 1962] Austin, J. 1962. *How to Do Things With Words*. Oxford University Press.

[Baker *et al.*, 1999] Baker, A.; Parunak, H.; y Erol, K. 1999. Agents and the internet: Infraestructure for mass customization. *IEEE Internet Computing* 3(5):62–69.

[Beck, 2000] Beck, K. 2000. *Extreme Programming Explained : Embrace Change*, volume XXI. Addison-Wesley.

[Bellifemine *et al.*, 2001] Bellifemine, F.; Poggi, A.; y Rimassa, G. 2001. Developing multi-agent systems with JADE. In *Intelligent Agents VII. Ed. Castelfranchi, C. and Lesperance, Y. 1571*, 89–103.

[Bengoa, 1996] Bengoa, A. 1996. An Aproach to Holonic Components in Control of Machine Tools. *Annals of CIRP* 45(1).

[Biswas *et al.*, 1995] Biswas, G.; Sugato, B.; y Saad, A. 1995. Holonic Planning and Scheduling for Assembly Tasks. *TR CIS-95-01, Center for Intelligent Systems, Vanderbilt University.*

[Bongaerts *et al.*, 1997] Bongaerts, L.; Valckenaers, V.; y Peeters, P. 1997. Reactive Scheduling in Holonic Manufacturing Systems: Architecture, Dynamic Model and Cooperation Strategy. In *Proceedings of the Advanced Summer Institute of the Network of Excellence on Intelligent Control and Integrated Manufactruing Systems.*

[Booch, 1994] Booch, G. 1994. *Object Oriented Analysis and Design with Applications.* Addison Wesley.

[Botti *et al.*, 1999] Botti, V.; Carrascosa, C.; Julian, V.; y Soler, J. 1999. *MAAMAW'99*, volume 1647 - LNAI. Springer Verlag. chapter Modelling agents in hard real-time environments, 63–76.

[Botti y Giret, 2002] Botti, V., y Giret, A. 2002. *Agentes Inteligentes: Sistemas Multiagentes y Aplicaciones.* Editorial Club Universitario - Universidad Internacional Menéndez Pelayo. chapter Aplicaciones de los sistemas multiagente (SMA) en Industria, 83–96.

[Brazier *et al.*, 2002] Brazier, F.; Jonker, C.; y Treur, J. 2002. Principles of Component-Based Design of Intelligent Agents. *Data and Knowledge Engineering* 41:1–28.

[Brennan *et al.*, 1997] Brennan, R.; Balasubramanian, S.; y Norrie, D. 1997.

Dynamic Control Architecture for Advanced Manufacturing Systems. *In Proceedings of International Conference on Intelligent Systems for Advanced Manufacturing.*

[Brennan *et al.*, 2003] Brennan, R. W.; Hall, K.; Marík, V.; Maturana, F.; y Norrie, D. H. 2003. A Real-Time Interface for Holonic Control Devices. In Marík, V.; McFarlane, D.; y Valckenaers, P., eds., *Holonic and Multi-Agent Systems for Manufacturing*, volume LNAI 2744, 25–34. Springer-Verlag.

[Brennan y Norrie, 2001] Brennan, R. W., y Norrie, D. H. 2001. Agents, Holons and Functions Blocks: Distributed Intelligent Control in Manufacturing. *Journal of Applied Systems Studies Special Issue on Industrial Applications of Multi-Agent and Holonic Systems* 2(1):1–19.

[Brennan y Norrie, 2003] Brennan, R. W., y Norrie, D. H. 2003. *From FMS to HMS.* Springer-Verlag. 31–49.

[Brown y McCarragher, 1998] Brown, J., y McCarragher, B. 1998. Maintenance resource allocation using decentralised cooperative control. Technical report, ANU.

[Bruckner *et al.*, 1998] Bruckner, S.; Wyns, J.; Peeters, P.; y Kollingbaum, M. 1998. Designing agents for manufacturing process control. In *Proceedings of Artificial Intelligence and Manufacturing Research Planning Workshop.*

[Buhr, 1998] Buhr, R. 1998. Use case maps as architectural entities for complex systems. *IEEE Transactions on Software Engineering* 24(12):1131–1155.

[Burmeister, 1996] Burmeister, B. 1996. Models and methodologies for agentoriented analysis and design. Technical report d-96-06, DFKI.

[Bussmann *et al.*, 2004] Bussmann, S.; Jennings, N.; y Wooldridge, M. 2004. *Multiagent Systems for Manufacturing Control. A design Methodology.* Springer Verlag.

[Bussmann, 1998] Bussmann, S. 1998. An Agent-Oriented Architecture for Holonic Manufacturing Control. *Proc. of 1st Int. Workshop on Intelligent Manufacturing Systems, EPFL* 1–12.

[Caire *et al.*, 2001] Caire, G.; Leal, F.; Chainho, P.; Evans, R.; Garijo, F.; Gomez-Sanz, J.; Pavon, J.; Kerney, P.; Stark, J.; y Massonet, P. 2001. *Agent Oriented Analysis using MESSAGE/UML*, volume 2222 - LNCS. Springer Verlag. 119–135.

[Castillo y Smith, 2002] Castillo, I., y Smith, J. 2002. Formal modelling methodologies for control of manufacturing cells: survey and comparison. *Journal of Manufacturing Systems* 21(1):40–57.

[Chirn y McFarlane, 1999] Chirn, J., y McFarlane, D. 1999. A Holonic Component-Based Architecture for Manufacturing Control Systems. In *Proceedings of MAS99.*

[Christensen, 1994] Christensen, J. 1994. Holonic Manufacturing Systems: Initial Architecture and Standards Directions. *in First European Conference on Holonic Manufacturing Systems, European HMS Consortium, Hanover, Germany.*

[Christensen, 2003] Christensen, J. 2003. *Agent-Based Manufacturing. Advances in the Holonic Approach.* Springer Verlag. chapter HMS/FB Architecture and Its Implementation, 53–88.

[Collinot *et al.*, 1996] Collinot, A.; Drogoul, A.; y Benhamou, P. 1996. Agent-oriented design of soccer robot team. In *Proceedings of the Second International Conference on Multi Agent Systems (ICMAS'96)*, 41–47. AAAI Press.

[Colombo *et al.*, 2002] Colombo, A.; Neubert, R.; y Sussmann, B. 2002. A coloured Petri net-based approach towards a fomral specification of agent-controlled production systems. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics.*

[Corkill y Lander, 1998] Corkill, D., y Lander, S. 1998. Diversity in agent organizations. *Object Magazine* 8(4):41–47.

[Cpckburn y Jennings, 1999] Cpckburn, D., y Jennings, N. 1999. ARCHON: a DAI system for industrial applications. *In Foundations of Distributed Artificial Intelligence, eds. G. OHare and N. Jennings, Wiley* 319–344.

[Cutkosky *et al.*, 1993] Cutkosky, M.; Engelmore, R.; Fikes, R.; Genesereth, M.; Gruber, T.; Mark, W.; Tenenbaum, J.; y Weber, J. 1993. PACT: An Experiment in Integrating Concurrent Engineering Systems. *IEEE Computer* 26(1):28–37.

[Deen y Fletcher, 2000] Deen, S. M., y Fletcher, M. 2000. Temperature equilibrium in multi-agent manufacturing systems. In *Proceedings of the 11th International Workshop on Database and Expert Systems Applications*, 259. IEEE Computer Society.

[Deen, 1993] Deen, S. 1993. Cooperation issues in Holonic Manufacturing Systems. *Information Infrastructure Systens for Manufacturing* B-14:401–412.

[Deen, 1994] Deen, S. 1994. A cooperation framework for holonic interactions in manufacturing. *In Proceedings of the Second International Working Conference on Cooperating Knowledge Based Systems (CKBS'94.*

[Dellarocas y Klein, 1999] Dellarocas, C., y Klein, M. 1999. Civil agent societies: Tools for inventing open agent-mediated electronic marketplaces. In *ACM Conference on Electronic Commerce (EC-99) (at IJCAI'99).*

[Demarco y Plauger, 1979] Demarco, T., y Plauger, P. J. 1979. *Structured Analysis and System Specification.* Prentice-Hall.

[Dignum *et al.*, 2002] Dignum, V.; Meyer, J.; Weigand, H.; y Dignum, F. 2002. An organization-oriented model for agent societies. In Lindemann, D. Moldt, M. P. . B. Y. E., ed., *International Workshop on Regulated Agent-Based Social Systems: Theory and Applications (RASTA'02)*, 31–50.

[Dilts *et al.*, 1991] Dilts, D.; Boyd, N.; y Whorms, H. 1991. The Evolution of Control Architectures for Automated Manufacturing Systems. *Journal of Manufacturing Systems* 10(1):79–93.

[Doumeingts, 1998] Doumeingts, G. 1998. Grai grid decisional modelling. In Bernus, P., ed., *Handbook on Architecture of Information System*, 313–337. Springer-Verlag.

[Duffie y Prabhu, 1994] Duffie, N., y Prabhu, V. 1994. Real-time distributed scheduling of heterarchical manufacturing systems. *Journal of Manufacturing Systems* 13(2):94–107.

[Elammari y Lalonde, 1999] Elammari, M., y Lalonde, W. 1999. An agent-oriented methodology: high-level and intermediate models. In *Proceedings of First Bi-Conference. Workshop on Agetn-Oriented Information Systems (AOIS'99).*

[ESPRIT, 1993] ESPRIT. 1993. *CIMOSA: Open Systen Architecture for CIM*, volume 1 of Research Reports ESPRIT, Project 688/5288 AMICE. Springer-Verlag.

[Esteva *et al.*, 2001] Esteva, M.; Rodriguez-Aguilar, J.; Sierra, C.; Arcos, J.; y Garcia, P. 2001. *On the Formal Specification of Electronic Institutions.* Lecture Notes in Artificial Intelligence 1991. Springer-Verlag. 126–147.

[EURESCOM, 2000] EURESCOM. 2000. MESSAGE: Methodology for engineering systems of software agents. Initial methodology. Technical report p907-d1, EURESCOM.

[EURESCOM, 2001a] EURESCOM. 2001a. Final guidelines for the identification of relevant problem areas where agent technology is appropiate. Technical report p907-d2, EURESCOM.

[EURESCOM, 2001b] EURESCOM. 2001b. MESSAGE: Methodology for engineering systems of software agents (Final). Technical report p907-ti1, EURESCOM.

[Feldmann y Colombo, 1998] Feldmann, K., y Colombo, A. 1998. Material flow and control sequence specification of flexible production systems using coloured Petri nets. *International Journal of Advanced Manufacturing Technology* 14:760–774.

[Ferber *et al.*, 2004] Ferber, J.; Gutknecht, O.; y Michel, F. 2004. From Agents to Organizations: an Organizational View of Multi-Agent Systems. In Giorgini, P.; Muller, J.; y Odell, J., eds., *Agent-Oriented Software Engineering VI*, volume LNCS 2935 of *Lecture Notes in Computer Science*, 214–230. Springer-Verlag.

[Ferber y Gutknecht, 1998] Ferber, J., y Gutknecht, O. 1998. A meta-model for the analysis and design of organizations in multi-agent systems. In *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS'98)*, 128–135. IEEE Computer Society.

[Ferber, 1999] Ferber, J. 1999. *Multi-Agent Systems*. Addison-Wesley.

[Ferguson, 1995] Ferguson, I. 1995. On the role of BDI modeling for integrated control and coordinated behavior in autonomous agents. *Applied Artificial Intelligence* 4(9):421–448.

[Finin *et al.*, 1994] Finin, T.; Fritzson, R.; McKay, D.; y McEntire, R. 1994. KQML as an Agent Communication Language. In *Proceedings of the 3rd International Conference on Information and Knowledge Management CIKM'94*, 456–463. Gaithersburg, Maryland: ACM Press.

[FIPA Methodology TC, 2005] FIPA Methodology TC. 2005. FIPA Methodology Technical Committee. *http://www.fipa.org/activities/methodology.html*.

[FIPA Modeling TC, 2005] FIPA Modeling TC. 2005. Agent Class Superstructure Metamodel - Working Documment. *httt://www.auml.org/auml/documments/*.

[FIPA, 2002] FIPA. 2002. Arcol. Foundation for Intelligent Physical Agents. *http://www.fipa.org/*.

[FIPA, 2005] FIPA. 2005. Foundation for Intelligent Physical Agent. *httt://www.fipa.org/.*

[Fischer *et al.*, 1996] Fischer, K.; Müller, J.; y Pischel, M. 1996. AGenDA: A general testbed for distributed artificial intelligence applications. *In G. M. P. OHare and J. N. R, editors, Foundations of Distributed Artificial Intelligence* 401–427.

[Fischer *et al.*, 2003] Fischer, K.; Schillo, M.; y Siekmann, J. 2003. Holonic Multiagent Systems: A Foundation fo Organisation of Multiagent Systems. *Holonic and Multi-Agent Systems for Manufacturing. LNAI 2744. ISSN 0302-9743* 71–80.

[Fischer, 1998] Fischer, K. 1998. An Agent-Based Approach to Holonic Manufacturing Systems. *in L. M. Camarinha-Matos, H. Afsarmanesh and v. Marik (Eds.) Intelligent Systems for Manufacturing. Multi-Agent Systems and Virtual Organisations* 3–12.

[Fischer, 1999] Fischer, K. 1999. Agent-Based Design of Holonic Manufacturing Systems. *Journal of Robotics and Autonomous Systems* 27(1-2):3–13.

[Fletcher *et al.*, 2000] Fletcher, M.; Garcia-Herreros, E.; Chritensen, J.; Deen, S.; y Mittmann, R. 2000. An Open Architecture for Holonic Cooperation and Autonomy. *Proceeding of HoloMAS'2000.*

[Fletcher y Deen, 2001] Fletcher, M., y Deen, M. S. 2001. Fault-tolerant holonic manufacturing systems. *Concurrency and Computation: Practice and Experience* 13(1):43–70.

[Fox *et al.*, 1993] Fox, M.; Chionglo, J.; y Barbuceanu, M. 1993. The Integrated Supply Chain Management System. *Internal Report, Dept. of Industrial Engineering, Univ. of Toronto.*

[Franklin y Graesser, 1996] Franklin, S., y Graesser, A. 1996. It is an agent, or just a Program?: A Taxonomy for Autonomous Agents. *Proceedings of*

the Third International Workshop on Agent Theories, Architectures, and Languages. Springer-Verlag.

[Galliers, 1988] Galliers, J. R. 1988. *A Theoretical Framework for Computer Models of Cooperative Dialogue, Acknowledging Multi-Agent Conflict.* Tesis Doctoral, Open University, UK.

[Gasser, 1992] Gasser, L. 1992. Boundaries, Identity and Aggregation: Plurality Issues in Multi-Agent Systems. *Decentralized A.I.-3* 199–213.

[Gasser, 2001] Gasser, L. 2001. *Perspective on Organizations in Multi-Agent Systems.* Springer-Verlag. 1–16.

[Gayed *et al.*, 1998] Gayed, N.; Jarvis, D.; y Jarvis, J. 1998. A Strategy for the Migration of Existing Manufacturing Systems to Holonic Systems. In *Proceedings of IEEE International Conference on Systems, Man and Cybernetics*, 319–324.

[Georgeff y Ingrand, 1989] Georgeff, M. P., y Ingrand, F. F. 1989. Monitoring and control of spacecraft systems using procedural reasoning. *Technical Report 03, Australian Artificial Intelligence Institute, Melbourne, Australia.*

[Georgeff y Rao, 1995] Georgeff, M., y Rao, A. 1995. BDI agents: From theory to practice. In *Proceedings of the First International Conference on Multi-Agents Systems (ICMAS-95).*

[Giret y Botti, 2004a] Giret, A., y Botti, V. 2004a. *Multiagent System Technologies.* LNAI 3187. chapter On the definition of meta-models for analysis of large-scale MAS, 273–286.

[Giret y Botti, 2004b] Giret, A., y Botti, V. 2004b. Towards a Recursive Agent Oriented Methodology for Large-Scale MAS. *Agent-Oriented Software Engineering IV.* LNCS 2935:25–35.

[Gómez y Pavón, 2002] Gómez, J., y Pavón, J. 2002. Meta-modelling in Agent Oriented Software Engineering. *8th Ibero-American Conference on AI*

(Iberamia 2002) : F.J. Garijo, J.C. Riquelme, M. Toro. Advances in Artificial Intelligence, LNAI 2527 606–615.

[Gómez, 2002] Gómez, J. J. 2002. *Modelado de Sistemas Multi-agente*. Tesis Doctoral, Universidad Complutense de Madrid. Facultad de Informática.

[Gmytrasiewics y Durfee, 1995] Gmytrasiewics, P., y Durfee, E. 1995. A rigorous, operational formalization of recursive modeling. *In Proceedings of the First International Conference on Multi-Agent Systems* 125–132.

[Gou *et al.*, 1994] Gou, L.; Hasegawa, T.; Luh, P.; Tamura, S.; y Oblak, J. 1994. Holonic Planning and Scheduling for a Robotic Assembly Testbed. In *Proceedings of the 4th Rensselaer International Conference on Computer Integrated Manufacturing and Automation Technology.*

[Gou *et al.*, 1998] Gou, L.; Luh, P.; y Kyoya, Y. 1998. Holonic Manufacturing Scheduling: architecture, cooperation, mechanism, and implementation. *Computers in Industry* 37:213–231.

[Grasia, 2005] Grasia. 2005. INGENIAS IDE. http://ingenias.sourceforge.net/.

[Groover, 1987] Groover, M. 1987. *Automation, Production Systems, and Computer Integrated Manufacturing.* Prentice Hall: Englewood Cliffs, NY.

[Gruver *et al.*, 2003] Gruver, W.; Kotak, D.; Van Leeuwen, E.; y Norrie, D. 2003. Holonic Manufacturing Systems: Phase II. *Holonic and Multi-Agent Systems for Manufacturing. Lecture Notes in Artificia Intelligence, Springer-Verlag. ISBN 3-540-40751-0* LNAI 2744.

[Haddadi y Sundermeyer, 1996] Haddadi, A., y Sundermeyer, K. 1996. Belief-desire-intention agent architectures. *In G. M. P.OHare and J. N. R, editors, Foundations of Distributed Artificial Intelligence* 169–186.

[Hadeli *et al.*, 2004] Hadeli; Valckenaers, P.; Kollingbaum, M.; y Van Brussel, H. 2004. Multi-agent coordination and control using stigmergy. *Computers in Industry* 53(1):75–96.

[Halpern y Moses, 1986] Halpern, J., y Moses, Y. 1986. Knowledge and common knowledge in a dsitributed environment. *In Proceedings 3rd Annual ACM Conference on Principles of Distributed Computing* 50–61.

[Hasegawa *et al.*, 1994] Hasegawa, T.; Gou, L.; Tamura, S.; Luh, P.; y Oblak, J. 1994. Holonic Planning and Scheduling Architecture for Manufacturing. *In Proceedings of the 2nd International Working Conference on Cooperating Knowledge-based Systems*.

[Hatvany, 1985] Hatvany, J. 1985. Intelligence and cooperation in heterarchic manufacturing systems. *Robotics and Computer-Integrated Manufacturing* 2(2):101–104.

[Heikkila *et al.*, 1995] Heikkila, T.; Jarviluoma, M.; y Hasemann, J. 1995. Holonic Control of a Manufacturing Robot Cell. Technical report, VTT Automation.

[Heikkila *et al.*, 1997] Heikkila, T.; Jarviluoma, M.; y Juntunen, T. 1997. Holonic Control for Manufacturing Systems: Design of a Manufacturing Robot Cell. *Integrated Computer Aided Engineering* 4:202–218.

[Hitomi, 1994] Hitomi, K. 1994. *Handbook of DEsign, Manufacturing and Automation*. John Wiley & Sons: New York. chapter Analysis and design of manufacturing systems, 405–433.

[HMS, 1994] HMS, P. R. 1994. *HMS Requirements*. http://hms.ifw.uni-hannover.de/: HMS Server.

[Huhns y Singh, 1998] Huhns, M., y Singh, M. 1998. Readings in Agents. *Morgan Kaufman Publ.*

[IDEF, 2004] IDEF. 2004. IDEF Family of Methods. http://www.idef.com/.

[IEC, 2000] IEC. 2000. *International Electrotechnical Commission: Function Blocks, Part 1 - Architecture.PAS 61499-1.*

[IEC, 2001] IEC. 2001. *International Electrotechnical Commission: Function Blocks, Part 1 - Software Tool Requirenments.PAS 61499-2.*

[Iglesias *et al.*, 1998] Iglesias, C.; Garijo, M.; Gonzalez, J.; y Velasco, J. 1998. Analysis and design of multi agent systems using MAS-CommonKADS. *In Singh, M. P., Rao, A., and Wooldridge, M.J. (eds.) Intelligent Agentd IV LNAI, Springer Verlag* 1365.

[Iwata y Onosato, 1994] Iwata, K., y Onosato, M. 1994. Random Manufacturing System: a New Concept of Manufacturing Systems for production to order. *Annals of the CIRP* 43/1/1994.

[Jacobson *et al.*, 1999] Jacobson, I.; Booch, G.; y Rumbaugh, J. 1999. *The Unified Software Development Process.* Addison Wesley.

[Jacobson, 1992] Jacobson, I. 1992. *Object-Oriented Software Engineering: A Use Case Driven Approach.* Addison Wesley.

[JADE, 2005] JADE. 2005. Java Agent DEvelopment Framework. *http://jade.tilab.com/.*

[Jennings *et al.*, 1992] Jennings, N. R.; Mamdani, E.; Laresgoiti, I.; Pérez; y Correa, J. 1992. GRATE: A general framework for cooperative problem solving. *IEE-BCS Journal of Intelligent Systems Engineering* 1(2):102–114.

[Jennings *et al.*, 1995] Jennings, N. R.; Corera, J.; y Laresgoiti, I. 1995. Developing Industrial Multi-Agent Systems. *In Proceedings of ICMAS'95* 423–430.

[Jennings y Wooldridge, 1998] Jennings, N. R., y Wooldridge, M. 1998. Applications of Intelligent Agents. *Agent Technology: Foundations, Applications, and Markets* 3–28.

[Jones y McLean, 1996] Jones, A., y McLean, C. 1996. A Proposed Hierarchical Control Model for Automated Manufacturing Systems. *Journal of Manufacturing Systems* 5(1).

[Joshi y Smith, 1994] Joshi, S., y Smith, J. 1994. Computer Control of Flexible Manufacturing Systems, Research and Development. *Chapman & Hall, New York*.

[Juan *et al.*, 2002] Juan, T.; Pierce, A.; y Sterling, L. 2002. Roadmap: Extending the GAIA methodology for complex open systems. In *The First International Joint Conference on Autonomous Angents and Multiagent Systems*, 3–10. ACM Press, ISBN 1-58113-480-0.

[Julián, 2002] Julián, V. J. 2002. *RT-MESSAGE: Desarrollo de Sistemas Multiagente de Tiempo Real*. Tesis Doctoral, Universidad Politécnica de Valencia. Departamento de Sistemas Informáticos y Computación.

[Kalpakjian y Schmid, 2002] Kalpakjian, S., y Schmid, S. 2002. *Manufactura, ingeniería y tecnología*. Pearson Educación.

[Kendall *et al.*, 1996] Kendall, E.; Malkoun, M.; y Jiang, C. 1996. A methodology for developing agent based systems. In Zhang, C., y Lukose, D., eds., *Distributed Artificial Intelligence - Architecture and Modelling*, volume 1087 - LNAI, 85–99. Springer Verlag.

[Kimura, 1993] Kimura, F. 1993. A Product and Process Model for Virtual Manufacturing Systems. *Annals of the CIRP* 42/1/1993.

[Kinny y Georgeff, 1997] Kinny, D., y Georgeff, M. 1997. *Modelling and Design of Multi-Agent Systems*. Springer-Verlag.

[Koestler, 1971] Koestler, A. 1971. *The Ghost in the Machine*. Arkana Books.

[Kripke, 1963] Kripke, S. 1963. Semantical analysis of modal logic. In *Z. Math. Logik Grundl. Math 9*, 67–96.

[Kruchten, 2000] Kruchten, P. 2000. *The Rational Unified Process : An Introduction*, volume XVIII. Addison-Wesley.

[Leitao y Restivo, 2002] Leitao, P., y Restivo, F. 2002. Holonic Adactive Production Control Systems. *In Proceedings of special session on Agent-based Intelligent Automation and Holonic Control Systems of the 28th Anual Conference of the IEEE Industrial Society* 2968–2973.

[Leitao y Restivo, 2003a] Leitao, P., y Restivo, F. 2003a. An Approach to the Formal Specification of Holonic Control Systems. *Holonic and Multi-Agent Systems for Manufacturing. LNAI 2744. ISSN 0302-9743* 59–70.

[Leitao y Restivo, 2003b] Leitao, P., y Restivo, F. 2003b. Identification of ADACOR Holons for Manufacturing Control. In *Proceedings of the 7th IFAC Workshop on Intelligent Manufacturing Systems*, 109–114.

[Lin y Solberg, 1992] Lin, G.-J., y Solberg, J. 1992. Integrated Shop Floor Control Using Autonomous Agents. *IIE Transactions: Design and Manufacturing* 26(3):57–71.

[Lin y Solberg, 1994] Lin, G.-J., y Solberg, J. 1994. Autonomous Control for Open Manufacturing Systems. In Joshi, S., y Smith, J., eds., *Computer Control of Flexible Manufacturing Systems*, 169–206.

[Lind, 1999] Lind, J. 1999. *MASSIVE: Software Engineering for Multi-agent Systems.* Tesis Doctoral, DFKI.

[Lyytinen y Rossi, 1999] Lyytinen, K. S., y Rossi, M. 1999. METAEDIT+ - A Fully Configurable Multi User and Multi Tool CASE and CAME Environment. *Springer Verlag LGNS 1080.*

[MacMAS, 2005] MacMAS. 2005. MaCMAS. http://www.tdg-seville.info/joaquinp/MaCMAS/index.htm.

[Marcus *et al.*, 1996] Marcus, A.; Kis Vancza, T.; y Monostori, L. 1996. A market approach to holonic manufacturing. *Annals of CIRP* 45:433–436.

[Matsatsinis *et al.*, 2003] Matsatsinis, N.; Moraïtis, P.; Psomatakis, V.; y Spanoudakis, N. 2003. An Agent-Based System for Products Penetration Strategy Selection. *Applied Artificial Intelligence* 17:901–925.

[Maturana y Norrie, 1996] Maturana, F., y Norrie, D. 1996. Multi-Agent Mediator Architecture for Distributed manufacturing. *Journal of Intelligent Manufacturing* 7:257–270.

[Maturana y Norrie, 1997] Maturana, F., y Norrie, D. 1997. Distributed decision-making using the contract net within a mediator architecture. *Decision Support Systems 20* 53–64.

[McFarlane y S., 2003] McFarlane, D., y S., B. 2003. *Agent-Based Manufacturing. Advances in the Holonic Approach.* Springer-Verlag. chapter Holonic Manufacturing Control: Rationales, Developments and Open Issues, 301–326.

[McFarlane, 1995] McFarlane, D. 1995. Holonic Manufacturing Systems in Continuos Processing: Concepts and Control Requirements. In *Proceedings of ASI 95.*

[MESA, 1995] MESA. 1995. MESA and International. Control Definition and MES to Controls Data Flow Possibilities. *White Paper No 3.*

[Meyer y van der Hoek, 1993] Meyer, J.-J., y van der Hoek, W. 1993. A Default Logic Based on Epistemic States. *in Symbolic and Quantitative Approaches to Reasoning and Uncertainty. Proc. ECSQARU'93. Springer Verlag* 265–273.

[Miles *et al.*, 2001] Miles, S.; Joy, M.; y Luck, M. 2001. *Agent-Oriented Software Engineering*, volume 1657 - LNAI. Springer Verlag. chapter Designing agent-oriented systems by analysing agent interactions, 171–183.

[Müller, 1996] Müller, J. P. 1996. The design of intelligent agents: a layered approach. *Volume 1177 of Lecture notes in artificial intelligence. Springer.*

[Monceyron y Barthes, 1992] Monceyron, E., y Barthes, J. 1992. Architecture for ICAD systems: an example form harbor design. *Revue Sciences et Techniques de la Conception* 1(1):49–68.

[Moulin y Brassard, 1996] Moulin, B., y Brassard, M. 1996. *Distributed Artificial Intelligence - Architecture and Modelling*, volume 1087 - LNAI. Springer Verlag. chapter A scenario-based design method and an environment for the development of multiagent systems, 216–232.

[Murata, 1989] Murata, T. 1989. Petri nets: properties, analysis, and applications. In *Proceedings of the IEEE*, volume 77, 541–580.

[Mylopoulos y Castro, 2001] Mylopoulos, J.and Kolp, M., y Castro, J. 2001. UML for agent-oriented software development: The TROPOS proposal. In *Proceedings of the 4th Int. Conf. on th Unified Modeling Language UML'01.*

[Ng *et al.*, 1996] Ng, A.; Yeung, R.; y Cheung, E. 1996. HSCS - the design of a holonic shopfloor control system. In *Proceedings of IEEE Conference on Emerging technologies and Factory Automation*, 179–185.

[Nwana, 1996] Nwana, H. S. 1996. Software Agents: An Overview. *Intelligent Systems Research. AA&T, BT Laboratories.*

[Occello, 2000] Occello, M. 2000. Towards a Recursive Generic Agent Model. *In Proceedings of International Conference on Artificial Intelligence.*

[Odell *et al.*, 2005] Odell, J.; Nodine, M.; y Levy, R. 2005. A Metamodel for Agents, Roles, and Groups. In James Odell, P. Giorgini, J. M., ed., *Agent-Oriented Software Engineering (AOSE) V*, Lecture Notes in Computer Science. Springer.

[OHare y Jennings, 1996] OHare, G., y Jennings, N. 1996. Foundation of Distributed Artificial Intelligence. *John Wiley & Sons.*

[Okino, 1993] Okino, N. 1993. Bionic Manufacturing Systems. *Flexible Manufacturing Systems, Past, Present, Future* 73–95.

[OMG, 2002] OMG, O. M. G. 2002. Software Process Engineering Metamodel Specification Version 1.0. *http://www.omg.org/docs/formal/02-11-14.pdf*.

[OMG, 2004] OMG, O. M. G. 2004. MOF. Meta Object Facility. *http://www.omg.org/technology/documents/formal/mof.htm*.

[OMG, 2005] OMG, O. M. G. 2005. UML. Unified Modeling Language. *http://www.uml.org/*.

[Overmars y Toncich, 1996] Overmars, A., y Toncich, D. 1996. Hybrid FMs Control Architectures Based on Holonic Principles. *International Journal of Flexible Manufacturing Systems* 8:263–278.

[Padgham y Winikoff, 2002] Padgham, L., y Winikoff, W. 2002. Prometheus: A methodology for developing intelligent agents. In *Proceedings of the Third International Workshop on AgentOriented Software Engineering, at AAMAS 2002*, 135–145.

[Park *et al.*, 1993] Park, H.; Tenenbaum, M.; y Dove, R. 1993. Agile Infrastructure for Manufacturing Systems (AIMS): A Vision for Transforming the US Manufacturing Base. *Defence Manufacturing Conference*.

[Parunak *et al.*, 1997] Parunak, V.; Ward, A.; Fleischer, M.; y Sauter, J. 1997. A Marketplace of Design Agents for Distributed Concurrent Set-Based Design. *In Proceedings of the Fourth ISPE International Conference on Concurrent Engineering: Research and Applications*.

[Parunak *et al.*, 2002] Parunak, V. D.; Breuckner, S.; Fleischer, M.; y Odell, J. 2002. Co-X: Defining what Agents Do Together. *Proceedings of the AAMAS 2002 Workshop on Teamwork and Coalition Formation, Onn Shehory, Thomas R. Ioerger, Julita Vassileva, John Yen, eds.*

[Parunak y Odell, 2002] Parunak, V. D., y Odell, J. 2002. Representing Social Structures in UML. *In Agent-Oriented Software Engineering II, M. Wooldridge, G. Weiss, and P. Ciancarini, eds. Springer Verlag* 1–16.

[Parunak, 1987] Parunak, V. 1987. Manufacturing Experience with the Contract Net. *Distributed Artificial Intelligence, Huhns, M.N. ed., Pitman* 285–310.

[Pavón y Gómez, 2003] Pavón, J., y Gómez, J. 2003. Agent Oriented Software Engineering with INGENIAS. *3rd International Central and Eastern European Conference on Multi-Agent Systems (CEEMAS 2003) : V. Marik, J. Müller, M. Pechoucek:Multi-Agent Systems and Applications II, LNAI 2691* 394–403.

[Peña *et al.*, 2002] Peña, J.; Corchuelo, R.; y Arjona, J. 2002. Towards Interaction Protocol Operations for Large Multi-agent Systems. In *Proceedings of the Second International Workshop on Formal Approaches to Agent-Based Systems*, volume LNCS 2699, 79–91. Springer-Verlag.

[Peña *et al.*, 2003] Peña, J.; Corchuelo, R.; y Arjona, J. 2003. A Top Down Approach for MAS Protocol Descriptions. In *Proceedings of the ACM Symposium on Applied Computing SAC'03*, 45–50. ACM.

[Pechoucek *et al.*, 2000] Pechoucek, M.; Marík, V.; y Stepankova, O. 2000. Coalition Formation in Manufacturing Multi-Agent Systems. In *11th International Workshop on Database and Expert Systems Applications (DEXA'00)*, 241–246. IEEE Computer Society DL.

[Peng *et al.*, 1998] Peng, Y.; Finin, T.; Labrou, Y.; Chu, B.; Long, J.; Tolone, W.; y Boughannam, A. 1998. A Multi-Agent System for Enterprise Integration. *In Proceedings of PAAM'98.*

[Prosser y Buchanan, 1994] Prosser, P., y Buchanan, I. 1994. Intelligent Scheduling: past, present and future. *Intelligent Systems Engineering.*

[Purvis *et al.*, 2002] Purvis, M.; Cranefield, S.; y Nowostawski, M. 2002. A Multi-level Infrastructure for Agent-Oriented Development. *Proceedings of the First International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2002). ACM Press* 88–89.

[Ramos, 1996] Ramos, C. 1996. A holonic approach for task scheduling in manufacturing systems. In *Proceedings of IEEE Conference on Robotics and Automation*, 2511–2516.

[Rannanjarvi y Heikkila, 1998] Rannanjarvi, L., y Heikkila, T. 1998. Software Development for Holonic Manufacturing Systems. *Computers in Industry* 37(3):233–253.

[Rao y Georgeff, 1992] Rao, A., y Georgeff, M. 1992. An abstract architecture for rational agents. In *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning*, 439–449.

[Ritter *et al.*, 2002] Ritter, A.; Baum, W.; Hopf, M.; y Westkamper, E. 2002. Agentification for production systems. In *Second International Workshop on Integration of Specification Techniques for Applications in Engineering.*

[Robinson, 2000] Robinson, D. J. 2000. A Component Based Approach to Agent Specification. School of Engineering. Master's thesis, Air Force Institute of Technology.

[Russell y Norvig, 1995] Russell, S., y Norvig, P. 1995. *Artificial Intelligence: A Modern Approach*. Englewood Cliffs, New Jersey: Prentice Hall.

[Saad *et al.*, 1995] Saad, A.; Biswas, G.; Kawamura, K.; Johnson, M.; y Salama, A. 1995. Evaluation of Contract Net-Based Heterarchical Scheduling for Flexible Manufacturing Systems. *In Proceedings of the 1995 International Joint Conference on Artificial Intelligence (IJCAI'95)* 310–321.

[Schreiber, 1999] Schreiber, G. 1999. *Knowledge Engineering & Management: The CommonKADS Methodology*. MIT Press.

[Seidel, 1994] Seidel, D. 1994. HMS - Strategies. *Vol0, Overview, IMS/HMS TC5 deliverable.*

[Sen y Weiss, 1999] Sen, S., y Weiss, G. 1999. Learning in multi-agent systems. *In G. Weiss, editor, Multi-Agent Systems: A modern Approach to Distributed AI* 259–298.

[Shehory y Kraus, 1998] Shehory, O., y Kraus, S. 1998. Methods for task allocation via agent coalition formation. *Artificial Intelligence Journal* 101(1-2):165–200.

[Shen *et al.*, 1998] Shen, W.; Maturana, F.; y Norrie, D. 1998. Learning in Agent-Based Manufacturing Systems. *In Proceedings of AI & Manufacturing Research Planning Workshop.*

[Shen *et al.*, 2001] Shen, W.; Norrie, D.; y Barthes, J. 2001. *Multi-agent Systems for Concurrent DEsign and Manufacturing.* Taylor and Francis.

[Shen y Norrie, 1999] Shen, W., y Norrie, D. 1999. Agent-Based Systems for Intelligent Manufacturing: A State-of-the-Art Survey. *Knowledge and Information Systems, an Internatinal Journal* 1(2):129–156.

[Sierra *et al.*, 2004] Sierra, C.; Rodríguez-Aguilar, J.; Noriega, P.; Esteva, M.; y Arcos, J. 2004. Engineering multi-agent systems as electronic institutions. *European Journal for the Informatics Professional* 4.

[Sikora y Shaw, 1997] Sikora, R., y Shaw, M. J. P. 1997. Coordination Mechanisms for Multi-agent Manufacturing Systems: Application to Integrated Manufacturing Scheduling. *IEEE Transactions on Engineering Management* 44(2):175–187.

[Simon, 1990] Simon, H. 1990. *The Science of the Artificial.* (second ed.) MIT Press Cambridge (Mass.).

[Smith, 1980] Smith, R. 1980. The contrat net protocol: High-level communication and control in a distributed problem solver. *IEEE Transactions on Computers* 29(12):1104–1113.

[Soler *et al.*, 2002] Soler, J.; Julian, V.; Rebollo, M.; Carrascosa, C.; y Botti,

V. 2002. Towards a real-time MAS architecture. In *Proceedings of Challenges in Open Agent Systems. AAMAS'02.*

[Spur *et al.*, 1996] Spur, G.; Mertins, K.; y Jochem, R. 1996. *Integrated Enterprise Modelling.* Beuth Verlag GmbH.

[Sugimura *et al.*, 1996] Sugimura, N.; Hiroi, M.; Moriwaki, T.; y Hozumi, K. 1996. A study on holonic scheduling for manufacturing systems of composite parts. In *Proceedings of Japan/USA Symposium on Flexible Manufacturing*, 1407–1410.

[Tambe, 1995] Tambe, M. 1995. Recursive agent and agent-group tracking in a real-time dynamic environment. In Lesser, V., y Gasser, L., eds., *Proceedings of the First International Conference on Multiagent Systems (ICMAS'95)*, 368–375. San Francisco, CA, USA: AAAI Press.

[Tanaya *et al.*, 1997] Tanaya, P.; Detand, J.; y Kruth, J. 1997. Holonic Machine Controller: A Study and Implmentation of Holonic Behaviour to Current NC Controller. *Computers in Industry* 33:325–333.

[Tanaya *et al.*, 1998] Tanaya, P.; Detand, J.; Kruth, J.; y Valckenaers, P. 1998. Object-Oriented Execution Model For a Machine Controller Holon. *European Journal of Control* 4(4):345–361.

[Ueda, 1993] Ueda, K. 1993. A Genetic Approach toward Future Manufacturing Systems. *Flexible Manufacturing Systems, Past, Present, Future* 221–228.

[UEML, 2003] UEML. 2003. Unified Entreprize Modelling Language. http://www.ueml.org/.

[Ulieru, 2001] Ulieru, M. 2001. FIPA technology overview: holonic enterprise. *FIPA Inform* 2(2):3–4.

[Valckenaers *et al.*, 1994] Valckenaers, P.; Van Brusel, H.; Bongaerts, L.; y Wyns, J. 1994. *Results of the Holonic Control System Bechmark at the KULeuven.* Proceedings of the CIMAT Conference.

[Valckenaers *et al.*, 1995] Valckenaers, P.; Van Brusel, H.; Bongaerts, L.; y Bonneville, F. 1995. Programming, Scheduling and Control of Flexible Assembly Systems. *Computers in Industry* 26:209–218.

[Van Brussel *et al.*, 1998] Van Brussel, H.; Wyns, J.; Valckenaers, P.; Bongaerts, L.; y Peeters, P. 1998. Reference Architecture for Holonic Manufacturing Systems: PROSA. *Computers In Industry* 37:255–274.

[Van Brussel *et al.*, 1999] Van Brussel, H.; Bongaerts, L.; Wyns, J.; Valckenaers, P.; y Van Ginderachter, T. 1999. A Conceptual Framework for Holonic Manufacturing Systems: Identification of Manufacturing Holons. *Journal of Manufacturing Systems* 18(1):35–52.

[Van der Hoek y Meyer, 1991] Van der Hoek, W., y Meyer, J. 1991. Graded Modalities for Epistemic Logic. *Logique et Analyse 133-134* 251–270.

[Vernadat, 1996] Vernadat, F. 1996. *Enterprise modeling and integration: principles and applications*. Chapman & Hall.

[W3C, 2002] W3C. 2002. XML. World-Wide Web Consortium XML. http://www.w3.org/.

[Wagner, 2001] Wagner, G. 2001. Agent-Oriented Analysis and Design of Organizational Information Systems. *in J. Barzdins and A. Caplinskas (Eds.), Databases and Information Systems. Kluwer Academic Publishers*. 111–124.

[Wang *et al.*, 2001] Wang, L.; Brennan, R.; Balasubramanian, S.; y Norrie, D. 2001. Realizing holonic control with function blocks. *Integrated Computer-Aided Engineering* 8(1).

[Warnecke, 1992] Warnecke, H. 1992. Die Fraktale Fabrik, Revolution Unternehmenskultur. *Springer Verlag*.

[Wood, 2000] Wood, M. F. 2000. Multiagent Systems Engineering: A Methodology for Analysis and Design of Multiagent Systems. Master's thesis, Air Force Institute of Technology.

[Wooldridge *et al.*, 2000] Wooldridge, M.; Jennings, N. R.; y Kinny, D. 2000. The Gaia Methodology for Agent-Oriented Analisys and Design. *Journal of Autonomous Agents and Multi-Agent Systems* 15.

[Wooldridge y Jennings, 1995] Wooldridge, M., y Jennings, N. R. 1995. Intelligent Agents - Theories, Architectures, and Languages. *Lecture Notes in Artificia Intelligence, Springer-Verlag. ISBN 3-540-58855-8* 890.

[Wooldridge, 1997] Wooldridge, M. 1997. Agent-Based Software Enginnering. In *IEE Proceedings of Software Engineering*, volume 14, 26–37.

[Yu, 1996] Yu, E. 1996. *Modelling Strategic Relationships for Process Reengineering.* Tesis Doctoral, Department of Computer Science. University of Toronto.

[Zambonelli *et al.*, 2003] Zambonelli, F.; Jennings, N. R.; y Wooldridge, M. 2003. Developing Multiagent Systems: the Gaia Methodology. *ACM Transactions on Software Engineering and Methodology (TOSEM).ISSN:1049-331X* 12(3):317–370.

[Zhang *et al.*, 2000] Zhang, H.; Balasubramanian, S.; Brennan, R.; y Norrie, D. 2000. Design and implementation of a real-time holonic control system. *Information Science. Special Issue on Computational Intelligence for Manufacturing* 127(1-2):23–44.

[Zhang y Norrie, 1999a] Zhang, H., y Norrie, D. 1999a. Autonomous Control for Open Manufacturing Systems. In *Proceedings of IPMM99.*

[Zhang y Norrie, 1999b] Zhang, H., y Norrie, D. 1999b. Holonic Control at the Production and Controler Levels. In *Proceedings of IMS99.*

[Zweben y Fox, 1994] Zweben, M., y Fox, M. 1994. *Intelligent Scheduling.* Morgan Kaufman.