Proyecto Castelldefels: Definición formal de los perfiles de usuario del Campus y ejecución de los mismos en una herramienta de simulación.

Javier Sánchez Capellán
Ingeniería Técnica Informática de Sistemas
Consultor: Pau Fonseca Casas. Ph. D.
16 de Noviembre de 2011

1 Resumen

El presente Trabajo Fin de Carrera, en adelante TFC, se encuentra definido dentro del área de Simulación de redes y sistemas informáticos de la Universidad Oberta de Cataluña.

Una de las líneas de investigación en este área es el Proyecto Castelldefels. Este proyecto tiene como objetivo, crear un modelo informático que simule el Campus Virtual de la UOC. Dentro de este objetivo general se encuadra este TFC, que partiendo de la base de trabajos y proyectos anteriores completara la definición formal de los perfiles de usuario del Campus Virtual y la ejecución de los mismos en una herramienta de simulación.

Por una parte se ampliara el número de perfiles identificados y se asociara cada perfil a las diferentes funcionalidades del Campus. Dichas funcionalidades serán descritas por primera vez y seran ponderadas en función del impacto de las mismas en el rendimiento del Campus Virtual.

Por otra parte y utilizando el conocimiento anterior, se ampliaran los escenarios de tipo "what-if" ya definidos, mediante un Sistema Multi-agente, siendo SDL el lenguaje utilizado para la definicion formal y el plug-in SanDriLa para la herramienta Microsoft Visio, la herramienta que generara el modelo en el lenguaje SDL.

Para la ejecución del modelo generado, utilizaremos la herramienta SDLPS. SDLPS es un simulador distribuido que permite la definición de los modelos mediante SDL. Esta definición es completa y representa el comportamiento del modelo y la estructura, lo que permite la simulación sin la necesidad de implementar el modelo, simplificando la validación y verificación.

Este TFC finalizara con las conclusiones extraidas tras su elaboración y las futuras lineas de trabajo que se desprenden de los resultados obtenidos.

Palabras clave: TFC, simulación de redes, simulación social, Castelldefels, formalización, perfiles de usuario, SDL, SDLPS.

Nombre de area: Simulación de redes y Sistemas Informaticos

Contenidos

Índice de ilustraciones

Índice de tablas

2 Introducción

2.1 Justificación

La justificación del presente TFC se centra, en la necesidad de completar los esfuerzos ya realizados en la simulación de diferentes diseños de la arquitectura de comunicaciones del Campus.

Para conseguirlo, es necesario realizar simulaciones del comportamiento del Campus frente a la interacción de los usuarios del mismo. Esta será la linea a seguir por este TFC, la definición formal de los perfiles de los usuarios del Campus y su ejecución en una herramienta de simulación.

Mientras más real sea la simulación del comportamiento social, más real será el modelo y mejor será la simulación a nivel de arquitectura de comunicaciones. Esto posibilita plantear escenarios cercanos a la realidad y de esta forma detectar el comportamiento de todos los elementos de la arquitectura, anticipándose a las necesidades y problemas futuros.

2.2 Objetivos

De manera general tenemos los siguientes objetivos:

- Identificar y definir el comportamiento de todos los usuarios del Campus Virtual.
- Modelar este comportamiento de los usuarios mediante SDL.
- Utilizar herramientas informáticas avanzadas de creación y ejecución de modelos SDL.

2.3 Enfoque y metodología

En el presente TFC seguiremos la metodología del Modelo en cascada. Esta metodología es un proceso secuencial en el que los pasos son vistos hacia abajo (como en una cascada de agua) a través de las fases de análisis de requisitos, diseño, modelización y entrega. El TFC por tanto estará dividido en fases secuenciales, con cierta superposición y splashback aceptable entre fases.

Dentro de la fase de modelización, utilizaremos la metodología iterativa para la mejora constante de los productos obtenidos.

2.4 Planificación

En este apartado se muestra el cumplimiento de la planificación propuesta durante la primera fase y reflejada en el Plan de Trabajo del presente TFC.

2.4.1 Etapas del proyecto

2.4.1.1 Plan de Trabajo

El objetivo de la primera fase del TFC, era el de delimitar el alcance del proyecto y definir la metodología ha seguir durante el desarrollo del mismo.

2.4.1.2 Análisis de los Requisitos

La fase de análisis, se centró en estudiar la documentación disponible sobre la tecnología del TFC y las aproximaciones anteriores al modelo. Identificando usuarios y funcionalidades

2.4.1.3 Diseño

A lo largo de esta fase se realizó el diseño UML y el diseño de los bloques, agentes, eventos, canales y procedimientos del modelo. En paralelo al diseño, se comenzó el aprendizaje de las herramientas donde se construiría el mismo.

2.4.1.4 Modelización del comportamiento

En esta fase se realizaron 8 versiones del modelo, hasta conseguir el definitivo. Esta ultima versión se etiquetó como 0200. Durante esta fase se encontraron diversos problemas que se fueron solucionando en las diferentes versiones.

En particular en la versión 0100, que partía del modelo del anterior TFC, aparecieron problemas en la carga y compilación en la herramienta SDLPS. Estos problemas se debían a errores en la definición de los procedimientos y variables del agente definido, la solución consistió en la redefinición de todo el modelo.

En la versión 0106 apareció el problema del máximo número de procedimientos que permitía cargar la macro para Visio, la solución consistió en la modificación de la misma aumentando el array definido para dichos procedimientos hasta 50. A continuación se presenta el código anterior y el modificado en la Macro:

Procedimiento: `Public Sub ExpXML()`

Código Anterior: `Dim ArrayProcedures(10) As Procedure`
Código Modificado: `Dim ArrayProcedures(50) As Procedure`

Función: `Private Function ClearProcedures(ArrayProcedures() As Procedure)`

Código Anterior: `i = 0 To 9`
Código Modificado: `i = 0 To 49`

Función: `Private Function ProcBuscar(ArrayProcedures() As Procedure, Nom As String) As Procedure`

Código Anterior: `j = 0 To 9`
Código Modificado: `j = 0 To 49`

Función: `Private Function ProcExisteix(ArrayProcedures() As Procedure, Nom As String) As Boolean`

Código Anterior: `i = 0 To 10`
Código Modificado: `i = 0 To 50`

2.4.1.5 Finalización del TFC

Durante la última fase se completo la documentación del TFC, incluyendo la generación de la presentación.

2.4.2 Diagrama de Gantt

	Nombre	Duración	Inicio	Terminado
1	Inicio del TFC	0 days	3/10/11 8:00	3/10/11 8:00
2	**Plan de Trabajo**	7 days	**3/10/11 8:00**	**12/10/11 8:00**
3	Eleccion del proyecto	3 days	3/10/11 8:00	5/10/11 17:00
4	Definicion del alcance del proyecto	3 days	6/10/11 8:00	10/10/11 17:00
5	Elaboracion del plan de proyecto	1 day	11/10/11 8:00	11/10/11 17:00
6	Entrega de la PAC1	0 days	12/10/11 8:00	12/10/11 8:00
7	**Analisis de los Requisitos**	15 days	**12/10/11 8:00**	**1/11/11 17:00**
8	Estudio de PFCs/TFCs anteriores	3 days	12/10/11 8:00	14/10/11 17:00
9	Identificacion de los perfiles de usuarios	3 days	17/10/11 8:00	19/10/11 17:00
10	Estudio del escenario inicial	2 days	20/10/11 8:00	21/10/11 17:00
11	Valoración de las funcionalidades	3 days	24/10/11 8:00	26/10/11 17:00
12	Justificación de la importancia de las funcionalidades	2 days	27/10/11 8:00	28/10/11 17:00
13	Estudio de las relaciones de integración	2 days	31/10/11 8:00	1/11/11 17:00
14	**Diseño**	11 days	**2/11/11 8:00**	**17/11/11 8:00**
15	Elaboracion del diagrama UML del sistema	5 days	2/11/11 8:00	8/11/11 17:00
16	Definición de las funcionalidades críticas	1 day	9/11/11 8:00	9/11/11 17:00
17	Tablas de relaciones	1 day	10/11/11 8:00	10/11/11 17:00
18	Ciclos comunes del plan de simulacion	1 day	11/11/11 8:00	11/11/11 17:00
19	Definición de las caracterisitcas de la simulación	1 day	14/11/11 8:00	14/11/11 17:00
20	Redaccion del documento	2 days	15/11/11 8:00	16/11/11 17:00
21	Entrega de PAC2	0 days	17/11/11 8:00	17/11/11 8:00
22	**Modelizacion del comportamiento**	26 days	**17/11/11 8:00**	**23/12/11 8:00**
23	Definicion formal de los perfiles	12 days	17/11/11 8:00	2/12/11 17:00
24	Refinamiento del los perfiles	5 days	5/12/11 8:00	9/12/11 17:00
25	Ejecucion en SDLPS	4 days	12/12/11 8:00	15/12/11 17:00
26	Redaccion del documento	5 days	16/12/11 8:00	22/12/11 17:00
27	Entrega de PAC3	0 days	23/12/11 8:00	23/12/11 8:00
28	**Finalizacion del TFC**	18 days	**23/12/11 8:00**	**18/01/12 8:00**
29	Elaboracion de las conclusiones	9 days	23/12/11 8:00	4/01/12 17:00
30	Descripcion de los productos obtenidos	7 days	5/01/12 8:00	13/01/12 17:00
31	Elaboracion de la presentacion del TFC	2 days	16/01/12 8:00	17/01/12 17:00
32	Entrega de la memoria	0 days	18/01/12 8:00	18/01/12 8:00

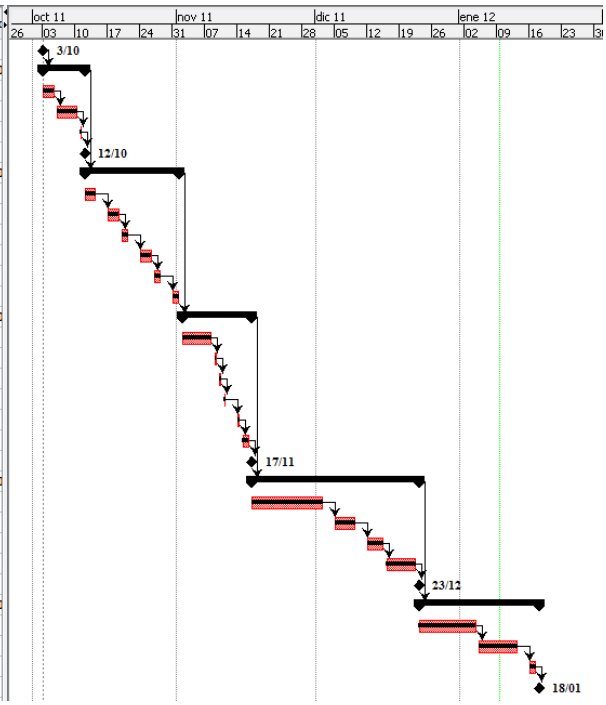


Ilustración 1: Diagrama de Gantt

2.5 Productos obtenidos

Los productos obtenidos en el presente TFC son los ficheros que conforman el modelo de simulación, tanto los diagramas (en formato Visio 2010) como el fichero con la sintaxis SDL, el código C y la DLL del modelo. Todos los productos se entregan con esta memoria.

2.6 Descripción del resto de capítulos

En el capitulo 3 se realiza una breve introducción a la problemática de la simulación de modelos sociales.

En el capitulo 4 se presentan los antecedentes del presente TFC dentro del Área.

En el capitulo 5 se define el ámbito del modelo que se desarrolla, mostrando la complejidad del sistema.

En el capitulo 6 se presenta el lenguaje en el que se desarrolla el TFC, el lenguaje SDL.

En el capitulo 7 se detallan los pasos para la construcción de un modelo SDL y su ejecución en la herramienta SDLPS.

En el capitulo 8 se realiza el estudio del sistema que se ha modelado, presentando todas las funcionalidades del mismo.

En el capitulo 9 se presenta la hipótesis del modelo, donde se delimita el mismo.

En el capitulo 10 se muestran todos los diagramas del modelo y se explican brevemente cada uno de los mismos.

3 La simulación de modelos sociales

Podemos ver la simulación como la experimentación con un modelo basado en la realidad. Los objetivos se pueden centrar en:

- Describir el comportamiento del modelo ante determinados inputs.
- Construir y confirmar o rechazar hipótesis a partir de este comportamiento.
- Predecir sucesos en base a los resultados obtenidos.

La simulación de un modelo social no resulta fácil de acotar, ya que el mundo que representan es amplio. El comportamiento humano está influido por numerosos factores y las sociedades forman redes de acontecimientos que pueden resultar interminables.

Para cualquier modelo de simulación, es imprescindible formular una serie de hipótesis simplificadoras que delimiten el sistema y permitan una abstracción factible del mundo real.

El tipo de simulación que ocupa este TFC debe verse como un sistema multiagente, donde la gran parte son inteligentes: reciben estímulos de otros agentes, actúan en consecuencia y adquieren conocimiento [FON1]

4 Antecedentes

El presente TFC se alimenta de anteriores proyectos y trabajos fin de carrera, donde se realizó una definición inicial de los perfiles de usuarios del Campus.

Los perfiles documentados fueron los siguientes:

Perfil	
Nombre	Descripción
Estudiante	Alumnos del Campus, opcionalmente se pueden dividir en diferentes perfiles en función de la carga de uso del Campus
Profesor	Tutores, Profesores y Consultores
Gestor	Personal de la UOC
Visitante	Acceso público al Campus

Tabla 1: Perfiles definidos inicialmente [MIQM]

Partiendo de esta base, en el presente TFC se ha completado dicho estudio inicial hasta definir todos los perfiles existentes en el Campus.

Perfil	
Nombre	Descripción
Estudiante	Persona que es alumna del Campus, se podrían dividir en diferentes perfiles en función de la carga de uso del Campus
Alumni	Persona Graduada de la UOC
Profesor	Persona responsable de las titulaciones de grado y posgrado de la UOC y que vela por la calidad de los cursos y asignaturas y es activa en la investigación.
Consultor	Persona que presenta, guía, planifica, estimula, orienta, dinamiza, cohesiona y evalúa los procesos de aprendizaje del estudiante a través de una actitud proactiva
Tutor	Persona que, de forma individualizada, acoge, acompaña y orienta al estudiante durante toda su vida académica
Administrador	Persona responsable de las tareas de mantenimiento del Campus
Gestor	Persona responsables de las tareas administrativas del Campus
Visitante	Persona que accede a la zona pública del Campus

Tabla 2: Perfiles definidos en el presente TFC

5 Ámbito de estudio

El ámbito del presente TFC se centra en La Universitat Oberta de Catalunya (UOC). Esta universidad con sede en Barcelona imparte toda su docencia a través de Internet.

Para prestar dicho servicio cuenta con un portal (www.uoc.edu) al que acceden todos los usuarios. Dicho portal cuenta con todos los recursos de una universidad tradicional.

Dado que la UOC tiene unos 40.000 usuarios entre los pertenecientes a la Comunidad Académica y al Equipo de Gestión, nos encontramos con un sistema informático muy complejo (el Campus Virtual) que debe dar respuesta a requerimientos muy estrictos en lo referente a minimizar el tiempo de respuesta a una petición, minimizar el tiempo de downtime del servicio, maximizar el número de usuarios concurrentes, y otros.

6 Implementación en SDL (Specification and Description Language)

La implementación en SDL de un modelo está formada de los siguientes elementos

- Estructura: la notación dedicada a la estructura permite subdividir el problema en instancias más simples, utilizando el enfoque de arriba a abajo clásico de la ingeniería. Los elementos que conforman la estructura del sistema en orden de generalidad y nivel de abstracción son:

    - Sistema: el sistema es el ambiente donde se desarrolla la existencia del producto y el producto mismo.
    - Bloques.
    - Procesos: los procesos son el equivalente de máquinas de estado finito extendidas, capaces de controlar el disparo de las transiciones con guardas
    - Servicios: un servicio es similar a un proceso, pero no posee un espacio privado de variables.

- Comunicación:

    - Señales
    - Canales de comunicación

- Comportamiento (procesos).

- Datos (tipos de datos abstractos).

- Relaciones de herencia (especialización)

7 Construcción del modelo SDL

7.1 El software: Visio 2010 + Sandrila Plugin y SDLPS

SDLPS es un simulador que permite ejecutar modelos diseñados en SDL. Para ello, utiliza una representación en XML del modelo diseñado gráficamente en SDL.

Para construir la parte SDL del modelo seguimos estos dos pasos:

1. Diseño de los diagramas SDL con Microsoft Visio 2010. Utilización del plug-in Sandrila, que permite la construcción de los diagramas de forma correcta.

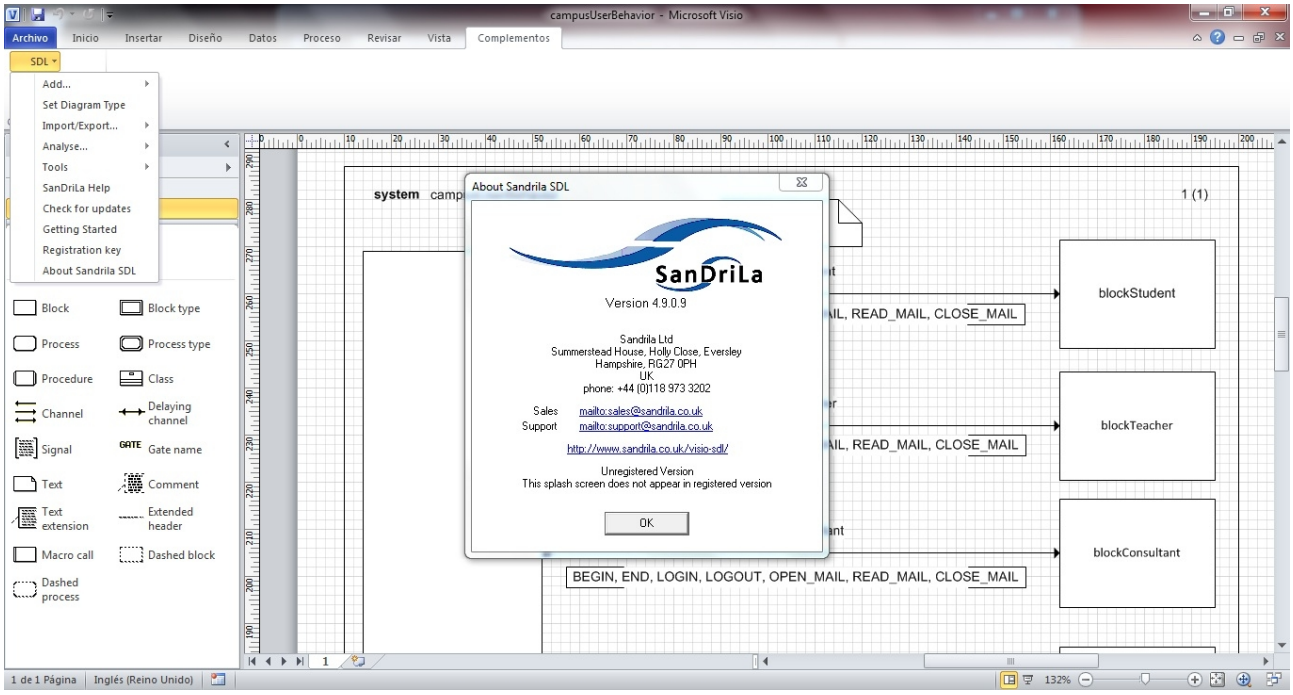


Ilustración 2: plug-in Sandrila

2. Transformación en XML. Junto con Sandrila, se utiliza una macro diseñada para Visio que transforma el sistema de diagramas SDL en una representación XML. El fichero generado tiene extensión .sdlps y se utiliza para cargar el modelo en la herramienta.

   En la figura siguiente se muestra la pantalla principal del SDLPS.

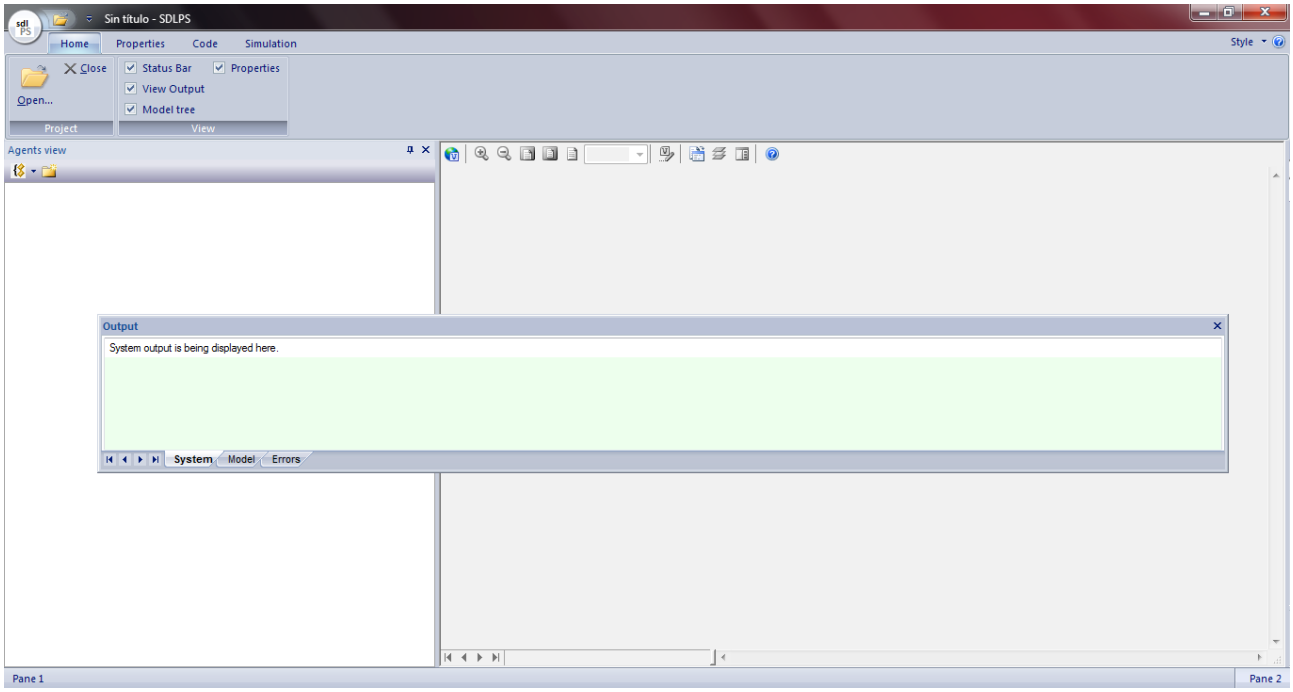


Ilustración 3: Pantalla inicial de la herramienta SDLPS

La herramienta dispone de una vista principal donde se muestra el diagrama principal del Sistema.

El panel izquierdo permite ver el árbol de agentes que forman el Sistema.

En primer termino podemos ver la pantalla de Output, que muestra los mensajes de error (o corrección) de la carga, compilación y enlace, así como de la ejecución del modelo.

La barra superior dispone de las operaciones necesarias para hacer funcionar el modelo. En el apartado siguiente se explicarán con más detalle, junto con los pasos a seguir para ejecutar el modelo.

7.2 Pasos de ejecución del Modelo

Los pasos a realizar para hacer funcionar el modelo empleando SDLP, quedan descritos en orden a continuación:

1. Carga del modelo (Home -> Open)

   Se debe seleccionar el archivo .sdlps del modelo. Si es sintácticamente correcto, el modelo quedará cargado en el programa.

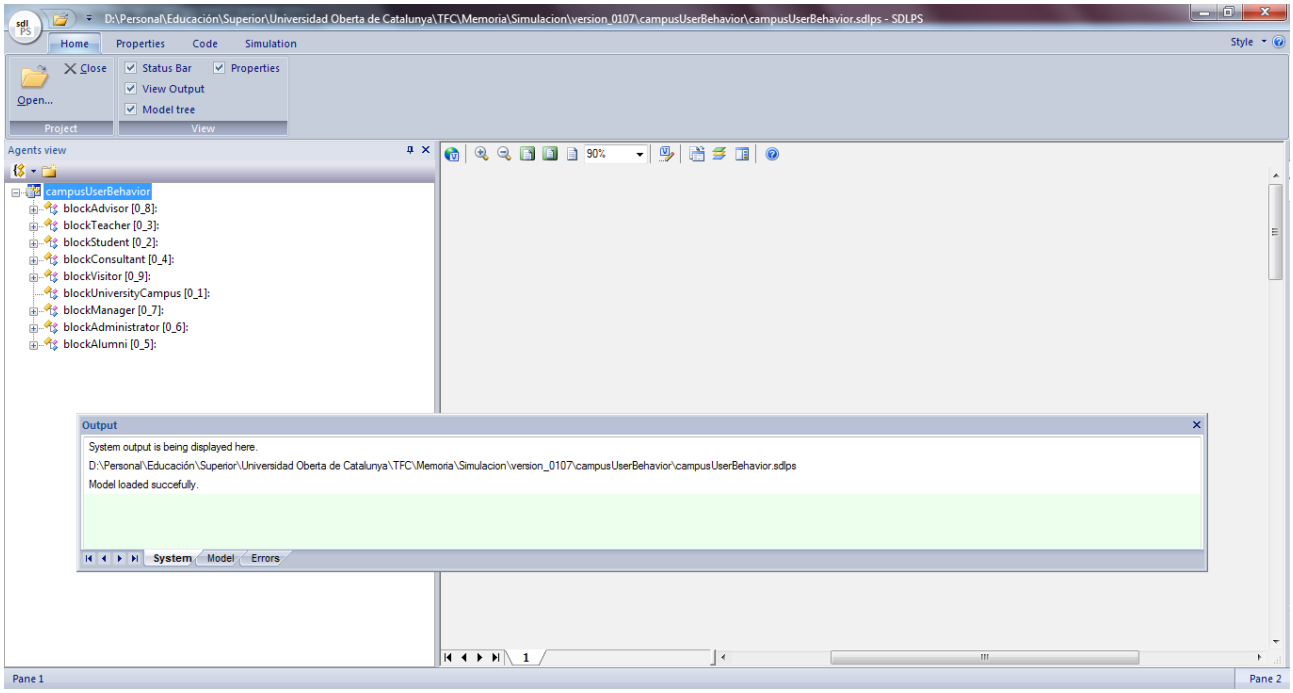


Ilustración 4: Carga del modelo SDL en la herramienta SDPLS

2. Creación, compilación y enlace del código (Code-> Create, Compile and Link DLL)

   Genera el código C correspondiente al fichero .sdlps, y lo compila junto con el código externo y lo enlaza en una única librería .dll.

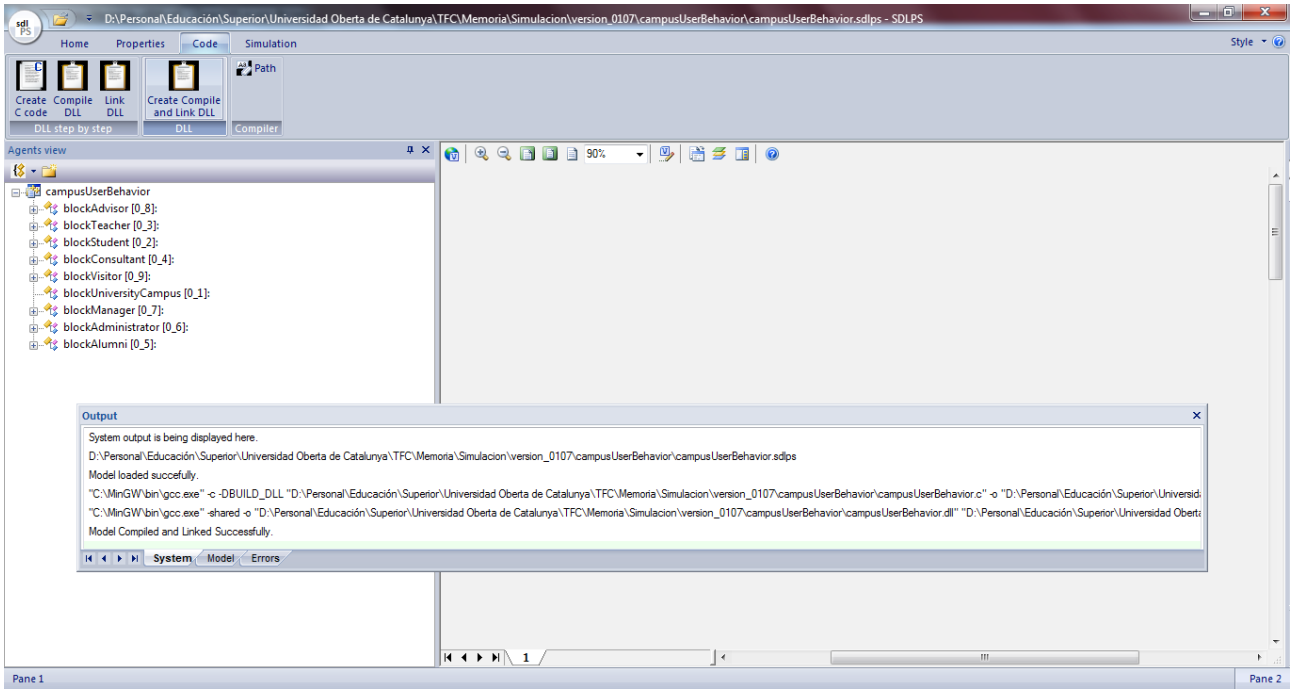


Ilustración 5: Creación, compilación y enlace del código en la herramienta SDPLS

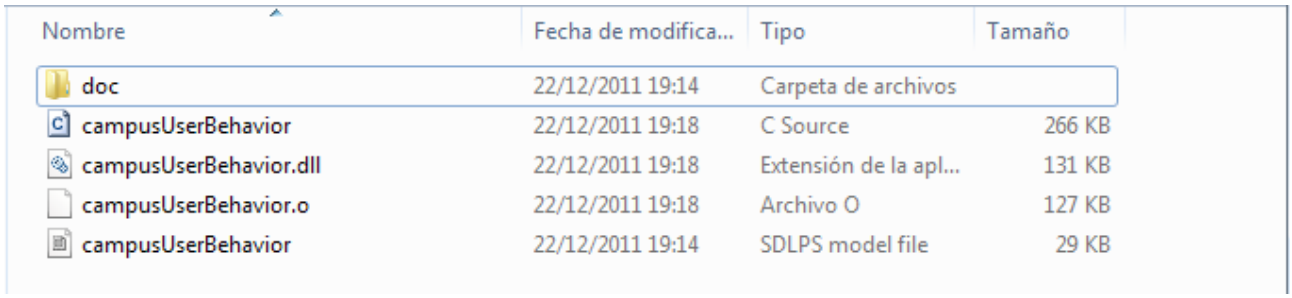


Ilustración 6: Ficheros generados tras la creación, compilación y enlace del código en la herramienta SDPLS

3. Establecimiento del tiempo de ejecución (Simulation-> Simulation time)

   Establece el tiempo de simulación. Se puede determinar la duración o mantener el valor por defecto, que producirá una simulación infinita. En este ultimo caso la simulación se deberá detener con el botón Stop.

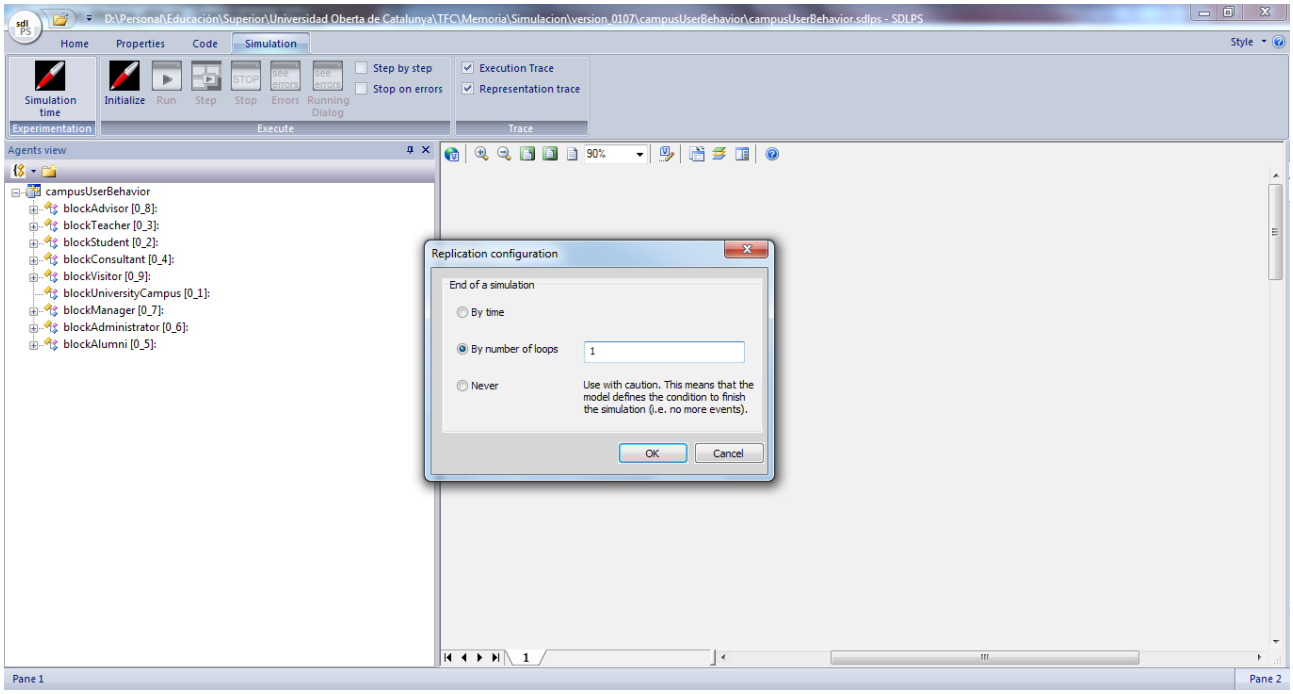


Ilustración 7: Establecimiento del tiempo de ejecución en la herramienta SDPLS

4. (Simulation -> initialize)

   Inicializa el modelo. Consiste en la ejecución de todos los procesos definidos en el SDL desde el estado "Start" hasta el cambio a otro estado definido en el proceso.

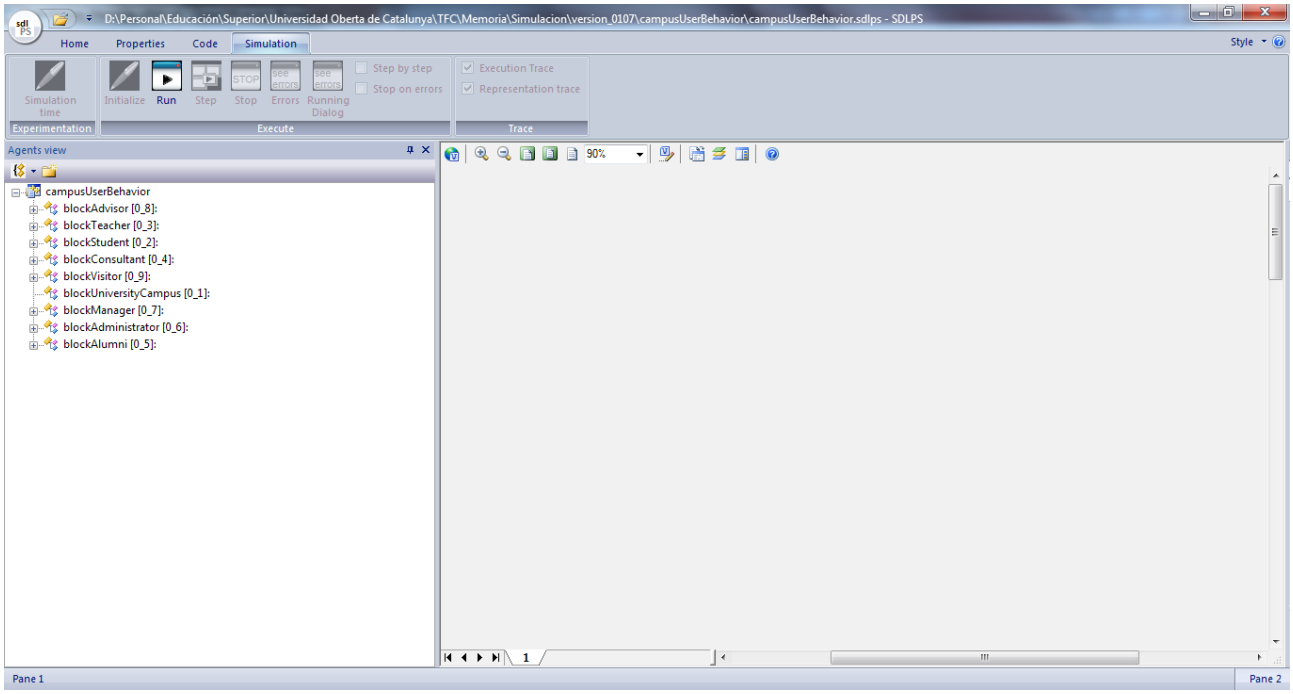


Ilustración 8: Inicialización de la simulación en la herramienta SDPLS

5. Ejecución de la simulación

   Podemos seguir dos procedimientos:

   - Ejecución continua: haciendo clic en el botón Run se ejecutará el modelo hasta el final.
   - Paso a paso: previamente se habrá de marcar la casilla "Step by step". El botón Step ejecutará un paso de la simulación cada vez que sea pulsado.

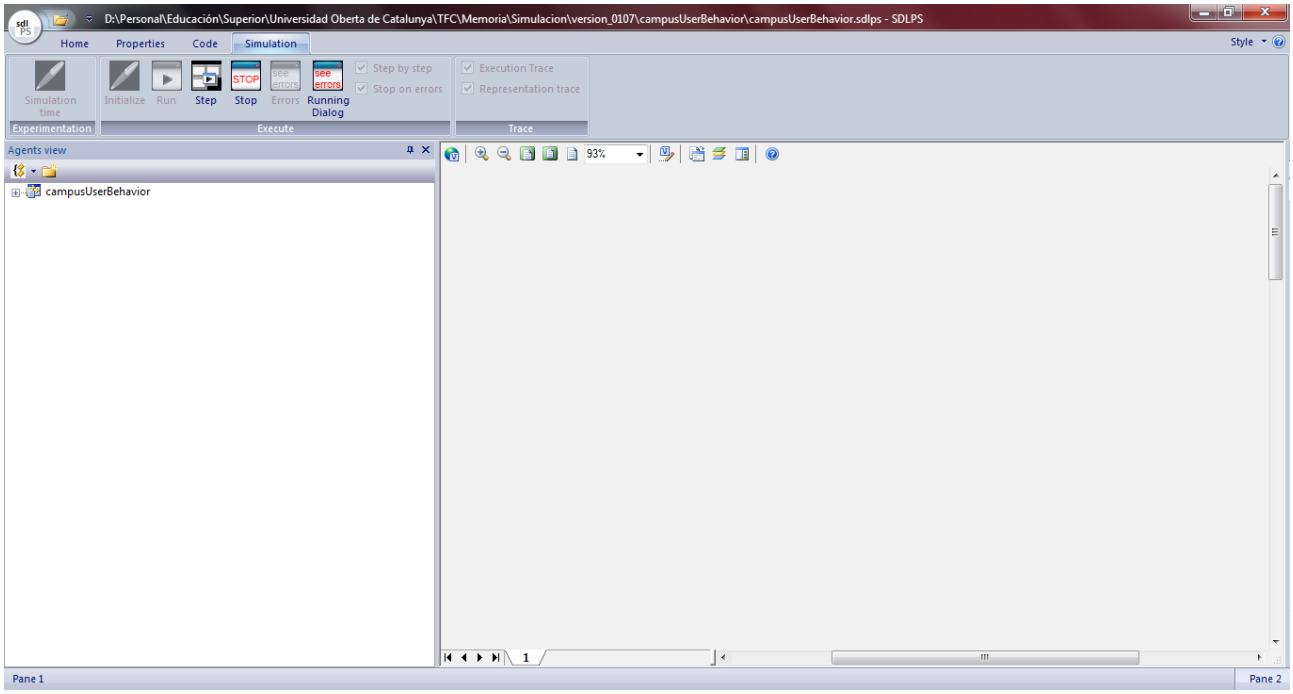
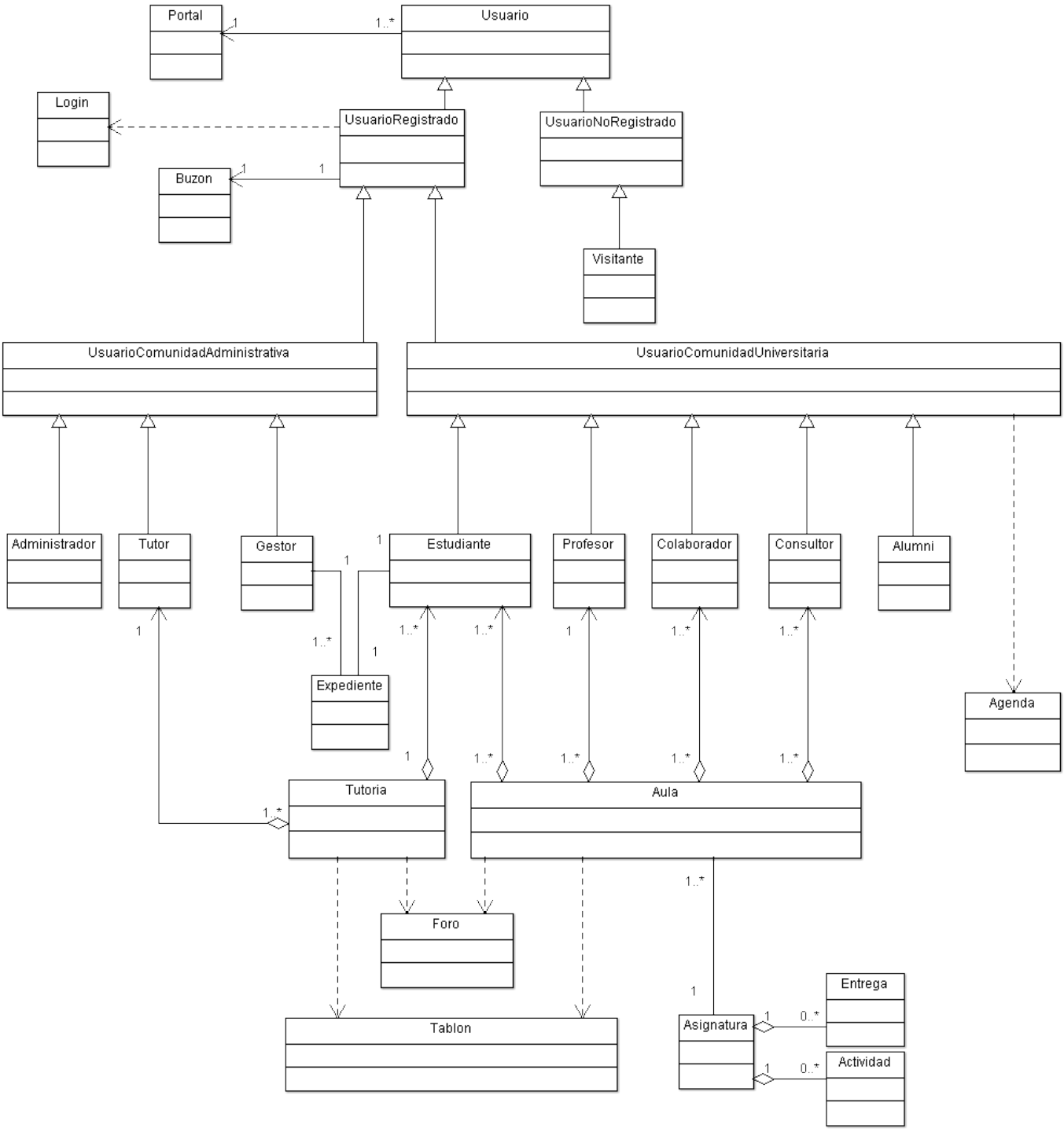


Ilustración 9: Ejecución de la simulación en la herramienta SDPLS

8 Arquitectura del Modelo

8.1 Diagrama del Sistema



8.2 Funcionalidades

Las funcionalidades del Sistema que podemos extraer del Diagrama se muestran en la siguiente tabla. A cada una de ellas se le ha asignado un valor entero para la variable EventCode que luego será utilizada en los diagramas del modelo.

Funcionalidad			
EventCode	Event	Descripción	Entidad
1	BEGIN	Inicio de la navegación	Portal
2	END	Final de la navegación	Portal
3	OPEN_PAGE	Abrir página del portal	Portal
4	CLOSE_PAGE	Cerrar página del portal	Portal
5	LOGIN	Hacer login en el portal	Login
6	LOGOUT	Salir del portal	Login
7	OPEN_MAIL	Abrir el buzón de correo	Buzón
8	READ_MAIL	Leer un mensaje de correo del buzón	Buzón
9	CLOSE_MAIL	Cerrar correo	Buzón
10	SEND_MAIL	Enviar correo	Buzón
11	EDIT_MAIL	Editar correo	Buzón
12	OPEN_CLASS	Abrir aula	Aula
13	CLOSE_CLASS	Cerrar aula	Aula
14	OPEN_SUBJECT	Abrir actividad	Actividad
15	CLOSE_SUBJECT	Cerrar actividad	Actividad
16	OPEN_DELIVERIES	Abrir entregas	Entrega
17	MAKE_DELIVERY	Hacer entrega	Entrega

Funcionalidad			
EventCode	Event	Descripción	Entidad
18	VIEW_DELIVERY	Ver entrega	Entrega
19	OPEN_FORUM	Abrir foro	Foro
20	READ_POST	Leer post del foro	Foro
20	READ_POST	Dar por leído un post del foro	Foro
21	EDIT_POST	Editar post del foro	Foro
22	SEND_POST	Enviar post al foro	Foro
23	READ_NOTES	Leer notas	Foro
24	READ_ALL_POST	Leer todos los post del foro	Foro
25	ORDER_POSTS	Ordenar todos los post del foro	Foro
26	EXPORT_POSTS	Exportar el foro	Foro
27	VIEW_POST_HISTORY	Ver historial del foro	Foro
28	OPEN_BOARD	Abrir tablón	Tablón
29	READ_MESSAGE	Leer mensaje del tablón	Tablón
29	READ_MESSAGE	Dar por leído un mensaje del tablón	Tablón
30	EDIT_MESSAGE	Editar un mensaje del tablón	Tablón
31	SEND_MESSAGE	Responder o reenviar un mensaje del tablón	Tablón
32	READ_ALL_MESSAGES	Leer todos los mensajes del tablón	Tablón
33	ORDER_MESSAGES	Ordenar los mensajes del tablón	Tablón
34	EXPORT_BOARD	Exportar el tablón	Tablón
35	VIEW_MESSAGE_HISTORY	Ver historial de mensajes del tablón	Tablón
36	OPEN_CALENDAR	Abrir agenda	Agenda

Funcionalidad			
EventCode	Event	Descripción	Entidad
37	READ_CALENDAR_EVENT	Consultar una cita de la agenda	Agenda
38	CREATE_CALENDAR_EVENT	Crear una cita en la agenda propia	Agenda
39	CREATE_SHARED_CALENDAR_EVENT	Crear una cita en la agenda de varios usuarios	Agenda
40	READ_CALENDAR_MONTH	Consultar citas en la agenda para un mes	Agenda
41	OPEN_TUTORSHIP	Abrir tutoría	Tutoría
42	CLOSE_TUTORSHIP	Cerrar tutoría	Tutoría
43	OPEN_EXPEDIENT	Abrir expediente	Expediente
44	CLOSE_EXPEDIENT	Cerrar expediente	Expediente
45	VIEW_EXPEDIENT	Ver expediente	Expediente
46	EDIT_EXPEDIENT	Modificar expediente	Expediente

Tabla 3: Funcionalidades del sistema

8.3 Funcionalidad por Perfil

En el presente apartado realizaremos uno de los puntos fundamentales del presente TFC, la definición de las relaciones entre las diferentes funcionalidades y los perfiles identificados para nuestro modelo.

Funcionalidad			Usuario							
			UsuarioRegistrado							UsuarioNoRegistrado
EventCode	Event	Entidad	UsuarioComunidadUniversitaria				UsuarioComunidadUniversitaria			Visitante
			Estudiante	Profesor	Consultor	Alumni	Administrador	Gestor	Tutor	
1	BEGIN	Portal	x	x	x	x	x	x	x	x
2	END	Portal	x	x	x	x	x	x	x	x
3	OPEN_PAGE	Portal	x	x	x	x	x	x	x	x
4	CLOSE_PAGE	Portal	x	x	x	x	x	x	x	x
5	LOGIN	Login	x	x	x	x	x	x	x	
6	LOGOUT	Login	x	x	x	x	x	x	x	
7	OPEN_MAIL	Buzón	x	x	x	x	x	x	x	
8	READ_MAIL	Buzón	x	x	x	x	x	x	x	
9	CLOSE_MAIL	Buzón	x	x	x	x	x	x	x	
10	SEND_MAIL	Buzón	x	x	x	x	x	x	x	
11	EDIT_MAIL	Buzón	x	x	x	x	x	x	x	

Funcionalidad			Usuario							
			UsuarioRegistrado							UsuarioNoRegistrado
EventCode	Event	Entidad	UsuarioComunidadUniversitaria				UsuarioComunidadUniversitaria			Visitante
			Estudiante	Profesor	Consultor	Alumni	Administrador	Gestor	Tutor	
12	OPEN_CLASS	Aula	x	x	x		x			
13	CLOSE_CLASS	Aula	x	x	x		x			
14	OPEN_SUBJECT	Actividad	x	x	x		x			
15	CLOSE_SUBJECT	Actividad	x	x	x		x			
16	OPEN_DELIVERIES	Entrega	x	x	x		x			
17	MAKE_DELIVERY	Entrega	x	x	x		x			
18	VIEW_DELIVERY	Entrega	x	x	x		x			
19	OPEN_FORUM	Foro	x	x	x		x		x	
20	READ_POST	Foro	x	x	x		x		x	
20	READ_POST	Foro	x	x	x		x		x	
21	EDIT_POST	Foro					x			
22	SEND_POST	Foro	x	x	x		x		x	
23	READ_NOTES	Foro	x	x	x		x		x	
24	READ_ALL_POST	Foro	x	x	x		x		x	

Funcionalidad			Usuario							
			UsuarioRegistrado							UsuarioNoRegistrado
EventCode	Event	Entidad	UsuarioComunidadUniversitaria				UsuarioComunidadUniversitaria			Visitante
			Estudiante	Profesor	Consultor	Alumni	Administrador	Gestor	Tutor	
25	ORDER_POSTS	Foro	x	x	x		x		x	
26	EXPORT_POSTS	Foro	x	x	x		x		x	
27	VIEW_POST_HISTORY	Foro	x	x	x		x		x	
28	OPEN_BOARD	Tablón	x	x	x		x		x	
29	READ_MESSAGE	Tablón	x	x	x		x		x	
29	READ_MESSAGE	Tablón	x	x	x		x		x	
30	EDIT_MESSAGE	Tablón	x	x	x		x		x	
31	SEND_MESSAGE	Tablón	x	x	x		x		x	
32	READ_ALL_MESSAGES	Tablón	x	x	x		x		x	
33	ORDER_MESSAGES	Tablón	x	x	x		x		x	
34	EXPORT_BOARD	Tablón	x	x	x		x		x	
35	VIEW_MESSAGE_HISTORY	Tablón	x	x	x		x		x	
36	OPEN_CALENDAR	Agenda	x	x	x	x	x	x	x	
37	READ_CALENDAR_EVENT	Agenda	x	x	x	x	x	x	x	

Funcionalidad			Usuario							
			UsuarioRegistrado							UsuarioNoRegistrado
EventCode	Event	Entidad	UsuarioComunidadUniversitaria				UsuarioComunidadUniversitaria			Visitante
			Estudiante	Profesor	Consultor	Alumni	Administrador	Gestor	Tutor	
38	CREATE_CALENDAR_EVENT	Agenda	x	x	x	x	x	x	x	
39	CREATE_SHARED_CALENDAR_EVENT	Agenda		x			x			
40	READ_CALENDAR_MONTH	Agenda	x	x	x	x	x	x	x	
41	OPEN_TUTORSHIP	Tutoría	x				x		x	
42	CLOSE_TUTORSHIP	Tutoría	x				x		x	
43	OPEN_EXPEDIENT	Expediente	x				x	x		
44	CLOSE_EXPEDIENT	Expediente	x				x	x		
45	VIEW_EXPEDIENT	Expediente	x				x	x		
46	EDIT_EXPEDIENT	Expediente					x	x		

Tabla 4: Funcionalidades por perfil

8.4 Valoración de las funcionalidades

Debido a la gran cantidad de funcionalidades identificadas, se hace necesario realizar una valoración de las mismas para ponderar su relevancia en el modelo. En función de esta valoración escogeremos las funcionalidades que se representaran finalmente en el mismo. La valoración se realizara en función del número de perfiles relacionados con cada funcionalidad:

Funcionalidad		Perfiles
EventCode	Event	
1	BEGIN	8
2	END	8
3	OPEN_PAGE	8
4	CLOSE_PAGE	8
5	LOGIN	7
6	LOGOUT	7
7	OPEN_MAIL	7
8	READ_MAIL	7
9	CLOSE_MAIL	7
10	SEND_MAIL	7
11	EDIT_MAIL	7
12	OPEN_CLASS	4
13	CLOSE_CLASS	4
14	OPEN_SUBJECT	4
15	CLOSE_SUBJECT	4
16	OPEN_DELIVERIES	4

Funcionalidad		Perfiles
EventCode	Event	
17	MAKE_DELIVERY	4
18	VIEW_DELIVERY	4
19	OPEN_FORUM	5
20	READ_POST	5
20	READ_POST	5
21	EDIT_POST	1
22	SEND_POST	5
23	READ_NOTES	5
24	READ_ALL_POST	5
25	ORDER_POSTS	5
26	EXPORT_POSTS	5
27	VIEW_POST_HISTORY	5
28	OPEN_BOARD	5
29	READ_MESSAGE	5
29	READ_MESSAGE	5
30	EDIT_MESSAGE	5
31	SEND_MESSAGE	5
32	READ_ALL_MESSAGES	5
33	ORDER_MESSAGES	5
34	EXPORT_BOARD	5
35	VIEW_MESSAGE_HISTORY	5

Funcionalidad		Perfiles
EventCode	Event	
36	OPEN_CALENDAR	7
37	READ_CALENDAR_EVENT	7
38	CREATE_CALENDAR_EVENT	7
39	CREATE_SHARED_CALENDAR_EVENT	2
40	READ_CALENDAR_MONTH	7
41	OPEN_TUTORSHIP	3
42	CLOSE_TUTORSHIP	3
43	OPEN_EXPEDIENT	3
44	CLOSE_EXPEDIENT	3
45	VIEW_EXPEDIENT	3
46	EDIT_EXPEDIENT	2

Tabla 5: Número de perfiles por funcionalidad

9 Hipótesis del modelo

Para poder evaluar y extraer conclusiones de cualquier modelo de simulación es necesario documentar las hipótesis del modelo.

Debido a la complejidad del sistema, se ha decido modelar solo las funcionalidades que utilizan mas de 4 perfiles y aquellas funcionalidades que por su impacto son relevantes en el rendimiento del Campus:

Funcionalidad		Utilización
EventCode	Event	
1	BEGIN	8
2	END	8
3	OPEN_PAGE	8
4	CLOSE_PAGE	8
5	LOGIN	7
6	LOGOUT	7
7	OPEN_MAIL	7
8	READ_MAIL	7
9	CLOSE_MAIL	7
10	SEND_MAIL	7
11	EDIT_MAIL	7
16	OPEN_DELIVERIES	4
17	MAKE_DELIVERY	4
19	OPEN_FORUM	5
20	READ_POST	5
22	SEND_POST	5

Funcionalidad		Utilización
EventCode	Event	
23	READ_NOTES	5
24	READ_ALL_POST	5
25	ORDER_POSTS	5
26	EXPORT_POSTS	5
27	VIEW_POST_HISTORY	5
28	OPEN_BOARD	5
29	READ_MESSAGE	5
30	EDIT_MESSAGE	5
31	SEND_MESSAGE	5
32	READ_ALL_MESSAGES	5
33	ORDER_MESSAGES	5
34	EXPORT_BOARD	5
35	VIEW_MESSAGE_HISTORY	5
36	OPEN_CALENDAR	7
37	READ_CALENDAR_EVENT	7
38	CREATE_CALENDAR_EVENT	7
40	READ_CALENDAR_MONTH	7

Tabla 6: Valoración de las funcionalidades por su utilización

Las funcionalidades escogidas por su impacto, son aquellas relacionadas con la Entrega de Actividades en el Campus. Se ha podido comprobar que en periodo de entregas el Campus ha tenido problemas de rendimiento.

Quedando por tanto los siguientes eventos por perfil

Funcionalidad		Perfiles
EventCode	Event	
1	BEGIN	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor Visitante
2	END	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor Visitante
3	OPEN_PAGE	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor Visitante
4	CLOSE_PAGE	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor Visitante
5	LOGIN	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
6	LOGOUT	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
7	OPEN_MAIL	Estudiante Profesor Consultor Alumni Administrador

Funcionalidad		Perfiles
EventCode	Event	
		Gestor Tutor
8	READ_MAIL	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
9	CLOSE_MAIL	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
10	SEND_MAIL	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
11	EDIT_MAIL	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
16	OPEN_DELIVERIES	Estudiante Profesor Consultor Administrador
17	MAKE_DELIVERY	Estudiante Profesor Consultor Administrador
19	OPEN_FORUM	Estudiante Profesor Consultor Administrador Tutor
20	READ_POST	Estudiante Profesor Consultor Administrador Tutor
22	SEND_POST	Estudiante Profesor Consultor Administrador

Funcionalidad		Perfiles
EventCode	Event	
		Tutor
23	READ_NOTES	Estudiante Profesor Consultor Administrador Tutor
24	READ_ALL_POST	Estudiante Profesor Consultor Administrador Tutor
25	ORDER_POSTS	Estudiante Profesor Consultor Administrador Tutor
26	EXPORT_POSTS	Estudiante Profesor Consultor Administrador Tutor
27	VIEW_POST_HISTORY	Estudiante Profesor Consultor Administrador Tutor
28	OPEN_BOARD	Estudiante Profesor Consultor Administrador Tutor
29	READ_MESSAGE	Estudiante Profesor Consultor Administrador Tutor
30	EDIT_MESSAGE	Estudiante Profesor Consultor Administrador Tutor
31	SEND_MESSAGE	Estudiante Profesor Consultor Administrador Tutor
32	READ_ALL_MESSAGES	Estudiante Profesor Consultor Administrador Tutor
33	ORDER_MESSAGES	Estudiante

Funcionalidad		Perfiles
EventCode	Event	
		Profesor Consultor Administrador Tutor
34	EXPORT_BOARD	Estudiante Profesor Consultor Administrador Tutor
35	VIEW_MESSAGE_HISTORY	Estudiante Profesor Consultor Administrador Tutor
36	OPEN_CALENDAR	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
37	READ_CALENDAR_EVENT	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
38	CREATE_CALENDAR_EVENT	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor
40	READ_CALENDAR_MONTH	Estudiante Profesor Consultor Alumni Administrador Gestor Tutor

Tabla 7: Eventos por perfil utilizados en el modelo

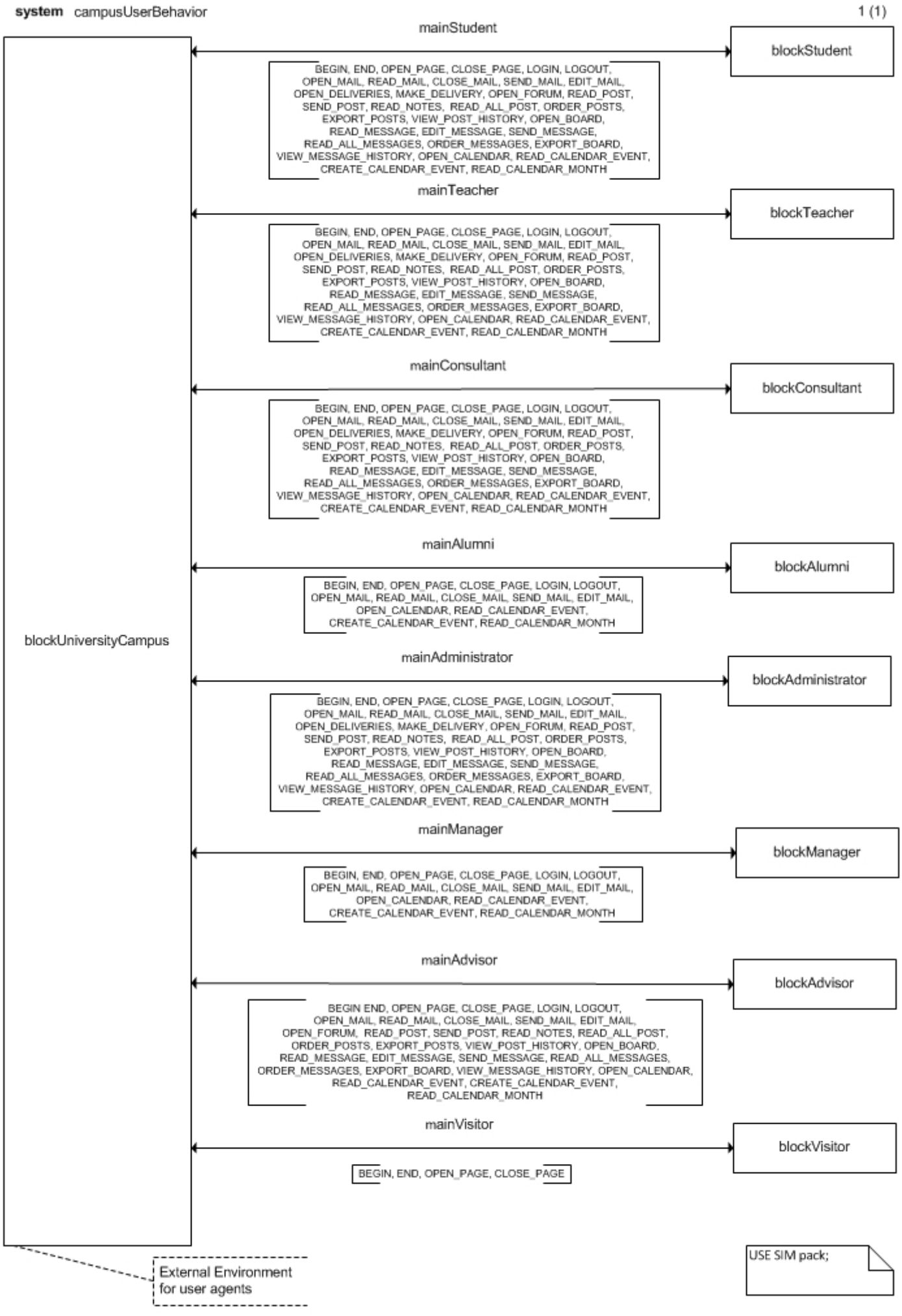
10 Modelo SDL propuesto

Consideraciones iniciales:

• Los diagramas SDL han sido construidos con Microsoft Visio 2010 + Plugin SANDRILA SDL [SAND]. Serán necesarios ambos para visualizarlos o editarlos.

10.1 Sistema

En este diagrama podemos observar la definición de los bloques de Usuario, los canales de comunicación y los diferentes eventos definidos para cada Usuario. Todos los nombres de ficheros, variables, bloques, procesos, procedimientos, canales de comunicación y eventos del modelo se han definido utilizando el idioma Ingles para la estandarización del mismo. El archivo de este diagrama es: campusUserBehavior.vsd.


system campusUserBehavior
blockUniversityCampus
mainStudent
blockStudent
BEGIN, END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_DELIVERIES, MAKE_DELIVERY, OPEN_FORUM, READ_POST, SEND_POST, READ_NOTES, READ_ALL_POST, ORDER_POSTS, EXPORT_POSTS, VIEW_POST_HISTORY, OPEN_BOARD, READ_MESSAGE, EDIT_MESSAGE, SEND_MESSAGE, READ_ALL_MESSAGES, ORDER_MESSAGES, EXPORT_BOARD, VIEW_MESSAGE_HISTORY, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainTeacher
blockTeacher
BEGIN, END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_DELIVERIES, MAKE_DELIVERY, OPEN_FORUM, READ_POST, SEND_POST, READ_NOTES, READ_ALL_POST, ORDER_POSTS, EXPORT_POSTS, VIEW_POST_HISTORY, OPEN_BOARD, READ_MESSAGE, EDIT_MESSAGE, SEND_MESSAGE, READ_ALL_MESSAGES, ORDER_MESSAGES, EXPORT_BOARD, VIEW_MESSAGE_HISTORY, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainConsultant
blockConsultant
BEGIN, END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_DELIVERIES, MAKE_DELIVERY, OPEN_FORUM, READ_POST, SEND_POST, READ_NOTES, READ_ALL_POST, ORDER_POSTS, EXPORT_POSTS, VIEW_POST_HISTORY, OPEN_BOARD, READ_MESSAGE, EDIT_MESSAGE, SEND_MESSAGE, READ_ALL_MESSAGES, ORDER_MESSAGES, EXPORT_BOARD, VIEW_MESSAGE_HISTORY, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainAlumni
blockAlumni
BEGIN, END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainAdministrator
blockAdministrator
BEGIN, END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_DELIVERIES, MAKE_DELIVERY, OPEN_FORUM, READ_POST, SEND_POST, READ_NOTES, READ_ALL_POST, ORDER_POSTS, EXPORT_POSTS, VIEW_POST_HISTORY, OPEN_BOARD, READ_MESSAGE, EDIT_MESSAGE, SEND_MESSAGE, READ_ALL_MESSAGES, ORDER_MESSAGES, EXPORT_BOARD, VIEW_MESSAGE_HISTORY, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainManager
blockManager
BEGIN, END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainAdvisor
blockAdvisor
BEGIN END, OPEN_PAGE, CLOSE_PAGE, LOGIN, LOGOUT, OPEN_MAIL, READ_MAIL, CLOSE_MAIL, SEND_MAIL, EDIT_MAIL, OPEN_FORUM, READ_POST, SEND_POST, READ_NOTES, READ_ALL_POST, ORDER_POSTS, EXPORT_POSTS, VIEW_POST_HISTORY, OPEN_BOARD, READ_MESSAGE, EDIT_MESSAGE, SEND_MESSAGE, READ_ALL_MESSAGES, ORDER_MESSAGES, EXPORT_BOARD, VIEW_MESSAGE_HISTORY, OPEN_CALENDAR, READ_CALENDAR_EVENT, CREATE_CALENDAR_EVENT, READ_CALENDAR_MONTH
mainVisitor
blockVisitor
BEGIN, END, OPEN_PAGE, CLOSE_PAGE
External Environment for user agents
USE SIM pack;

10.2 Bloques

10.2.1 Estudiante

En este diagrama podemos observar la definición del canal channelStudent y la llamada al agente processStudentUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockStudent.vsd.

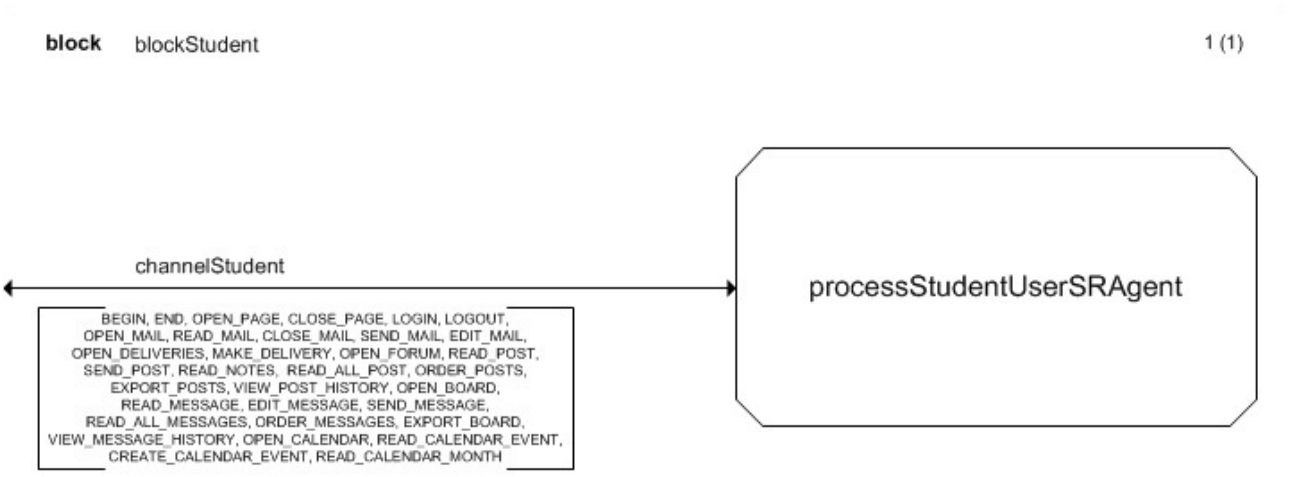


Ilustración 11: Diagrama del bloque Estudiante

10.2.2 Profesor

En este diagrama podemos observar la definición del canal channelTeacher y la llamada al agente processTeacherUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockTeacher.vsd.

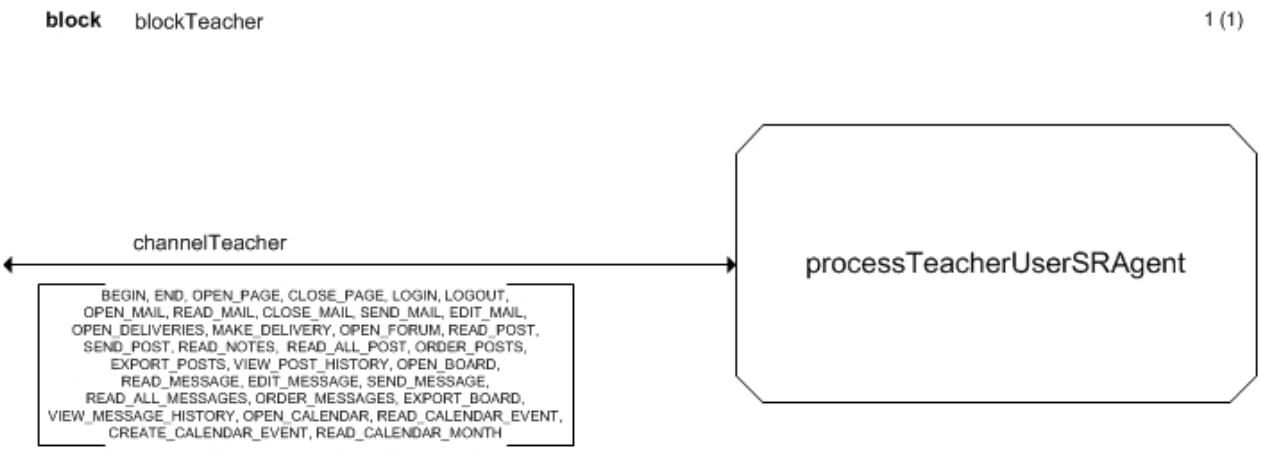


Ilustración 12: Diagrama del bloque Profesor

10.2.3 Consultor

En este diagrama podemos observar la definición del canal channelConsultant y la llamada al agente processConsultantUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockConsultant.vsd.

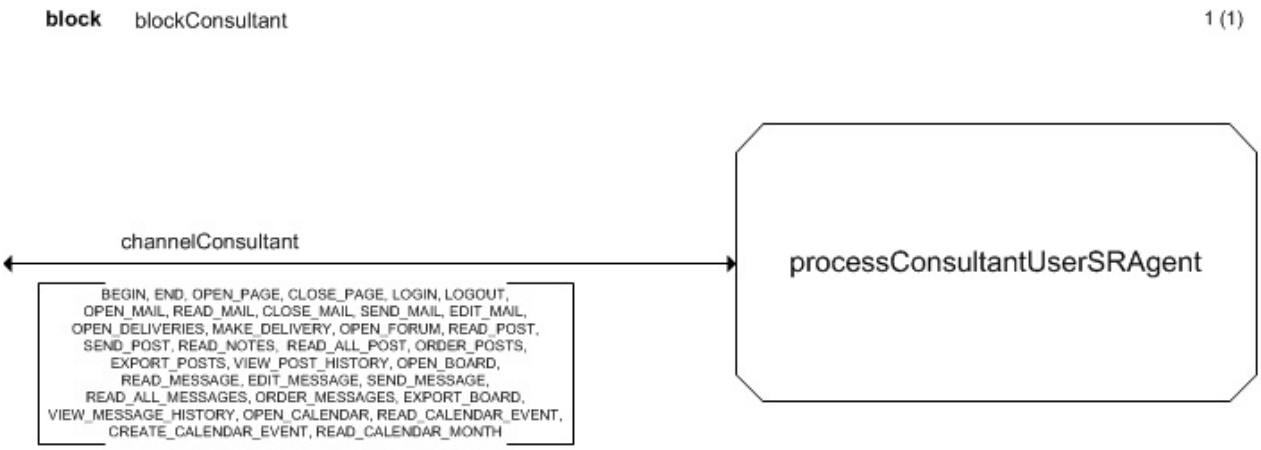


Ilustración 13: Diagrama del bloque Consultor

10.2.4 Alumni

En este diagrama podemos observar la definición del canal channelAlumni y la llamada al agente processAlumniUsersSRAgent con los eventos definidos durante el TFC. El archivo

de este bloque es blockAlumni.vsd.

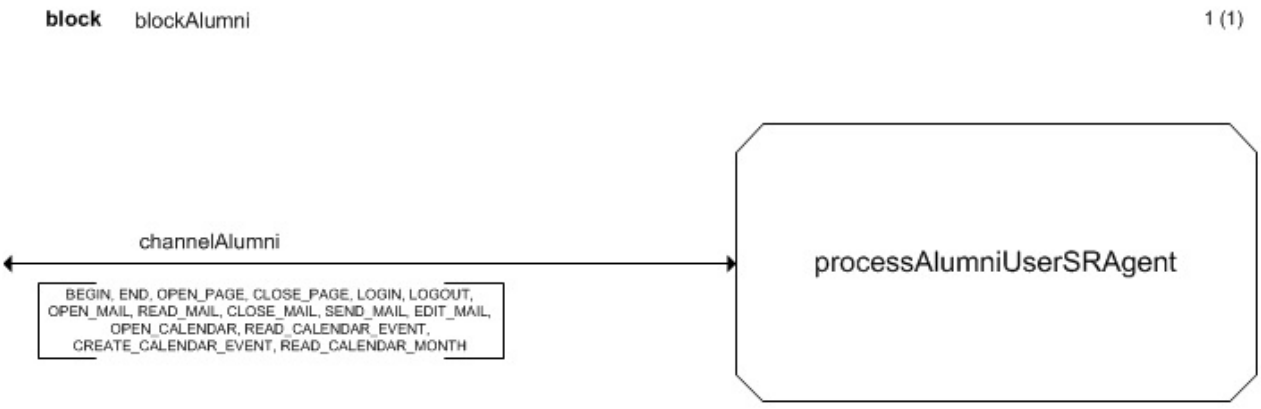


Ilustración 14: Diagrama del bloque Alumni

10.2.5 Administrador

En este diagrama podemos observar la definición del canal channelAdministrator y la llamada al agente processAdministratorUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockAdministrator.vsd.

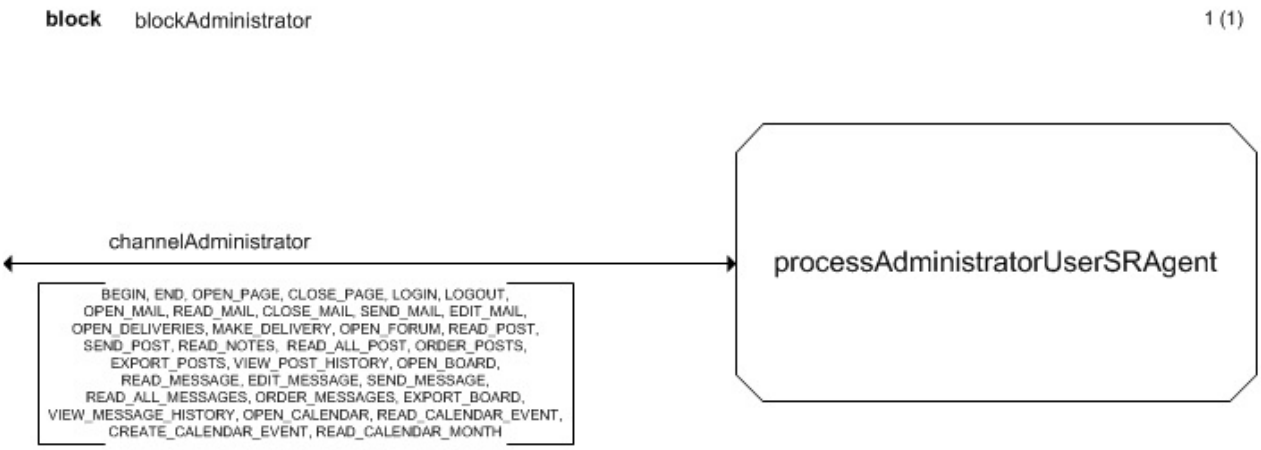


Ilustración 15: Diagrama del bloque Administrador

10.2.6 Gestor

En este diagrama podemos observar la definición del canal channelManager y la llamada al agente processManagerUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockManager.vsd.

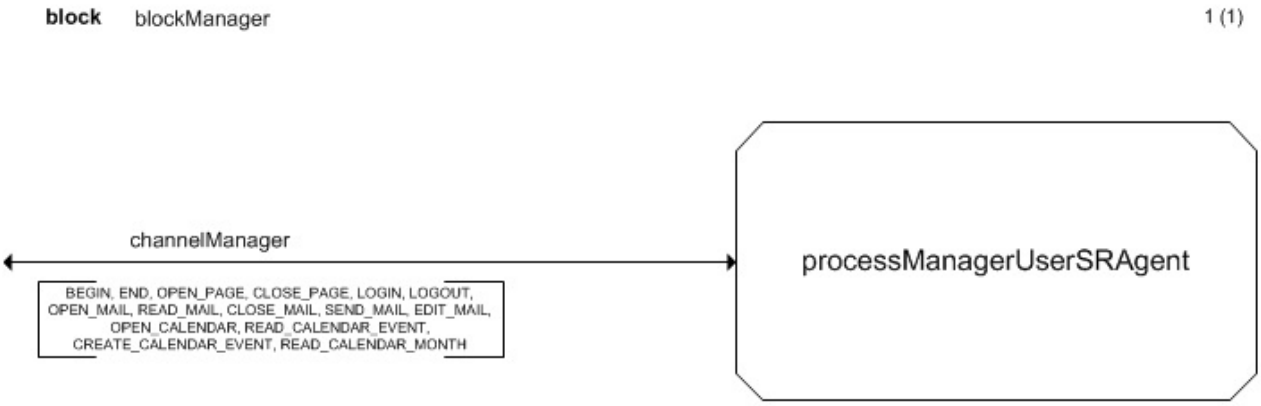


Ilustración 16: Diagrama del bloque Gestor

10.2.7 Tutor

En este diagrama podemos observar la definición del canal channelAdvisor y la llamada al agente processAdvisorUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockAdvisor.vsd.

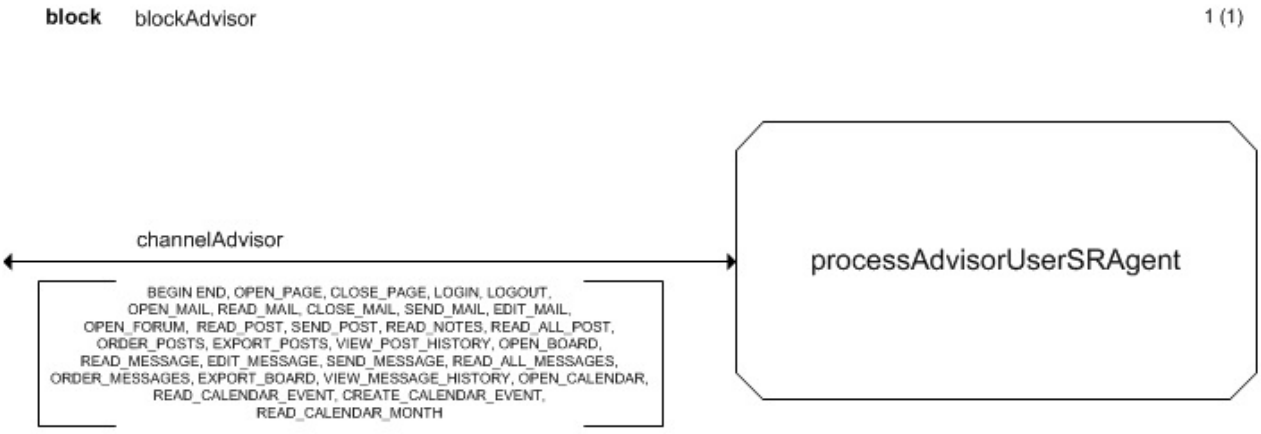


Ilustración 17: Diagrama del bloque Tutor

10.2.8 Visitante

En este diagrama podemos observar la definición del canal channelVisitor y la llamada al agente processVisitorUsersSRAgent con los eventos definidos durante el TFC. El archivo de este bloque es blockVisitor.vsd.

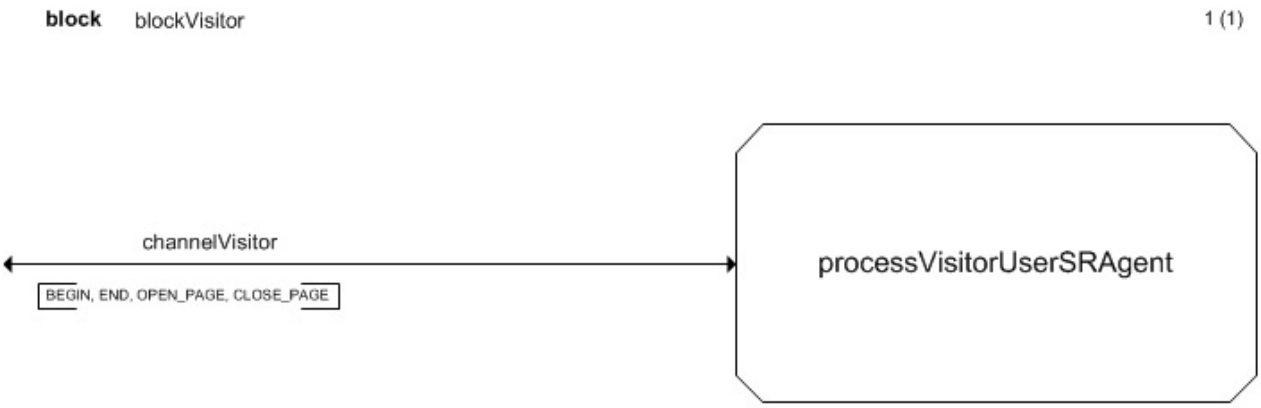


Ilustración 18: Diagrama del bloque Visitante

10.3 Procesos

10.3.1 Agente Estudiante

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- studentEventCode
- studentExecTime
- studentDelayTime
- studentInfoStack
- studentMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureStudentInitialize()
- procedureStudentReactionTime(int studentDelayTime, int studentEventCode)
- procedureStudentActionEffects()
- procedureStudentModifyingKnowledge(),
- procedureStudentExecutionTime(int studentExecTime, int studentEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processStudentUserSRAgent.vsd.

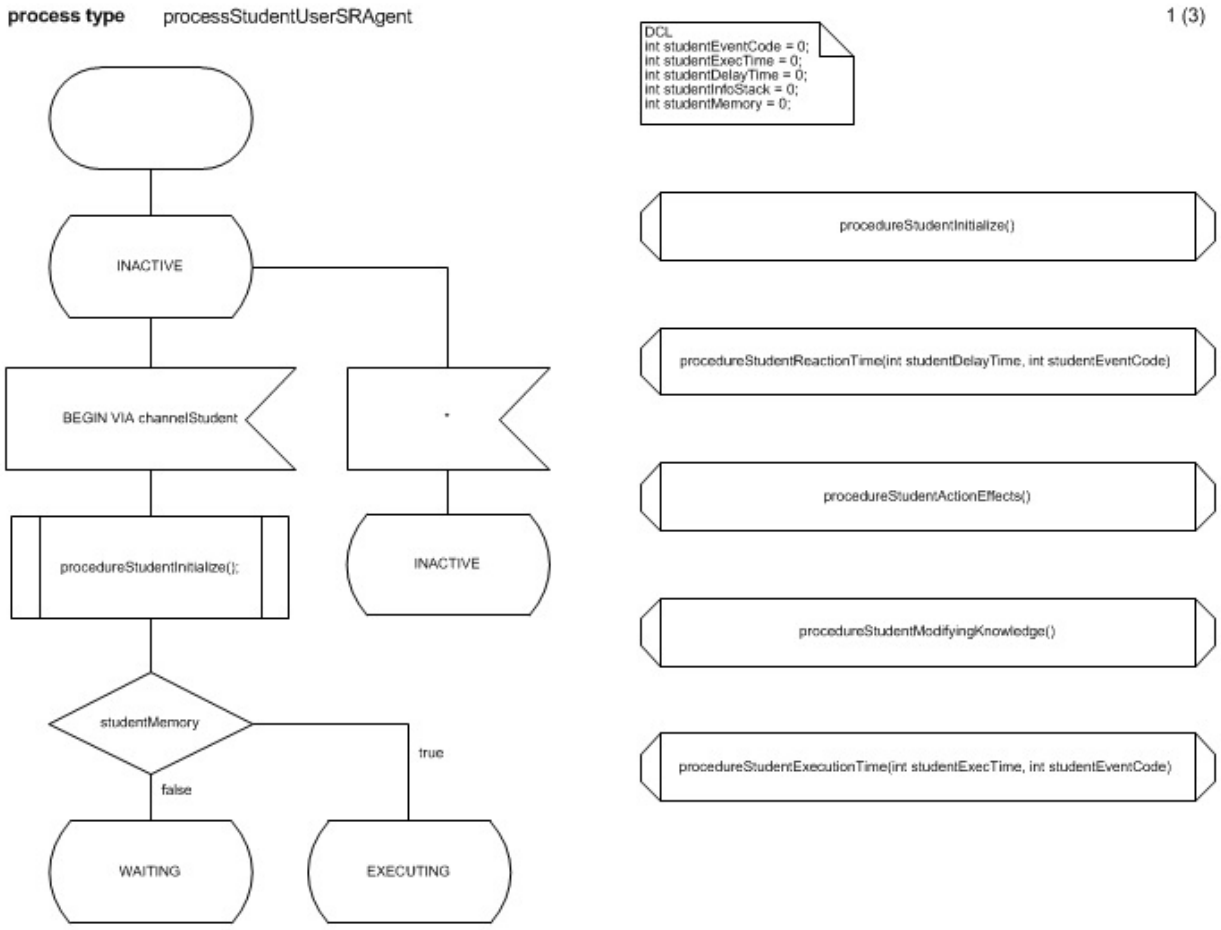


Ilustración 19: Diagrama del agente Estudiante (1 de 6)

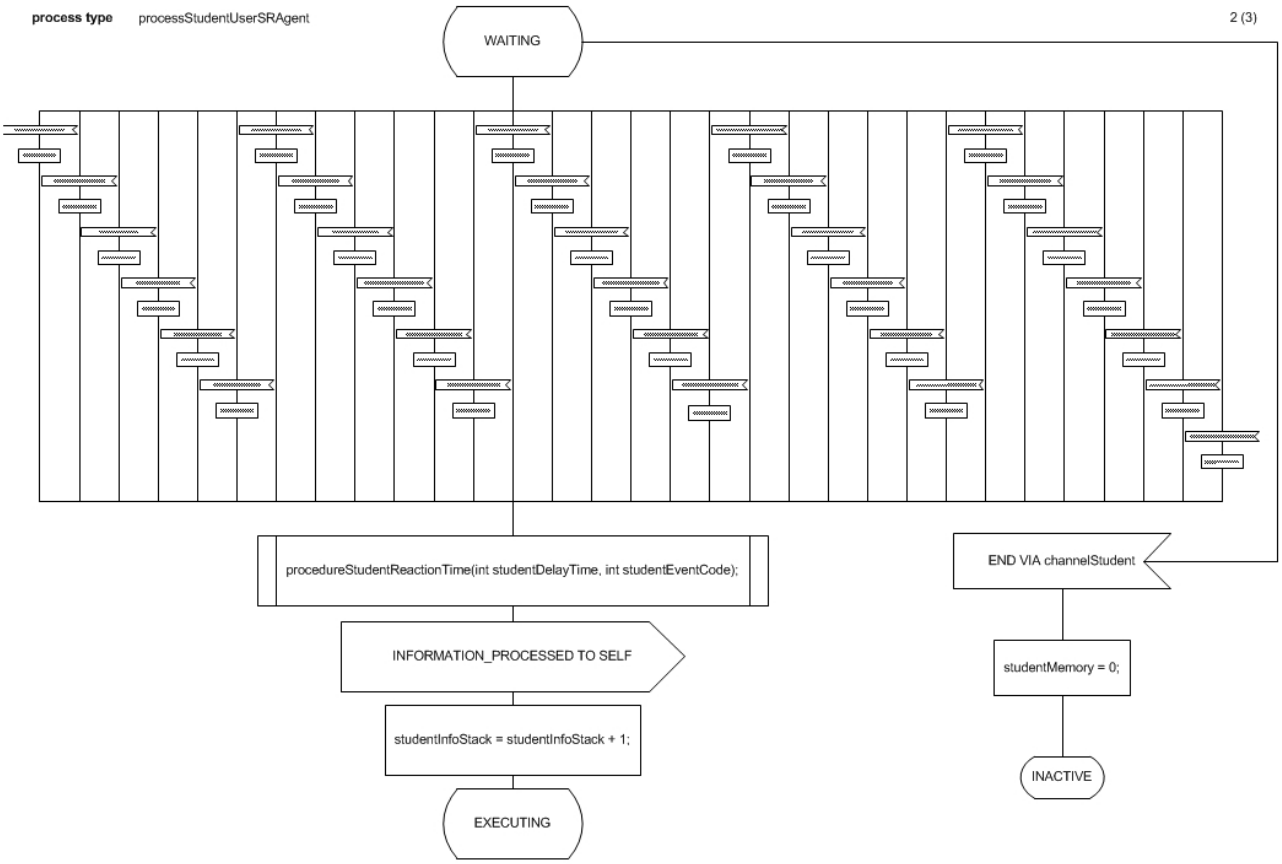


Ilustración 20: Diagrama del agente Estudiante (2 de 6)

En la siguiente ilustración se puede observar con mas detalle una parte del escenarios what-if

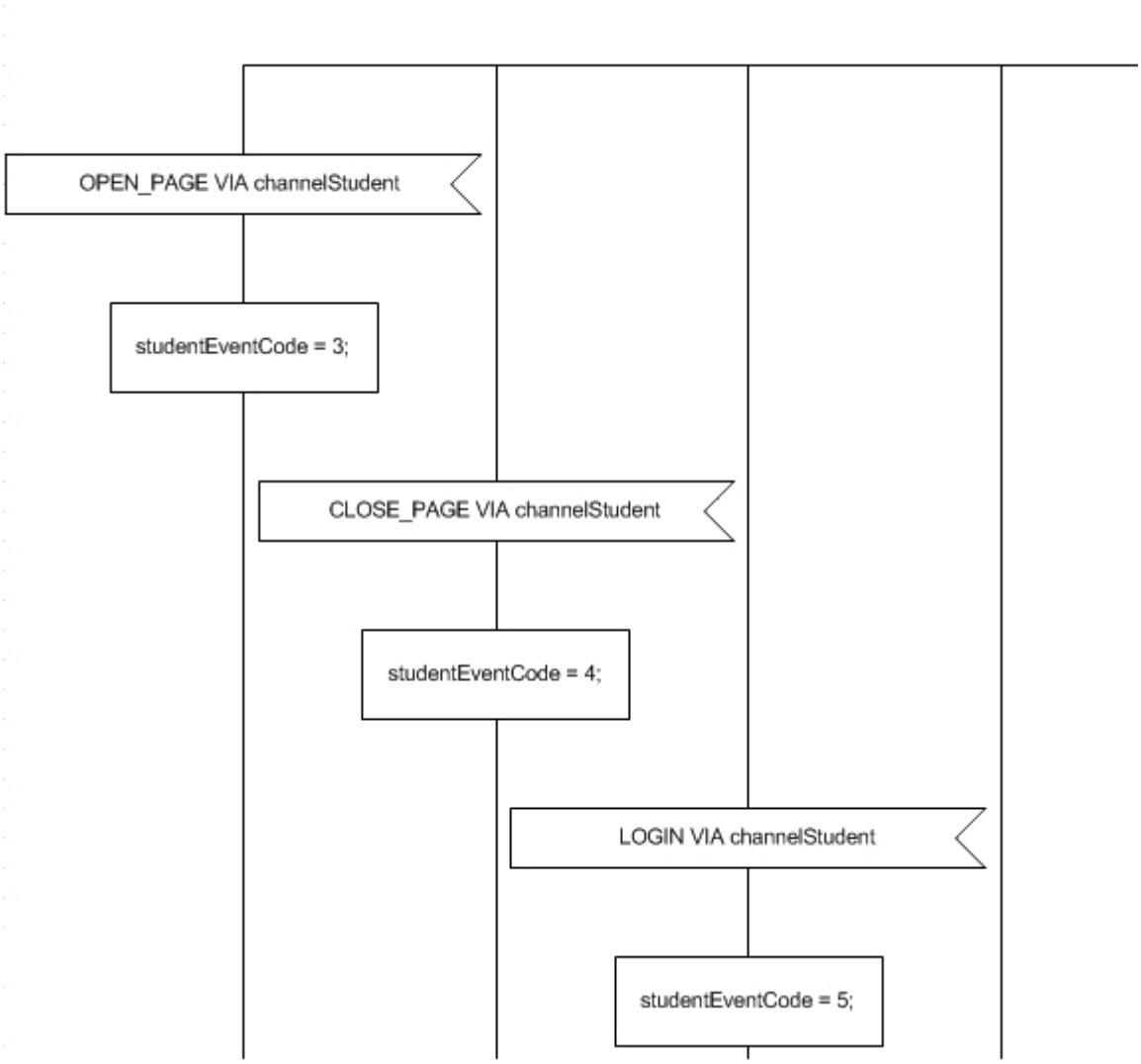


Ilustración 21: Diagrama del agente Estudiante (3 de 6)

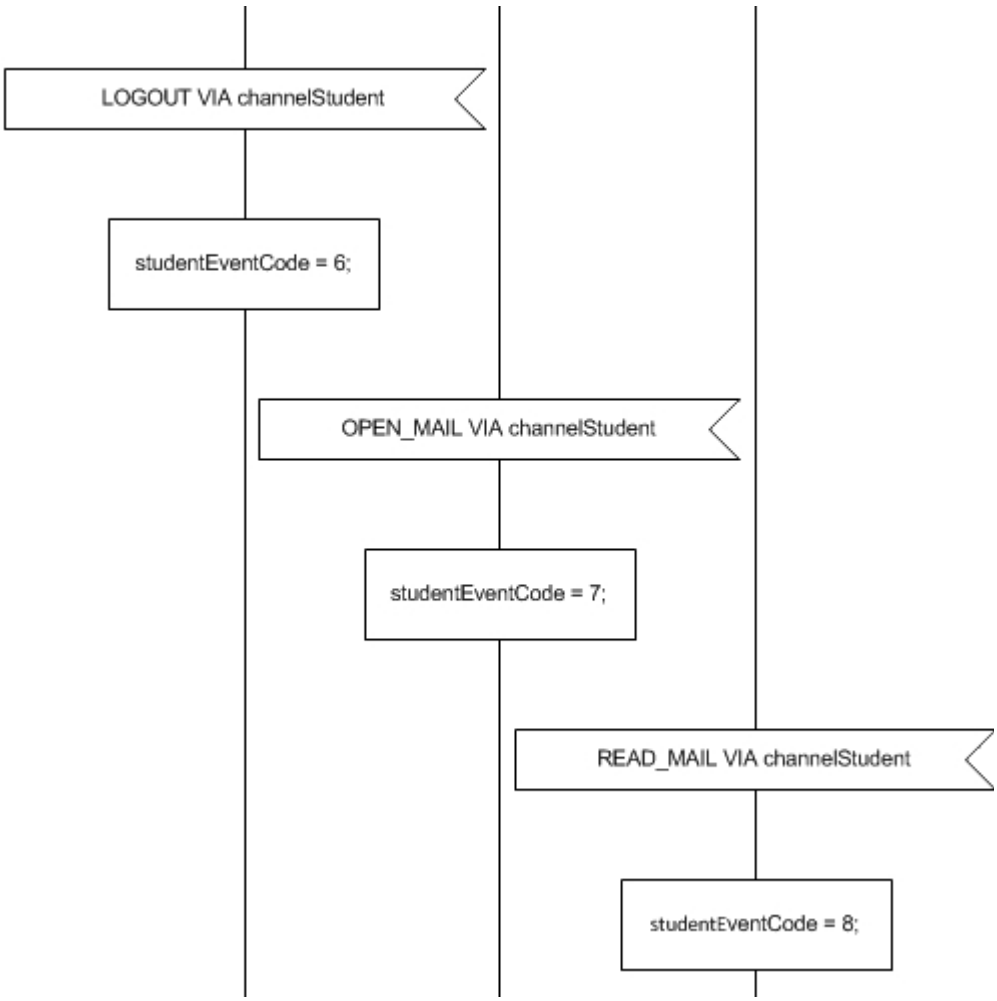


Ilustración 22: Diagrama del agente Estudiante (4 de 6)

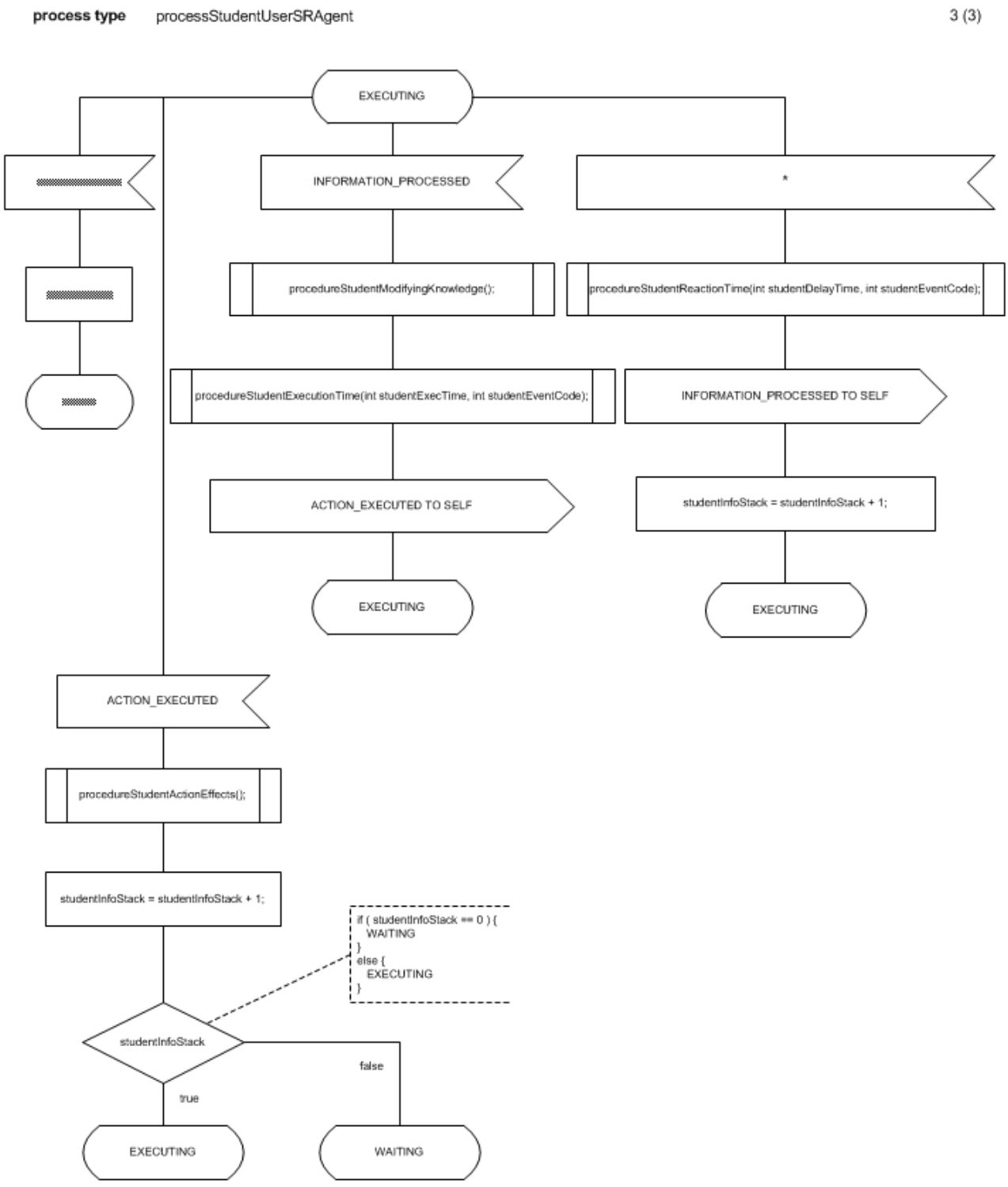


Ilustración 23: Diagrama del agente Estudiante (5 de 6)

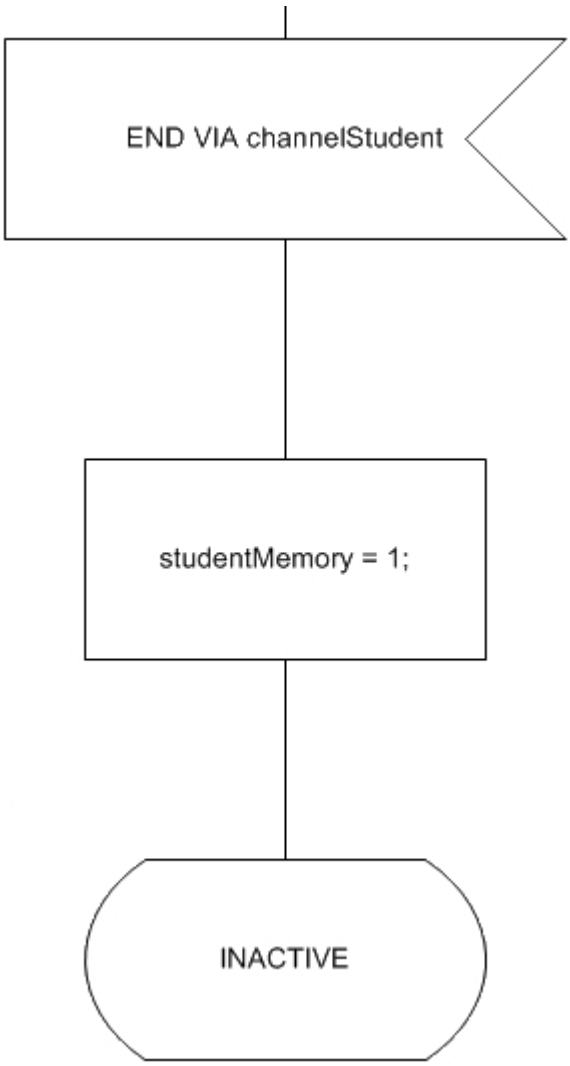


Ilustración 24: Diagrama del agente Estudiante (6 de 6)

10.3.2 Agente Profesor

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- TeacherEventCode
- TeacherExecTime
- TeacherDelayTime
- TeacherInfoStack
- TeacherMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureTeacherInitialize()
- procedureTeacherReactionTime(int TeacherDelayTime, int TeacherEventCode)
- procedureTeacherActionEffects()
- procedureTeacherModifyingKnowledge(),
- procedureTeacherExecutionTime(int TeacherExecTime, int TeacherEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processTeacherUserSRAgent.vsd.

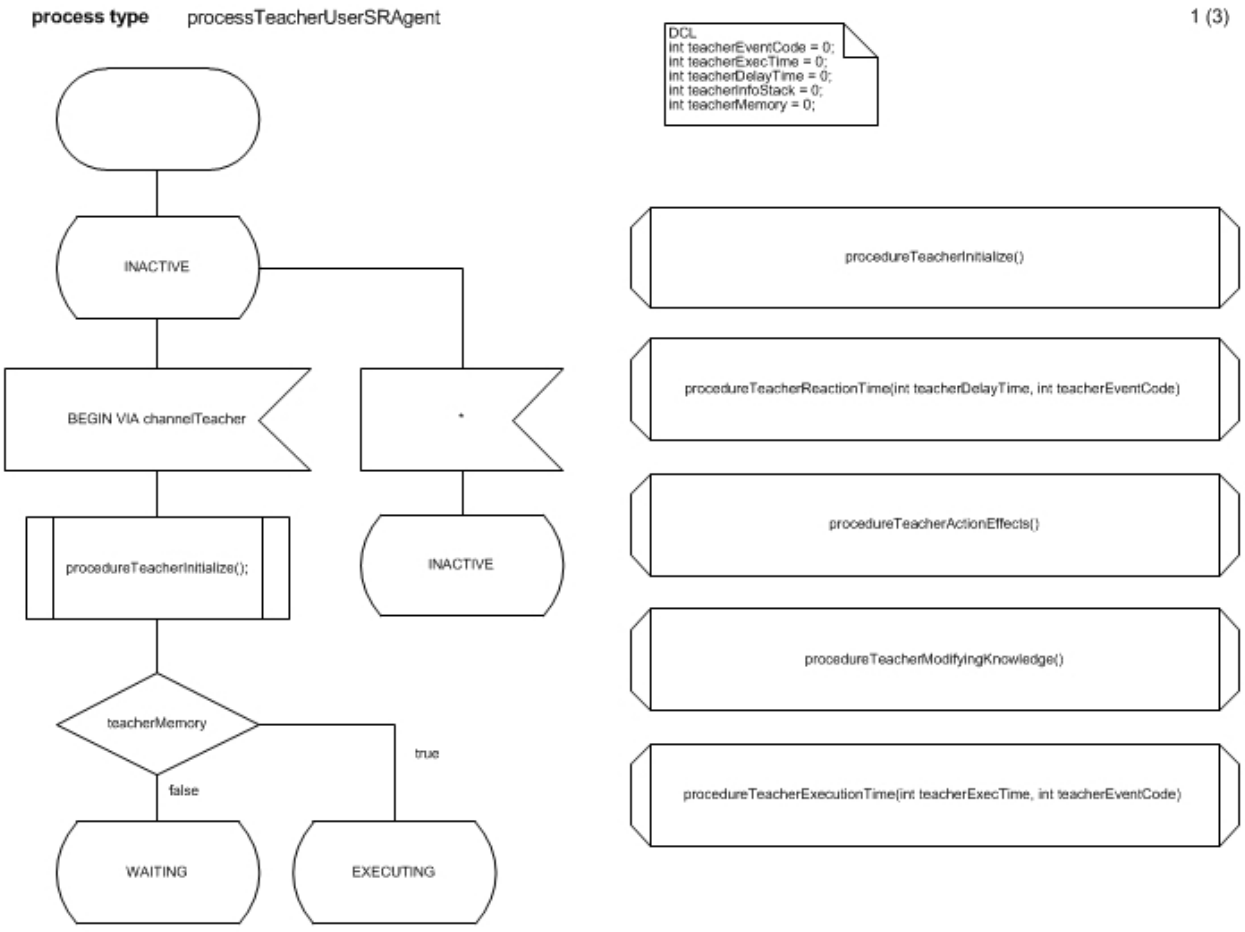


Ilustración 25: Diagrama del agente Profesor (1 de 6)

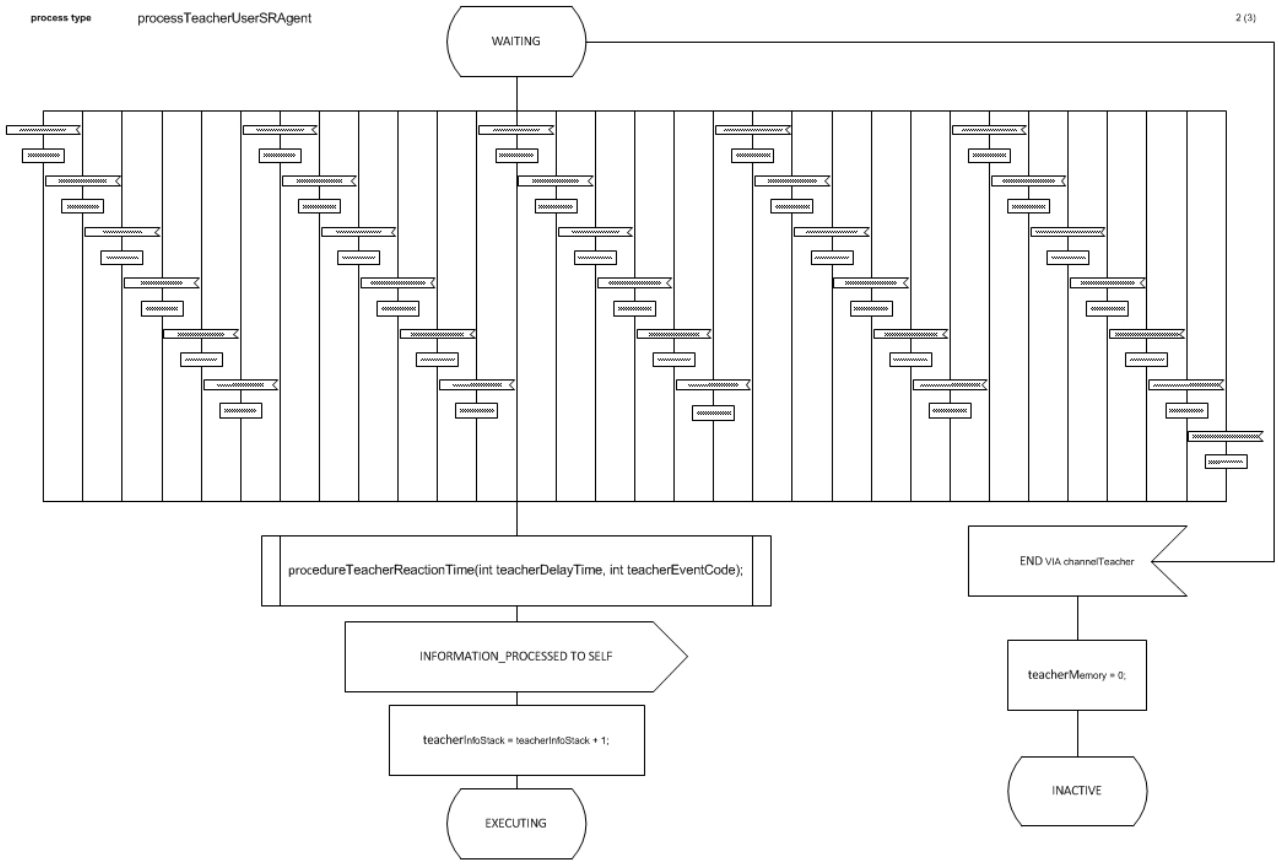


Ilustración 26: Diagrama del agente Profesor (2 de 6)

En la siguiente ilustración se puede observar con mas detalle una parte del escenario what-if

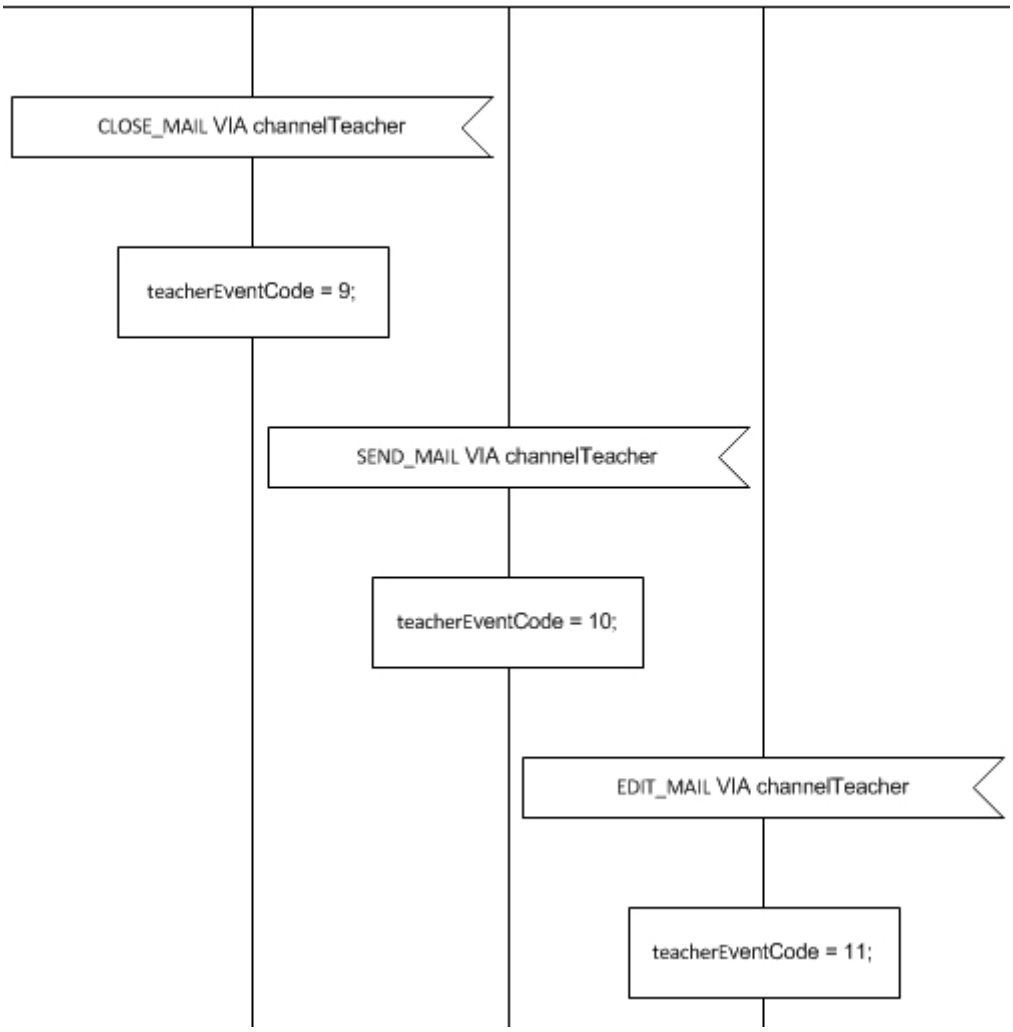


Ilustración 27: Diagrama del agente Profesor (3 de 6)

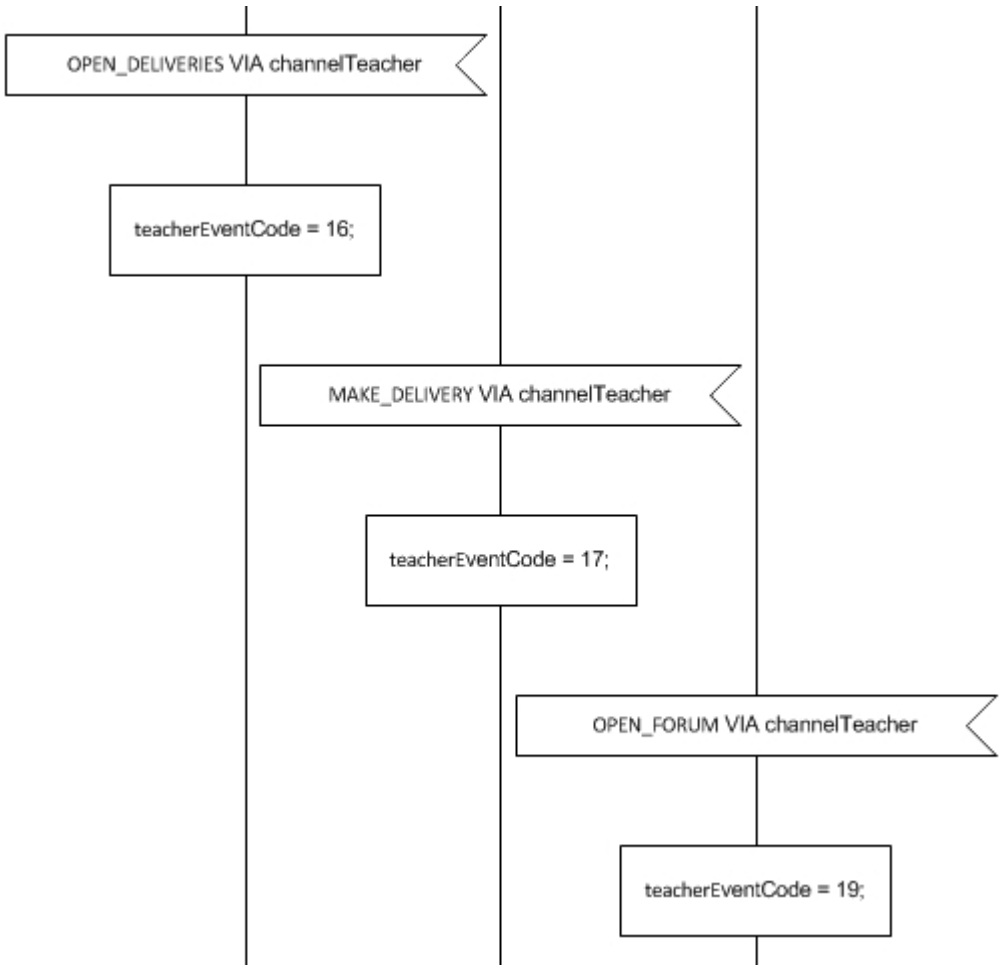


Ilustración 28: Diagrama del agente Profesor (4 de 6)

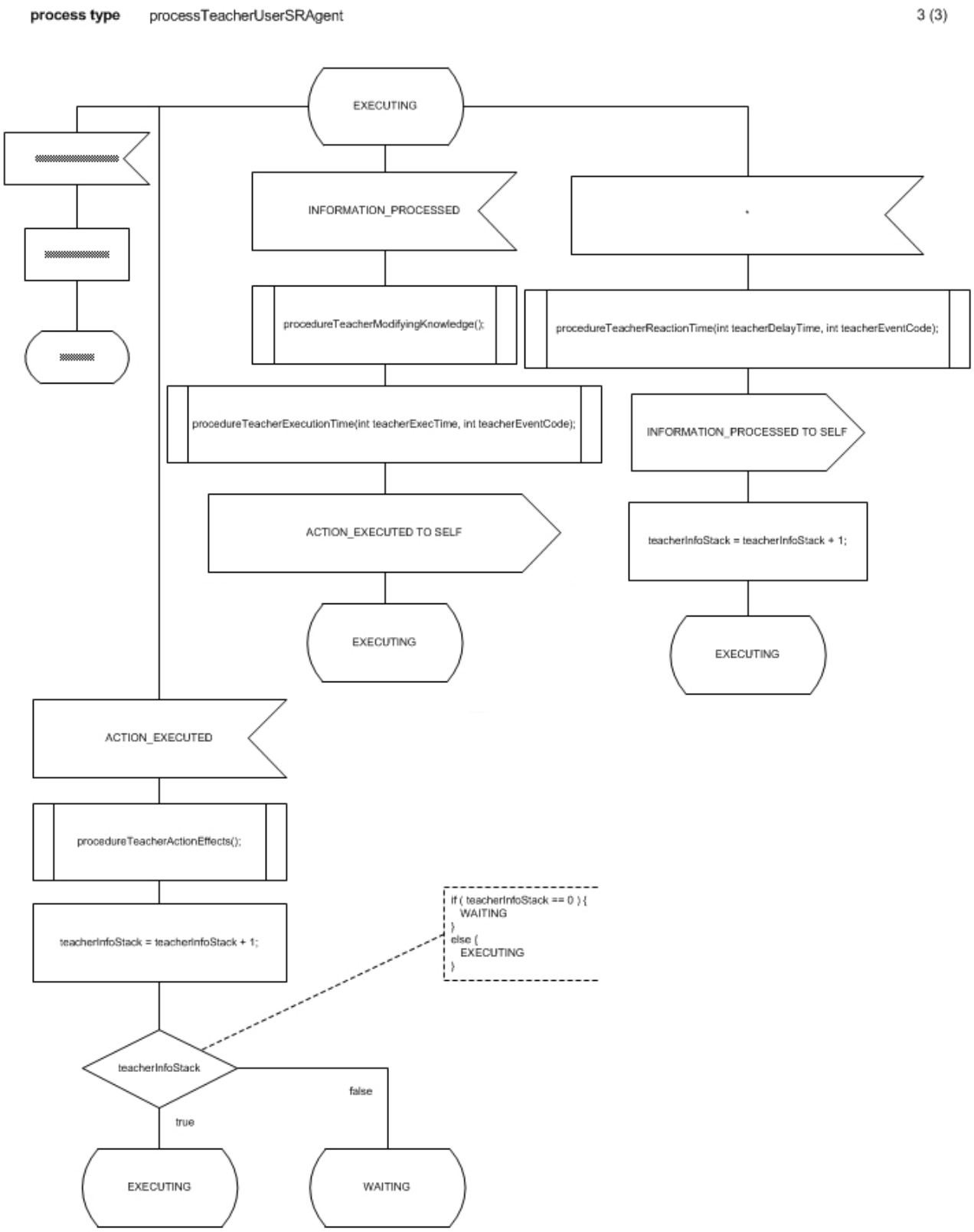


Ilustración 29: Diagrama del agente Profesor (5 de 6)

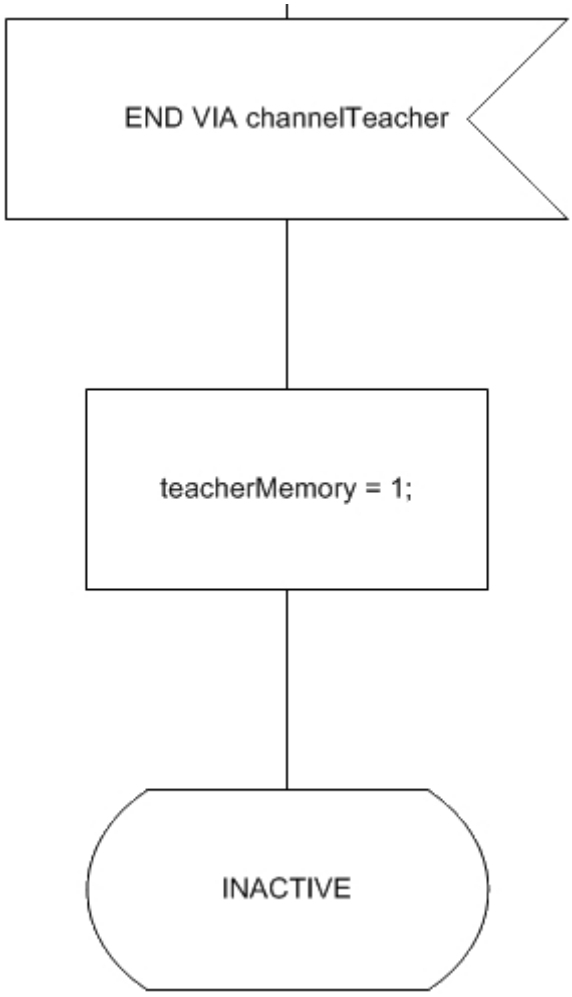


Ilustración 30: Diagrama del agente Profesor (6 de 6)

10.3.3 Agente Consultor

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- ConsultantEventCode
- ConsultantExecTime
- ConsultantDelayTime
- ConsultantInfoStack
- ConsultantMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureConsultantInitialize()
- procedureConsultantReactionTime(int ConsultantDelayTime, int ConsultantEventCode)
- procedureConsultantActionEffects()
- procedureConsultantModifyingKnowledge(),
- procedureConsultantExecutionTime(int ConsultantExecTime, int ConsultantEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processConsultantUserSRAgent.vsd.

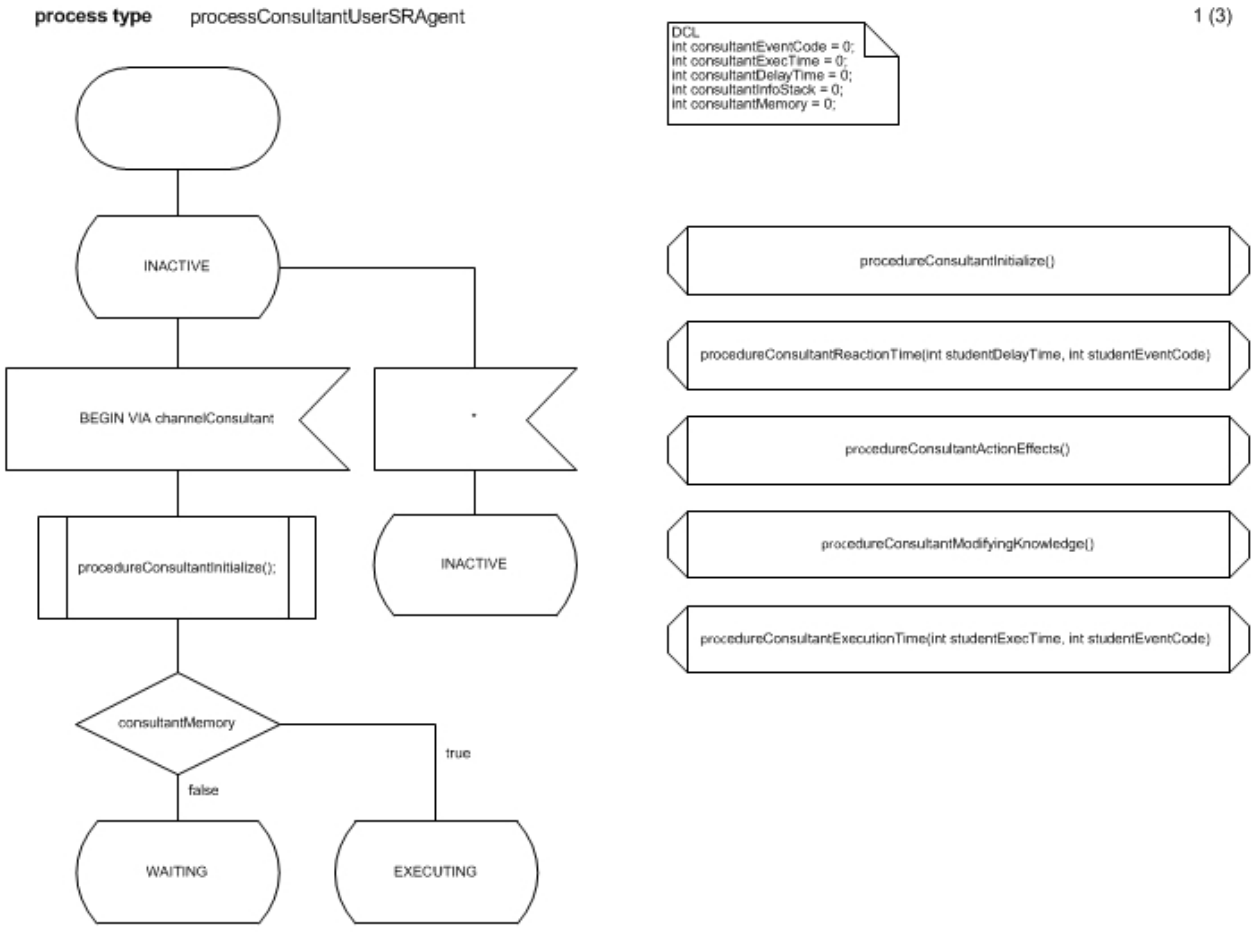


Ilustración 31: Diagrama del agente Consultor (1 de 6)

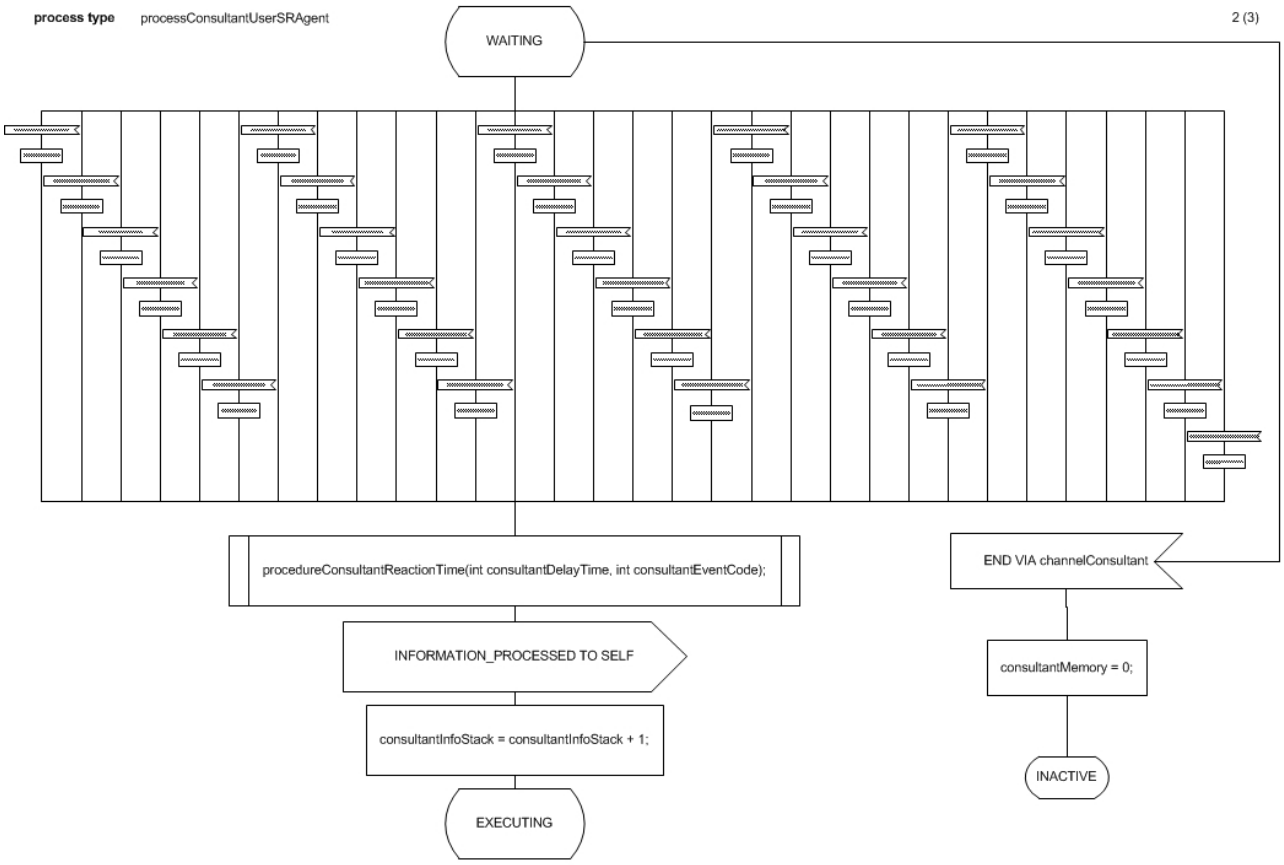


Ilustración 32: Diagrama del agente Consultor (2 de 6)

En la siguiente ilustración se puede observar con mas detalle una parte del escenario what-if

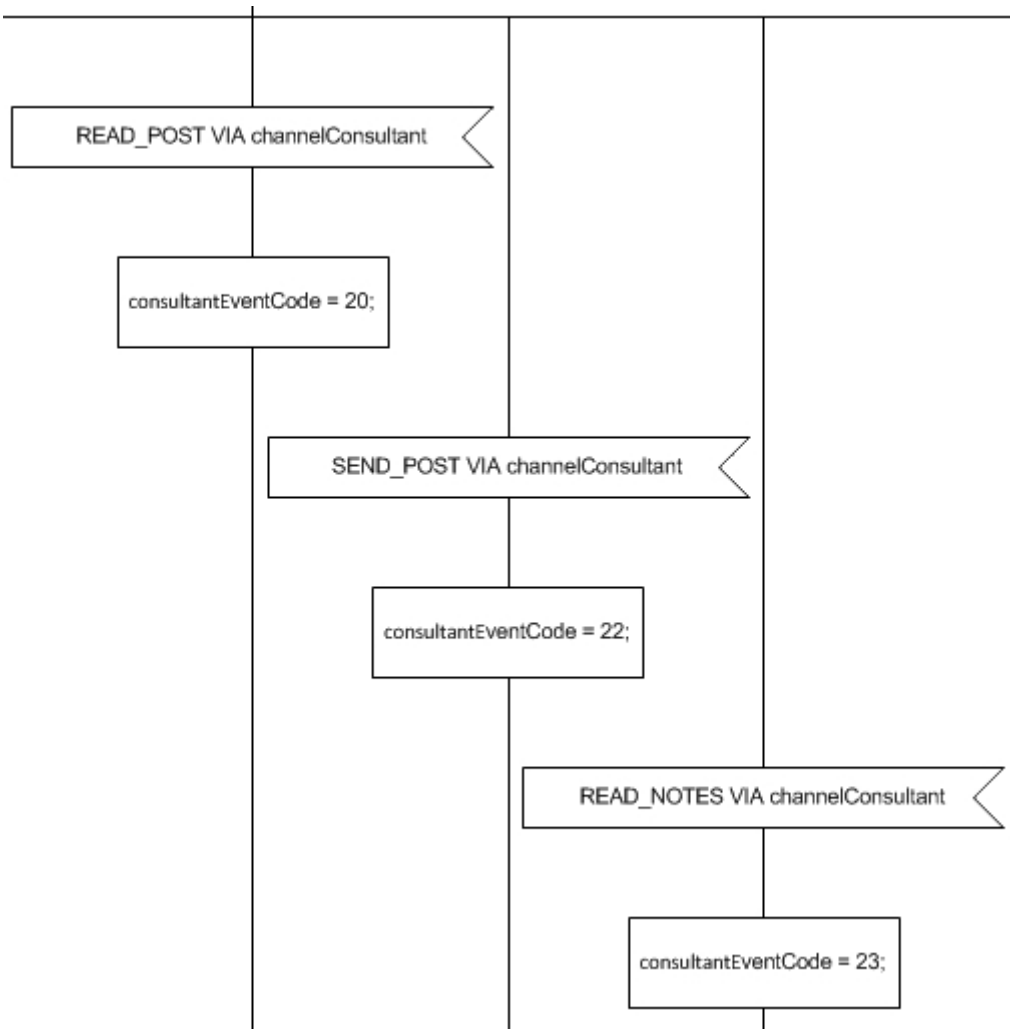


Ilustración 33: Diagrama del agente Consultor (3 de 6)

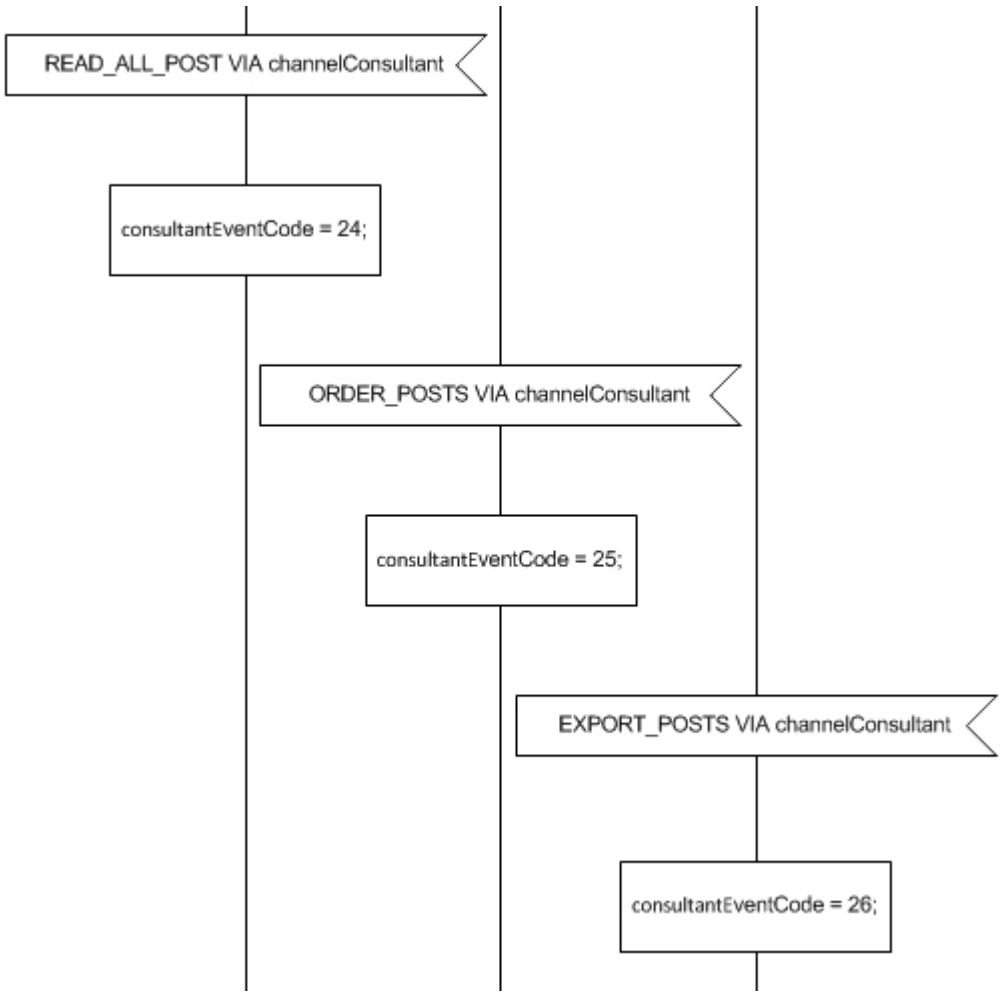


Ilustración 34: Diagrama del agente Consultor (4 de 6)

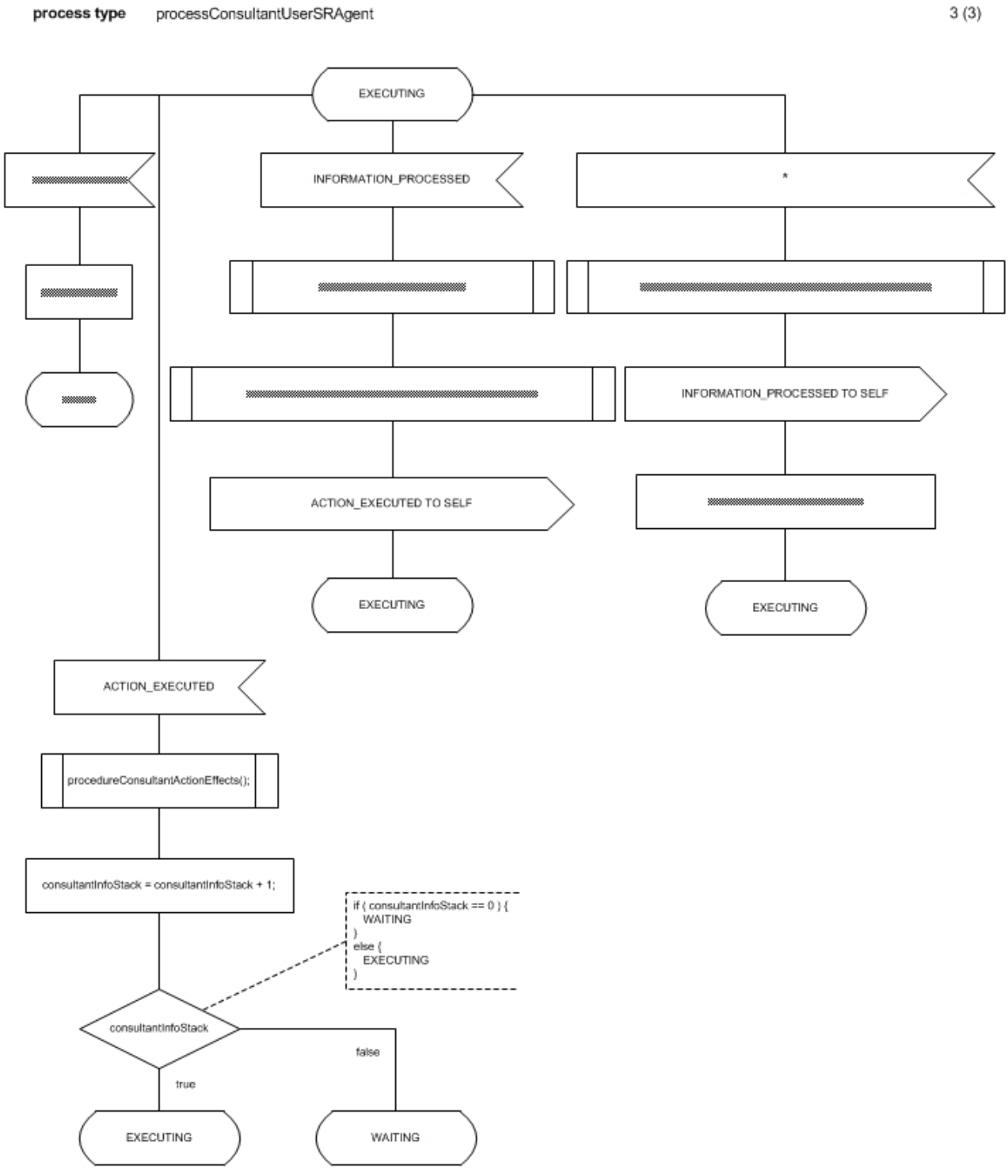


Ilustración 35: Diagrama del agente Consultor (5 de 6)

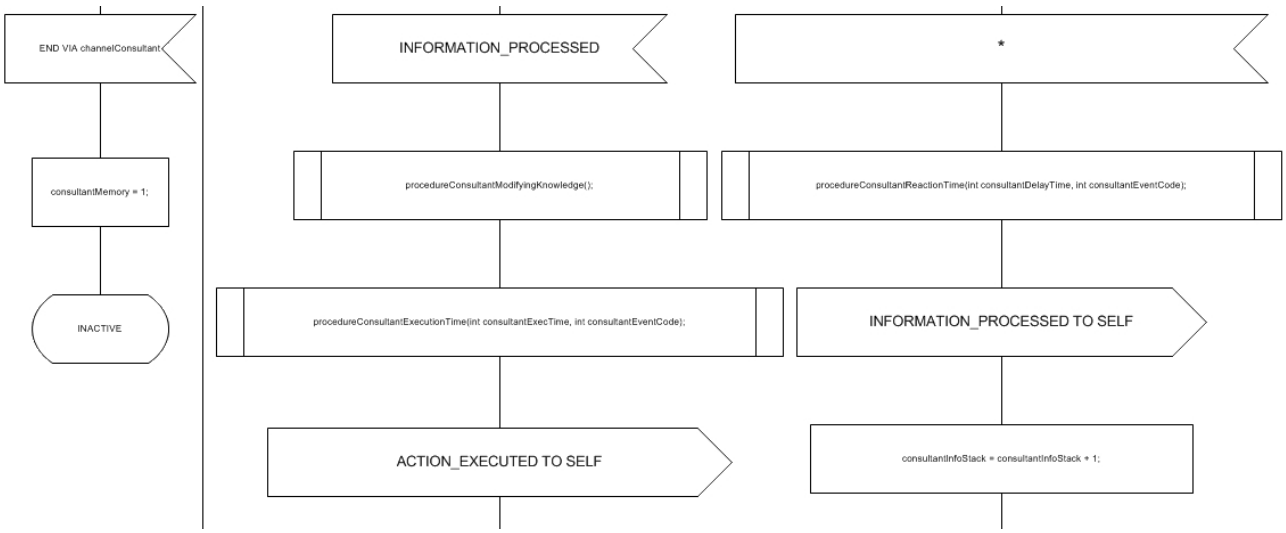


Ilustración 36: Diagrama del agente Consultor (6 de 6)

10.3.4 Agente Alumni

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- AlumniEventCode
- AlumniExecTime
- AlumniDelayTime
- AlumniInfoStack
- AlumniMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureAlumniInitialize()
- procedureAlumniReactionTime(int AlumniDelayTime, int AlumniEventCode)
- procedureAlumniActionEffects()
- procedureAlumniModifyingKnowledge(),
- procedureAlumniExecutionTime(int AlumniExecTime, int AlumniEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processAlumniUserSRAgent.vsd.

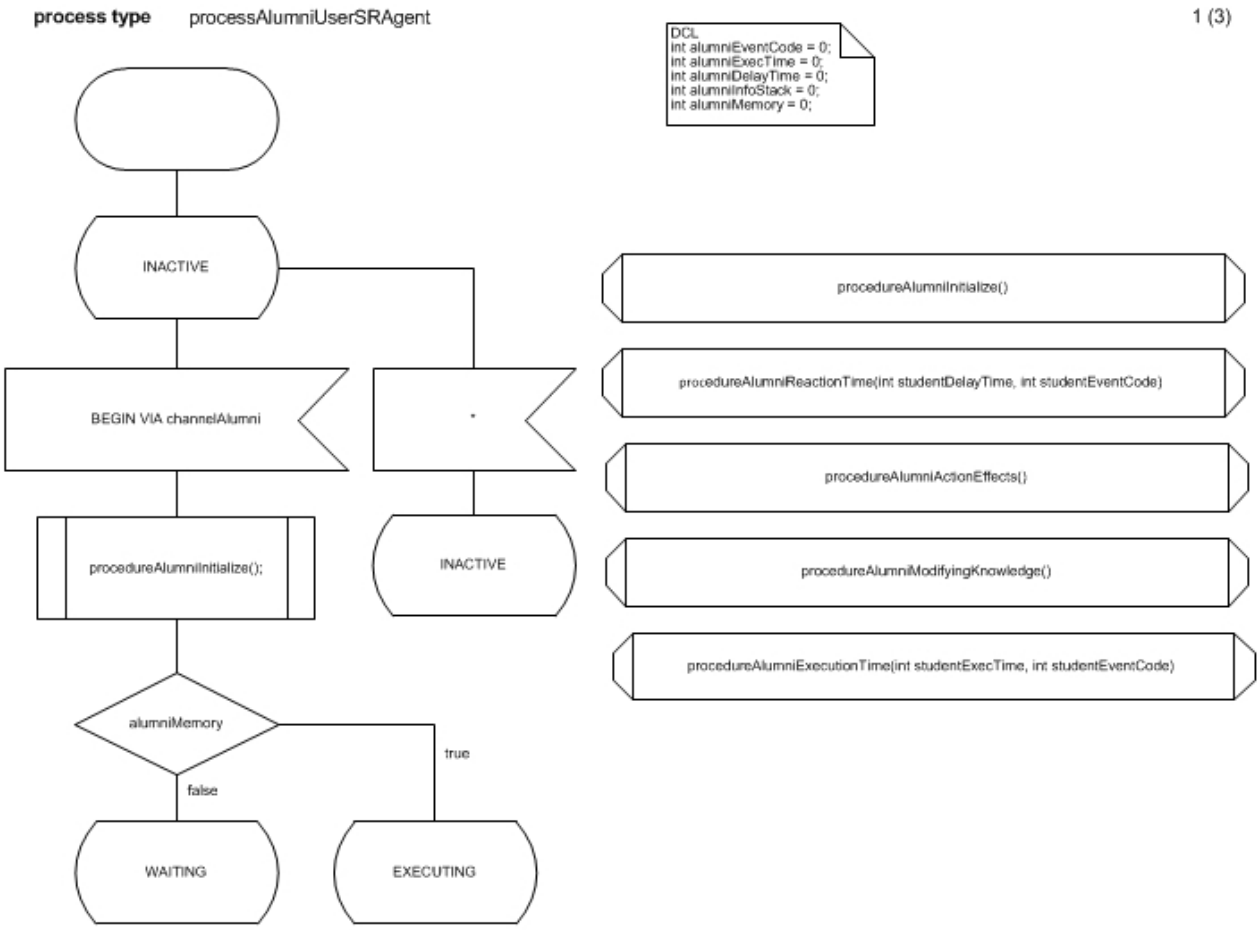


Ilustración 37: Diagrama del agente Alumni (1 de 5)

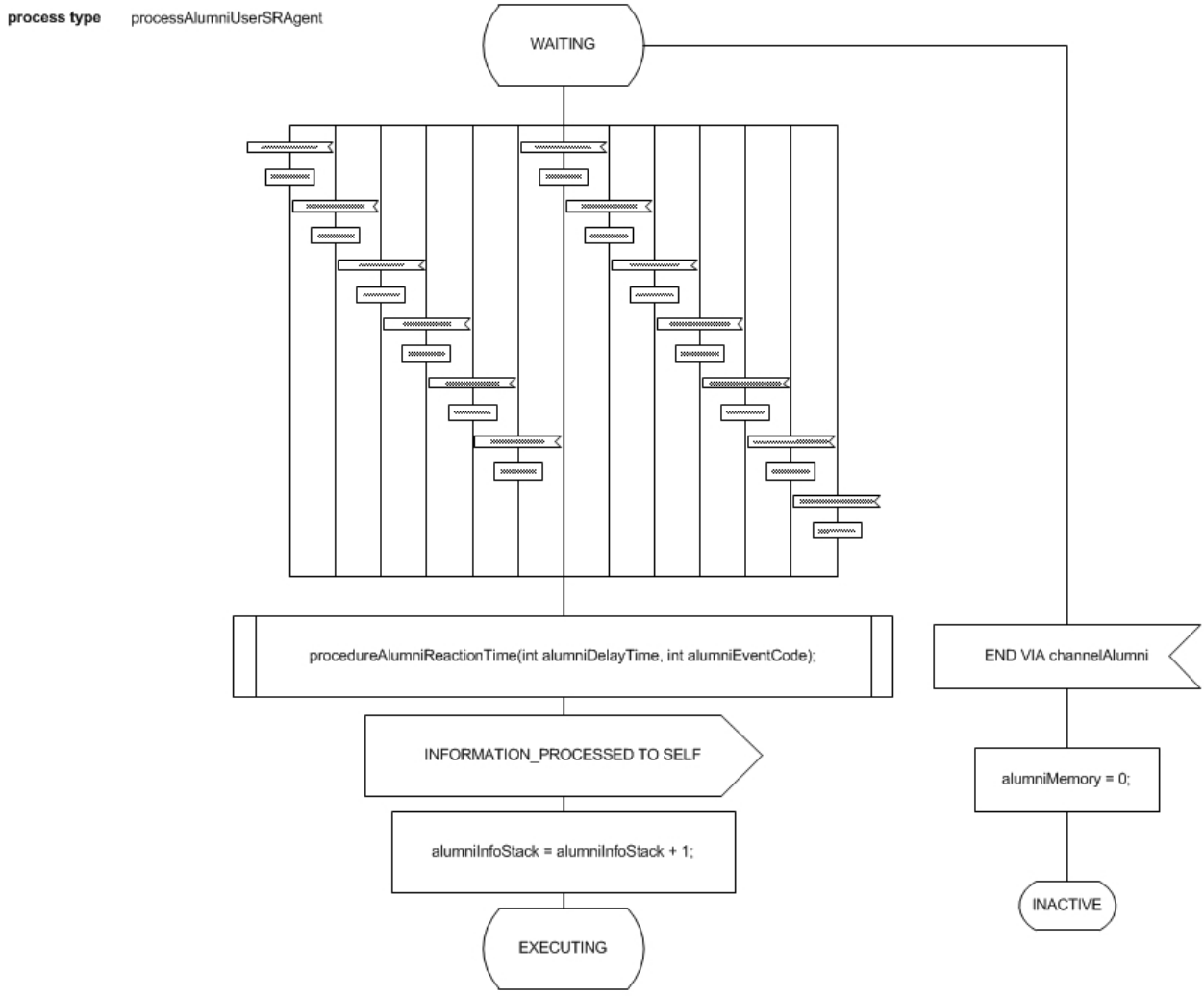


Ilustración 38: Diagrama del agente Alumni (2 de 5)

En la siguiente ilustración se puede observar con mas detalle una parte del escenario what-if

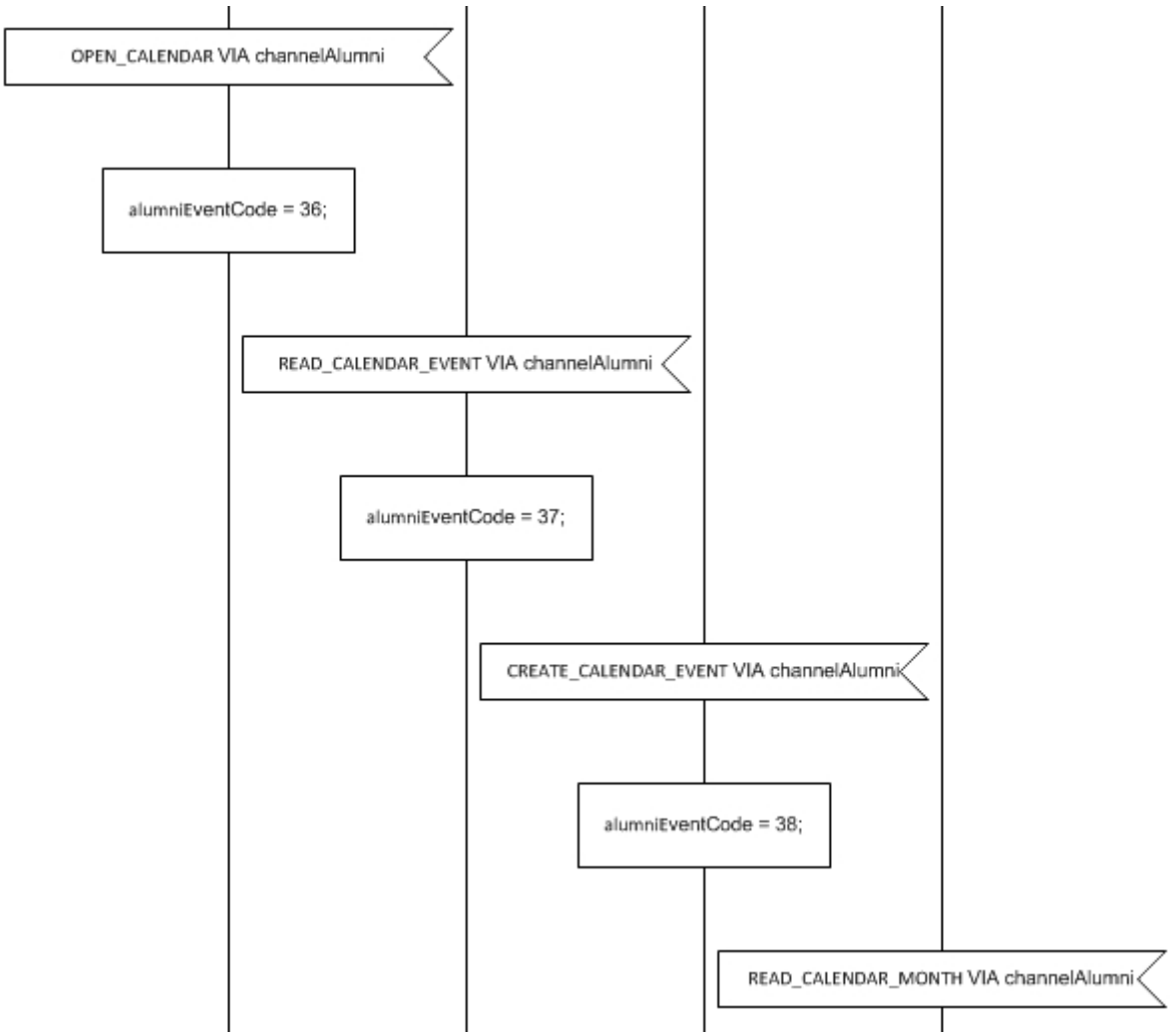


Ilustración 39: Diagrama del agente Alumni (3 de 5)

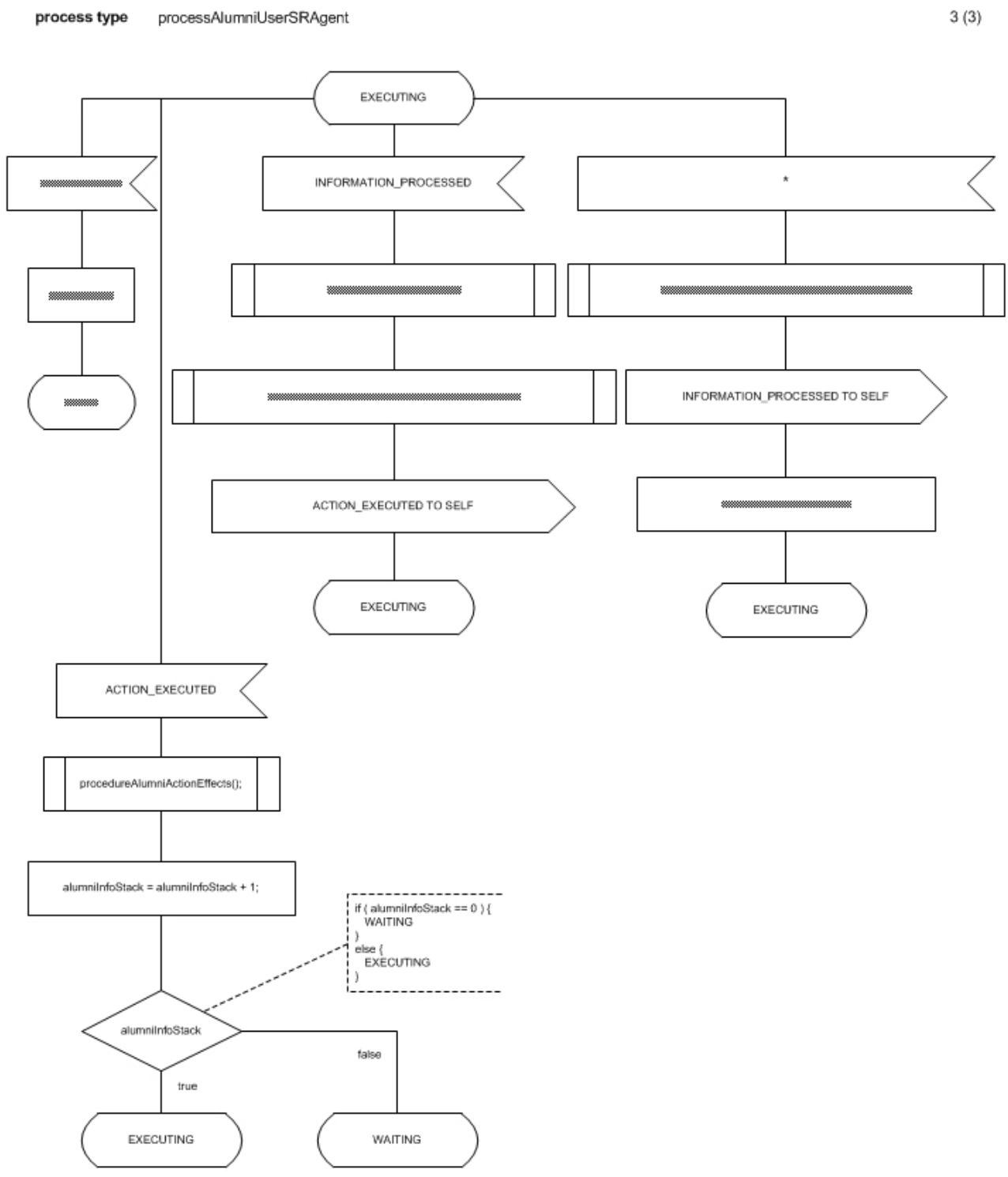


Ilustración 40: Diagrama del agente Alumni (4 de 5)

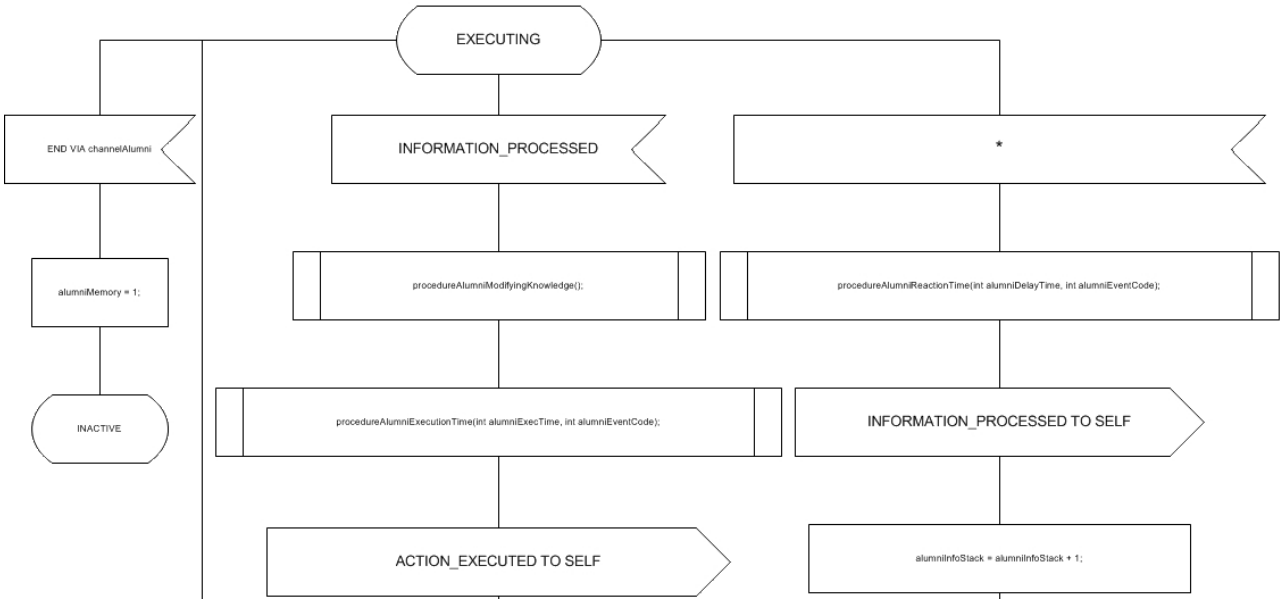


Ilustración 41: Diagrama del agente Alumni (5 de 5)

10.3.5 Agente Administrador

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- AdministratorEventCode
- AdministratorExecTime
- AdministratorDelayTime
- AdministratorInfoStack
- AdministratorMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureAdministratorInitialize()
- procedureAdministratorReactionTime(int AdministratorDelayTime, int AdministratorEventCode)
- procedureAdministratorActionEffects()
- procedureAdministratorModifyingKnowledge(),
- procedureAdministratorExecutionTime(int AdministratorExecTime, int AdministratorEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processAdministratorUserSRAgent.vsd.

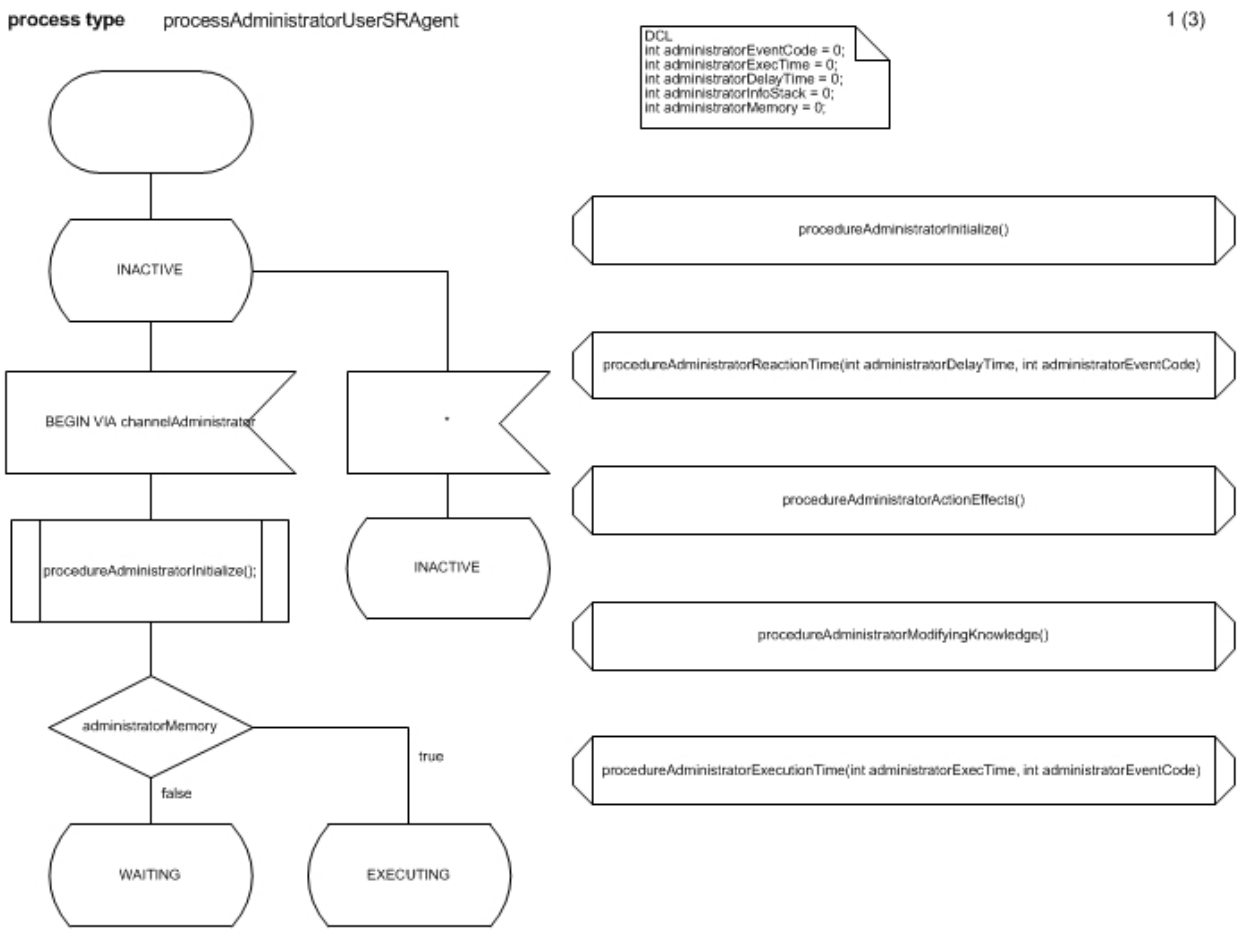


Ilustración 42: Diagrama del agente Administrador (1 de 3)

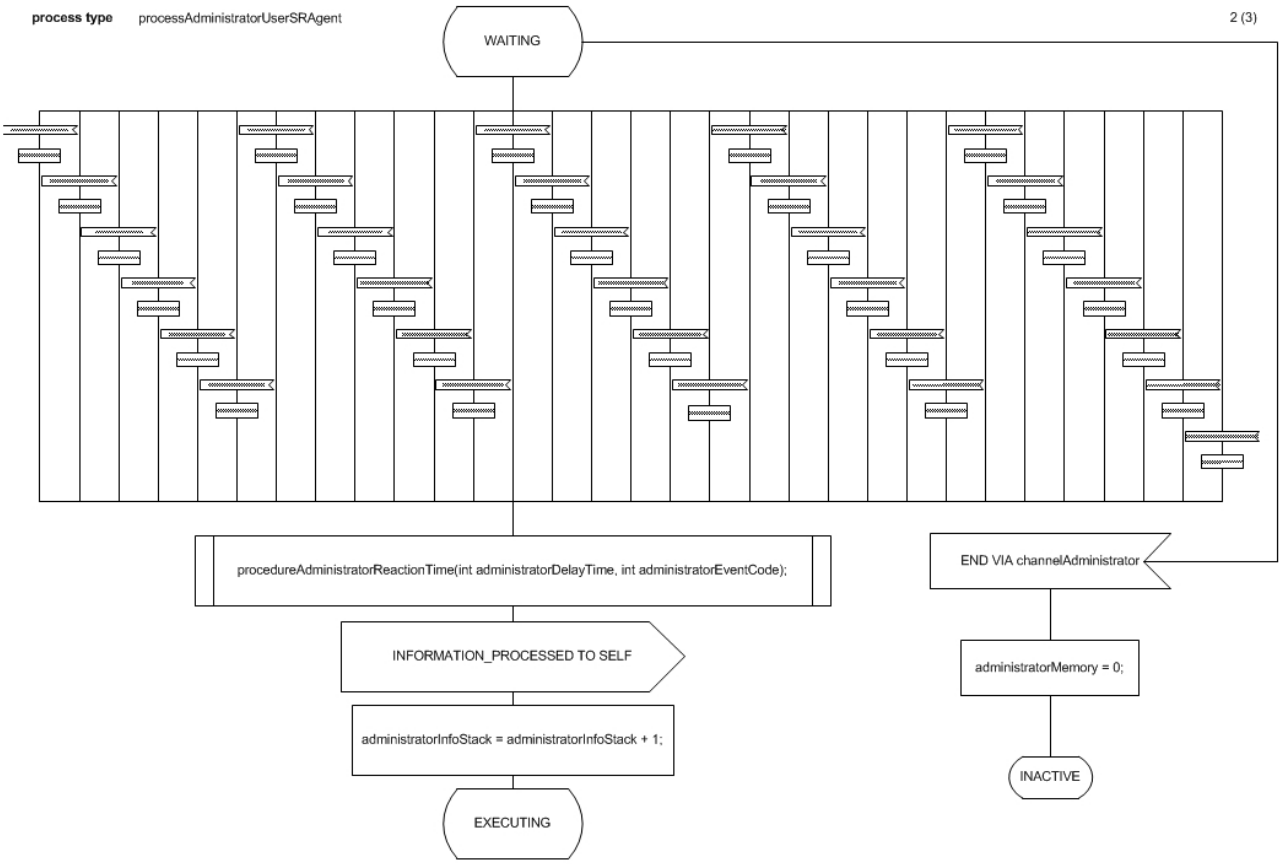


Ilustración 43: Diagrama del agente Administrador (2 de 3)

En la siguiente ilustración se puede observar con mas detalle una parte del escenario what-if

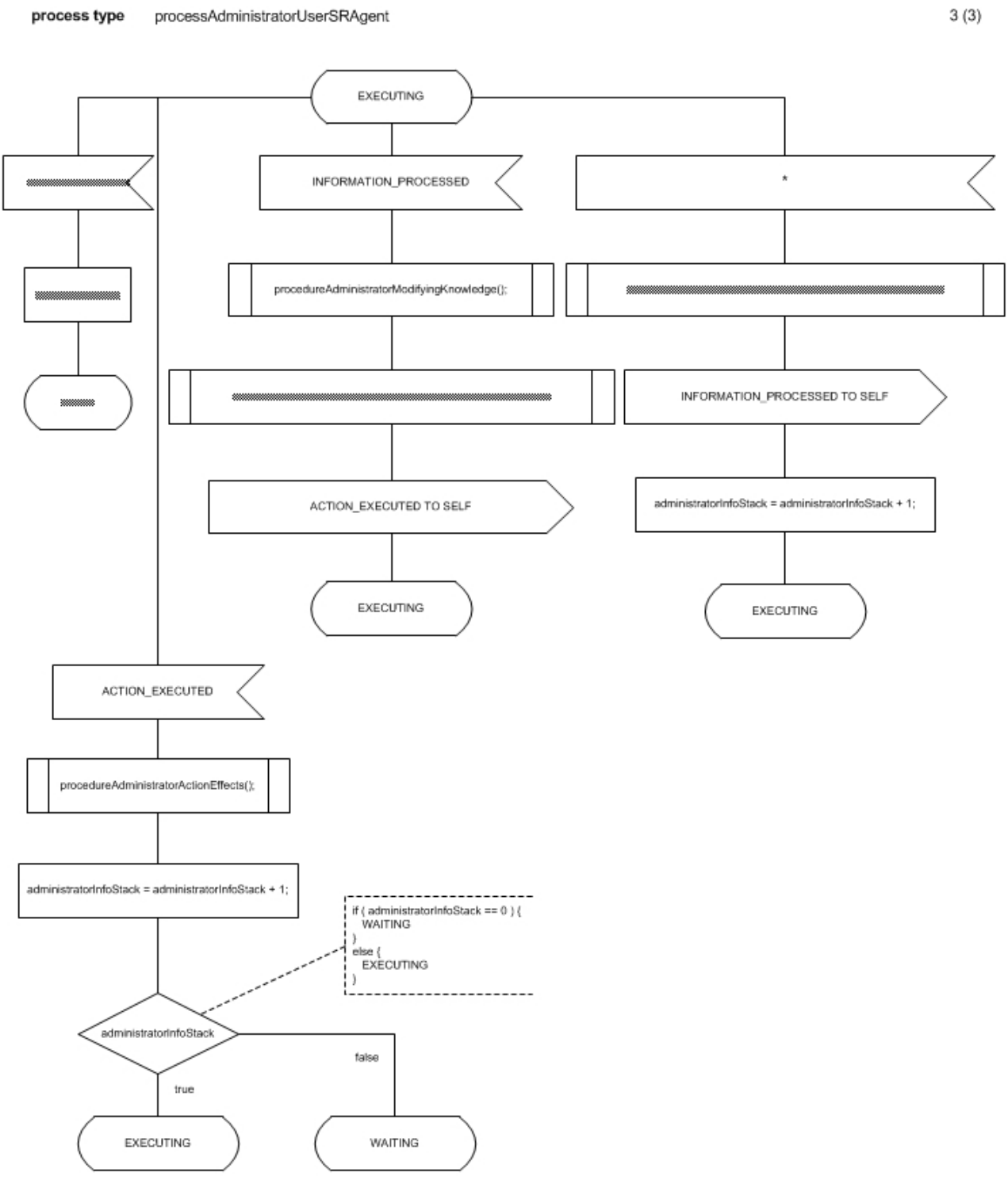


Ilustración 44: Diagrama del agente Administrador (3 de 3)

10.3.6 Agente Gestor

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- ManagerEventCode
- ManagerExecTime
- ManagerDelayTime
- ManagerInfoStack
- ManagerMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureManagerInitialize()
- procedureManagerReactionTime(int ManagerDelayTime, int ManagerEventCode)
- procedureManagerActionEffects()
- procedureManagerModifyingKnowledge(),
- procedureManagerExecutionTime(int ManagerExecTime, int ManagerEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processManagerUserSRAgent.vsd.

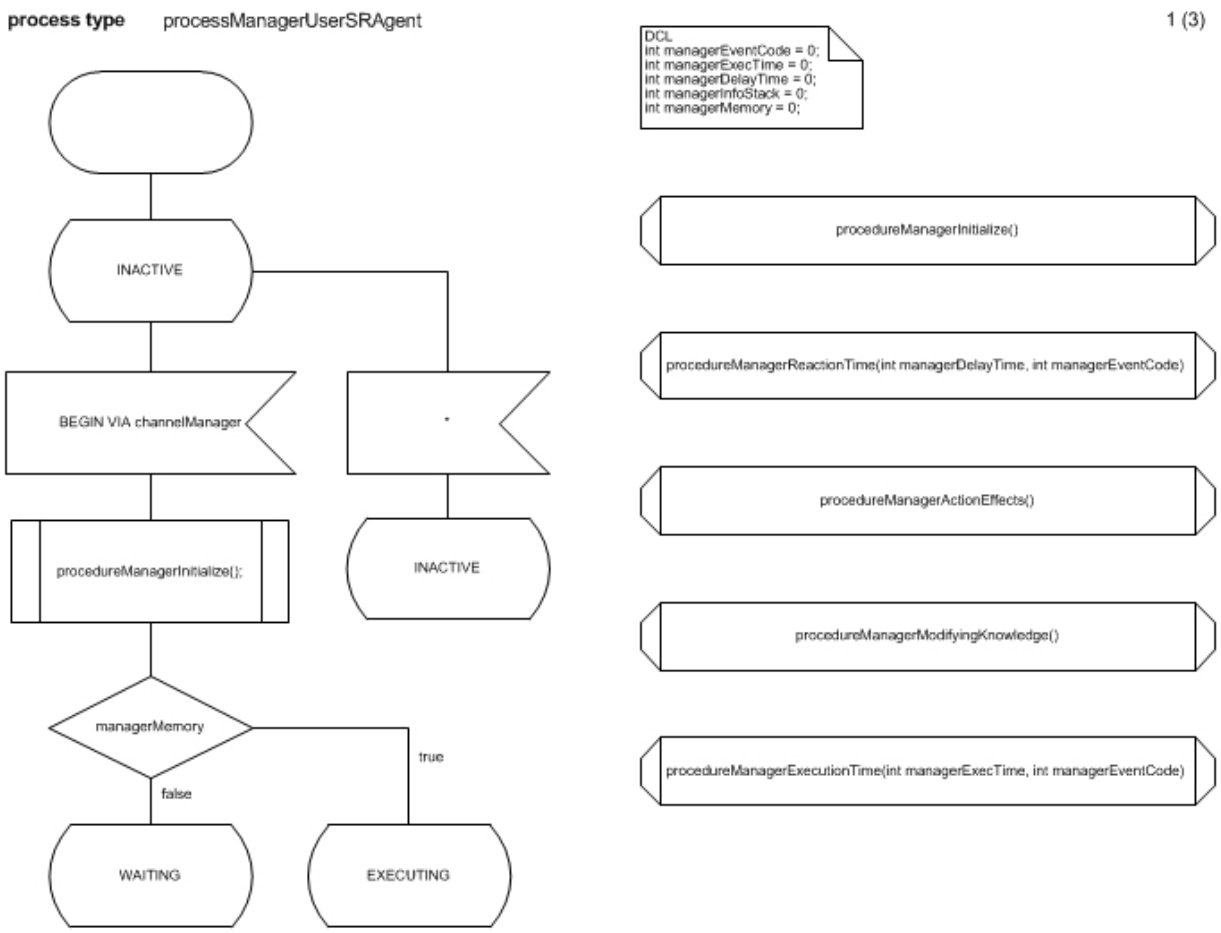


Ilustración 45: Diagrama del agente Gestor (1 de 3)

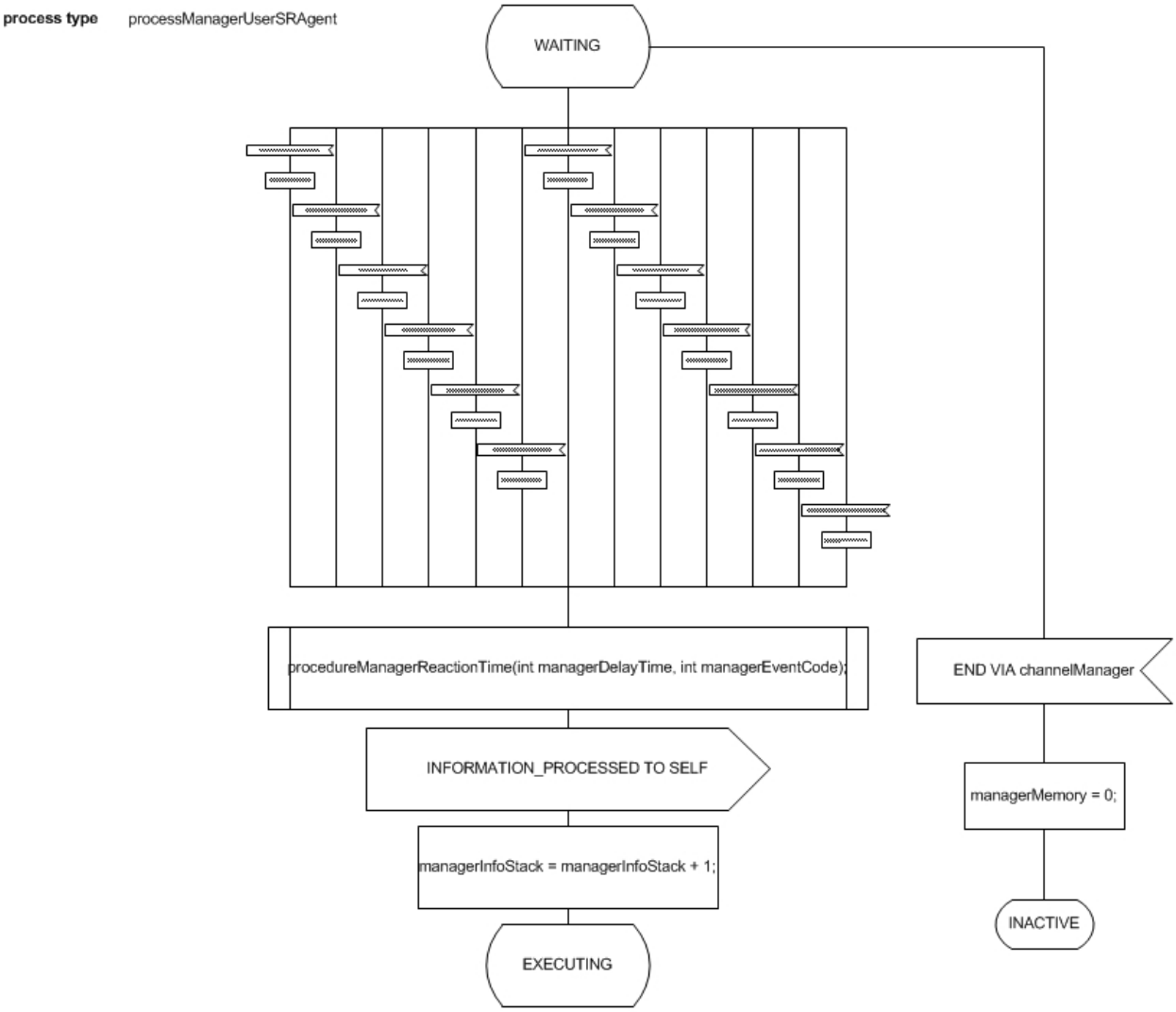


Ilustración 46: Diagrama del agente Gestor (2 de 3)

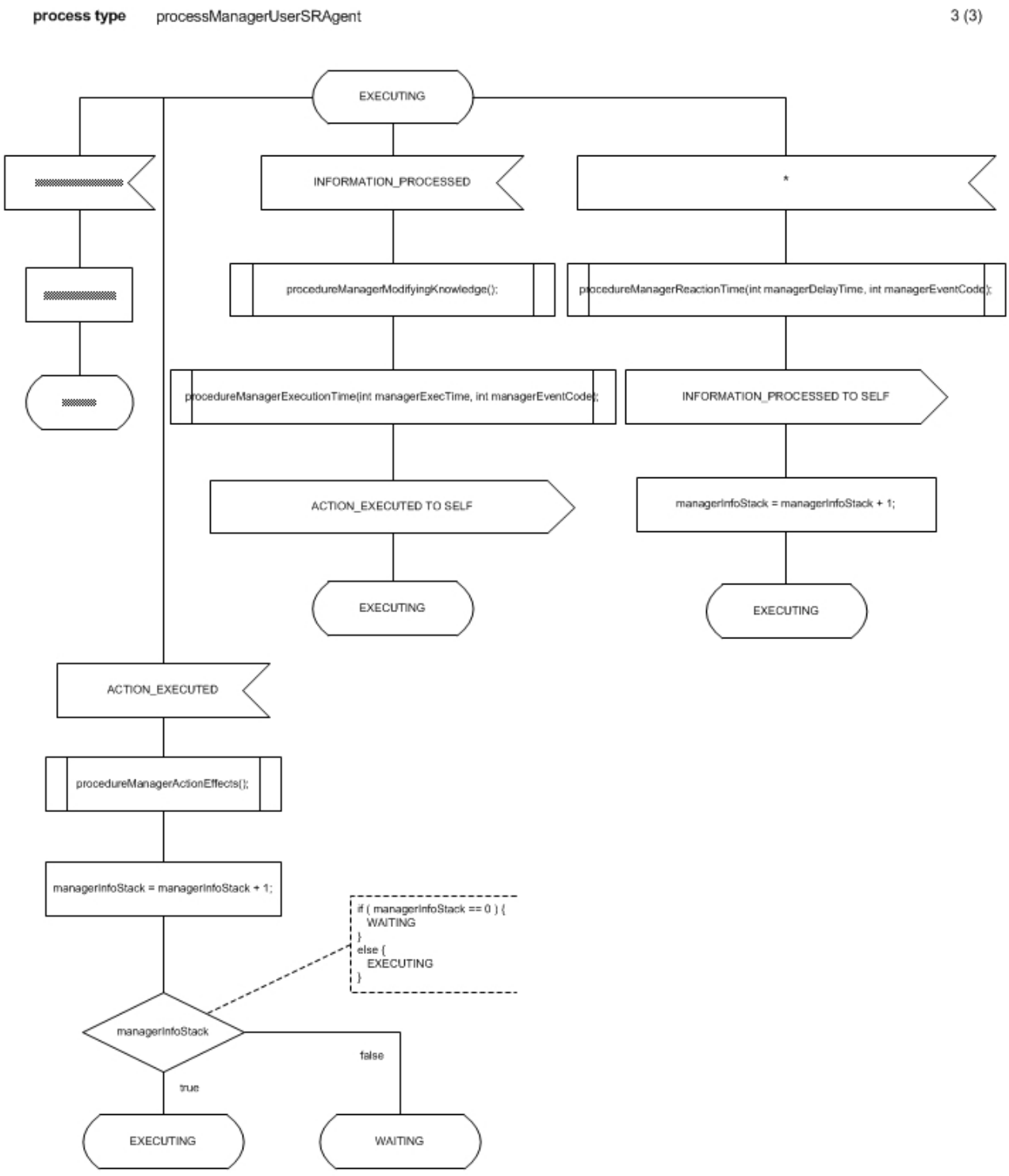


Ilustración 47: Diagrama del agente Gestor (3 de 3)

10.3.7 Agente Tutor

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- AdvisorEventCode
- AdvisorExecTime
- AdvisorDelayTime
- AdvisorInfoStack
- AdvisorMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureAdvisorInitialize()
- procedureAdvisorReactionTime(int AdvisorDelayTime, int AdvisorEventCode)
- procedureAdvisorActionEffects()
- procedureAdvisorModifyingKnowledge(),
- procedureAdvisorExecutionTime(int AdvisorExecTime, int AdvisorEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processAdvisorUserSRAgent.vsd.

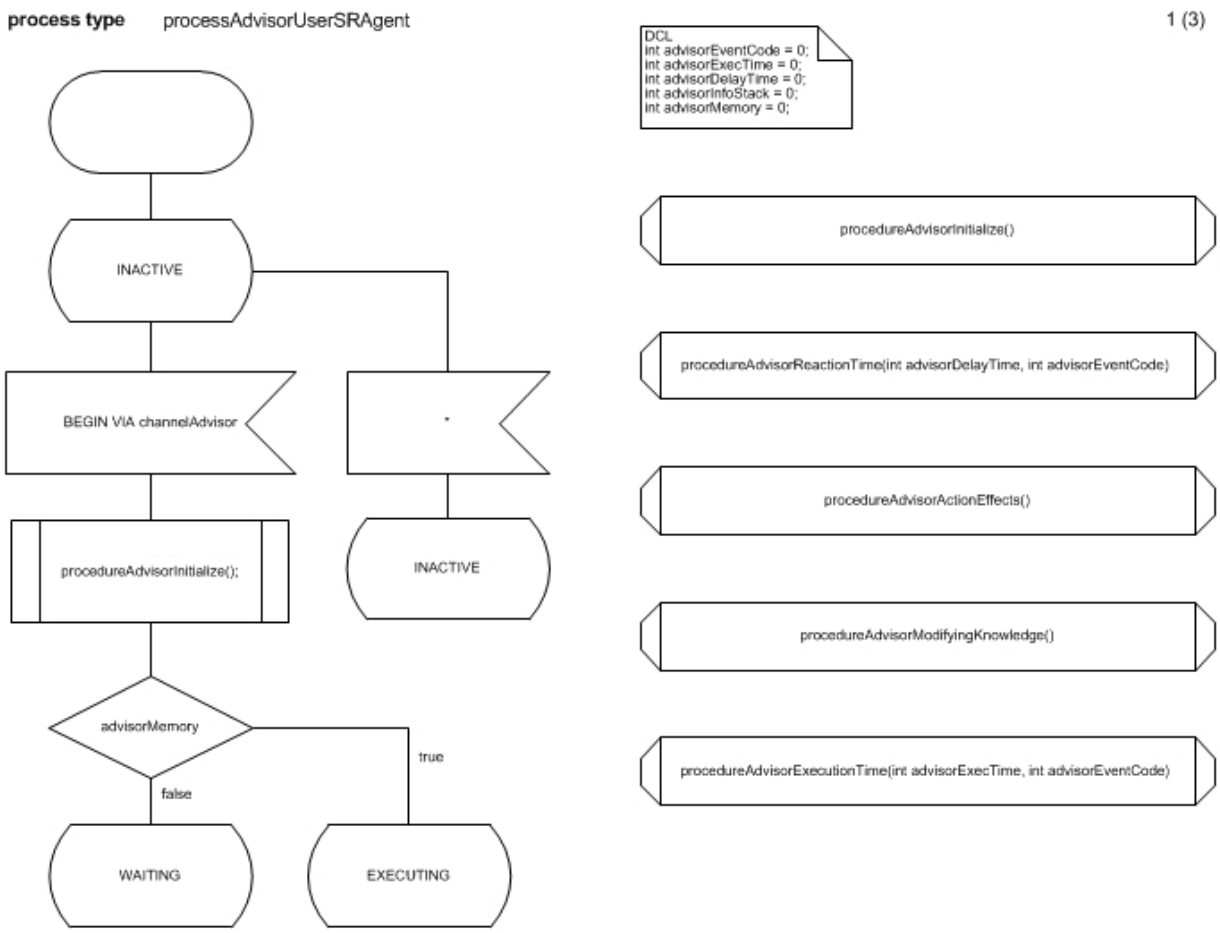


Ilustración 48: Diagrama del agente Tutor (1 de 3)

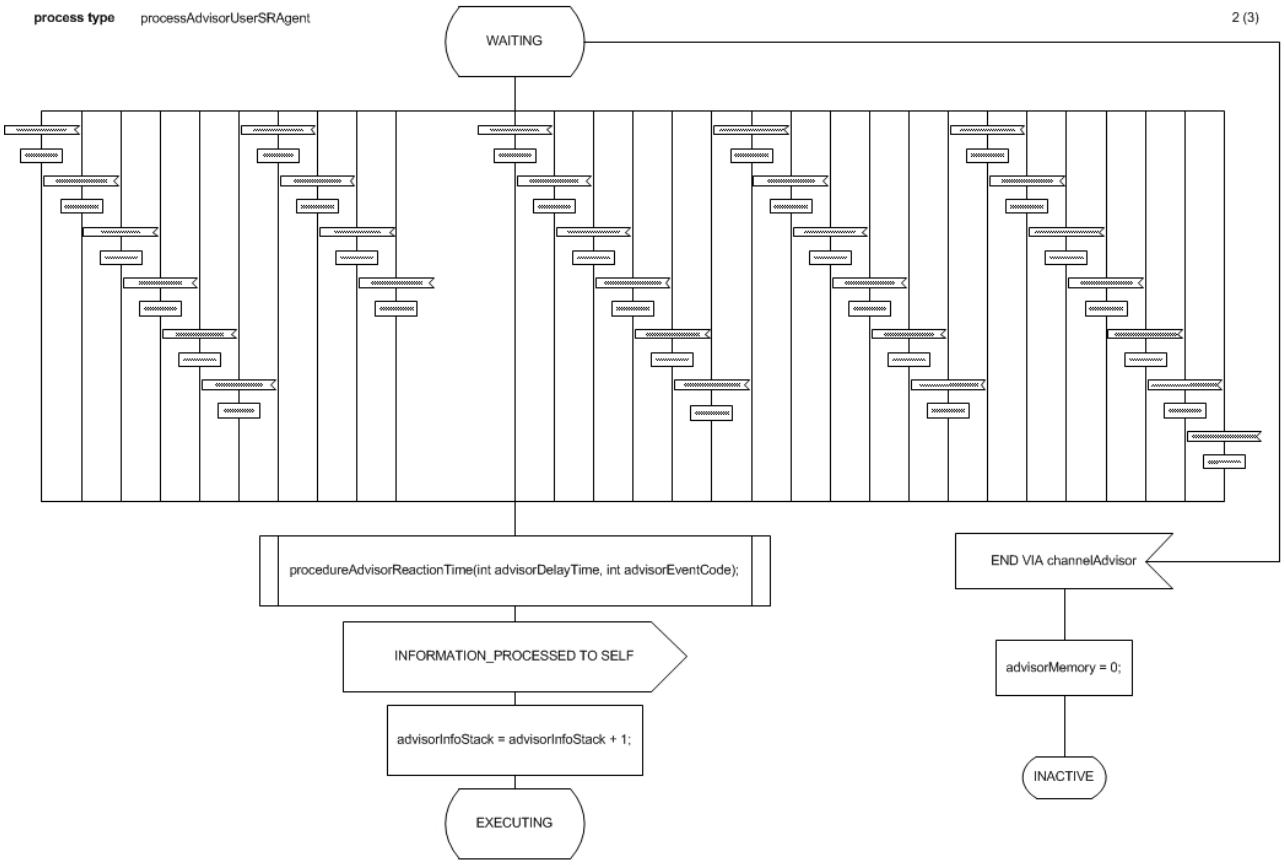


Ilustración 49: Diagrama del agente Tutor (2 de 3)

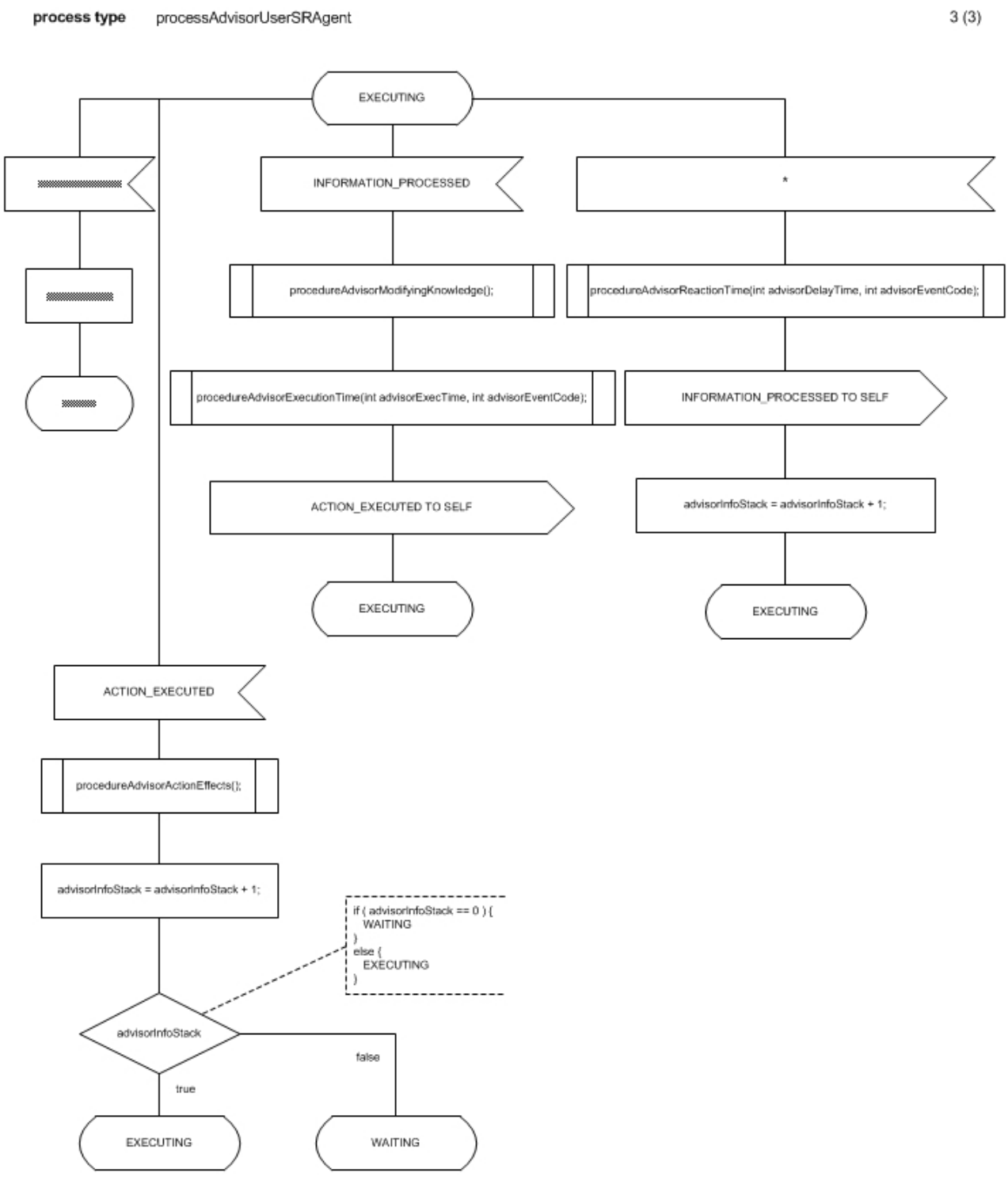


Ilustración 50: Diagrama del agente Tutor (3 de 3)

10.3.8 Agente Visitante

En este diagrama compuesto por 3 páginas. En la primera página se puede observar la inicialización del agente, la definición de las variables (todas de tipo entero) utilizadas:

- VisitorEventCode
- VisitorExecTime
- VisitorDelayTime
- VisitorInfoStack
- VisitorMemory

Y la definición de todos los procedimientos que se invocaran en el mismo:

- procedureVisitorInitialize()
- procedureVisitorReactionTime(int VisitorDelayTime, int VisitorEventCode)
- procedureVisitorActionEffects()
- procedureVisitorModifyingKnowledge(),
- procedureVisitorExecutionTime(int VisitorExecTime, int VisitorEventCode)

En la segunda página se encuentra definido el estado WAITING compuesto y en la tercera el estado EXECUTING del agente, ambos modelados mediante escenarios what-if. El fichero de este agente es processVisitorUserSRAgent.vsd.

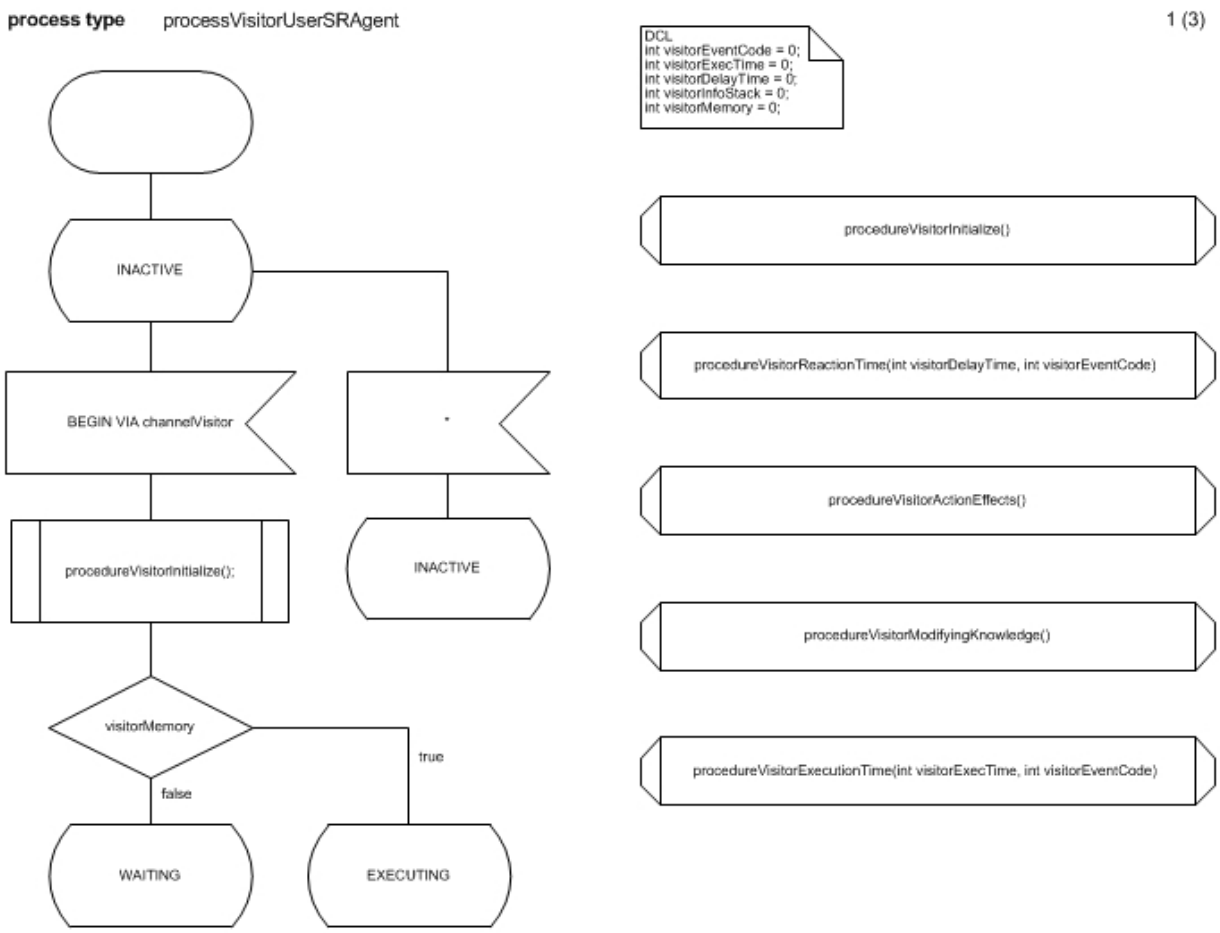


Ilustración 51: Diagrama del agente Visitante (1 de 3)

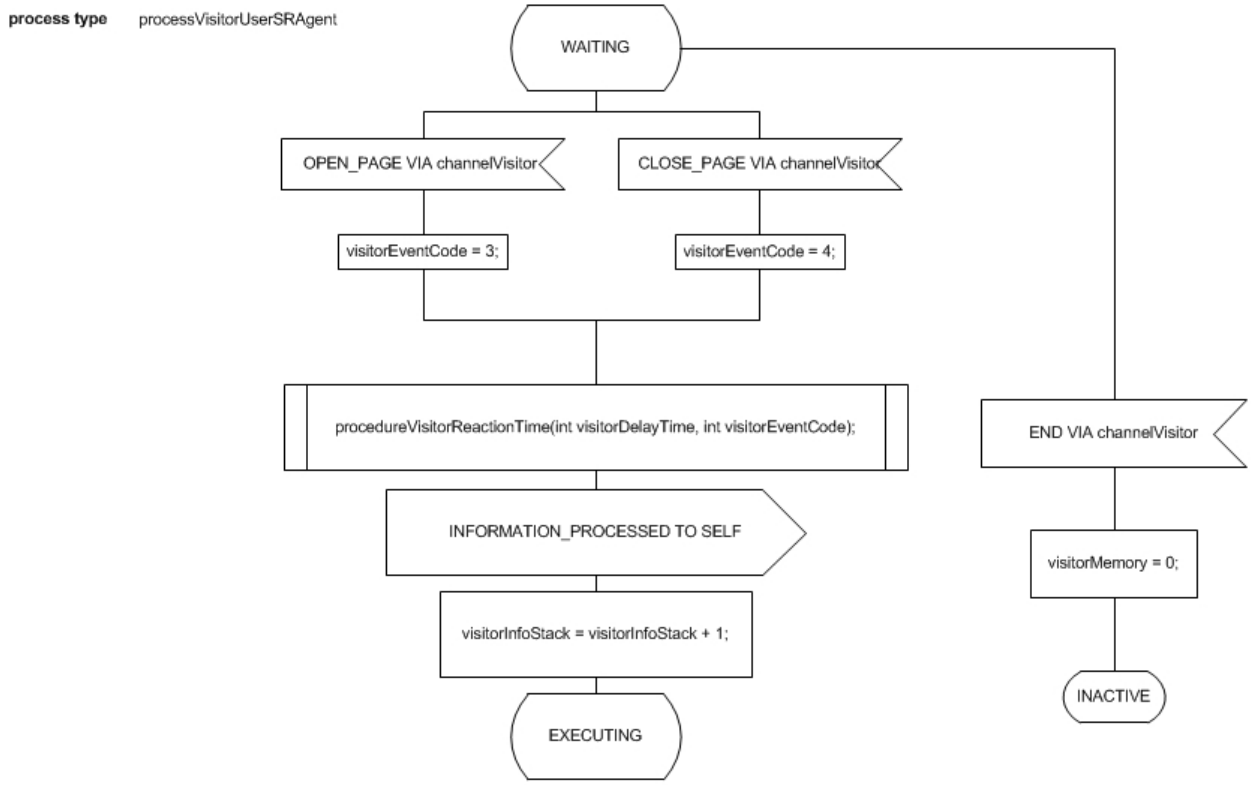


Ilustración 52: Diagrama del agente Visitante (2 de 3)

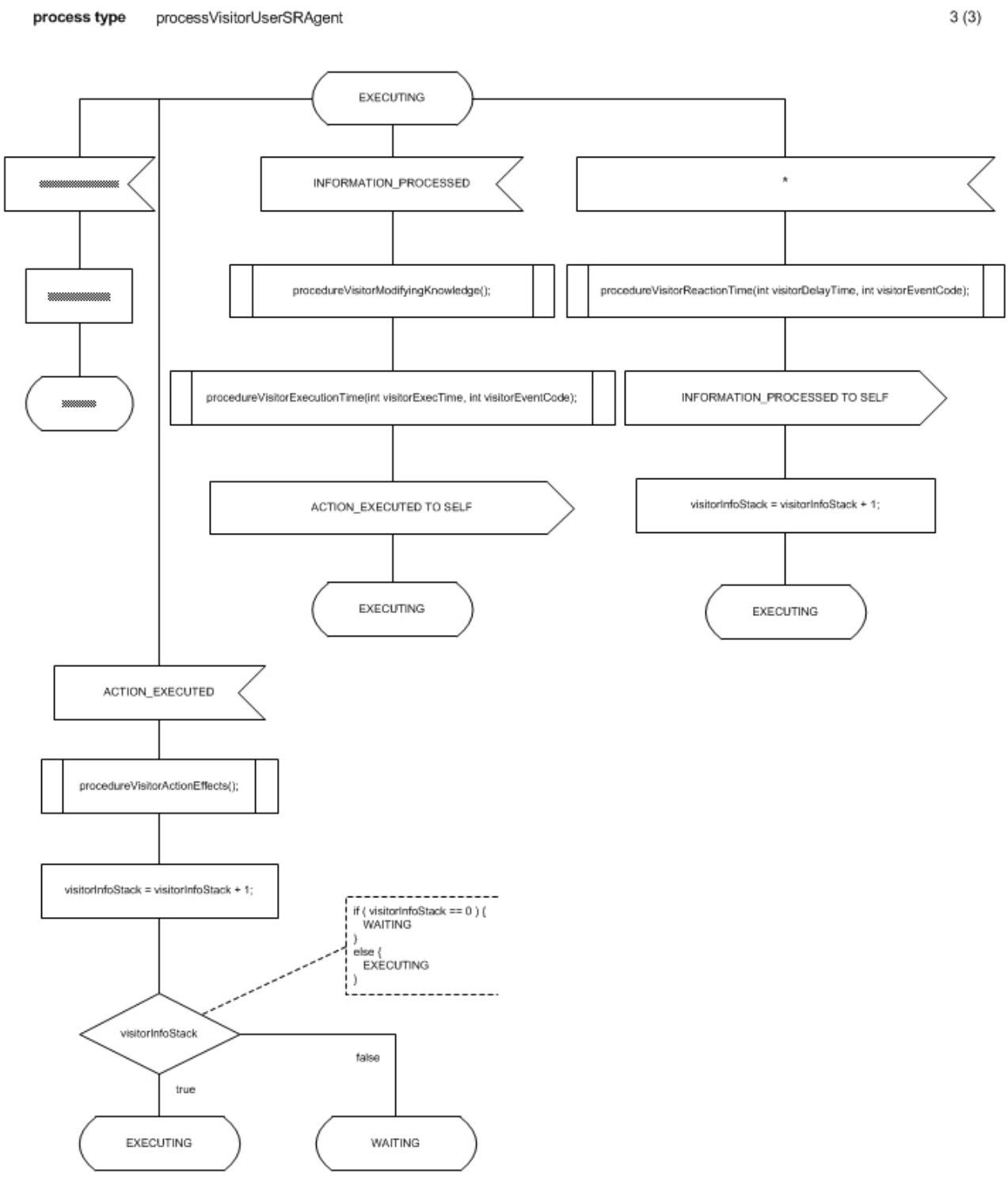


Ilustración 53: Diagrama del agente Visitante (3 de 3)

10.4 Procedimientos

10.4.1 Inicialización de un agente de tipo Estudiante

El archivo de este procedimiento es procedureStudentInitialize.vsd

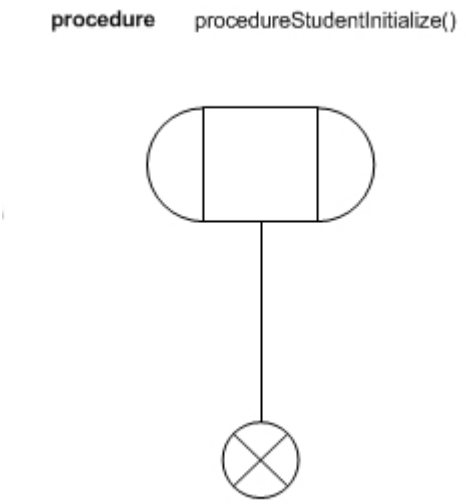


Ilustración 54: Diagrama del procedimiento de la Inicialización de un agente de tipo Estudiante

10.4.2 Efectos tras una accion de un agente de tipo Estudiante

El archivo de este procedimiento es procedureStudentActionEffects.vsd

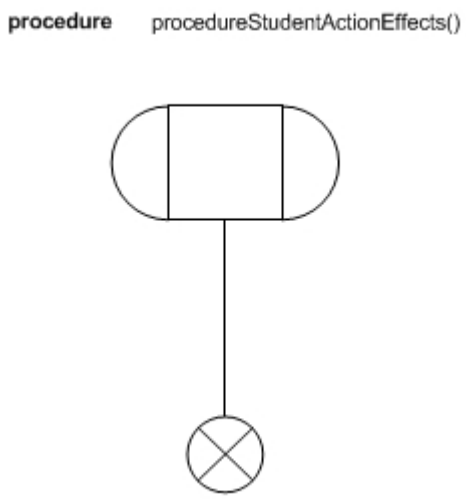


Ilustración 55: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Estudiante

10.4.3 Modificacion del conocimiento de un agente de tipo Estudiante

El archivo de este procedimiento es procedureStudentModifyingKnowledge.vsd

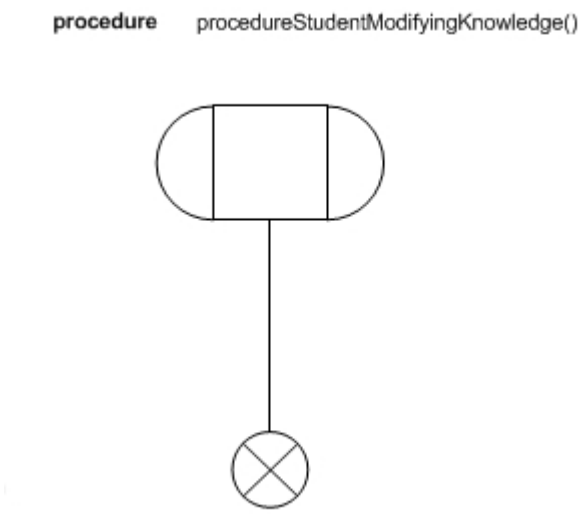


Ilustración 56: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Estudiante

10.4.4 Tiempo de ejecucion de un agente de tipo Estudiante

El archivo de este procedimiento es procedureStudentExecutionTime.vsd

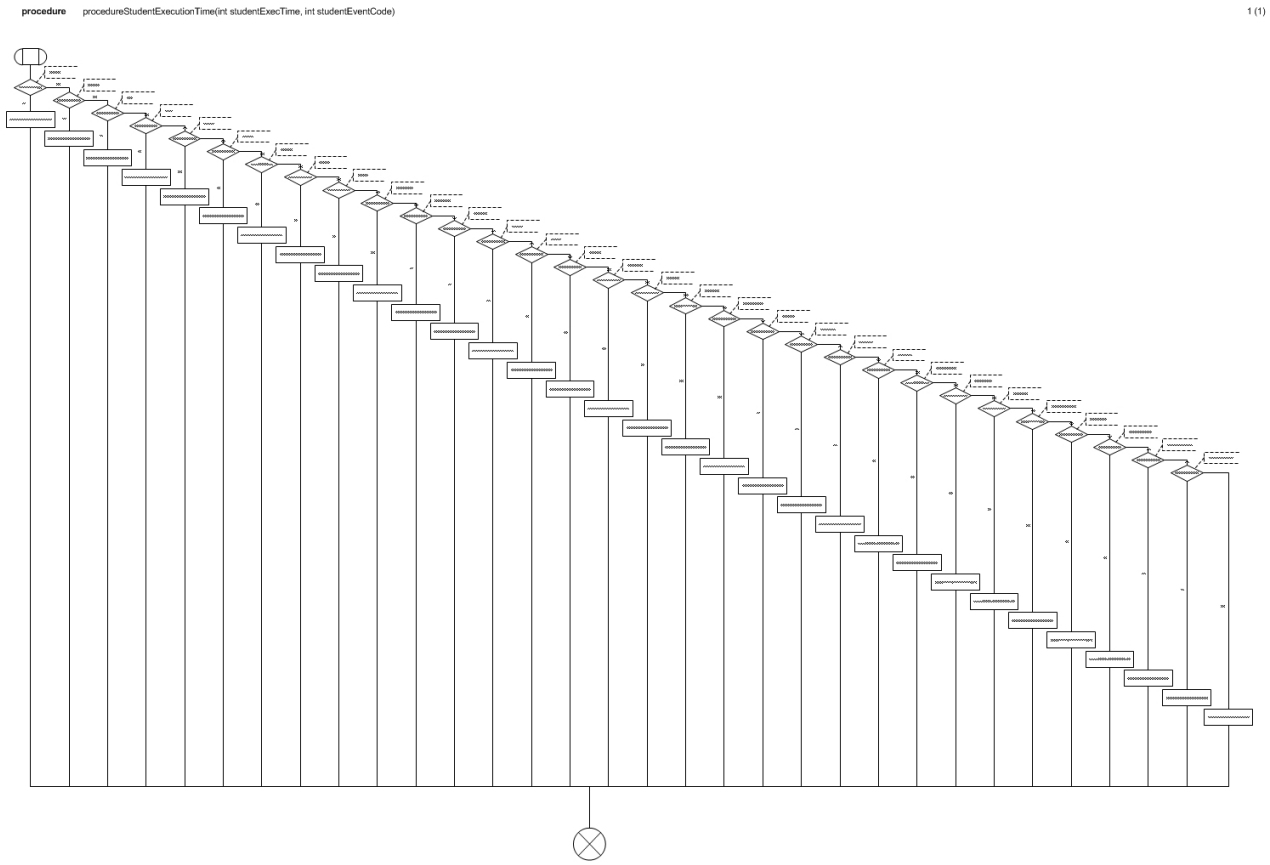


Ilustración 57: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Estudiante (1 de 2)

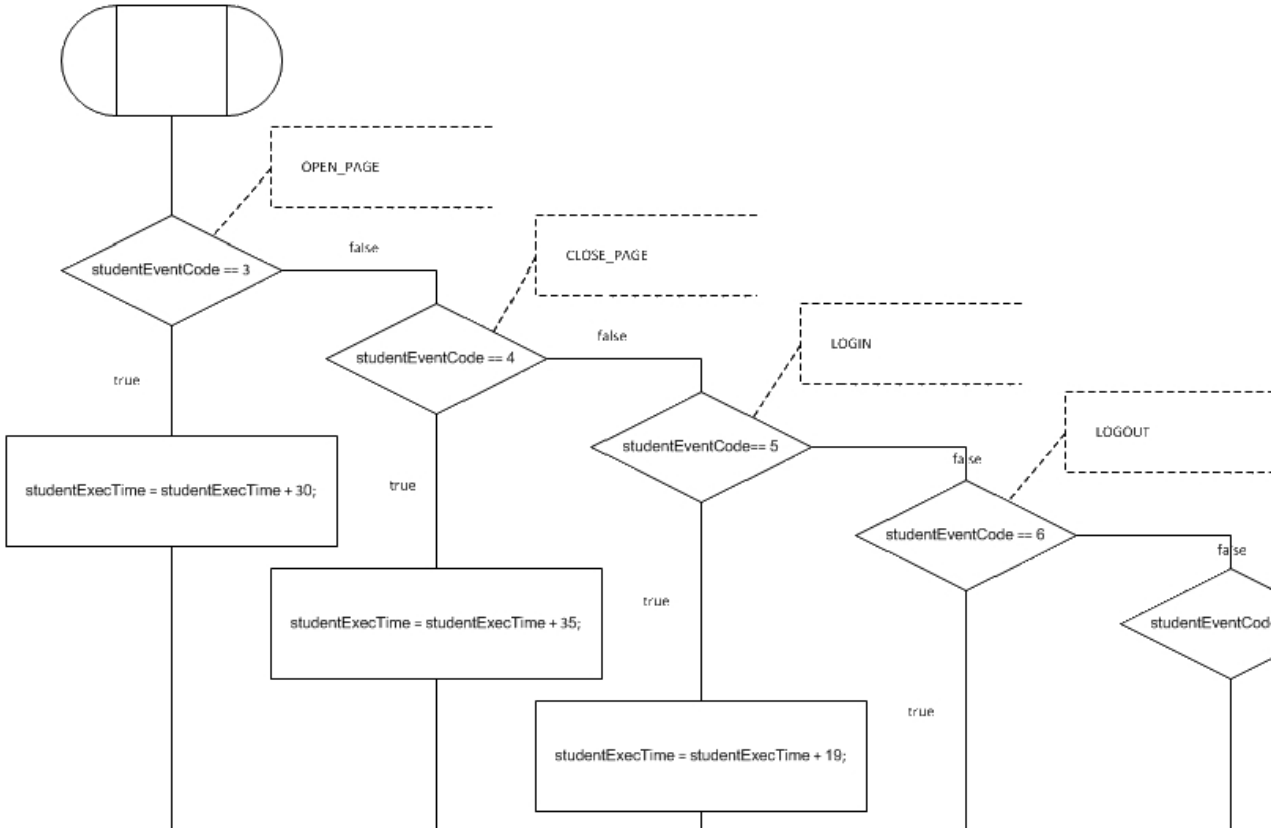


Ilustración 58: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Estudiante (2 de 2)

10.4.5 Tiempo de reaccion de un agente de tipo Estudiante

El archivo de este procedimiento es procedureStudentReactionTime.vsd

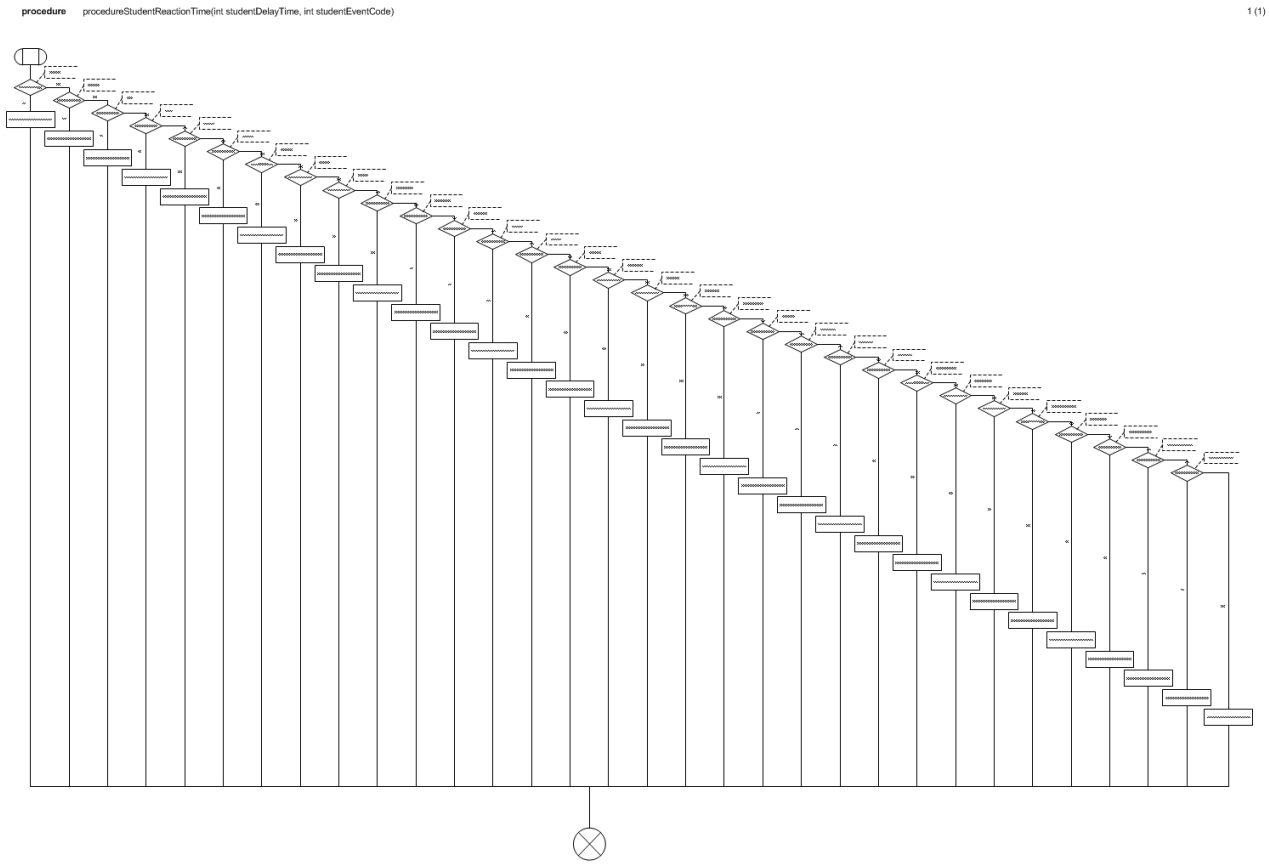


Ilustración 59: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Estudiante (1 de 2)

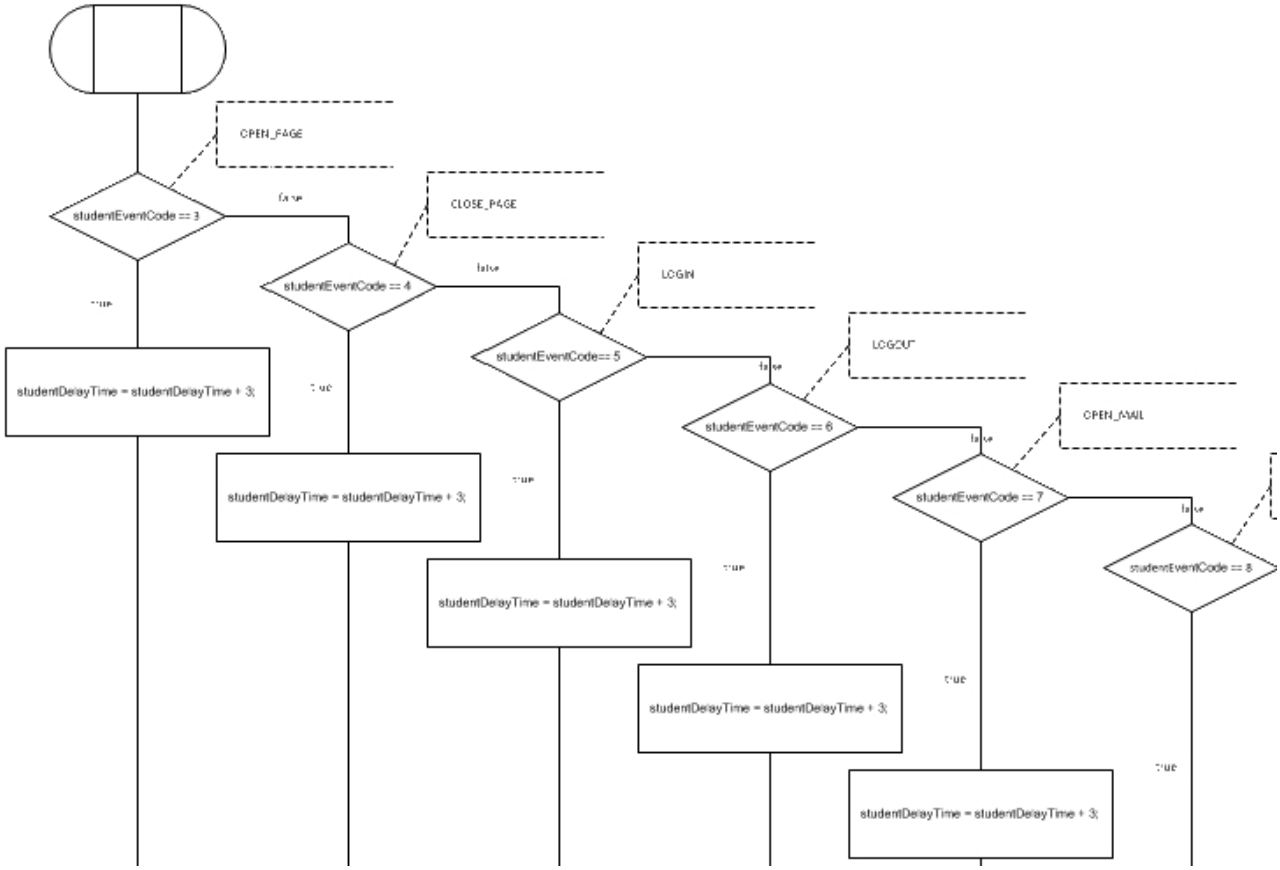


Ilustración 60: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Estudiante (2 de 2)

10.4.6 Inicialización de un agente de tipo Profesor

El archivo de este procedimiento es procedureTeacherInitialize.vsd

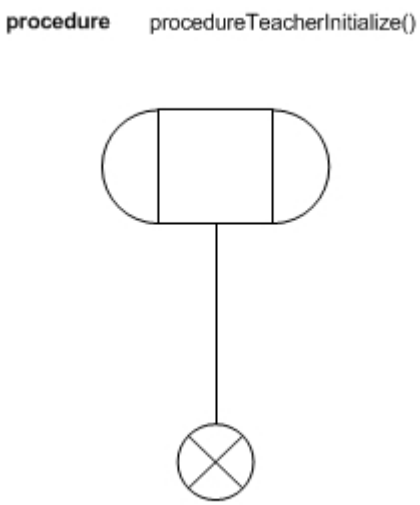


Ilustración 61: Diagrama del procedimiento de la Inicialización de un agente de tipo Profesor

10.4.7 Efectos tras una accion de un agente de tipo Profesor

El archivo de este procedimiento es procedureTeacherActionEffects.vsd

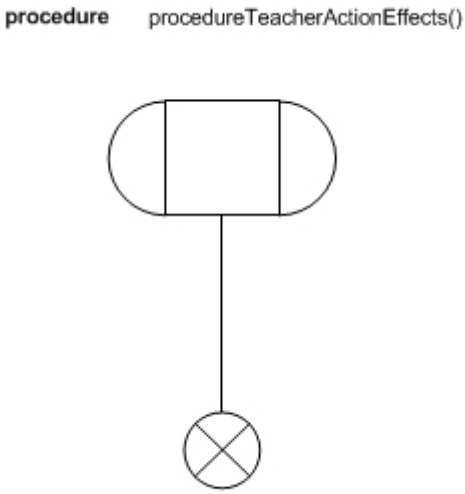


Ilustración 62: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Profesor

10.4.8 Modificacion del conocimiento de un agente de tipo Profesor

El archivo de este procedimiento es procedureTeacherModifyingKnowledge.vsd

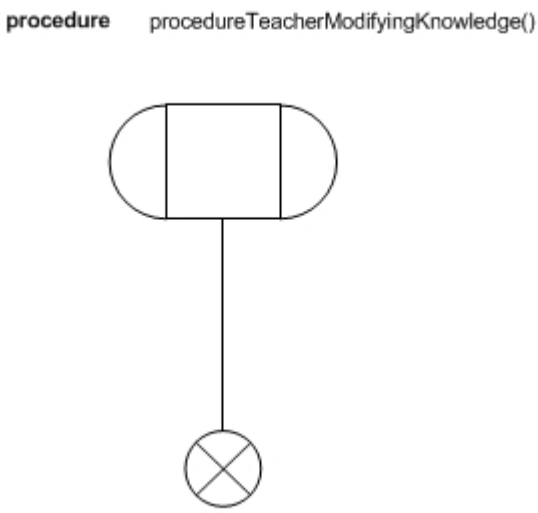


Ilustración 63: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Profesor

10.4.9 Tiempo de ejecucion de un agente de tipo Profesor

El archivo de este procedimiento es procedureTeacherExecutionTime.vsd

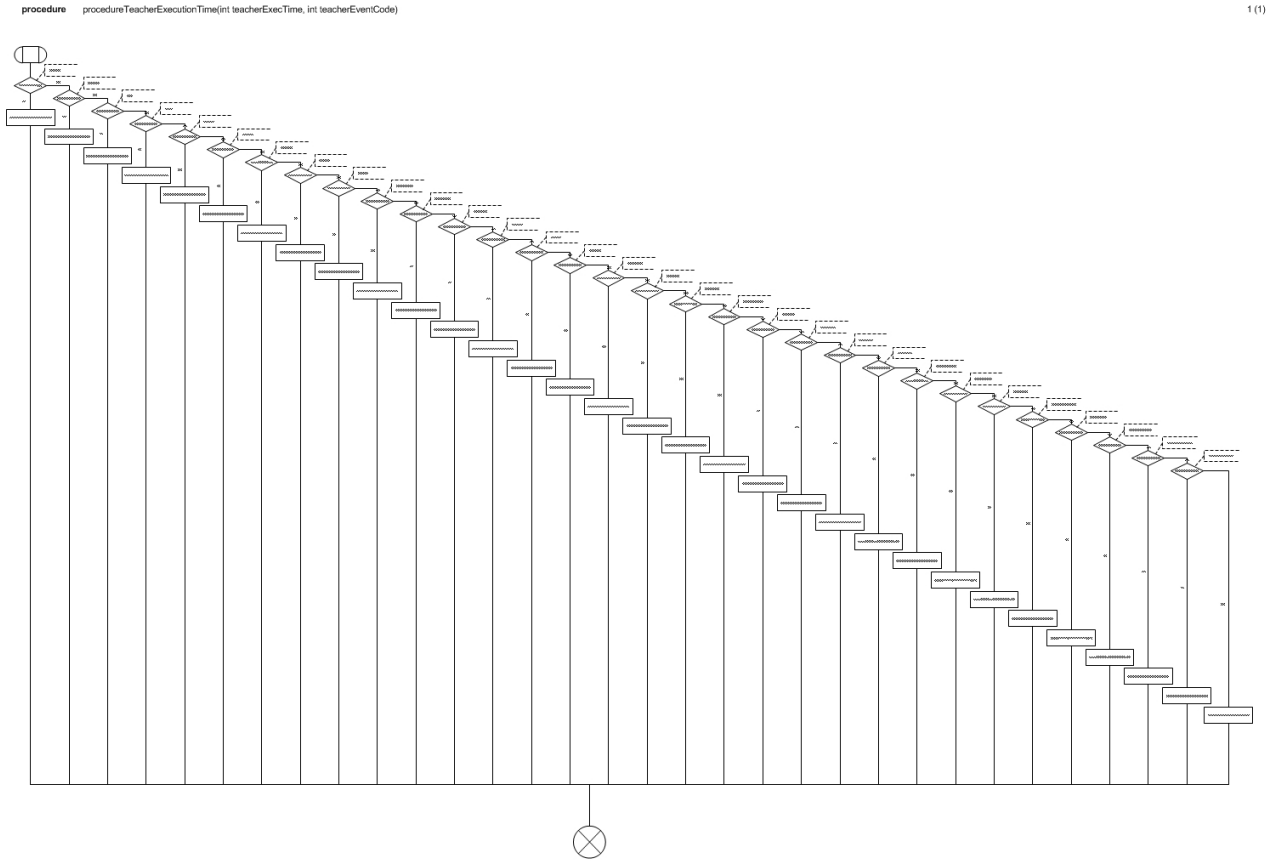


Ilustración 64: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Profesor

10.4.10 Tiempo de reaccion de un agente de tipo Profesor

El archivo de este procedimiento es procedureTeacherReactionTime.vsd

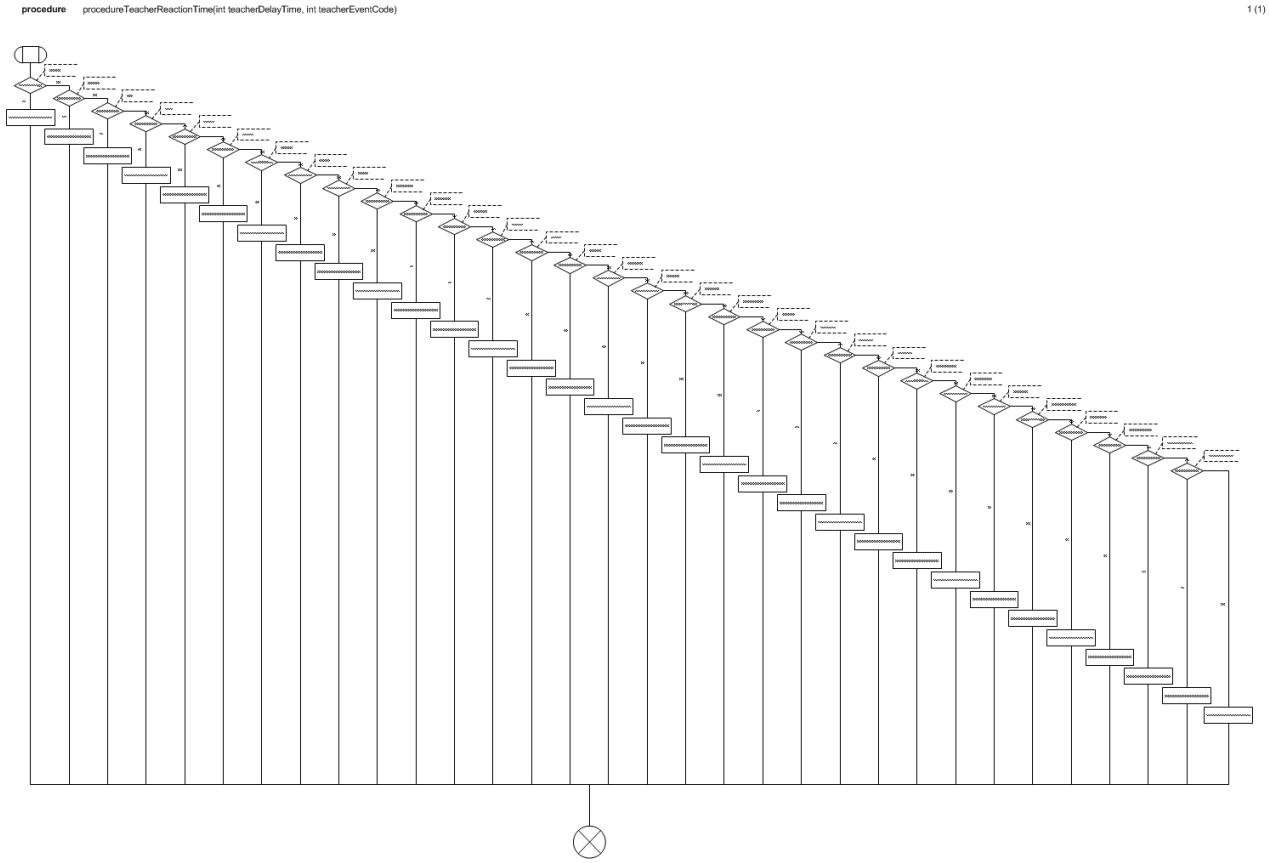


Ilustración 65: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Profeso

10.4.11 Inicialización de un agente de tipo Consultor

El archivo de este procedimiento es procedureConsultantInitialize.vsd

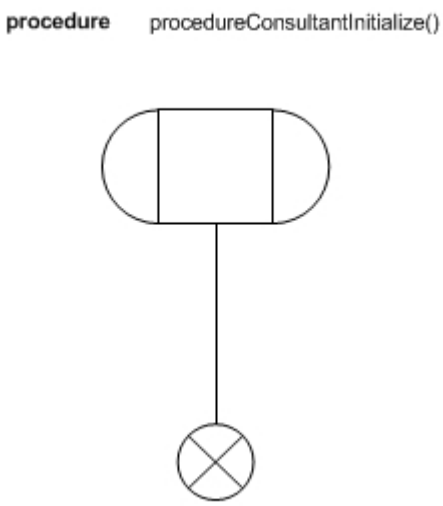


Ilustración 66: Diagrama del procedimiento de la Inicialización de un agente de tipo Consultor

10.4.12 Efectos tras una accion de un agente de tipo Consultor

El archivo de este procedimiento es procedureConsultantActionEffects.vsd

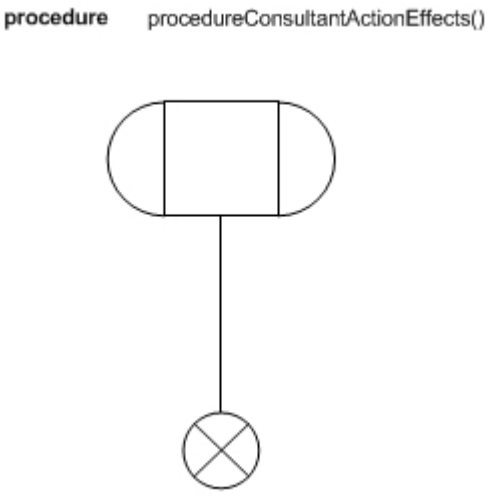


Ilustración 67: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Consultor

10.4.13 Modificacion del conocimiento de un agente de tipo Consultor

El archivo de este procedimiento es procedureConsultantModifyingKnowledge.vsd

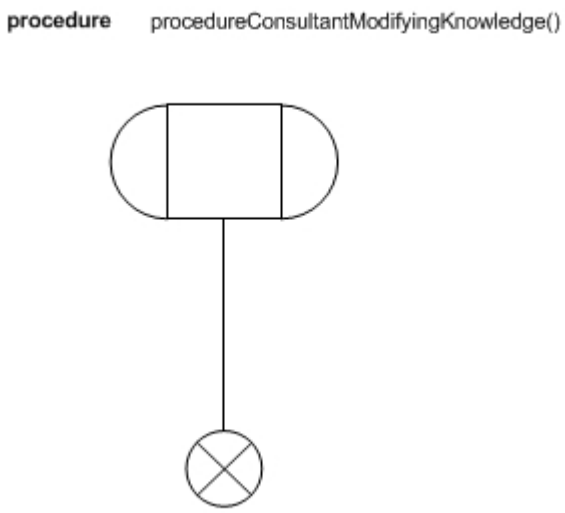


Ilustración 68: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Consultor

10.4.14 Tiempo de ejecucion de un agente de tipo Consultor

El archivo de este procedimiento es procedureConsultantExecutionTime.vsd

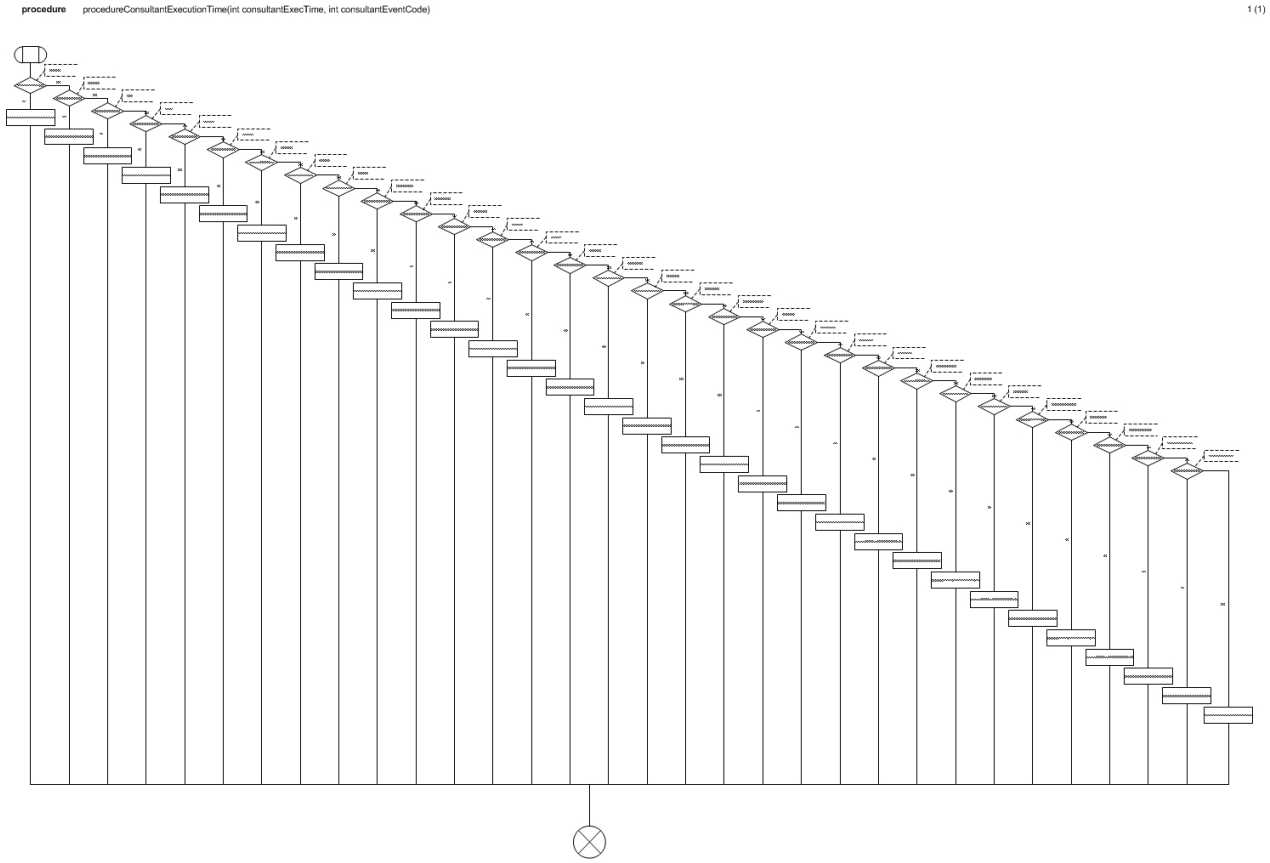


Ilustración 69: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Consultor

10.4.15 Tiempo de reaccion de un agente de tipo Consultor

El archivo de este procedimiento es procedureConsultantReactionTime.vsd

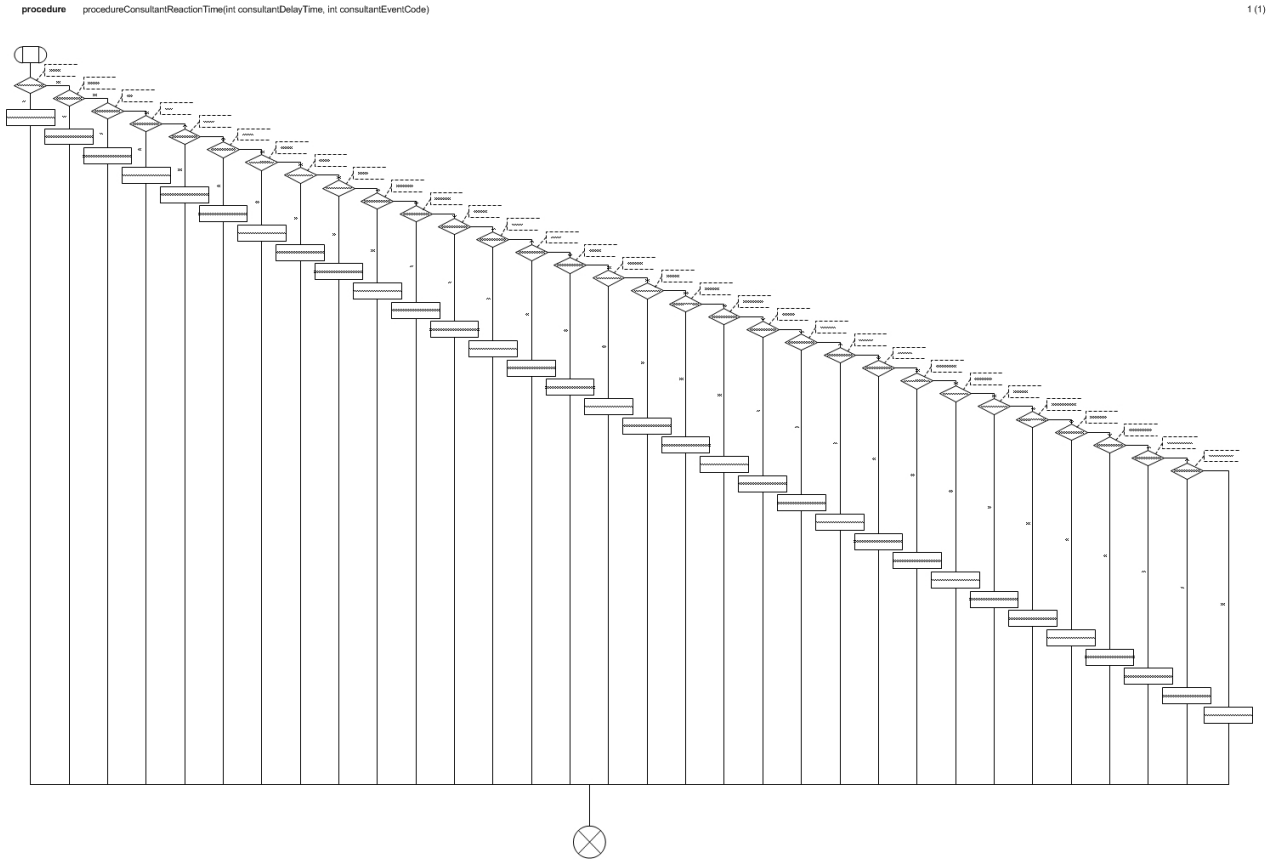


Ilustración 70: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Consultor

10.4.16 Inicialización de un agente de tipo Alumni

El archivo de este procedimiento es procedureAlumniInitialize.vsd

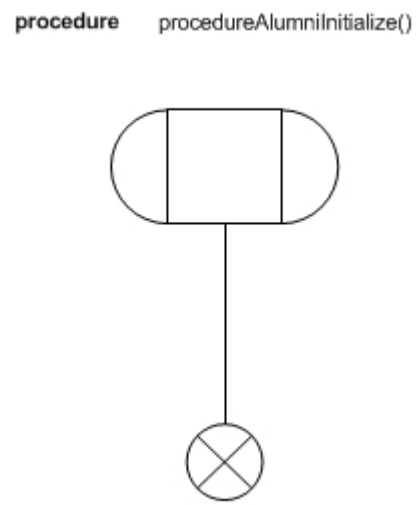


Ilustración 71: Diagrama del procedimiento de la Inicialización de un agente de tipo Alumni

10.4.17 Efectos tras una accion de un agente de tipo Alumni

El archivo de este procedimiento es procedureAlumniActionEffects.vsd

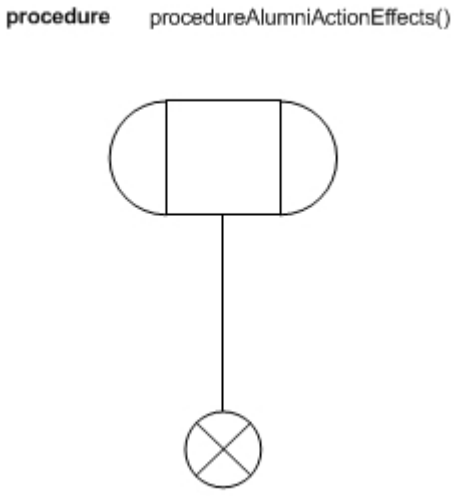


Ilustración 72: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Alumni

10.4.18 Modificacion del conocimiento de un agente de tipo Alumni

El archivo de este procedimiento es procedureAlumniModifyingKnowledge.vsd

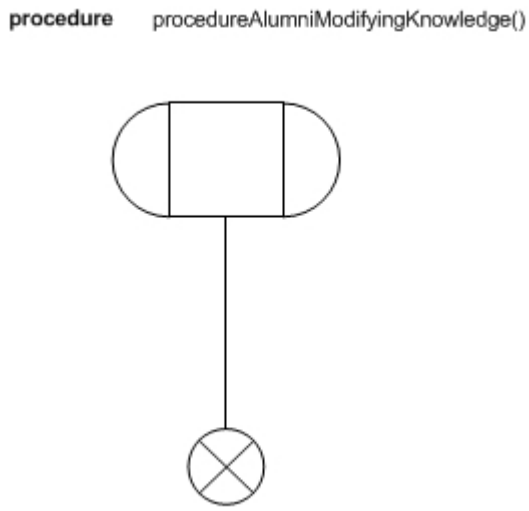


Ilustración 73: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Alumni

10.4.19 Tiempo de ejecucion de un agente de tipo Alumni

El archivo de este procedimiento es procedureAlumniExecutionTime.vsd

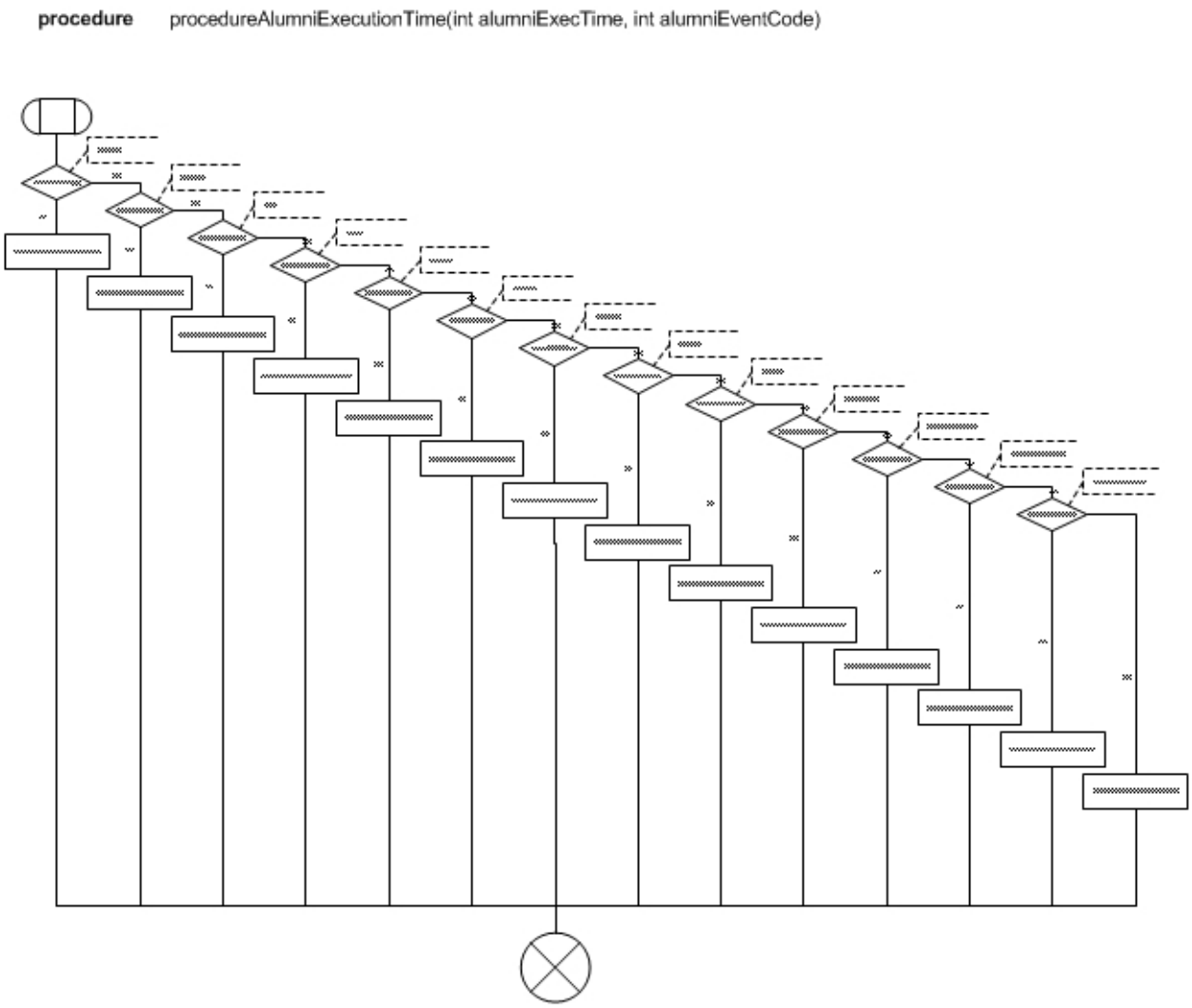


Ilustración 74: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Alumni

10.4.20 Tiempo de reaccion de un agente de tipo Alumni

El archivo de este procedimiento es procedureAlumniReactionTime.vsd

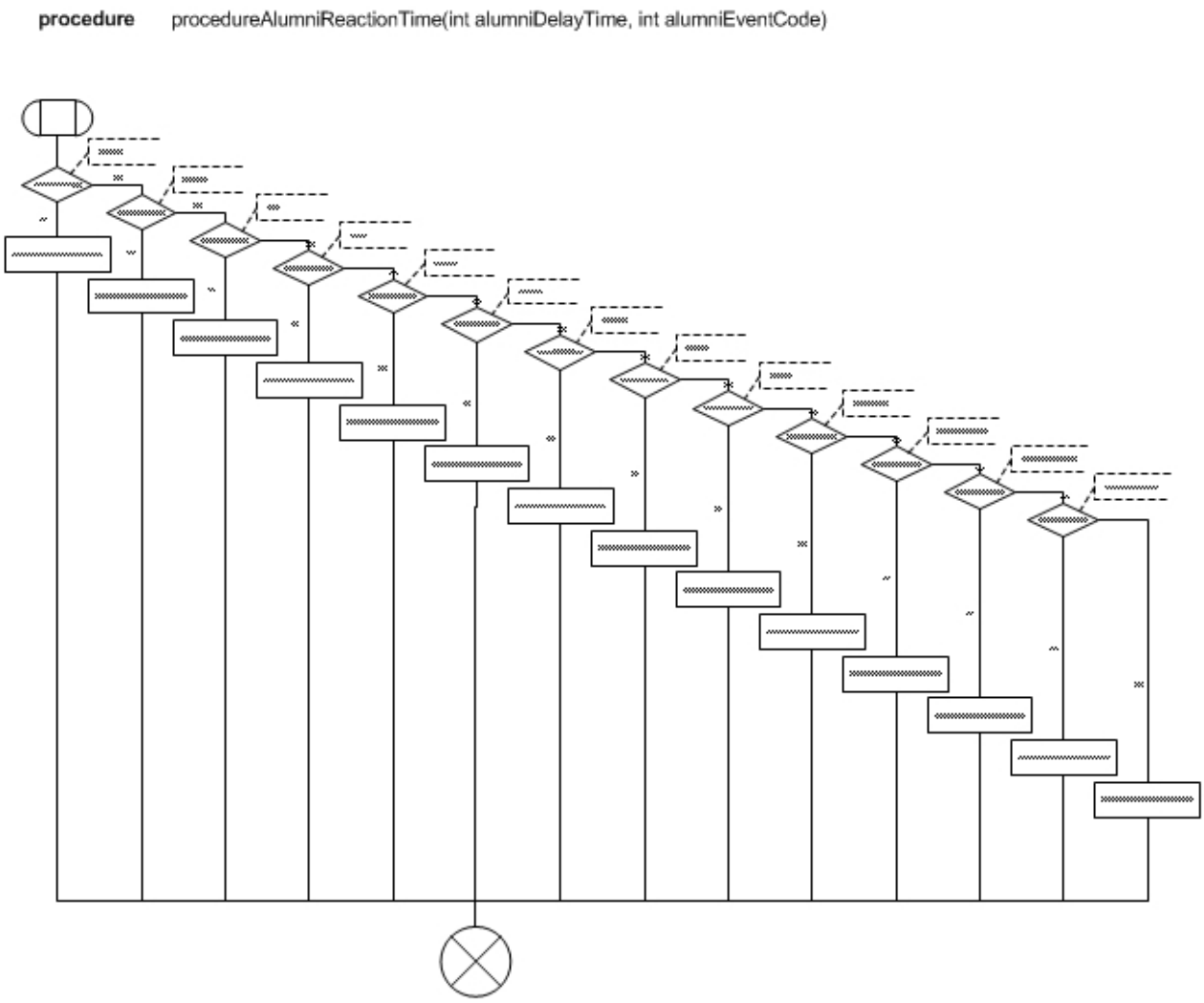


Ilustración 75: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Alumni

10.4.21 Inicialización de un agente de tipo Administrador

El archivo de este procedimiento es procedureAdministratorInitialize.vsd

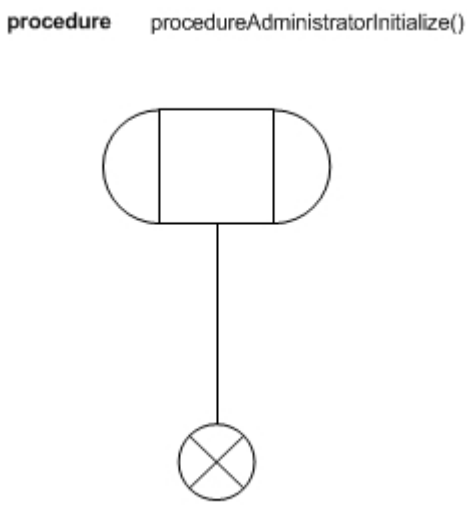


Ilustración 76: Diagrama del procedimiento de la Inicialización de un agente de tipo Administrador

10.4.22 Efectos tras una accion de un agente de tipo Administrador

El archivo de este procedimiento es procedureAdministratorActionEffects.vsd

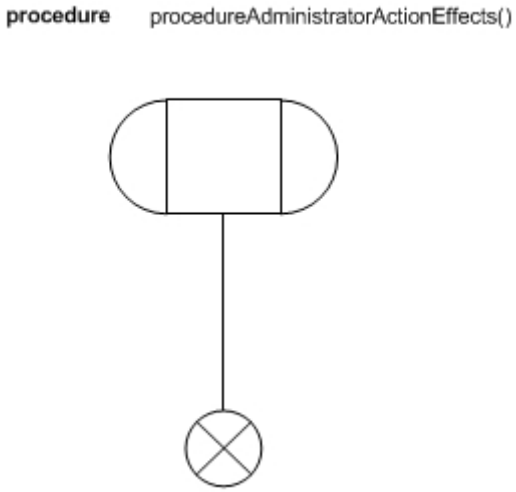


Ilustración 77: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Administrador

10.4.23 Modificacion del conocimiento de un agente de tipo Administrador

El archivo de este procedimiento es procedureAdministratorModifyingKnowledge.vsd

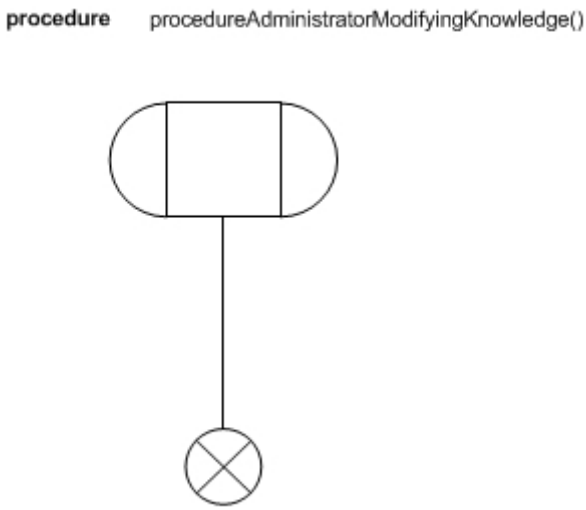


Ilustración 78: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Administrador

10.4.24 Tiempo de ejecucion de un agente de tipo Administrador

El archivo de este procedimiento es procedureAdministratorExecutionTime.vsd

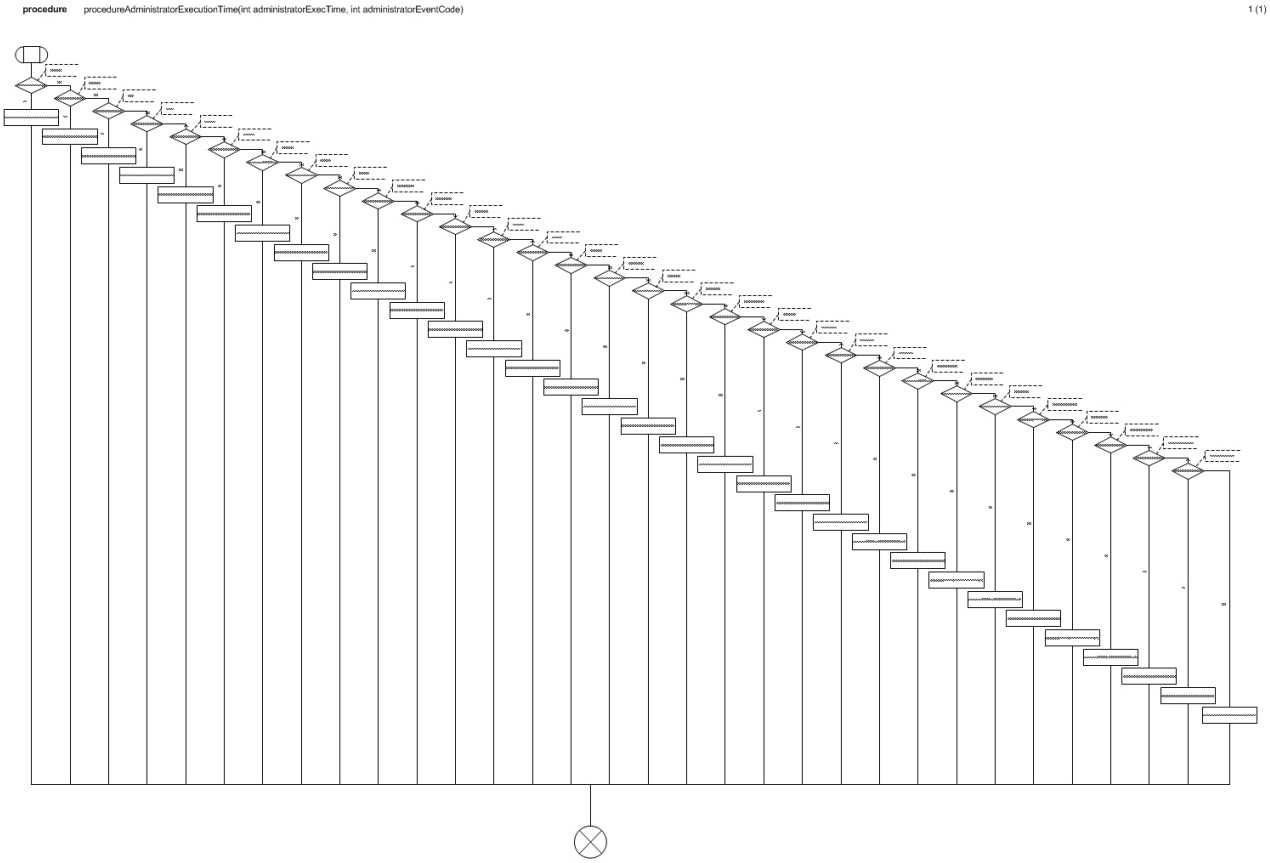


Ilustración 79: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Administrador

10.4.25 Tiempo de reaccion de un agente de tipo Administrador

El archivo de este procedimiento es procedureAdministratorReactionTime.vsd

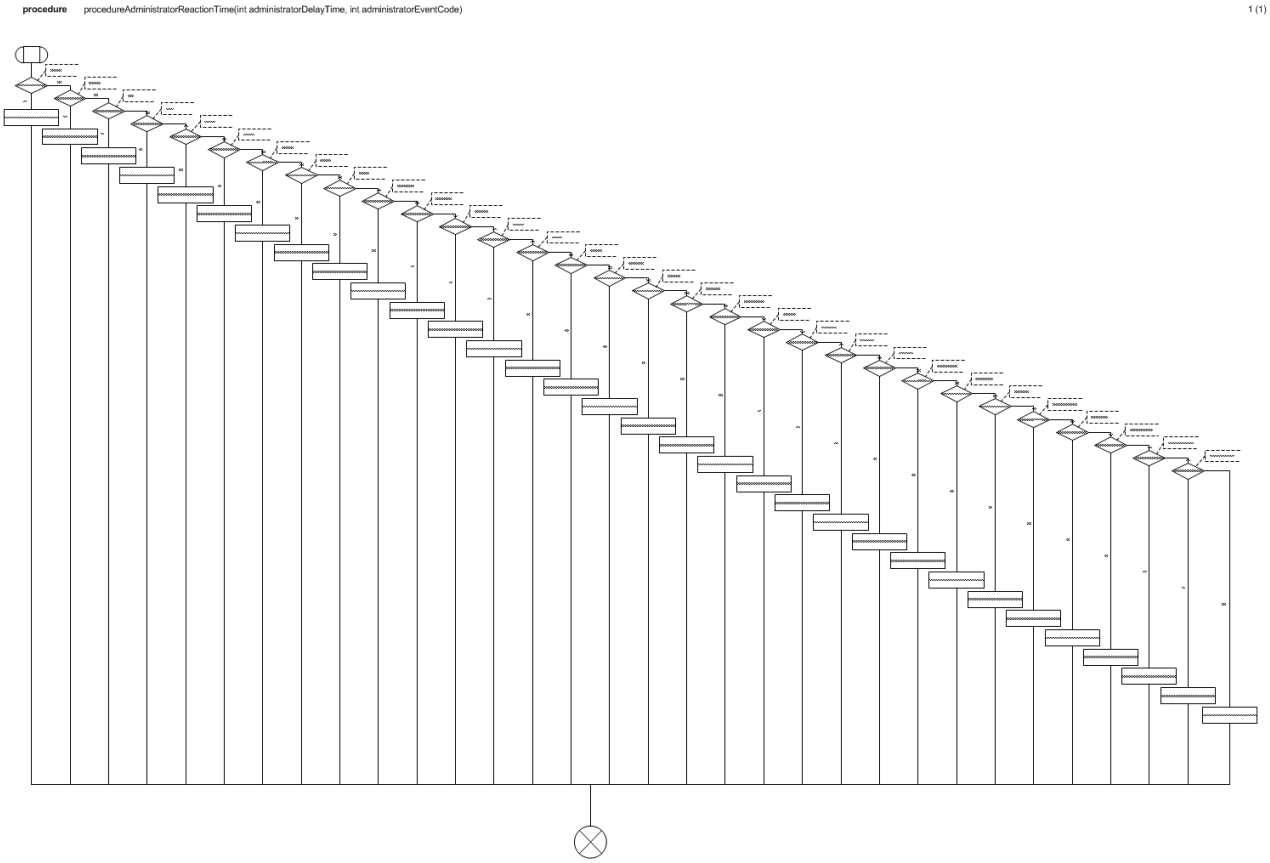


Ilustración 80: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Administrador

10.4.26 Inicialización de un agente de tipo Gestor

El archivo de este procedimiento es procedureManagerInitialize.vsd

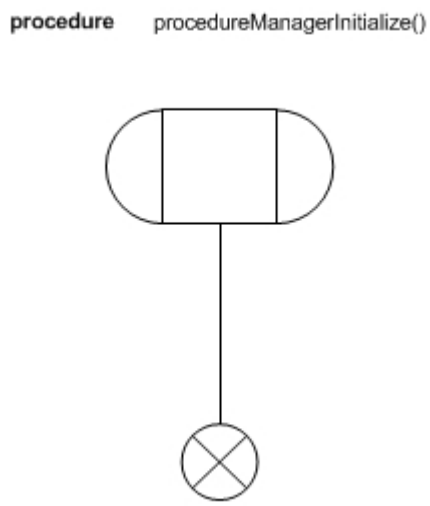


Ilustración 81: Diagrama del procedimiento de la Inicialización de un agente de tipo Gestor

10.4.27 Efectos tras una accion de un agente de tipo Gestor

El archivo de este procedimiento es procedureManagerActionEffects.vsd

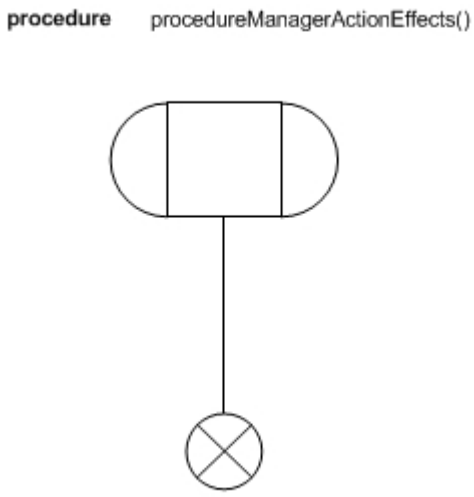


Ilustración 82: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Gestor

10.4.28 Modificacion del conocimiento de un agente de tipo Gestor

El archivo de este procedimiento es procedureManagerModifyingKnowledge.vsd

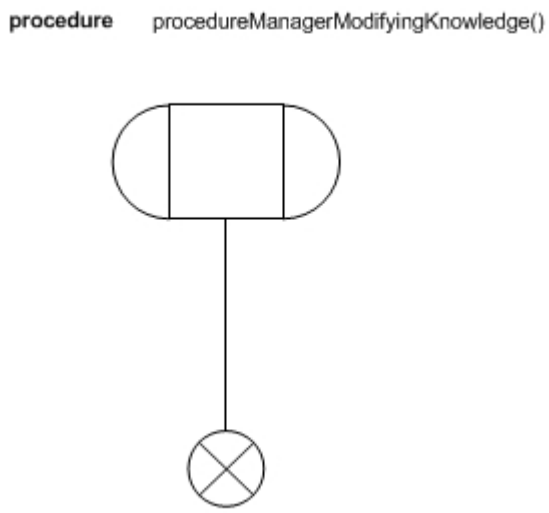


Ilustración 83: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Gestor

10.4.29 Tiempo de ejecucion de un agente de tipo Gestor

El archivo de este procedimiento es procedureManagerExecutionTime.vsd

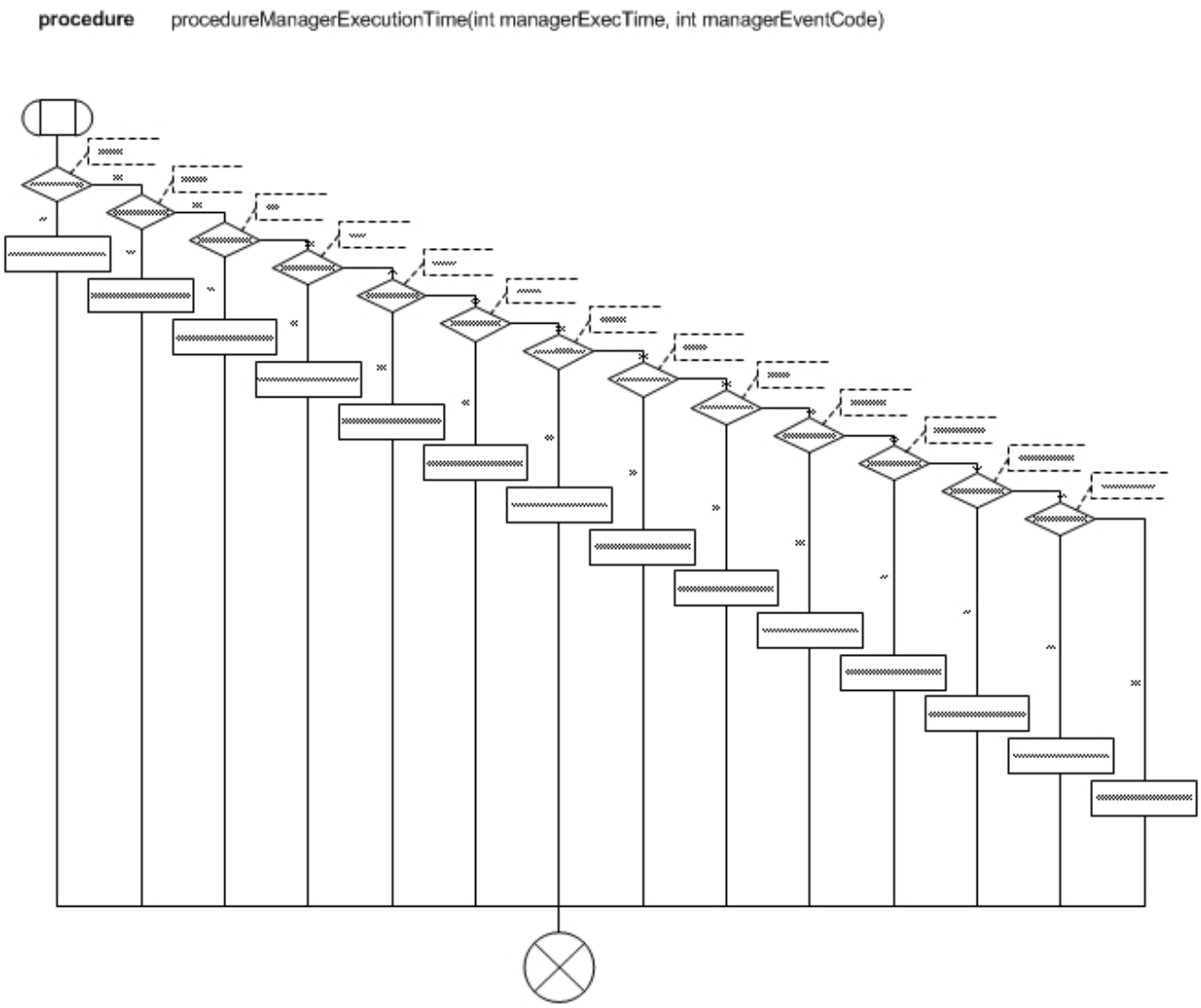


Ilustración 84: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Gestor

10.4.30 Tiempo de reaccion de un agente de tipo Gestor

El archivo de este procedimiento es procedureManagerReactionTime.vsd

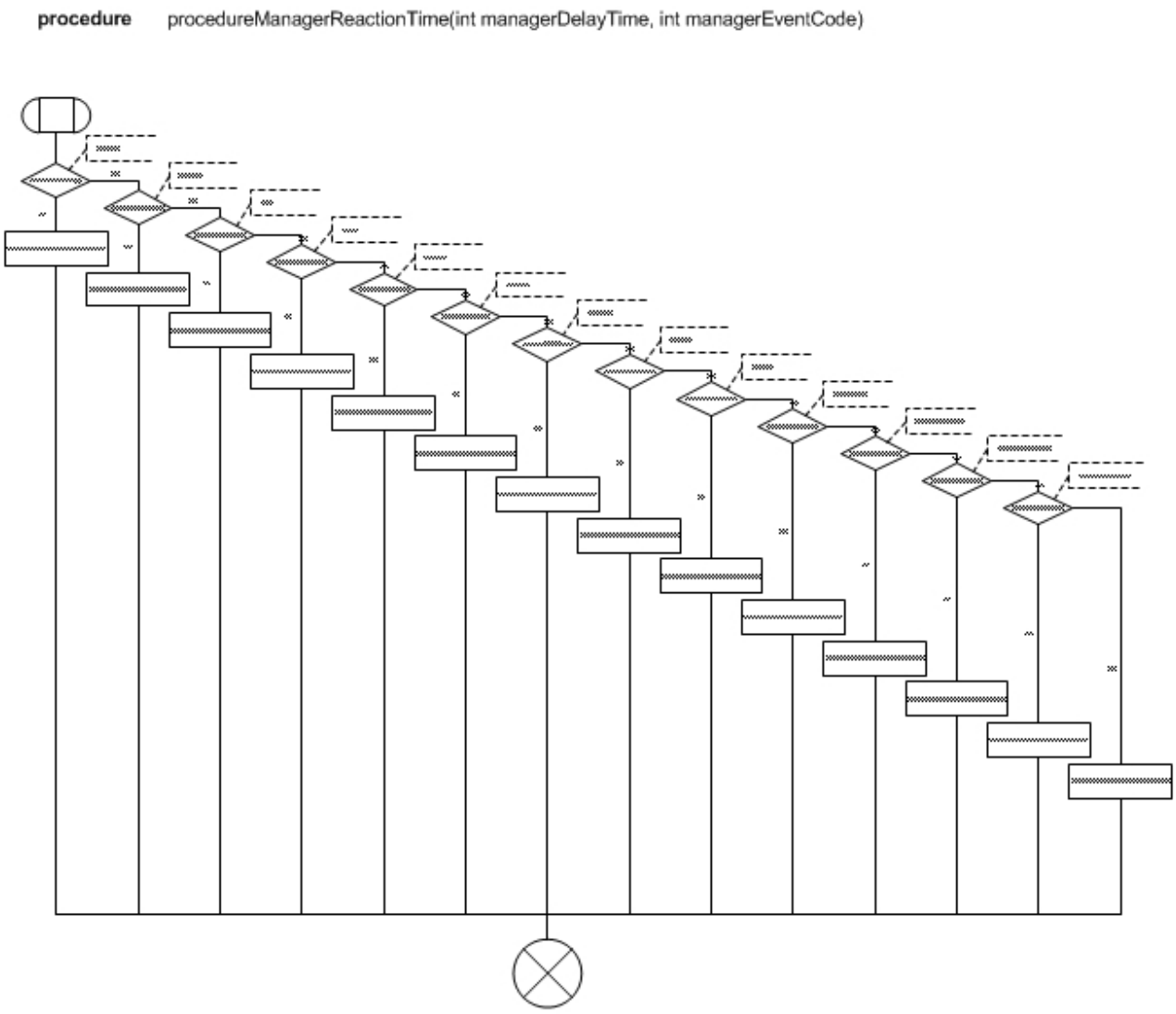


Ilustración 85: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Gestor

10.4.31 Inicialización de un agente de tipo Tutor

El archivo de este procedimiento es procedureAdvisorInitialize.vsd

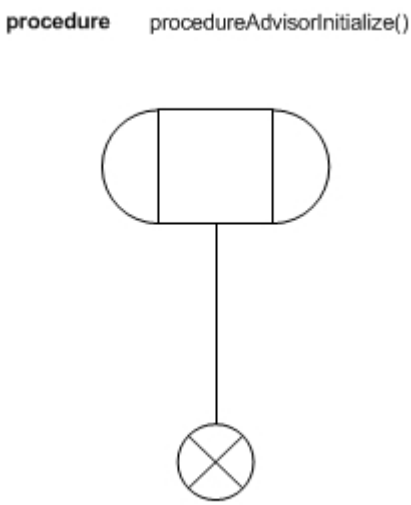


Ilustración 86: Diagrama del procedimiento de la Inicialización de un agente de tipo Tutor

10.4.32 Efectos tras una accion de un agente de tipo Tutor

El archivo de este procedimiento es procedureAdvisorActionEffects.vsd

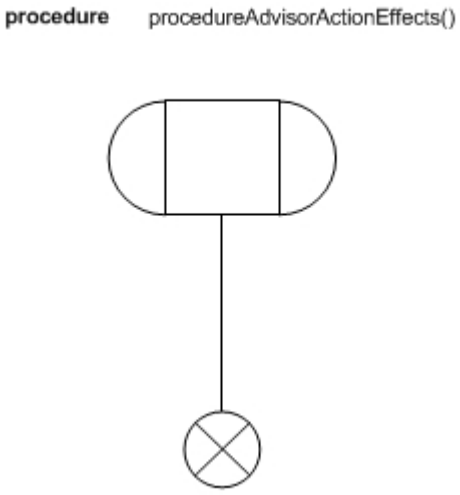


Ilustración 87: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Tutor

10.4.33 Modificacion del conocimiento de un agente de tipo Tutor

El archivo de este procedimiento es procedureAdvisorModifyingKnowledge.vsd

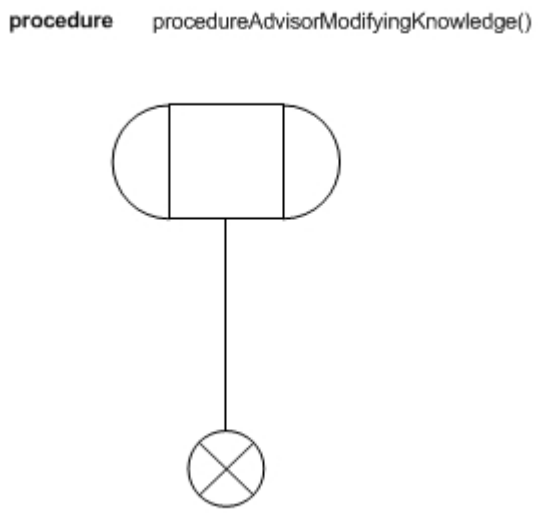


Ilustración 88: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Tutor

10.4.34 Tiempo de ejecucion de un agente de tipo Tutor

El archivo de este procedimiento es procedureAdvisorExecutionTime.vsd

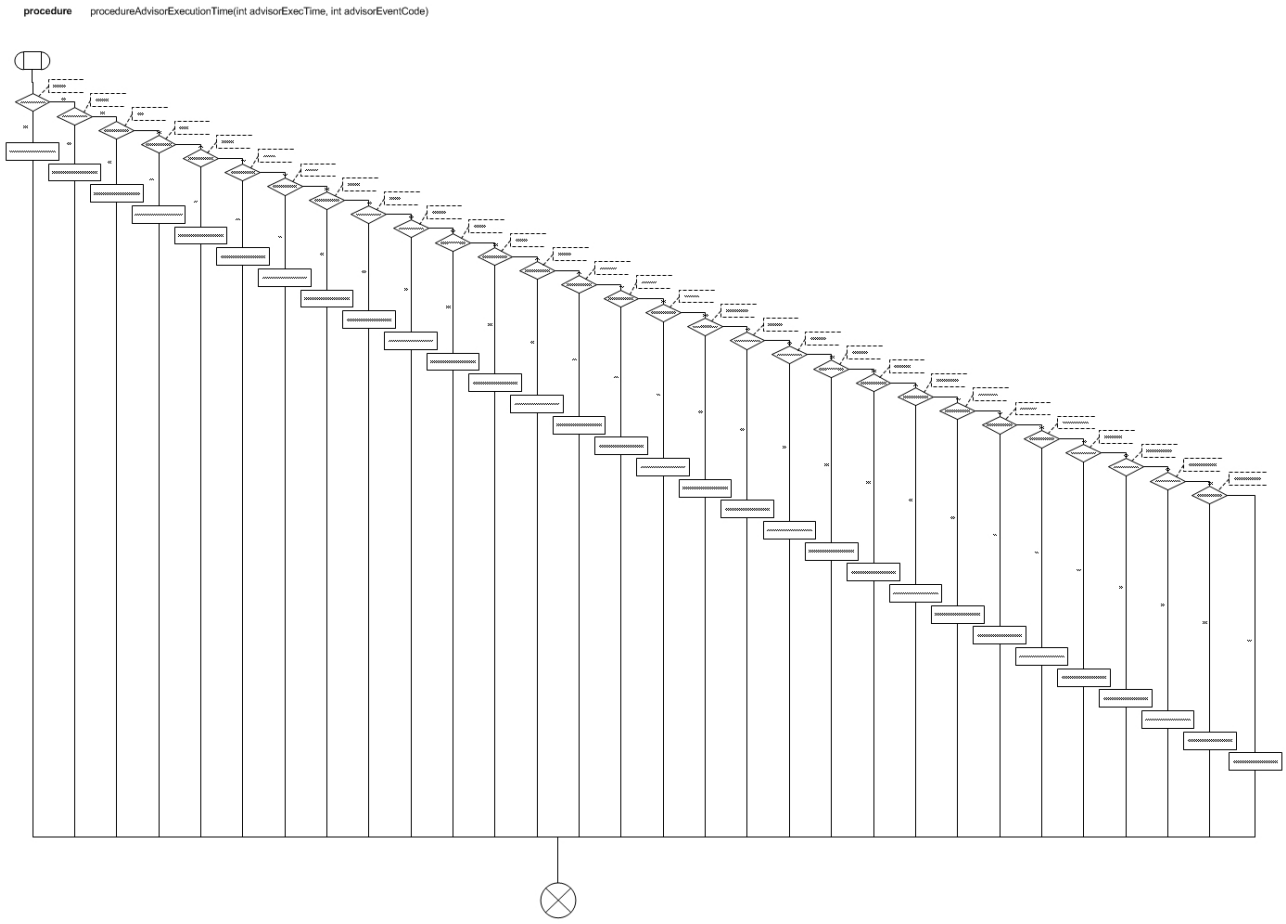


Ilustración 89: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Tutor

10.4.35 Tiempo de reaccion de un agente de tipo Tutor

El archivo de este procedimiento es procedureAdvisorReactionTime.vsd

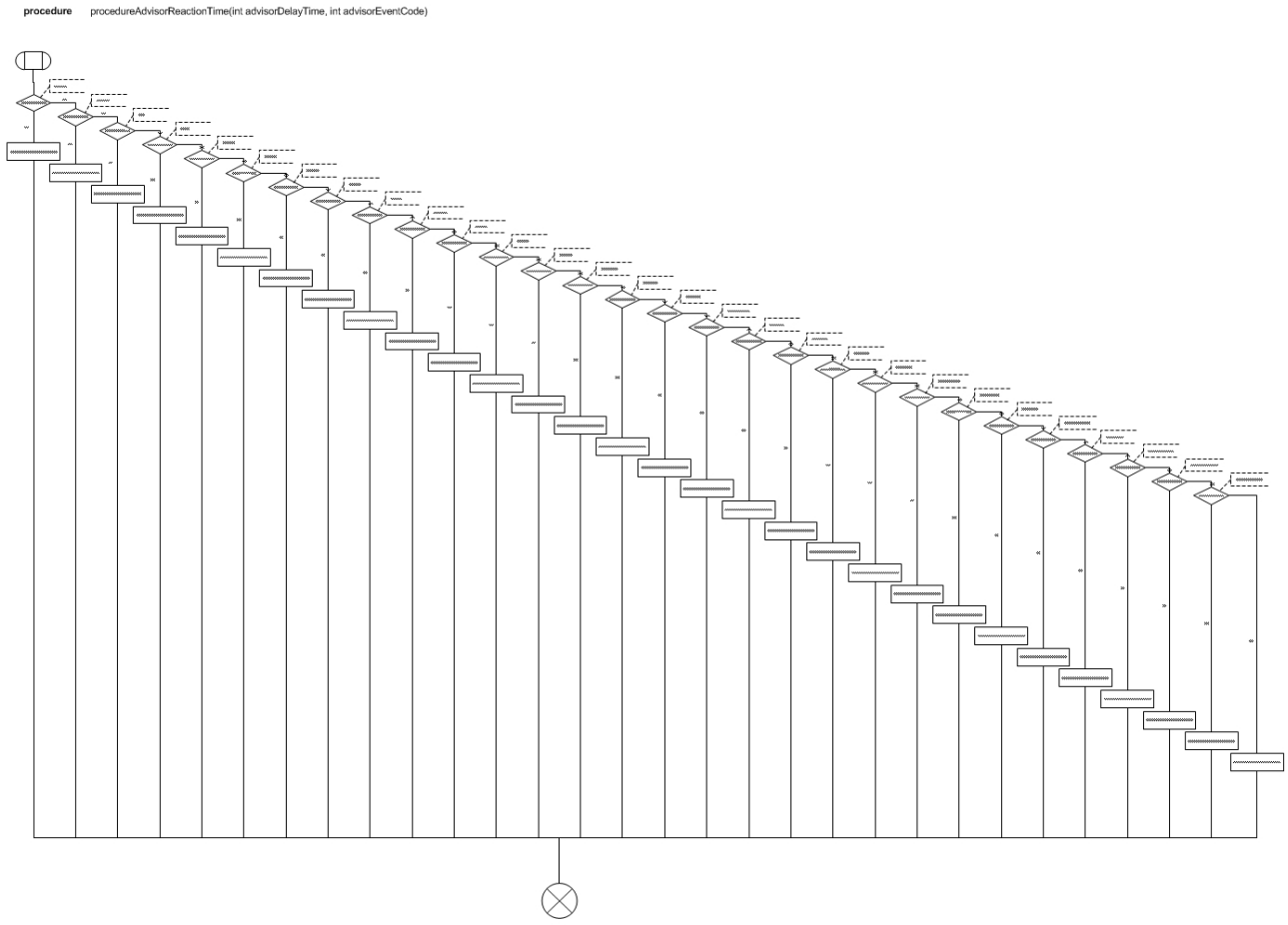


Ilustración 90: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Tutor

10.4.36 Inicialización de un agente de tipo Visitante

El archivo de este procedimiento es procedureVisitorInitialize.vsd

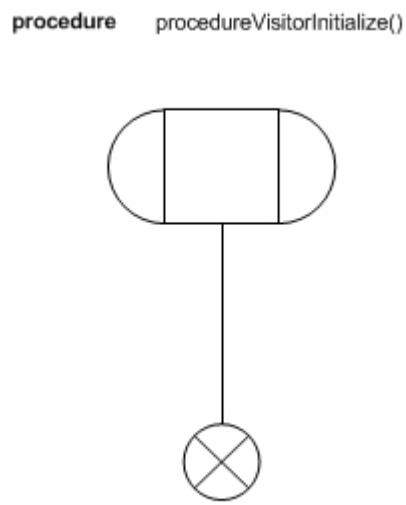


Ilustración 91: Diagrama del procedimiento de la Inicialización de un agente de tipo Visitante

10.4.37 Efectos tras una accion de un agente de tipo Visitante

El archivo de este procedimiento es procedureVisitorActionEffects.vsd

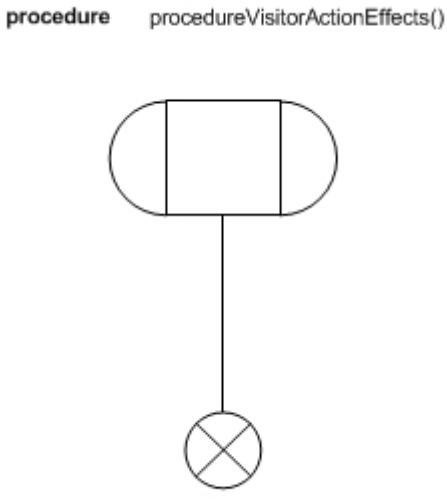


Ilustración 92: Diagrama del procedimiento de los efectos tras una accion de un agente de tipo Visitante

10.4.38 Modificacion del conocimiento de un agente de tipo Visitante

El archivo de este procedimiento es procedureVisitorModifyingKnowledge.vsd

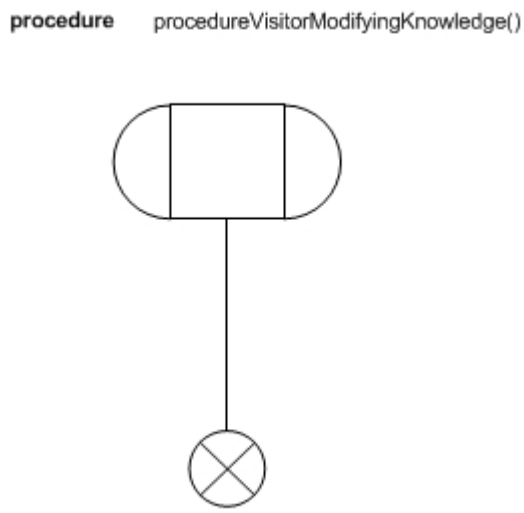


Ilustración 93: Diagrama del procedimiento de la modificacion del conocimiento de un agente de tipo Visitante

10.4.39 Tiempo de ejecucion de un agente de tipo Visitante

El archivo de este procedimiento es procedureVisitorExecutionTime.vsd

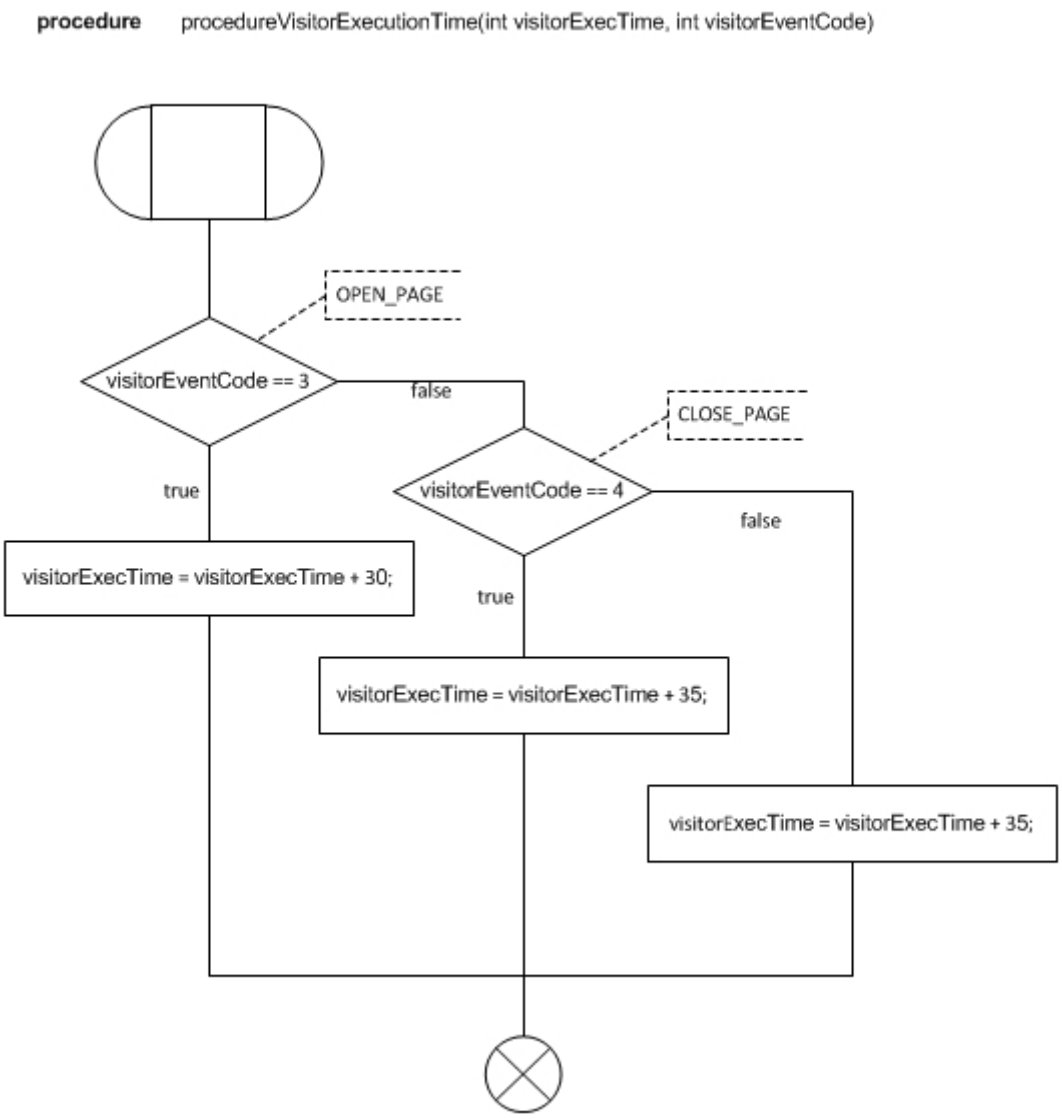


Ilustración 94: Diagrama del procedimiento del tiempo de ejecucion de un agente de tipo Visitante

10.4.40 Tiempo de reaccion de un agente de tipo Visitante

El archivo de este procedimiento es procedureVisitorReactionTime.vsd

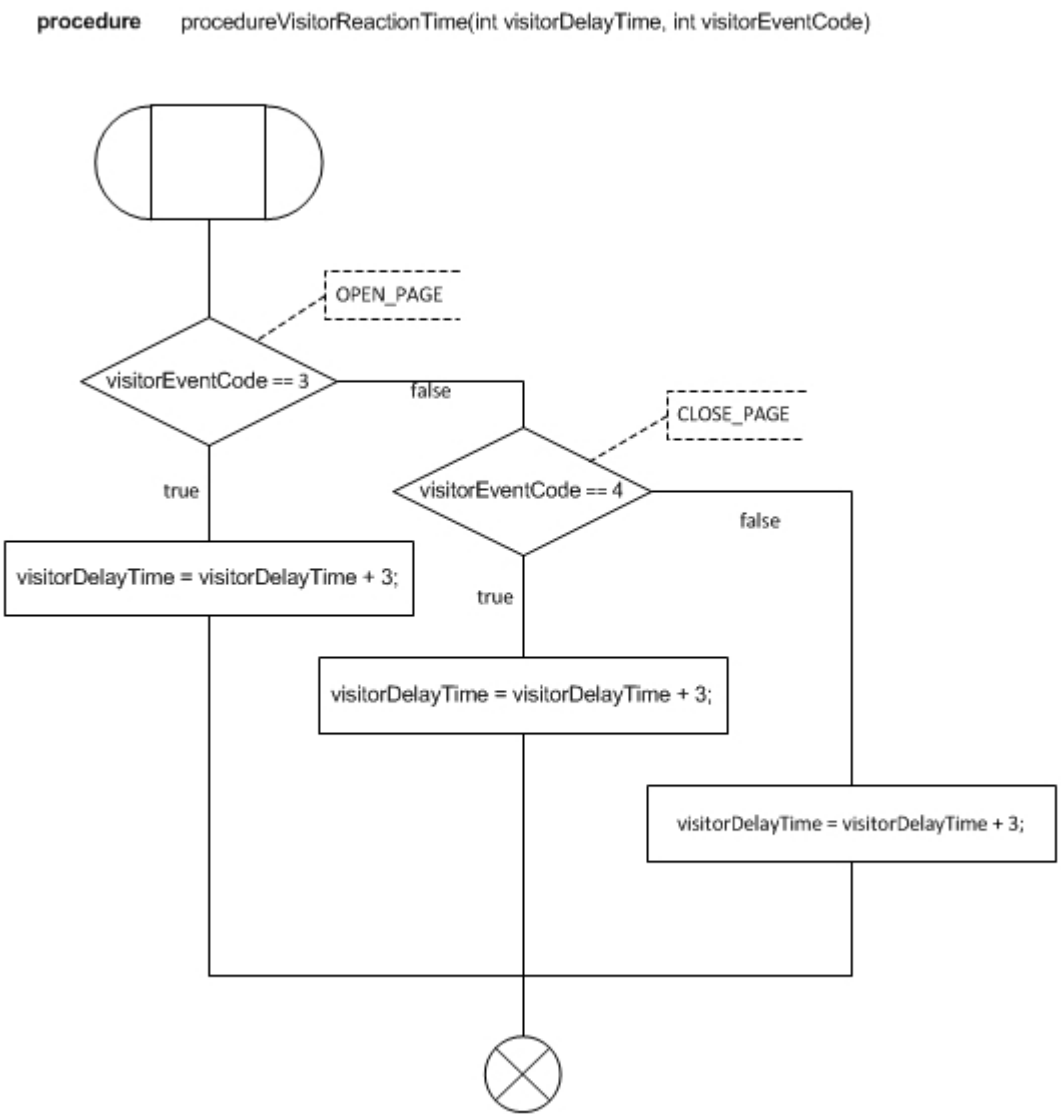


Ilustración 95: Diagrama del procedimiento del tiempo de reaccion de un agente de tipo Visitante

11 Conclusiones

En el presente TFC hemos definido formalmente los perfiles para simular el funcionamiento del Campus Virtual de la UOC.

Esta definición se ha realizado mediante la identificación de todos los perfiles existentes, la definición de las funcionalidades y la elaboración del modelo del sistema. Como conclusión podemos extraer que mediante la definición exhaustiva de las funcionalidades de un sistema es posible realizar un modelo de simulación, aunque el sistema sea de gran complejidad.

El refinamiento del modelo SDL, utilizando como base los modelos presentados en proyectos anteriores, ha sido posible gracias al software Microsoft Visio + SANDRILA plugin. Este software nos ha permitido ampliar la complejidad sin añadir dificultad a la modelización.

La herramienta SDLPS nos ha permitido comprobar el diseño y la corrección de nuestro modelo.

Las vías futuras de trabajo deberán consistir en la generación de interfaces con OMNet ++ y el desarrollo del bloque blockUniversityCampus para completar el modelo.

12 Glosario

Término	Definición
TFC	Trabajo Fin de Carrera
SDL	SDL (Specification and Description Language) es un lenguaje de especificación formal y visual normado por la ITU-T en el estándar Z.100, diseñado para la especificación de sistemas complejos.
Macro	Serie de instrucciones que se almacenan para que se puedan ejecutar de forma secuencial mediante una sola llamada u orden de ejecución
SDLPS	Simulador distribuido que permite la definición de modelos que utilizan el lenguaje SDL
DLL	Dynamic-link library, es la implementación de Microsoft del concepto de librería compartida en sistemas operativos Microsoft Windows
Perfil de usuario	Representación informática del modelo cognitivo de un usuarios humanos, incluidos los modelos de sus habilidades y conocimiento declarativo
Funcionalidad	Proceso, acción o tarea que un sistema es capaz de realizar
Modelo	Representación de procesos que conforman un conglomerado mayor o sistema, que pretende el análisis de interacción entre los mismos

13 Bibliografía

[MIQM] Miquel Mora Pérez, J. (2006). Castelldefels_05_149_TFC1: Modelatge i simulació del sistema informàtic que dona suport al Campus Digital UOC

[CABL] Cabeza Loriente, A. M. (2010). Castelldefels_10111_PFCamcabeza: Modeling user profile behavior using SDL

[BORR] Borràs Castillo, A. (2008). SDL Specification of Intelligent Agents to Model User Profiles. Treball de Fi de Carrera. Universitat Oberta de Catalunya. Barcelona.

[FON1] Fonseca i Casas, P. (2008). SDL, A Graphical Language Useful to Describe Social Simulation Models. In F. J. Quesada (Ed.), 2nd Workshop on Social Simulation and Artificial Societies Analysis (SSASA'08). Barcelona.

[FON2] Fonseca, P., Ramo, M., Juan, A. A. Using Specification and Description Language to represent users' profiles in OMNET++ simulations.

[FON3] Fonseca, P., Casanovas, J. Using Generic Event for a Simple Reflexive Intelligent Agent SDL Specification.

[SAND] Sandrila Ltd., SANDRILA SDL
http://www.sandrila.co.uk/visio-sdl/index.php

[SDLF] SDL Forum Society

http://www.sdl-forum.org/

[SDLP] SDLPS, a SDL distributed simulator
http://www-eio.upc.es/~pau/index.php?q=node/27