



Trabajo de Fin de Grado

GRADO DE INGENIERIA INFORMÁTICA

**Facultad de Matemáticas
Universidad de Barcelona**

**MODERACIÓN AUTOMÁTICA EN
COMUNIDADES VIRTUALES**

David Sánchez Pinsach

Directora: Dr. Maite López-Sánchez
Realizado en: Departamento de
Matemática Aplicada i
Análisis. UB

Barcelona, 09 de enero de 2013

Agradecimientos

Agradecer a Maite López por darme la oportunidad de poder trabajar en este proyecto y a Juan Antonio Rodríguez, Javier Morales y losu Mendizabal por ayudarme en todas las dudas que se han ido planteando a lo largo del desarrollo del proyecto.

Índice

1. Introducción	4
1.1. Abstract	4
1.2. Antecedentes	5
1.3. Definición del problema	8
1.4. Marco de investigación básica relacionada	10
1.5. Definición de los objetivos de trabajo	14
1.6. Plan de trabajo	15
1.7. Motivación personal para la realización del presente TFG	16
1.8. Estructura de la memoria	18
2. Estado del arte	19
2.1. KeepCon	19
2.2. Stilus Forum	21
2.3. SourPanel	22
2.4. Definición de moderador y tipos de moderación	23
2.4.1 La figura del moderador	23
2.4.2. Tipos de moderación	24
2.5. Conclusiones del estado del arte	25
3. Diseño	26
3.1 MAS	26
3.1.1. Definición	26
3.1.2. Roles	26
3.1.3. Acciones del rol de usuario	26
3.1.4. Acciones del rol del moderador	26
3.1.5. Tipos de perfiles del usuario	27
3.1.6. Ontología	28
3.2. Simulación del MAS	30
3.2.1. Arquitectura del proyecto	30
3.2.2. Contenidos, secciones, quejas y usuarios	31
3.2.3. Estructura de la Base de datos	35
3.2.4. Diagrama casos de uso	37
3.2.5. Casos de uso	37
3.3 Estructura del proyecto	39
3.4. Diseño de agentes en JAVA	40
3.5. Diseño de las acciones de los agentes en JAVA	40
3.5.1 Subir contenido (uploadContent)	40
3.5.2 Visitar y quejarse de contenidos (viewContent and complaintContent)	41
3.6. Diseño del Context builder	41
4. Desarrollo	43
4.1. Entornos de trabajo usados	43
4.2. Funcionamiento, configuración y curiosidades de Repast Symphony Java 2.0	44
4.3. Problemas detectados en el desarrollo	45
5. Resultados	47
6. Conclusiones y trabajo futuro	48
7. Referencia bibliográficas	50
8. Anexos	52
8.1. Manual de uso de la aplicación	52
8.2. Importar scripts MySQL con XAMPP	57
8.3. Cambiar iconos de los agentes, secciones, comentarios o quejas en la aplicación.	58

1. Introducción

1.1. Abstract

In the last decade, the number of the virtual communities, users and contents has multiplied. Due to this growth, appears a problem about the control of the content and the users of these virtual communities: huge number of the contents. This fact makes more problems to control the content because normally the content is moderate with human moderators. The human moderators sometimes are users and other times are employees but in both case it's a time or money cost. This general project would be a solution to reduce the cost of moderate full contents with automatic generation of norms and moderate the content with these norms.

This project is simple piece of a general project to automatic generation of norms in a virtual community. The general project pretends to give an answer about some problems, which occurs when the users generate content within the environment of the virtual communities of the soccer's followers. We want to simplify the problem about the moderate contents in a virtual community applying norms generated by the content complaints.

Initially, we need to simulate the virtual community. With this simulation we are going to make a representation of the virtual community and also of its actions. Our simulation will have different types of agent like moderated, spammer or troll. The agents can do some actions about the content. These actions are upload, view or complaint content. The agents can't interact directly with others agent but yes with the content.

With aim to simplify the system due to the absence of a content categorizer, we decide that the content has already categorized in a database.

This TFG with it simulations of the virtual community is the first step to solve our general problem about the automatic moderation control.

1.2. Antecedentes

La humanidad ha ido evolucionando con el paso del tiempo y de los años. Dicha evolución no ha sido solo basada en cosas al azar producidas por la propia naturaleza ni tampoco basada íntegramente en avances tecnológicos.

Los humanos tenemos una capacidad o habilidad; que no tienen muchos animales, que es la de hablar con los de nuestra misma especie. Esto se basa en el concepto de la "Comunicación". Según una típica definición del concepto: *"La comunicación es el proceso mediante el cual se puede transmitir información de una entidad a otra. Los procesos de comunicación son interacciones mediadas por signos entre al menos dos agentes que comparten un mismo repertorio de signos"*¹

Como se deduce de la propia definición del término, la comunicación es una acción entre dos individuos. Esta acción se hace mediante la transmisión de mensajes que contienen información. Gracias a este mecanismo que es capaz de trasladar la información de una persona a otra, la humanidad y los humanos como tal, hemos podido evolucionar. El conocimiento a circulado entre personas y no se ha quedado solamente en aquellas personas que han pensado en ir más allá.

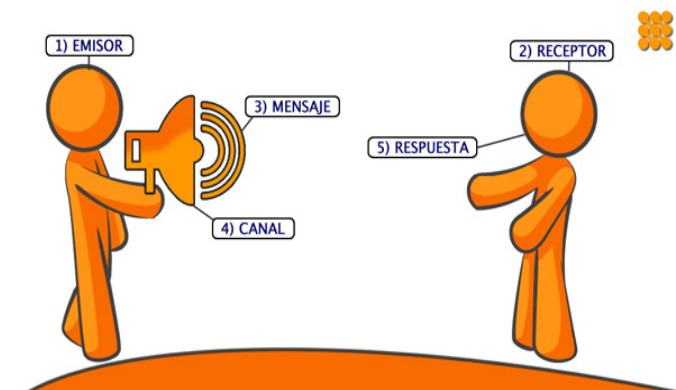


Figura 1: Estructura y partes que intervienen en la comunicación

La información hace que un individuo que no ha vivido ciertas cosas o desconoce otras tantas sea capaz de conocer cosas gracias a la transmisión de información de otra persona; que si que lo ha vivido o ha oído hablar de ello.

Otro de los pilares fundamentales de la evolución también ha sido el de evolucionar el concepto y las maneras o métodos para transmitir la información. Todo ello yendo más allá del mero hecho de transmitir información de forma oral entre dos personas que están cara a cara o que están presentes en un mismo espacio y en un mismo tiempo.

Entre estos pilares habría que mencionar dos invenciones que han tenido un peso importante en dicha evolución y en la historia de la humanidad.

La escritura es un sistema gráfico para representar una lengua. Esta invención apareció de manera simultánea en varias zonas del mundo como fueran Egipto, China o Mesopotamia o incluso en civilizaciones como los Mayas en Centro-América allá el cuarto milenio a.C. A partir de ese momento se empezó a escribir y documentar cosas o hechos que tenían lugar en esos lugares. Esa fue una forma diferente a la de transmitir experiencias o conocimientos entre humanos a la que había entonces como fuera el habla.

Por otra parte no había muchas personas que supieran escribir e incluso era bastante costoso el hacer copias de algunas determinadas escrituras o libros ya que se tenía de hacer de forma manual y por personas cualificadas.

¹ Definición de Wikipedia: <http://es.wikipedia.org/wiki/Comunicaci%C3%B3n>

Hasta que no apareció la imprenta la copias de determinadas obras era muy costoso; sobretodo hablando en términos de tiempo. La imprenta revolucionó medio mundo ya que dió la oportunidad de hacer copias de libros de forma fidedigna, en tiempo y costos muy bajos. La invención data de mediados del siglo XV aunque siglos anteriores hubieran sistemas rudimentarios que intentaran lograr el mismo objetivo.

Allá el 1969 aparecería otro proyecto que cambiaria la manera en que nos comunicariamos en el futuro. Dicho proyecto era nombrado como ARPANET. ARPANET fue la primera red interconectada de ordenadores. Sus inicios fueron los de unir las universidades de UCLA y Stanford en los Estados Unidos de América por medio de una línea telefónica. Muchos rumores de esa época dicen que fue pensado para soportar desastres nucleares dado que fue desarrollado por ARPA (actualmente llamado DARPA²). Teniendo en cuenta que muchos de los avances de la humanidad se han producido con fines como protegerse o atacar en casos de guerras la idea no resultada tan descabellada.

Pero como todo lo relacionado con los humanos y las sociedades, el proyecto ARPANET también fue evolucionando y haciendo que en esa red cada vez hubieran más ordenadores conectados. También fueron evolucionando los protocolos de comunicación y creando estándares en los protocolos.

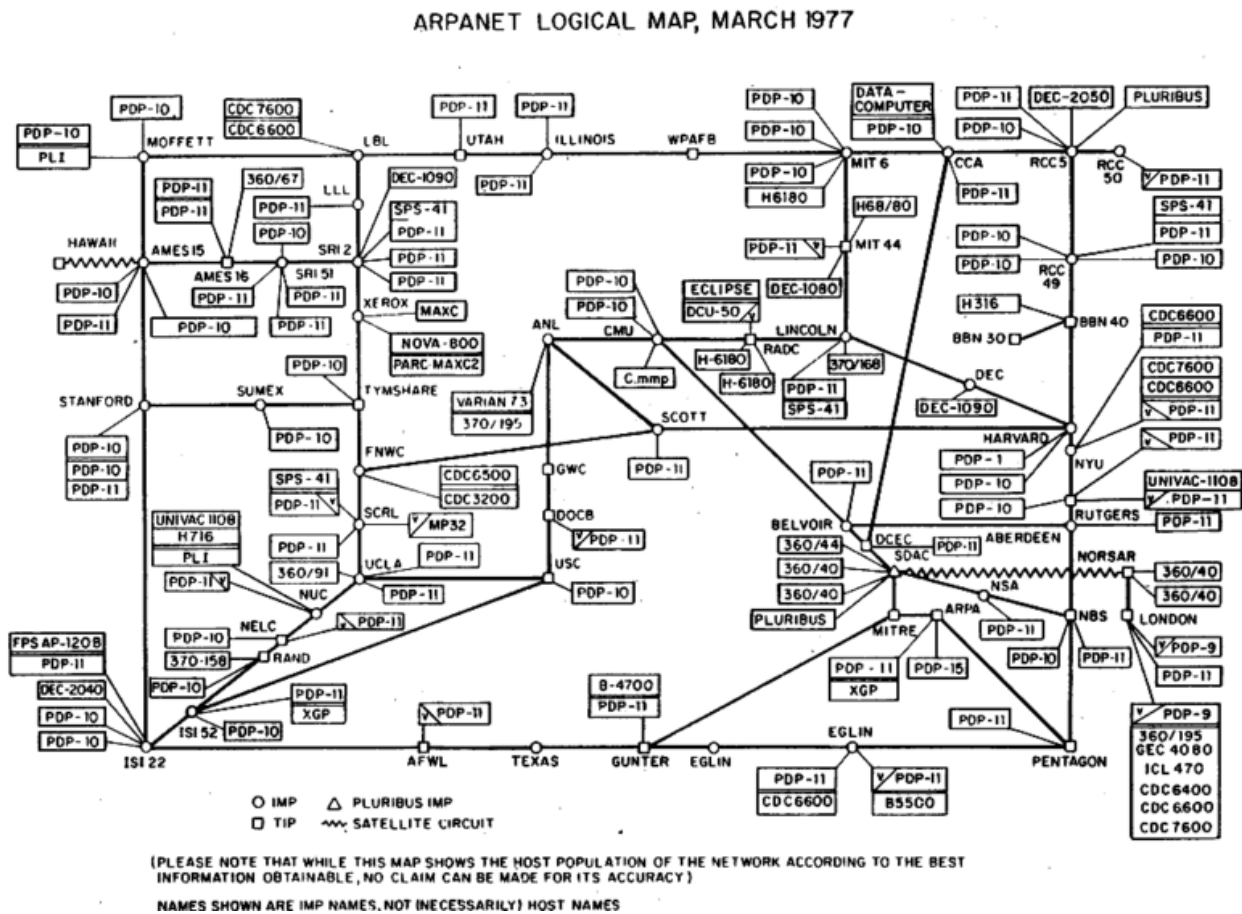


Figura 2: Estado de la red de ordenadores de ARPANET en 1977.

² Acrónimo de la expresión en inglés Defense Advanced Research Projects Agency (DARPA)

Con el paso del tiempo el proyecto ARPANET pasó a ser comercial y accesible para todas las personas y fue evolucionando a lo que hoy día se conoce como Internet.



Figura 3: Logo de MySpace.com

De los diferentes tipos de servicios que ofrece Internet que entre ellos hay (STMP, FTP, P2P, IRC, Telnet...) el WWW³ es uno de los que ha tenido y tiene más impacto en la vida de los humanos del siglo XXI. WWW es un sistema de distribución de información basado en formato hipertexto y accesible por medio de Internet. Gracias a un navegador web dicha información; escrita en formato hipertexto, se visualiza por pantalla produciendo o generando lo que comúnmente se conoce como “páginas webs”.

Con la aparición de la WWW empezaron a crearse las primeras redes sociales y los primeros blogs. Los primeros blogs eran como diarios personales donde un usuario comenta cosas que le intrigan o aspectos y vivencias del su día a día. Los otros usuarios entraban y leían e incluso comentaban lo que decía el blogger⁴.



Figura 4: Imagen donde aparece el logo de facebook en diferentes tipos de pantalla.

Por otra lado, unas de las primeras redes sociales que tubo mucho éxito y que tenía muchos usuarios fue MySpace. Era la primera gran comunidad virtual y pasó a ser una de las primeras comunidades virtuales donde la gente le dedicaba muchos minutos al día, haciendo que entre el 2005-2008 fuera la web más visitada en el mundo.

Pero en el camino de MySpace se cruzó un proyecto que redefiniría el concepto de comunidad virtual y lo evolucionaría en lo que hoy en día conocemos como red virtual. Facebook inicialmente fue pensada para unir en una comunidad a los estudiantes de diferentes redes sociales. Resulta curioso que ARPANET también inicialmente fue pensada para unir diferentes universidades. Como sucedió con ARPANET o Internet, Facebook en setiembre del 2006 se abre a todo tipo de usuarios.

Facebook tiene uno de los puntos claves que ha hecho crecer y fomentar la creación de nuevas comunidades virtuales. La mayoría de los usuarios de Facebook tiene perfiles reales. Eso quiere decir que las personas no son anónimas, sino que un perfil virtual se asocia a una persona real. Esto ha hecho que las personas tengan otra manera de expresarse o comunicarse con otras personas. Como con la idea de los blogs, las personas expresan vivencias y pensamientos en la red virtual. Lo curioso es que si dispones de un perfil privado sólo las personas que tengas agregadas o consideradas como amigos son los que podrán ver tus comentarios o fotos. Esto ha hecho que importa lo que digas en la red, ya que lo que digas en ese perfil virtual se te asociara directamente a ti como persona. Eso ha hecho que muchas personas midan directamente o indirectamente las cosas que dicen o publican en la red. Otras redes virtuales como

³ WWW. Acrónimo de World Wide Web.

⁴ Blogger es el usuario propietario de un blog.

puede ser Twitter también están basadas mayormente en perfiles virtuales que se asocian a personas reales.

Internet hoy en día es un sistema de transmisión de información con diferentes canales (video, imagen, texto...). Cualquier persona que disponga de acceso a Internet puede generar, crear, difundir y comentar información. Dicha información a su vez es visible por miles de personas que a su vez generan nuevo contenido sobre dicha información.

El proyecto final de carrera que se explica a continuación pretende dar respuestas a varios problemas que aparecen cuando los usuarios generan información en Internet dentro del entorno de las comunidades virtuales.

1.3. Definición del problema

Con el crecimiento de Internet y con su uso extendido entre todo tipo de personas y edades ha hecho que el contenido o información también aumentase. Y es que no solo hay mucha información en Internet si no que dicha información es potencialmente visible por muchas personas. En los últimos años ha habido un auge de las comunidades virtuales importante y eso se ve reflejado con el dato de donde pasan más tiempo los usuarios cuando están en Internet.

Las comunidades virtuales son una comunidad de personas que se comunican y relacionan mediante el uso de la red de Internet. Como si se tratase de una casa, no todas las personas pueden acceder a dichas comunidades sino tienen la llave. Normalmente estas llaves son el hecho de registrarse o el hecho de recibir una invitación en dicha comunidad. Hay otras comunidades que la entrada no requiere de una "llave" y que son visibles por cualquier persona. En este tipo de comunidades virtuales a veces se les da la posibilidad a personas que no estén registradas de poder poner nueva información en la comunidad pero no sucede en la mayoría de los casos. Pero el interés de este proyecto se centra en los usuarios que están registrados y por lo tanto identificados.

Otro problema es que no todos los perfiles de usuarios de estas redes virtuales se asocian a perfiles reales de personas. Hay muchos individuos que crean perfiles falsos para generar insultos, publicidad o intentar producir el error en otros para beneficiarse de muchas maneras con dicho error.

Por otra parte hay muchas comunidades virtuales que realmente no están pensadas a que los perfiles de usuario estén asociados a perfiles de personas reales. Por ejemplo, la cuenta virtual @pau_gasol de Twitter si que se asocia a la persona Pau Gasol. Entonces, todo lo que se dice desde esa cuenta va a nombre y se le asocia a Pau Gasol. Pero eso no sucede en muchas comunidades virtuales y suele ser complicado asociar una cuenta virtual a una persona física.

Eso hace que los usuarios no midan lo que publican en la red haciendo que los usuarios sean capaces de generar contenido que muchas veces no esta admitido en las comunidades virtuales. Como un usuario no puede saber quién es realmente el otro usuario eso hace que los usuarios sean capaces de hacer cualquier cosa en la comunidad.

Toda comunidad de personas siempre tiene que tener una persona o alguien que modere a las personas que permanecen en ella ya que en la mayoría de relaciones entre

personas se producen crispaciones y conflictos. Este problema surge porque las comunidades de personas son en muchos casos un cúmulo de personas de diferentes ideologías, culturas y edades. Hay conflictos porque son diferentes pero las cosas que tienen en común hacen que permanezcan en las comunidades. En el caso de los seguidores de fútbol este hecho se manifiesta con más fuerza. El fútbol es uno de los deportes que ha traspasado fronteras culturales, ideológicas y de edades. Esto produce que en las comunidades que se habla y se relacionan seguidores de fútbol hayan personas de todos tipos. En dichas comunidades siempre hay asociadas unas normas de uso y si no se siguen estas normas se producen castigos a los usuarios. Estos castigos varían según la importancia de la norma quebrantada y de antecedentes del usuario. Si eres un usuario que reiteradamente quebrantas o no sigues las normas de una determinada comunidad; aunque la norma no sea muy importante, serás castigado.

El cómo se determina que una persona ha quebrantado una norma y que tiene que ser castigado se hace mediante el uso de uno o varios moderadores. Los moderadores son personas que se encargan de que una comunidad de personas sigan las normas estipuladas. De este modo, revisan los contenidos y las quejas de los usuarios y aplican acciones a los usuarios o a sus contenidos cuando dichos usuarios no cumplen una determinada norma.

La moderación es necesaria para que en las comunidades no hayan cosas ilegales o fuera de lugar, o para controlar usuarios que “molestan” a otros.

Con el auge de las comunidades y con el crecimiento de las personas que acceden a él, esto ha hecho que hayan muchísimos usuarios y que a su vez genere mucho contenido en ellas. Esto hace que muchas tengan que necesitar tener muchos moderadores para poder controlar la comunidad.

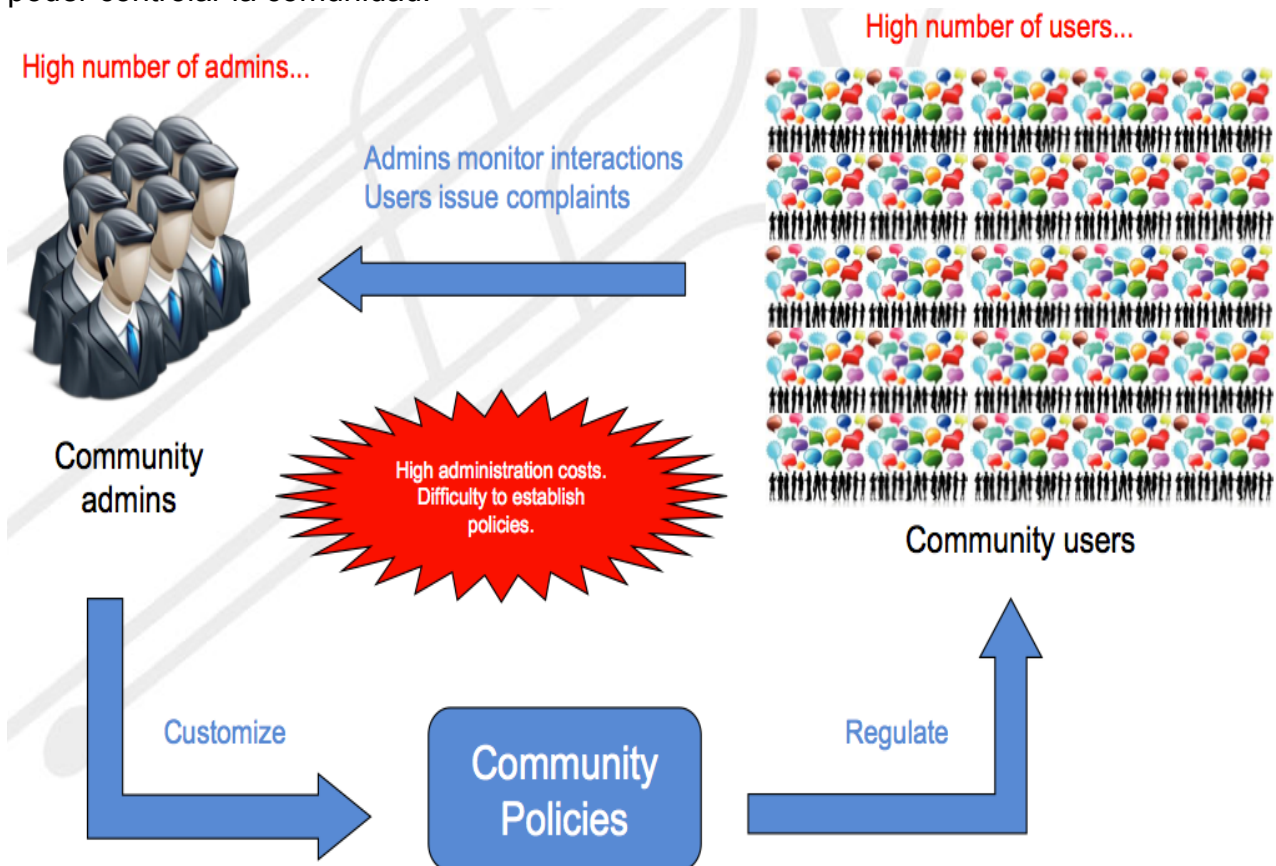


Figura 4: Diagrama donde se representa el coste de tener administradores o moderadores cuando el numero de usuarios de una comunidad virtual es grande.

En comunidades virtuales grandes el número de moderadores suele ser elevado. Los moderadores no dejan de ser empleados de la empresa que se encarga de administrar la comunidad virtual y que como todo empleado tiene su sueldo a fin de mes. Por eso en comunidades virtuales grandes el coste que supone mantener a tantos moderadores suele ser elevado. A eso se le junta que muchas veces es una faena que no suele valorarse mucho ya que no da nuevas funcionalidades a un sistema o mejorar las existentes.

En resumen, en las comunidades virtuales tienen un número elevado de usuarios este hecho genera un problema de costes y tiempo al tener a un gran número de moderadores para controlar el tráfico de la comunidad. De este modo, el problema se haya en intentar simplificar las tareas a los moderadores con un sistema automático de generación de normas para la comunidad virtual.

1.4. Marco de investigación básica relacionada.

El trabajo de investigación dentro del área de sistemas multi-agente en el que se sitúa el presente TFC, se basa en la extensión de la idea de una tesis doctoral. Esta tesis doctoral tiene como nombre *Using the experience to generate new regulations* y es de Javier Morales con supervisión de Maite López-Sánchez y Juan Antonio Rodríguez. El objetivo principal de la tesis es generar normas a partir de la experiencia en un contexto de tráfico.

El escenario es una cruce de diferentes carreteras cómo el que podríamos encontrar en cualquier ciudad. Por otro lado, el objetivo principal del sistema es el de evitar accidentes en este cruce y para ello el sistema generará normas que hagan y ayuden a evitar dichos accidentes. Las normas también tendrán una etapa de validación para determinar si eran necesarias y si se logra evitar el conflicto o accidente con ellas.

Todo este sistema funciona de manera online dado que tiene ciertas ventajas frente a un sistema offline. De manera offline, tiene la desventaja cómo puede ser que las condiciones del sistema varíen y que las normas que inicialmente funcionaban luego ya no. Otra desventaja en un funcionamiento del sistema offline es el número de estados posibles ya que es imposible generarlos todos, como que sería necesaria al hacer el sistema offline.

Los pasos básicos del sistema son los siguientes:

1. El sistema empieza a ejecutarse con o sin normas.
2. Se monitoriza el sistema de manera continuada, poniendo varias cámaras en la zona del cruce de carreteras.
3. Se identifican los estados no deseados(conflictos) a medida que suceden.
4. Se generan normas que impidan el acceso a estos estados conflictivos para que no tengan lugar.
5. Las normas se avalúan continuamente en función de si evitan los conflictos y de si realmente eran necesarias.

Utilitat de les normes

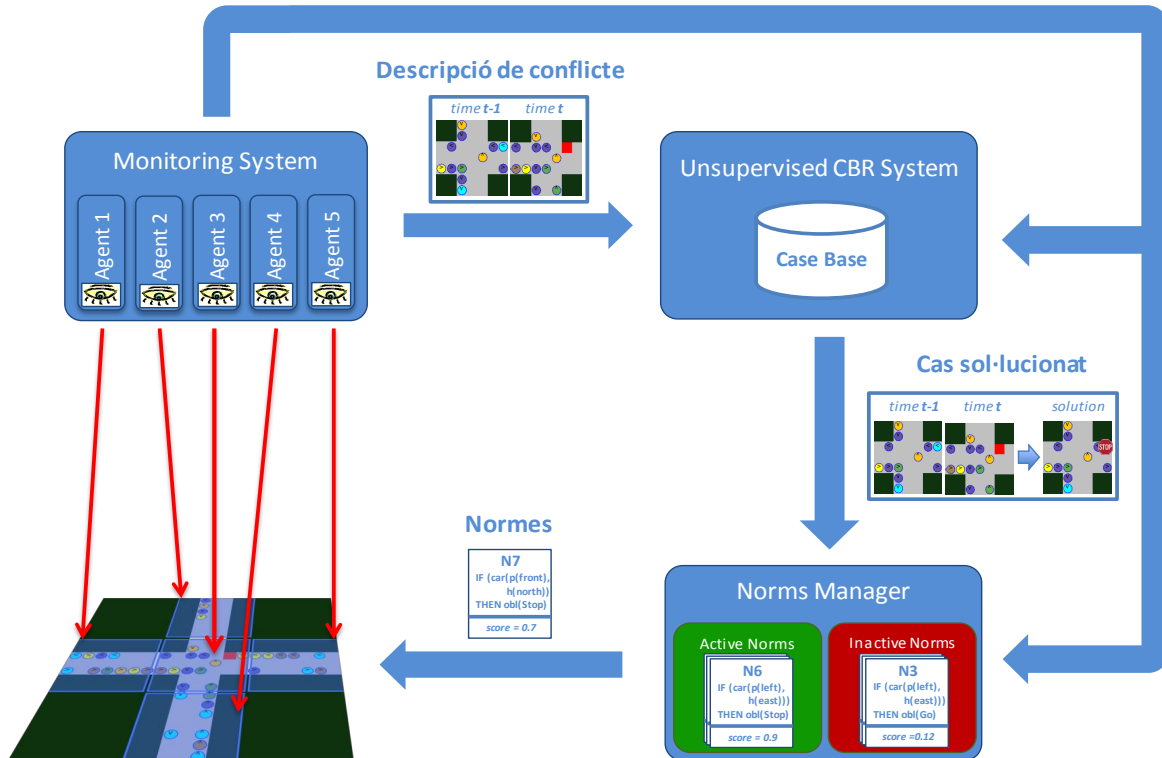


Figura 5: Diagrama de las partes de la tesis doctoral "Using the experience to generate new regulations" de Javier Morales.

Para hacer proceso de identificar estados y que de a partir de la experiencia de estos estados se generen normas el sistema usa un CBR⁵ no supervisado.

El CBR clásico describe un problema como:

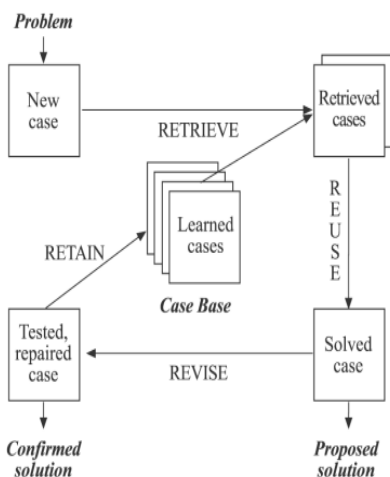
$$\text{Caso} = (\text{problema}, \text{solución})$$

Donde el problema se define como:

$$\text{problema} = (\text{problema}(t-1), \text{problema}(t))$$

Y la solución se define cómo el conjunto de todas las posibles soluciones con su puntuación. La puntuación es un valor que oscila entre [0,1].

$$\text{solución} = (\text{solución}(i), \text{puntuación}(i))$$



Todo CBR tiene cuatro etapas diferenciadas:

RECORDAR (retrieve). Recordar los casos similares al que analizamos.

REUTILIZAR (reuse). Reutilización de la información y el conocimiento que tenemos en este caso para resolver el problema.

REVISAR (revise). Revisión de la solución propuesta.

RETENER (retain) las partes de esta experiencia que nos puedan ser útiles para la resolución de futuros problemas.

Figura 6: Diagrama del funcionamiento de CBR

⁵ CBR. Acrónimo de Razonamiento basado en casos (Case Based Reasoning).

Una vez el sistema un caso en que se soluciona el conflicto, este caso se envía al Norms Manager para que haga la traducción del caso a norma.

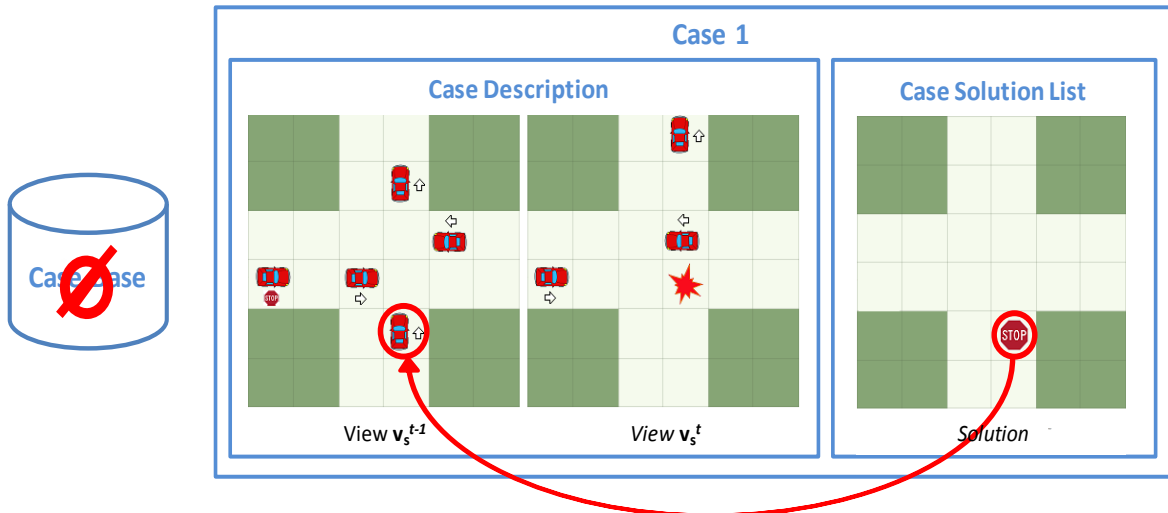


Figura 7: Diagrama donde se muestra la representación de un caso y de su solución.

Las normas se describe con el formato siguiente; donde la condición de la formula es para saber en que momento hay que aplicar la norma:

SI hay esta condición ENTONCES obligar hacer(acción)

Las normas también tienen asociada una puntuación que representa su rendimiento.

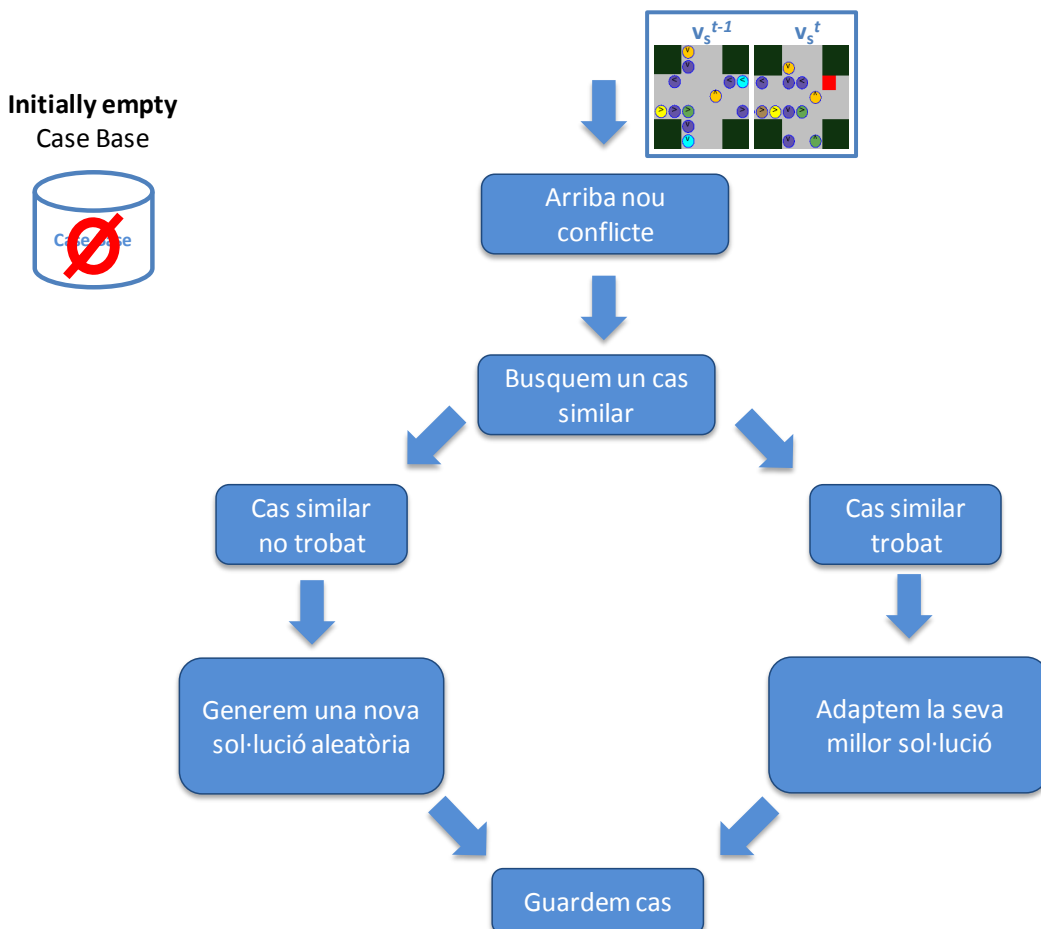


Figura 8: Diagrama donde se muestra el flujo de trabajo del CBR no supervisado.

Por otro lado tenemos todo un sistema de simulación de tráfico donde las calles tienen unos determinados sentidos y direcciones. Toda la simulación se ha hecho con el framework de Repast Simphony Java 2.0. En él hay una interface gráfica donde se representan las simulaciones de diferentes agentes (coches) y de sus posibles movimientos dentro del escenario.

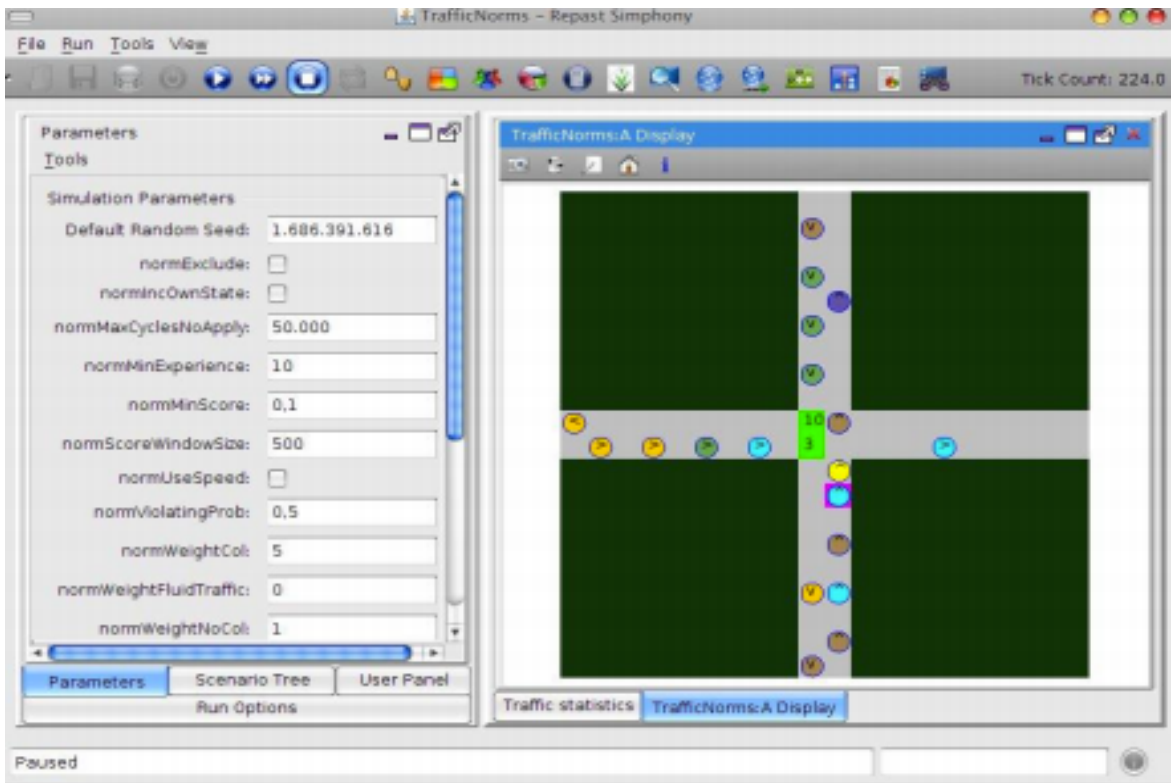


Figura 9: Imagen del simulador y de los parametros configurables.

Por otro lado, el simulador también dispone de utilidades gráficas para ver en forma de gráficas resultados o determinadas características del simulador.

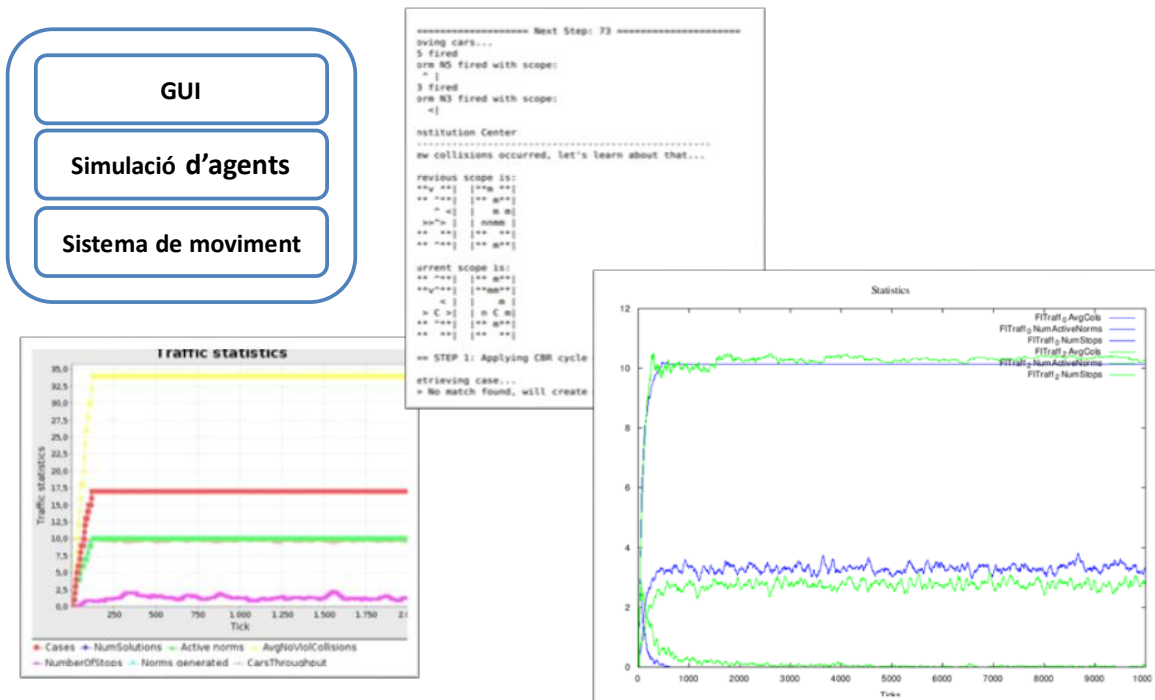


Figura 10: Imagen donde se muestran diferentes tipos de logs o gráficas. Estas gráficas o logs son útiles para analizar resultados de forma rápida y cómoda.

1.5. Definición de los objetivos de trabajo

El problema general es de la moderación automática en comunidades virtuales de seguidores de fútbol. La tesis de Javier Morales propone un método para generar normas de comportamiento (o convivencia) entre agentes dentro de un sistema multi-agente. La idea es cambiar el contexto de la aplicación (en inglés, use case); dado que Javier lo usa en uno de tráfico, a un contexto de comunidad virtual de seguidores de fútbol y donde los conflictos no serán accidentes de tráfico sino las quejas de los comentarios que hagan los usuarios.

Para resolver este problema general necesitamos un sistema que simule una comunidad virtual para posteriormente aplicar el método que recoge la tesis de Javier Morales.

El objetivo principal del trabajo es el de simular una comunidad virtual de seguidores de fútbol con Repast Symphony Java 2.0.

En dicha comunidad tendremos diferentes usuarios que podrán interactuar con la comunidad. Dicha interacción será mediante la generación de nuevo contenido, visitando contenido que haya sido puesto por otros usuarios o quejándose del contenido que se ya halle en la comunidad.

La interacción que realizan los usuarios en este entorno es indirecta, pues ésta se hace a través de la publicación de contenidos en un espacio compartido. La simulación es una recreación del comportamiento de los usuarios en las comunidades virtuales reales que permitirá estudiar los efectos de introducir normas.

El contexto de la aplicación será el de simular una web donde los contenidos están pre generados y están almacenados en una zona común para todos los usuarios. Los usuarios cogen contenidos de esta zona común y los suben a la web.

Se ha tomado la decisión que todo el contenido esté categorizado por tipos en una base de datos (la zona común de todos los usuarios). Esto implica una simplificación del problema ya que se carece de sistemas de categorización automáticos.

Los usuarios cogerán contenido pre-generado y lo publicaran en la comunidad. De esta manera quiere simplificarse y quitar la parte de analizadores de lenguajes natural para determinar que ha dicho tal usuario y de que tipo de contenido en consecuencia se trata.

La finalidad del trabajo no es detectar qué dice una persona sino hacer una simulación de varios usuarios generando, visitando y quejándose sobre contenidos. Todo esto para posteriormente generar de forma automática normas y aplicar acciones con a los actos producidos por los usuarios.

1.6. Plan de trabajo

Se ha diseñado un plan de trabajo para desarrollar el proyecto. Dicho plan de trabajo no se creo inicialmente ya que se desconocía el alcance del proyecto y que cosas se implementarían finalmente o no.

Con la idea de especie de diario o de lista de tareas se ha construido el plan de trabajo final de este proyecto:

Nombre	Duración	Inicio	Terminado
Llegir documentació administrada per la Maite	1 day?	14/09/12 8:00	14/09/12 17:00
Repasar la moderació de incondicionales.com	1 day?	14/09/12 8:00	14/09/12 17:00
Mirar l'implementació de la simulació web de la xarxa social	1 day?	14/09/12 8:00	14/09/12 17:00
Llegir la documentació aportada per JAR sobre xarxes socials	1 day?	14/09/12 8:00	14/09/12 17:00
Implementar exemple Repast	4 days?	14/09/12 8:00	19/09/12 17:00
Buscar i documentar informació sobre diferents tipus de moderació	4 days?	25/09/12 8:00	28/09/12 17:00
Implementar la part visual de simulació amb i diferents tipus d'agents	13 days?	28/09/12 8:00	16/10/12 17:00
Definir tipus de continguts, queixes i agents.	9 days?	28/09/12 8:00	10/10/12 17:00
Implementacio en java dels diferents tipus de comentaris	12 days?	28/09/12 8:00	15/10/12 17:00
Implentacio en java dels diferents agents	1 day?	28/09/12 8:00	28/09/12 17:00
Conectar la BBDD amb la simulació	2 days?	15/10/12 8:00	16/10/12 17:00
Creació dels icones dels agents i els tipus de comentaris	13 days?	28/09/12 8:00	16/10/12 17:00
Implementacio mysql de la base de dades	4 days?	16/10/12 8:00	19/10/12 17:00
Instalar i configurar servidor local amb XAMPP	2 days?	16/10/12 8:00	17/10/12 17:00
Crear script de creacio de BBDD	3 days?	16/10/12 8:00	18/10/12 17:00
Crear script de dades per la BBDD	3 days?	16/10/12 8:00	18/10/12 17:00
Testejar funcionament de la BBDD amb phpmyadmin	4 days?	16/10/12 8:00	19/10/12 17:00
Adaptar simulació perquè els agents fagin el post en seccions	5,5 days?	19/10/12 13:00	26/10/12 17:00
Definir colors pels seccions	2,5 days?	19/10/12 13:00	23/10/12 17:00
Separar el grid-world en seccions	2,5 days?	19/10/12 13:00	23/10/12 17:00
Crear imagnes de titulos	1 day?	19/10/12 13:00	22/10/12 13:00
Arreglar imagnes titulos	1,5 days?	25/10/12 13:00	26/10/12 17:00
Añadir seccion en el contendio de la BBDD	1,5 days?	25/10/12 13:00	26/10/12 17:00
Modificar codi de post en seccions ya que ahora estará en BBDD	1,5 days?	25/10/12 13:00	26/10/12 17:00
Redefinir el UPLOAD profile para que que un agente pueda hacer todos los tipos de compr	0,5 days?	26/10/12 13:00	26/10/12 17:00
Implementar VIEW profile en los agentes	1 day?	26/10/12 8:00	26/10/12 17:00
Implementar COMPLAINT profile en los agentes	1 day?	26/10/12 8:00	26/10/12 17:00
Arreglar el proyecto	2 days?	27/10/12 8:00	30/10/12 17:00
Ordenar y renombrar proyecto, packages...	2 days?	27/10/12 8:00	30/10/12 17:00
Crear PSD para las imagnes, iconos etc...	1 day?	27/10/12 8:00	29/10/12 17:00
Crear diferents tipus llistes per el viewprofile	6,25 days?	30/10/12 15:00	7/11/12 17:00
Possar random els upload i no a cada tick	6,25 days?	30/10/12 15:00	7/11/12 17:00
Implementar diferents configuracions de possar comentaris, visitar-los i queixar-se	6,25 days?	30/10/12 15:00	7/11/12 17:00
Acabar d'implementar el viewProfile amb diferents distribucions i llistes	9,25 days?	7/11/12 15:00	20/11/12 17:00
Canviar el modo d'execució multithread dels scheluled anotacion dels agents a un únic thr	4,25 days?	7/11/12 15:00	13/11/12 17:00
Arreglar bug los comentarios totales de los agentes	1 day?	14/09/12 8:00	14/09/12 17:00
Implementar distribución normal para viewprofile de las listas	5,25 days?	20/11/12 15:00	27/11/12 17:00
Integrar la nueva versió de la finestra de configuració d'agents	5,25 days?	20/11/12 15:00	27/11/12 17:00
Documentar i redactar la memòria del projecte	26 days?	1/12/12 8:00	7/01/13 17:00

Figura 11: Imagen donde se muestra el plan de trabajo de este TFG. Este plan de trabajo va por días disponibles para hacer cierta tarea. En él no se quiere recojer las horas que se han invertido para el trabajo.

1.7. Motivación personal para la realización del presente TFG

Para explicar la motivación del problema nos remontaremos a la adolescencia.

Una de las decisiones más difíciles que una persona adolescente tiene que tomar es qué quiere estudiar ya que el día de mañana es muy probable que tengas que ganarte la vida con los estudios que tienes.

Las primeras decisiones académicas empiezan en la ESO cuando uno tenía y tiene que elegir que optativas escoger. Pero como muchas de las decisiones de la vida esta decisión no tiene porque tener un impacto grande en la vida de los adolescentes. Muchas veces estas optativas sirven para hacer más ameno las asignaturas obligatorias de forma lúdica o siguiendo dinámicas de trabajo más alejadas de otras asignaturas.

Las primeras decisiones académicas importantes empiezan cuando hay un cruce de caminos. Uno camino te lleva hacer un ciclo formativo y otro hacer el bachillerato. Muchas veces esta decisión está condicionada con las ganas y la motivación que tienes para estudiar o con tener un objetivo académico que va más allá de aprobar la ESO.

Vas aclarando las dudas que tienes sobre la meta académica quieres alcanzar y te decides por estudiar un Grado de Ingeniería Informática por la Universidad de Barcelona.

Como muchas de las cosas de la vida, la informática abarca muchísimas cosas. La carrera te da una visión general de muchas de las cosas que puede llegar hacer y trabajar un informático. Y aunque muchas personas tienen un concepto equivocado o simple de lo que puede hacer un informático; como sería el de arreglar una impresora o un ordenador, somos capaces de muchas cosas más.

Por otra parte, la visión general que das en la carrera de muchos de los aspectos o cosas relacionadas con el mundo de la informática es a veces muy general. Hay cosas que aunque inicialmente desconocías su ámbito o alcance te acaban agradando más de lo que al principio creías. Usualmente sucede que el plan docente de la asignatura dice lo que harás durante el curso pero a veces tienes una visión diferente del que finalmente es.

Allá por el 2011 haría una de la asignaturas que más me ha gustado y que el simple hecho de haberla cursado se ha convertido en una de las principales piezas del puzle que me ha llevado a decidirme por hacer este trabajo. Y es que hay asignaturas que te gustan más que otras y otras que simplemente las haces porque son necesarias para aprobar. Normalmente las asignaturas que más te gustan a su vez son las que sacas más notas y son con las que más aprendes. Dicha asignatura fue Inteligencia artificial dada la teoría por Maite López y las prácticas por Javier Morales.

El camino para aprobarla no fue sencillo, requirió horas de estudio y horas de prácticas. Todo esto hizo que aumentaran con el paso de las horas de trabajo la estima y las ganas en trabajar con proyectos relacionados con el mundo de la inteligencia artificial. Y aunque la mayor dificultad de las practicas con el pequeño Pacman era el de entender un código que había creado otra persona, ahí se daban las primeras pinceladas de cómo pequeños algoritmos hacían que el Pacman pudiera conocer la mejor ruta para coger las bolas blancas o incluso fuera capaz de aprender a medida que iba jugando partidas. Hicimos el Pacman inteligente.

La primera pieza del puzle fue hacer Inteligencia Artificial pero eso no fue suficiente. Un puzle raramente se compone de una sola pieza, sino la forma varias. Aunque sueltas pueda parecer que no tienen relación luego encajan entre si dando forma a proyectos como este.

Llevado por la curiosidad y con las ganas de que mi proyecto de fin de carrera fuera relacionado con el mundo de la Inteligencia Artificial y antes de que empezara el último curso de la carrera hablé con la profesora Maite López.

De los varios proyectos que menciono que iba a publicar para que los alumnos pudieran trabajar en su proyecto de fin de carrera, hubo uno que destacaba de los demás. Era un proyecto que no pretendía continuar un código o un proyecto ya iniciado sino que iba a ser el principio de extensión de una idea que se hallaba en el proyecto de tesis doctoral del Javier Morales.

Poco a poco iban apareciendo e uniendo las piezas del puzle, formando el proyecto en el cual he trabajado y en el cual esta memoria quiere recoger con palabras y diagramas.

El proyecto estaba relacionado con el mundo de Internet y con la publicación de información que se produce en él. Otra pieza clave fue que el proyecto tocaba uno de esos puntos que personalmente encuentro que carece Internet y más concretamente las comunidades virtuales, que es el de moderaciones automáticas de contenidos y usuarios en comunidades virtuales.

Soy de los que piensa que las comunidades crecen y tienen éxito gracias a los usuarios y que los usuarios tiene que estar contentos en permanecer en ellas. Y es que la felicidad no solo consiste en poder acceder a un sitio o comunidad, sino que al permanecer en él no te resulte ofensivo o desagradable. El sentimiento que tendría que producirse al permanecer a una comunidad es el de alegría o felicidad, y eso se produce cuando uno está a gusto donde esta.

Esto hace que muchos usuarios de comunidades virtuales no estén realmente a gustos en ellas y que formen parte de dichas comunidades o porque tienes a sus conocidos ahí o porque les aporta algo que en otro sitio no encuentran.

Uniendo todas las piezas anteriormente mencionadas me han llevado a hacer este trabajo relacionado con la moderación automática de comunidades virtuales.

1.8. Estructura de la memoria

Esta memoria se estructura en diferentes capítulos. Esos capítulos son los siguientes:

1. Introducción

El primer capítulo es la introducción. En este capítulo se contextualiza el TFG y se define el problema para posteriormente poder sacar los objetivos de este trabajo.

Por otra parte, también hay una parte que hace referencia a la motivación personal.

2. Estado del arte

Segundo capítulo donde se hace un breve recorrido por proyectos que tienen relación con los objetivos que persigue este TFG. Como se verá en esa parte hay varios sistemas que tienen varios puntos en común con este trabajo. Después de mostrar y explicar cómo funcionan estos sistemas se extraen unas breves conclusiones sobre estos proyectos y sobre el parecido o los puntos en común con este proyecto.

En este capítulo también hay un espacio donde se definen conceptos relacionados con la moderación. Por una parte se define el concepto de moderador y por otra parte se clasifica de forma sistemática los diferentes tipos de moderación.

3. Diseño

El tercer capítulo es el diseño. Este capítulo hay toda la parte del diseño del trabajo. Se detalla la arquitectura del proyecto a nivel de ficheros y de esquema conceptual de los elementos que lo forman. También se define el sistema multi-agente que se pretende simular. Una vez definido cómo es y será el sistema multi-agente viene la parte de como se simulará. La parte de la simulación es una parte donde se explican consideraciones de diseño que tiene el simulador y todo aquello que se puede encontrar en él. Incluso esta parte muestra aspectos gráficos del simulador cómo puede ser iconos de determinadas cosas. Otra parte que se muestra es la que hace referencia al modelo entidad-relacional de la base de datos y los diagramas de casos de uso. Finalmente se explican los casos de uso de forma más detallada.

4. Desarrollo

Inicialmente este capítulo muestra los entornos que se han utilizado en este proyecto y se cuenta parte de la implementación de este proyecto. También se detallan algunos conceptos y se consideraciones para implementar ciertas funcionalidades con el Repast. Finalmente se muestran errores aparecidos en el desarrollo del trabajo y que soluciones se han llevado a cabo.

5. Resultados

Quinto capítulo donde se dan los resultados finales del simulador.

6. Conclusiones

Finalmente el sexto capítulo se dan las conclusiones del trabajo y se menciona los trabajos futuros relacionados con este proyecto que empezarán una vez termine este proyecto. También hay un espacio de trabajos futuros relacionados con la inclusión o la mejora de elementos del simulador.

Bibliografía

Parte en que se muestra el material externo de consulta utilizado.

Anexos

Parte que se muestra cómo se importa la base de datos y parte dónde hay un pequeño manual de uso de la aplicación.

2. Estado del arte

2.1. KeepCon

KeepCon es una empresa de Buenos Aires (Argentina) que nació alrededor del 2009 como unión de dos personas (Julio Guzman y Frenchman) que tenían intereses comunes en un mundo bastante desconocido. Estuvieron investigando durante un periodo de tiempo y descubrieron que apenas había camino recorrido en este sentido. Con todo esto decidieron crear un sistema de moderación automática de contenidos generados en comunidades virtuales cuyo nombre es el mismo que el de la empresa.

El sistema se divide en varias etapas. En la imagen de continuación se resumen cada una de las diferentes etapas.

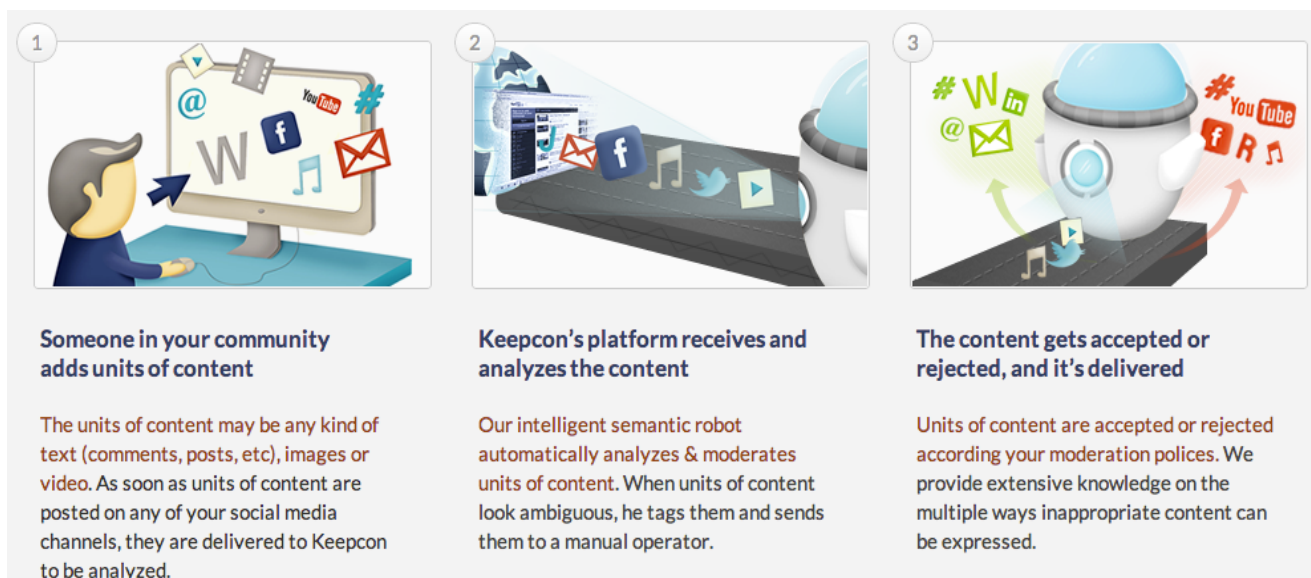


Figura 12: Fases de funcionamiento del sistema Keepcon.

1. La primera etapa es en la que el usuario genera contenido en una comunidad virtual. Dicho contenido pasa al Keepcon para ser analizado y clasificado.

2. El Keepcon recibe el contenido y lo analiza. El proceso de análisis para saber si este contenido se acepta o no a su vez se divide en diferentes etapas:

2.1. Etapa de comprensión: Es una etapa nada sencilla ya que internet el idioma no se usa del todo correcto. Se usan palabras con diferentes letras o caracteres para expresar lo mismo (como por ejemplo love y lov3), y muchas veces se usan palabras camufladas para insultar, que una persona normal interpreta que le están insultando pero que muchos sistemas automáticos no lo detectarían.

2.2 Etapa de clasificación: En esta etapa se combinan dos técnicas basadas en la inteligencia artificial.

2.2.1 Tecnología simbólica (Symbolic technology). Se redactan unas reglas de clasificación en función de las necesidades del cliente. De esta manera se establece que cosas están permitidas o no. (por ejemplo si no queremos ver se venden productos en nuestra comunidad).

2.2.2 Máquinas de aprendizaje (Learning machine). Entrenan a unos algoritmos con textos que pertenecen a un corpus. Luego testean el algoritmo con casos reales para medir la precisión y el reconocimiento.

El robot clasifica los contenidos y esto se complementa con un sistema manual de moderación (con moderadores humanos).

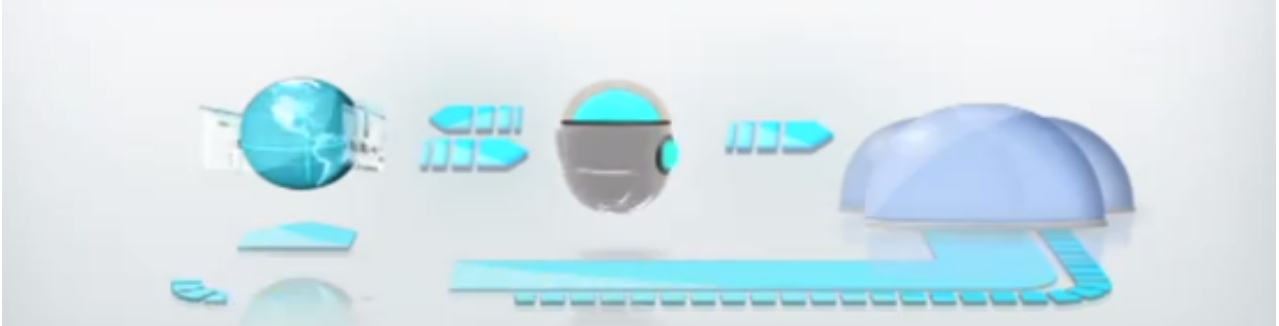


Figura 13: Diagrama que representa como el robot coje contenido de la red y lo gestiona. Algunos de estos contenidos los envía al Manual Moderation Factory para que personas humanas los examinen. Hay otros contenidos que el propio robot ya descarta si son claramente nocivos.



Figura 14: El Manual Moderation Factory es un sistema compuesto personas humanas para determinar si los contenidos son correctos o no. Para facilitar esta tarea el robot previamente ha categorizado los contenidos.

Por otra parte también usan la moderación para detectar sentimientos y detectar cosas inapropiadas como puede ser contenido ilegal donde aparecen menores.

Los precios varían según el volumen de contenido y de la comunidad.

En la web también sale un documento detallado de cómo podemos integrar Keepcon.

Hay varias grandes empresas que ya usan Keepcon cómo son: Walt Disney, Páginas amarillas, Mercado libre, Terra, Europa press...

<http://keepcon.com/>

2.2. Stilus Forum



Figura 15: Logo de la compañía Stilus Forum.

Stilus forum es un producto de la empresa DAEDALUS - DATA, DECISIONS AND LANGUAGE, S.A.

La empresa esta especializada en actividades entorno a las tecnologías de búsqueda, tecnologías del lenguaje y en la gestión avanzada del conocimiento (business intelligence).

Este producto esta basado en el filtrado automático de mensajes según su contenido

Y entre las principales características del producto cabe destacar:

- Filtrado de contenidos ofensivos, ilegales, inapropiados o, de alguna manera, objetables
- Varios niveles de filtrado (más o menos estricto)
- Cobertura del español peninsular y de las variantes del español de Latinoamérica
- Reconocimiento de textos en lenguaje abreviado, tipo SMS
- Integrable en gestores de contenidos o como servicio
- Procesamiento en tiempo real

La empresa no da muchos más detalle de su funcionamiento ni hay ningún video que se haga una demostración o se explique el proceso que utilizan para su funcionamiento.

<http://www.daedalus.es/blog-y-recursos/sobre-stilus/stilus-forum-moderacion-automatica-de-foros-de-participacion/>

2.3. SourPanel

SourPanel es un producto de la empresa Sourtech. Sourtech es una empresa española que tiene sedes en varios países de todo el mundo. Su modelo de negocio se divide en 4 partes (moderación, programación, contenidos, data-entry).

La parte de moderación según sus datos del 2010 moderaba 9.000.000 de contenidos. Tienen un sistema no 100% de moderación automática. Su sistema se divide en una parte previa de moderación automática y luego una moderación manual por personas.

Software	Personal
Unificar mensajes de diferentes fuentes (noticias, blogs, foros...)	Personal especializado y dedicado solo a la moderación.
Elimina mensajes claramente nocivos para la comunidad.	
Marca el contenido por temática para facilitar la faena al personal.	
Marca las expresiones o usuarios peligrosos para su inmediata atención	
Estadísticas detalladas de participación	

Delatan que se observa un claro aumento de la fluidez de conversaciones en una comunidad virtual cuando se aplica su sistema.

El sistema proporciona un panel para la administración con todas las opciones disponibles.

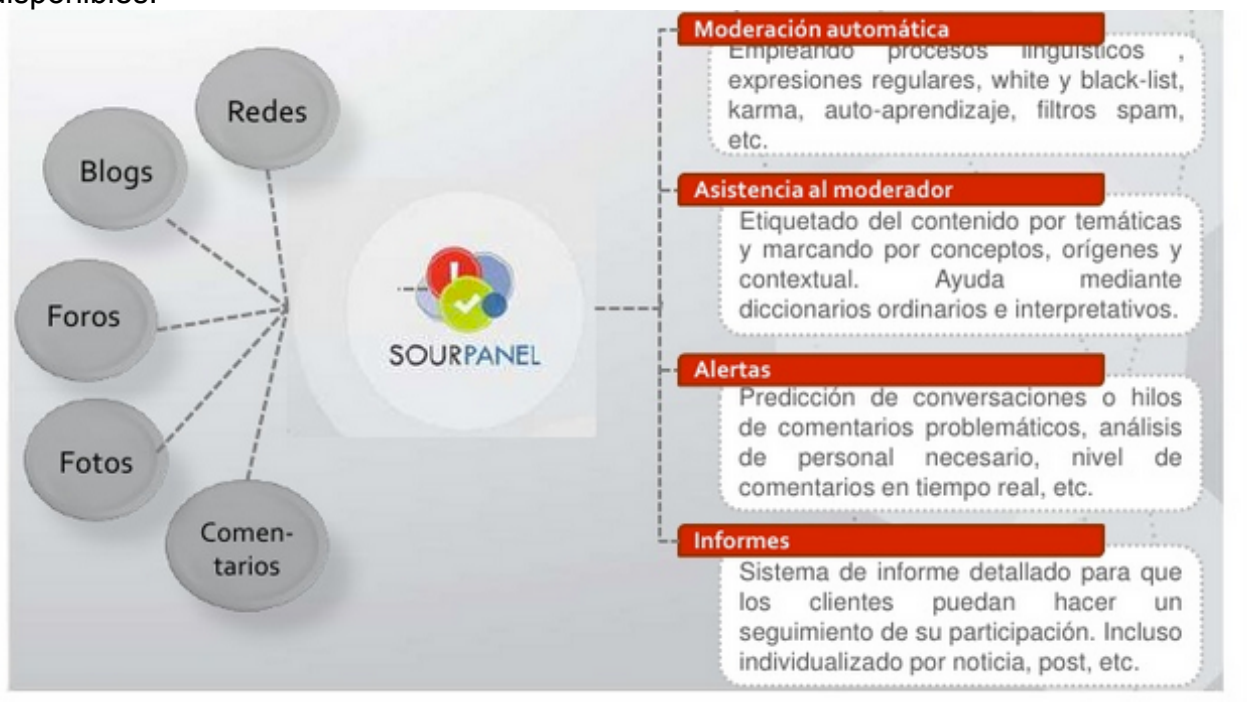


Figura 16: Panel de administración SOURPANEL. En el se dan herramientas para facilitar la moderación.

Y el proceso de trabajo del sistema es el siguiente:



Figura 17: Diagrama donde se muestra el funcionamiento del sistema y como los contenidos se descartan o se admiten.

Como vemos en el diagrama los contenidos pueden salir del SourPanel ya filtrados o simplemente clasificado para que un moderador humano dicte si ese contenido es aprobado o no lo es. Si es aprobado el contenido se queda en la web sino lo es el contenido se descarta de la web.

<http://www.sourtech.com/quienes-somos/>

2.4. Definición de moderador y tipos de moderación

2.4.1 La figura del moderador

Un moderador es un usuario o un empleado de un lugar; donde se relacionan las personas, que trabaja con la función de mantener un ambiente cordial y agradable entre todos los usuarios.

Su papel es el de detectar y dictar que reglas o normas; que la red tiene, han sido quebrantadas por un usuario o por los contenidos que dicho usuario a generado. También tiene un papel; aunque menor importante, de soporte o ayuda a determinadas dudas de los usuarios.

Entre las principales capacidades de un moderador hay la de editar, borrar, crear o mover contenidos en la red que hayan producidos otros usuarios.

En muchos casos y cuando la comunidad virtual que se desea moderar es grande existen organismos jerárquicos o sectoriales de moderadores. En esta jerarquía existen diferentes tipos o niveles de moderadores. En cada nivel los moderadores tienen unos privilegios o permisos para hacer determinadas acciones. Este tipo de sistema existen rangos que determinan el nivel que esta un moderador. Según el papel que tienen y de su importancia en la moderación de la red social los moderadores adquieren o suben de rango en la estructura jerárquica, adquiriendo nuevas responsabilidades y permisos en la comunidad.

2.4.2. Tipos de moderación

Hay diferentes tipos de moderación que se pueden separar según en que momento en que se hace el control de los contenidos (antes o después que sea publicado) y quién hace este control (humanos de forma manual/automática o máquinas de forma automática).

Pre-moderación

La pre-moderación es una dinámica de moderación que consiste en moderar el contenido antes de que sea publicado; visible por los otros usuarios. Es un sistema que el control se hace antes de ser publicado y en consecuencia todo lo que se publica ha pasado por estos filtros y se considera contenido adecuado. Sino pasa por este control o moderación, el contenido se descarta.

Esta dinámica de moderación no es muy habitual, ya que el tiempo entre que el usuario genera contenido hasta que este es revisado no es inmediato. Los usuarios normalmente quieren inmediatez en todo lo que hacen y publican. Por lo tanto, a los usuarios les gusta que todo se haga al momento y no en unos lapsus de tiempo.

Por otra lado, hay determinados casos que esta dinámica si que se utiliza, como en comunidades de alto riesgo legal; como puede ser famosos y temas relacionados con los menores.

En comunidades donde se informa sobre los famosos se tiene que controlar como se ha obtenido, de que forma, quién afecta y si dicho contenido infringe alguna ley o es castigado de alguna manera por los poderes judiciales.

En temas de niños la pre-moderación se utiliza para frenar casos de pederastia o de abuso de menores.

Post-moderación

La post-moderación es una dinámica de moderación que consiste en moderar el contenido después de ser publicado. Una vez el usuario genera nuevo contenido se le hace una moderación; un filtro. La moderación de este contenido puede tener dos resultados; o el contenido es correcto y se mantiene publicado o se suprime por no serlo. Las ventajas y inconvenientes difieren considerablemente de la dinámica de la pre-moderación.

La post-moderación aporta inmediatez a los contenidos que genera el usuario. El usuario genera contenido e inmediatamente se publica, generando debate, comentarios, controversia o todo tipo de contenidos de otros usuarios. La inmediatez es una característica que comparten la mayoría de comunidades virtuales, sobretodo en las que se mantienen conversaciones entre los usuarios. De todas formas no son todas ventajas para este tipo de moderación ya que tiene el inconveniente del tiempo de demora que tardan los moderadores en revisar el contenido y dictar si se tiene que suprimir o no. Este espacio de tiempo suele ser elevado y depende mucho del número de moderadores que hay y del contenido que tiene que moderar. Este problema aumenta considerablemente cuando más grande es la comunidad de usuarios que se quiere moderar, ya que el número de los contenidos que se genera en dicha comunidad también aumenta.

Tabla comparativa de la pre-moderación y la post-moderación

	¿De que es trata?	Ventajas	Desventajas
Pre-moderación	Revisión de los contenidos antes de ser publicados.	Publicación solo de contenido apropiado o adecuado.	Alto tiempo e espera del usuario hasta ver su contenido publicado..
Post-moderación	Revisión de los contenidos después publicados.	Buena experiencia de usuario (inmediatez del contenido que genera). Proceso simple i económico.	Alto tiempo de exposición de contenidos inapropiados. Riesgo de tener publicado contenidos que no están revisados.

2.5. Conclusiones del estado del arte

Haciendo una búsqueda para determinar el estado del arte me ha llevado a la conclusión que hay pocos y escasos sistemas de moderación automática de comunidades virtuales. Estos están más orientados a detectar y tratar contenidos de diferentes tipos, que a generar automáticamente normas a partir de las quejas de los usuarios.

También mencionar que hay poca información al respecto ya que es algo bastante novedoso y que la mayoría de los sistemas tienen menos de 10 años de vida. Cabe mencionar también que una moderación automática 100% no es contemplada en ninguno de los casos ya que no da mucha seguridad y no es bien visto por la comunidad de usuarios que un robot determine si lo que expresan es correcto o no.

Todos estos sistemas la moderación automática que se han analizado se utilizan para facilitar las tareas a los moderadores humanos, clasificando contenido, alertando de forma clara de contenidos o usuarios peligrosos o descartando contenidos que claramente son nocivos para la comunidad virtual.

3. Diseño

3.1 MAS

3.1.1. Definición

Un sistema MAS⁶ consiste en un conjunto de agentes inteligentes interactuando entre si dentro de un entorno. En este proyecto nuestro sistema MAS será el de una pequeña comunidad virtual de seguidores de fútbol donde los agentes serán los usuarios.

Por lo tanto, el entorno del MAS es un entorno una comunidad virtual de seguidores de fútbol.

3.1.2. Roles

Tendremos dos roles diferenciados.

En primer lugar consideramos los agentes de tipo usuario que son los que interactuarán dentro la comunidad virtual generando, visitando y quejándose de contenidos. Este rol tiene un perfil de agente asociado.

Por otro lado, tenemos la figura del moderador. Aunque en la simulación no esté implementado este rol de usuario hay que mencionarlo igualmente porque es parte de la definición de este MAS y será implementado e introducido en este sistema en extensiones de este proyecto. Los moderadores son los que mirarán que se cumplan las normas que salgan de esta simulación y aplicaran las acciones necesarias cuando éstas sean quebrantadas.

3.1.3. Acciones del rol de usuario

La interacción de los agentes no será directa entre agentes sino que ellos solo interactuaran con el contenido generado por otros agentes. Toda interacción será a través de contenidos. Así que las acciones permitidas por los agentes serán las siguientes:

- Subir contenido en la comunidad virtual.
- Ver contenido que haya en la comunidad virtual.
- Quejarse de contenido que haya en la comunidad de otros usuarios.

3.1.4. Acciones del rol del moderador

El moderador interactuará con el contenido y los agentes de la comunidad virtual. Su interacción se hará a modo de “juez” y las acciones que podrá hacer son:

- Avisar a usuario de su comportamiento incorrecto.
- Expulsar a un usuario.
- Eliminar contenido.
- Reubicar contenido.

⁶ MAS: Acrónimo de sistema multi-agente (Multi agent system)

3.1.5. Tipos de perfiles del usuario

Hay diferentes tipos de agentes que juegan el rol de usuario y cada tipo pretende recoger de forma explícita una caracterización de un tipo de usuario de una comunidad de seguidores de fútbol. De esta manera el perfil define el comportamiento de un tipo de usuario de la comunidad virtual. Cabe mencionar además que los perfiles se usan también para definir diferentes tipos de poblaciones de agentes de la comunidad virtual.

Finalmente se han creado agentes muy generales y que difieran entre ellos en la manera de comportarse en una comunidad virtual. Este comportamiento se recoge en su perfil de agente.

Perfil del agente (Agent profile)

El perfil de un agente está dividido en tres subperfiles; un perfil por cada acción que un agente puede hacer. Estos perfiles son los siguientes:

- **Perfil de subida de contenido (UploadProfile)**

Perfil que recoge el comportamiento que tiene un agente en el momento que tiene que subir un nuevo contenido a la comunidad virtual. Este perfil está dividido en dos partes. Una parte recoge la frecuencia con que un agente sube contenido (upload frequency). Este parámetro es un porcentaje que puede oscilar entre 0%-100%.

Por otro lado, la otra parte que tiene este perfil es el porcentaje de contenido que se sube de cada tipo(ver sección 3.2.2 apartado "Tipos de contenido" de la memoria). El total del porcentaje de los diferentes tipos de contenidos que va a subir oscila entre 0%-100%.

- **Perfil de visita de otros contenidos (ViewProfile)**

Perfil que recoge el comportamiento que tiene un agente cuando mira contenido de la comunidad virtual. Los agentes pueden tener diferentes preferencias cuando visitan unas determinadas secciones o cuando miran un determinado tipo de contenido por encima de otros contenidos o secciones. Este perfil está dividido en dos partes.

La primera parte es la que se expresa en qué secciones el agente prefiere mirar (section view). Como se ha explicado hay tres secciones diferentes(ver sección 3.2.2 apartado "Tipos de secciones" de la memoria). Este parámetro recoge la frecuencia en la que el agente visitará cada sección. Este parámetro oscila entre 0%-100% del total de las diferentes secciones.

Por otro lado tenemos el modo de visita del agente. Esta parte recoge en que modo el agente visita el contenido de una determinada sección. El agente quizá le interese visitar los últimos contenidos generados en la comunidad virtual o quizá le interese visitar los contenidos que son más populares (que tienen más visitas) o mirar contenido de forma aleatoria. Por este motivo se han definido estos tipos de visitas: últimos contenidos (ordenados), por el número de visitas o de forma aleatoria.

- **Perfil de quejas de contenidos (ComplaintProfile)**

Perfil que recoge con qué frecuencia el agente se queja de un determinado tipo de contenido. El agente visita un contenido y según del tipo que sea y de la frecuencia de quejas de sobre ese tipo de contenido, el agente se queja o no.

Estructura del perfil de agente

Agent profile

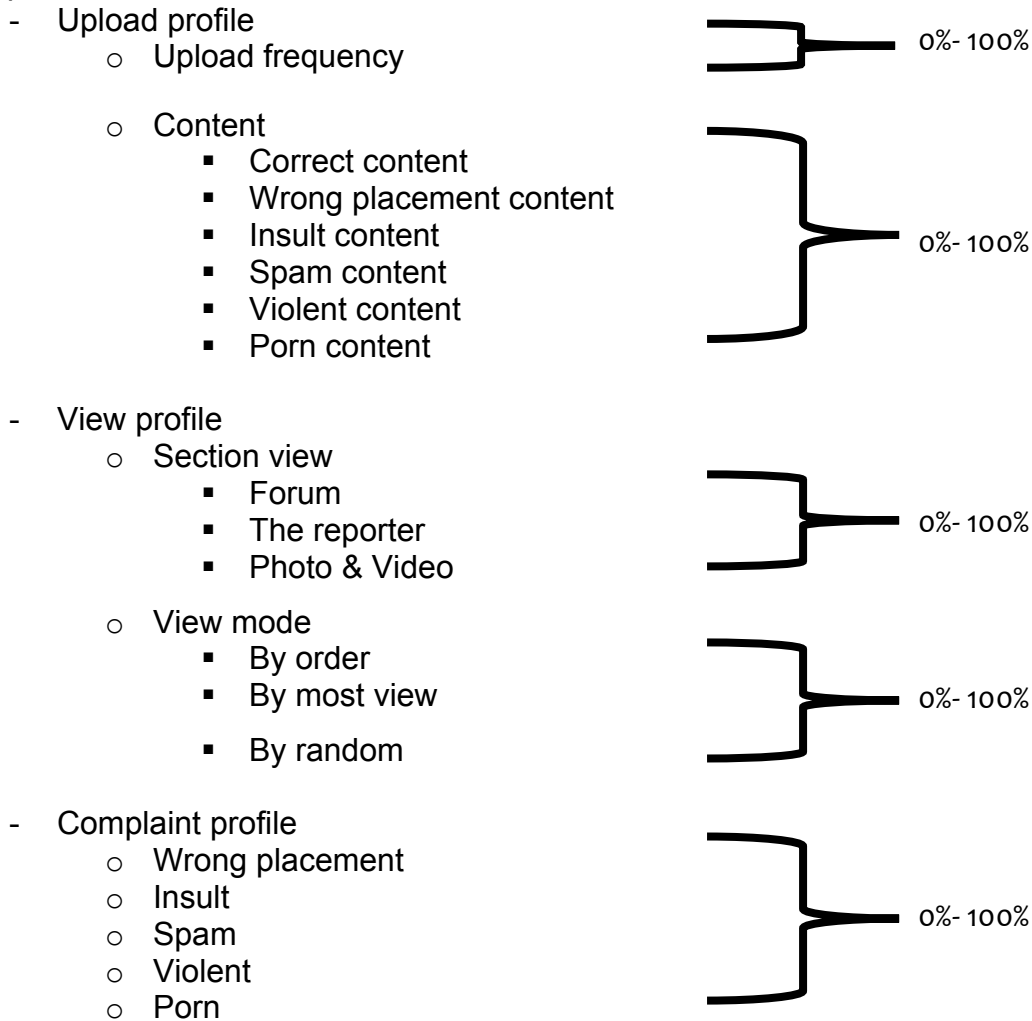


Diagrama 1: Se muestra la estructura del perfil de un agente y los subperfiles.

Como se ve en el diagrama, un agente tendrá un perfil de agente que se dividirá en subperfiles (uploadProfile, viewProfile i complaintProfile). Cada tipo de agente tendrá unos parámetros determinados en sus subperfiles de agente que van en función de su comportamiento en la comunidad virtual.

3.1.6. Ontología

La ontología es todo el lenguaje o palabras que el sistema y los agentes que forman parte de él conocen. Por otro lado, la ontología también sirve para definir la simulación, el entorno o contexto de aplicación. Dentro de nuestra simulación destacamos los siguientes conceptos:

Content: contenido que se sube en al comunidad virtual.

Complaint: queja de un contenido de la comunidad virtual.

Views: visitas que recibe de los agentes de un contenido.

Date of upload: fecha en que se ha subido un contenido.

3.2. Simulación del MAS

3.2.1. Arquitectura del proyecto

La simulación del MAS y este TFG es una pequeña parte de un problema mayor como ya se ha explicado. Esta es la primera etapa del proyecto mayor.

Para ver en qué parte encaja el simulador de comunidad virtual y para qué se quiere, a continuación se muestra un diagrama la arquitectura general.

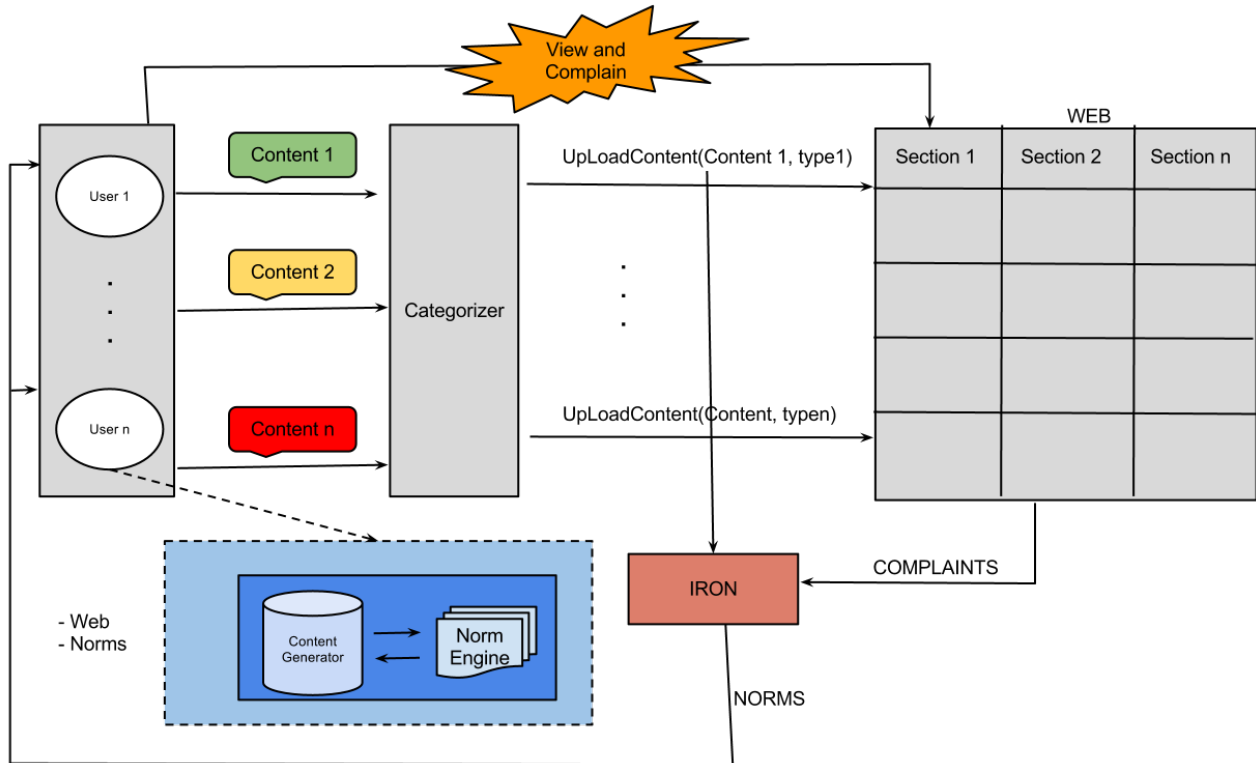


Figura 18: Diagrama en que se muestran las partes del proyecto general.

El simulador es el primer paso para todo el sistema de moderación automática en comunidades virtuales. El primer paso era el de crear un simulador de comunidad virtual en el que se pudiera experimentar y ejecutar paso a paso.

La parte de arriba del diagrama de tono gris es más o menos la parte que se ha creado con el simulador; excepto que se ha decidido no usar de momento un clasificador de contenidos y cogerlos directamente de BBDD ya categorizados.

La parte inferior hace referente más a la parte que otro compañero implementará a continuación de este TFG. Es la parte en que se generan las normas y que posteriormente se aplicaran. IRON es el método de la tesis doctoral del Javier Morales. Es un sistema que se encarga de generar las normas en función de las quejas que vayan teniendo los diferentes contenidos.

Una vez se ha visto que parte encaja el simulador se pretende explicar de forma visual que arquitectura tiene la simulación del MAS y este TFG.

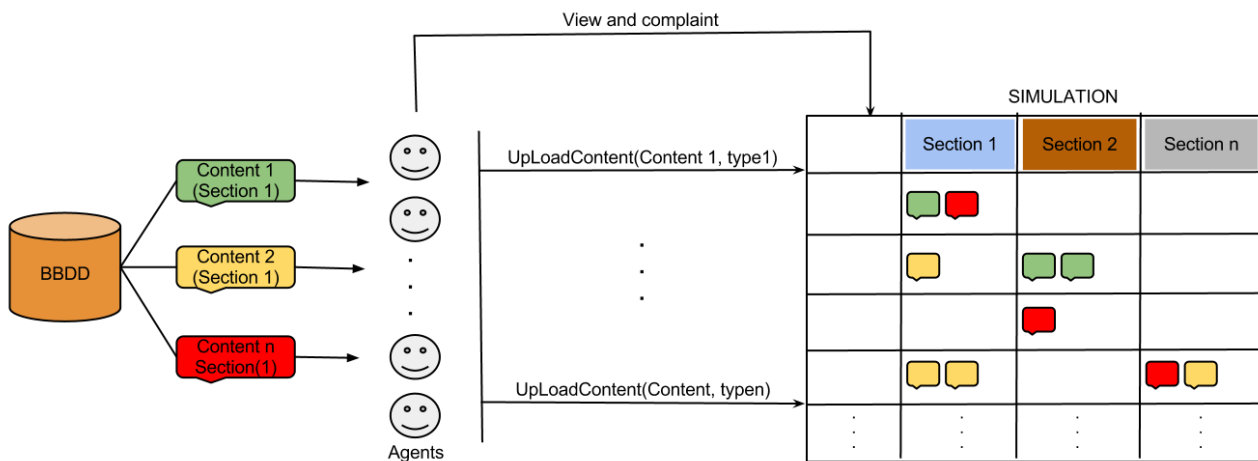


Figura 19: Diagrama donde se muestran las partes que componen el sistema y este TFG.

Como se ha detallado en la definición del problema, se ha tomado la decisión que todo el contenido esté categorizado por tipos en una base de datos. Esto es una simplificación del problema ya que se carece de sistemas de categorización automáticos. De esta manera tenemos contenido en una base de datos que ya está categorizado y que pertenece a una sección determinada.

Los agentes lo que hacen es coger contenido de esta base de datos y lo ponen en el simulador. A parte de subir contenido a la base de datos los agentes también visitan y se quejan de otros contenidos que haya en el simulador.

Como vemos donde acaba este simulador empieza el otro TFG que hará otro compañero ya que este TFG acaba cuando se generan quejas de contenidos; parte final de la simulación y cosa indispensable para empezar a generar normas. La generación normas estará en función de las quejas.

3.2.2. Contenidos, secciones, quejas y usuarios

A continuación se explicarán, se definirán y se mostrarán los diferentes tipos de contenidos, agentes, quejas, secciones y sus iconos asociados en el simulador:

Tipos de contenidos

Los contenidos pueden ser principalmente de tres formatos (texto, vídeo o imágenes). Por otro lado los contenidos se clasifican según la categoría a la que permanecen.

Correcto

Contenido que cumple con las normas y el estilo de la comunidad virtual. Suele ser contenido que no recibe quejas por otros usuarios.

Spam

Contenido en el que es promociona a una entidad, una persona o algún producto. Suele ser muy controlado este tipo de contenido ya que recibe un fuerte rechazo por parte de la comunidad virtual, ya que es molesto. También suele publicarse en intervalos cortos de tiempo, seguidos o repetitivos.

Pornográfico

Contenido en el que se muestra escenas o personas desnudas o haciendo sexo de forma implícita o explícita.

Violento

Contenido que incita a la violencia o que se muestra explícitamente. Suele ser rechazado por la comunidad. Los insultos graves se consideren como un determinado tipo de violencia. También se considera violencia comentarios ofensivos.

Mal ubicado

Contenido que es correcto pero que no esta bien colocado; no se ha puesto en la sección correcta o que no tiene cabida en la comunidad virtual.

Insultos

Los insultos son un tipo de contenido que esta entre camino de comentario y de violencia. Es un tipo de insulto leve ya que los insultos graves se consideran como un tipo de violencia.

	Correcto
	Contenido mal ubicado
	Insultos
	Spam
	Violento
	Pornográfico

Tabla 1: Íconos de la simulación de los diferentes tipos de contenidos

Tipos de secciones

El contenido que publican los usuarios en la simulación se ordena en tres diferentes secciones dentro de la comunidad virtual. Esto hace que el sistema se enriquezca y que lo complique un poco más. El objetivo de los diferentes nombres de las secciones era el de mantener una pequeña similitud con los nombres de las secciones a los de la red de Incondicionales.com (red de seguidores de fútbol) y que el contenido se organizara en diferentes secciones. De esta manera tendría sentido usar el concepto de contenido mal ubicado cuando se hace referencia a contenido que es incorrecto solo porque esta en una sección equivocada y no porque dicho contenido no respeta las normas de la comunidad virtual.

El foro



Figura 20: Logo de la sección forum.

El foro es la típica sección que tienen la mayoría de comunidades virtuales. Es una sección donde la gente da su opinión sobre diferentes cuestiones que van surgiendo. En él suelen existir varios contenidos que generan mucho debate o discusión.

El reportero

The reporter

Figura 21: Logo de la sección the reporter.

El reportero es el tipo de sección que se postean noticias de fuentes contrastadas. Son noticias que pueden resultar de interés para los usuarios de la comunidad virtual. También suele ser contenido de tipo texto.

Foto y video

Photo & Video

Figura 22: Logo de la sección photo & video.

Esta sección sería la propia donde los usuarios ponen contenidos que no son de tipo texto. Estos contenidos pueden ser foto y video y pueden ser de temática variada siempre que trate con cosas relacionadas con las normas de la comunidad virtual.

Quejas

Las quejas están asociadas directamente a los diferentes tipos de contenidos. Todo el contenido que no tiene el icono de color verde, y en consecuencia lo tiene de color amarillo o rojo puede tener un tipo de queja asociado a ese tipo de contenido. Por lo tanto, las quejas serán del mismo tipo que el contenido.



Tabla 2: Icono de cuando se produce una queja

Tipos de agentes

- **Moderado (moderate)**

Agente que se comporta correctamente y no altera la convivencia dentro de la red. A su vez se podría segmentar en dos tipos: el activo o el pasivo. El activo sería el que participa activamente en la red, aportando contenido nuevo o comentando el que hay. El pasivo normalmente es un tipo de agente que sigue el día a día de la red pero no participa activamente, sino que simplemente es un observador de la red social.

- **Spammer**

Agente que hace publicidad constantemente de empresas, páginas o cosas de su interés.

- **Troll**
Agente entra en la red y ha entrado para buscar confrontaciones de forma inmediata. Su intención es incendiar de forma rápida la red. Normalmente este tipo de agente es contrario a les ideas o contenidos que se publiquen en la red.
- **Pornográfico**
Agente que normalmente publica contenido para mayores de 18 años o que aparecen menores. Se trata de contenido considerado como pornográfico.
- **Violento**
Agente que normalmente ataca a otros usuarios con insultos graves, faltas de respeto, comentarios ofensivos...
- **Mal educado (rude)**
Agente que no busca peleas de forma directa pero que suele crearlas. No suele hablar de forma normal y muchas veces se cofunde su manera de hablar con insultos a otros usuarios.







M 	Moderado
S 	Spammer
T 	Gnomo
P 	Pornográfico
V 	Violento
R 	Mal educado

Tabla 3: Íconos de la simulación de los diferentes tipos de agentes

3.2.3. Estructura de la Base de datos

Esta aplicación/simulador hace uso de una base de datos; como se ha mencionado reiteradamente, donde tenemos datos de contenidos categorizados o poblaciones de agentes para la simulación.

Se han creado dos scripts SQL para que se fácilmente manipulable y que se pueda hacer una instalación rápida y sencilla de la base de datos⁷.

Modelo Entidad-Relación

La base de datos esta compuesta de dos partes. La primera parte hace referencia a los contenidos, usuarios y secciones. Por otro lado, la otra parte hace referencia a la configuración de poblaciones de agentes que queremos tener en el simulador.



Figura 23: Modelo entidad-relación donde se muestra la parte de la base de datos encargada de las poblaciones.

La imagen anterior muestra la parte de la base de datos que hace referencia a las poblaciones de agentes. Una población de agentes es un conjunto de diferentes agentes que serán simulados. De esta manera si queremos tener diez agentes en el simulador tendremos una población total de diez agentes. Se ha creado una base de datos para las poblaciones para poder guardar y cargar diferentes tipos de poblaciones que hayamos creado. De esta manera, en cualquier momento podemos cargar una población que hemos guardado en la base de datos y hacer o repetir su simulación cuando nos interese.

Como se puede ver en el diagrama, una población esta compuesta por el tipo de agente, el nombre de la población que hemos puesto y con el perfil del agente. En el perfil de

⁷ En el Anexo 1 se detalla como hacer la importación de la estructura de la base de datos y de los datos que contienen en un servidor local.

agente hay todos los parámetros que hacen referencia a un agent profile como explicado anteriormente. Para simplificar el número de entidades y de relaciones de nuestra base de datos, los datos de uploadProfile, viewProfile y complaintProfile se han puesto directamente en el agent profile.

El diagrama siguiente muestra la parte de la base de datos que hace referencia a los comentarios, comentarios y usuarios de la simulación. Como vemos es una base de datos que tiene mucha cohesión. Los contenidos, quejas i secciones son de un tipo. Por otro lado un contenido es de un tipo y pertenece a una sección. Como se ha mencionado anteriormente, los contenidos ya los tenemos clasificados por tipos pero también los tenemos clasificados por secciones.

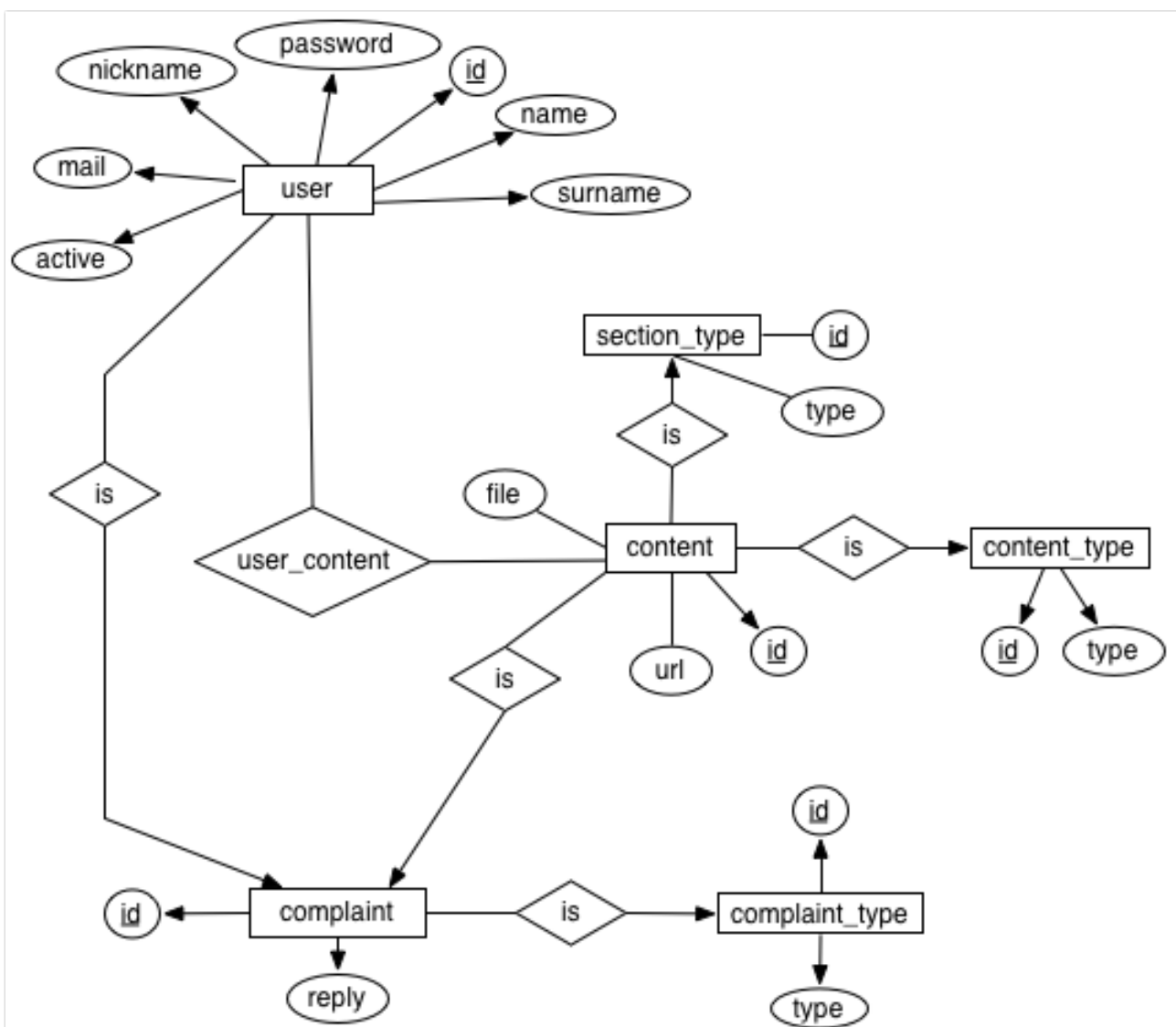


Figura 24: Modelo entidad-relación donde se muestra la parte de la base de datos encargada de administrar los contenidos ya clasificados, los tipos de secciones, las quejas...

3.2.4. Diagrama casos de uso

En el siguiente diagrama de uso se pueden ver las diferentes acciones o casos de uso que un usuario puede hacer con esta aplicación:

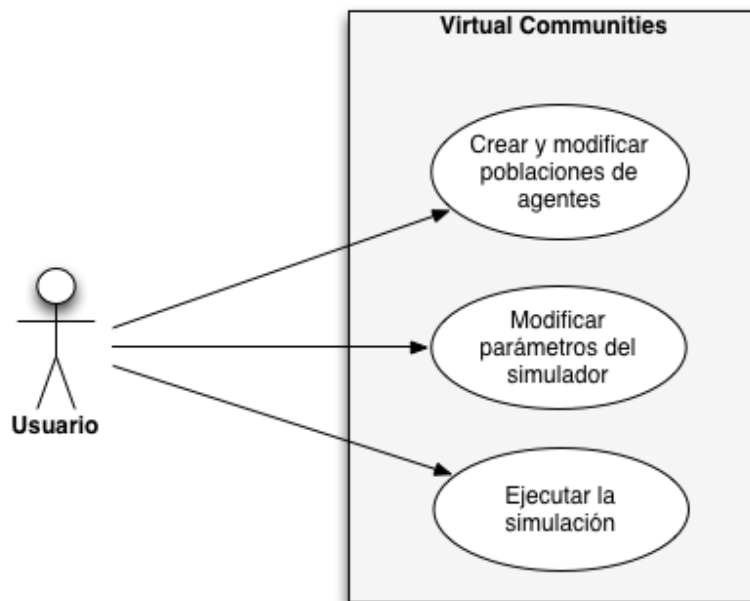


Figura 25: Figura donde se muestran los casos de uso del usuario.

3.2.5. Casos de uso

A continuación se describen los casos de uso:

UC1: Crear y modificar poblaciones de agentes
Descripción: El usuario puede modificar o crear poblaciones nuevas que se cargarán en el simulador.
Actores: Usuario
PreCondiciones: El sistema esta arrancado.
Flujo Básico: <ol style="list-style-type: none">1. El sistema muestra la ventana de configurar poblaciones.2. El usuario elige una población existente3. El sistema muestra los parámetros y numero de agentes de esta población.4. El usuario modifica algún parámetro.5. El usuario sale de la ventana de poblaciones6. El sistema inicializa la simulación con los agentes de la población elegida.
Flujo Alternativo: <ol style="list-style-type: none">2.1. El usuario escribe el nombre de la nueva población<ol style="list-style-type: none">2.1.1 El usuario modifica parámetros de agentes y los añade a la población.2.1.2 El usuario guarda la población.2.1.3 El usuario sale de la ventana de las poblaciones2.1.4 El sistema muestra la pantalla de simulación con los agentes de la población
PostCondiciones: La población ya esta en simulador cargada.

UC2: Modificar parámetros del simulador
Descripción: El usuario modifica parámetros del simulación (tiempo entre ticks, cuando acaba la simulación).
Actores: Usuario
PreCondiciones: El usuario ya ha seleccionado las poblaciones.
Flujo Básico: <ol style="list-style-type: none"> 1. El sistema muestra la simulación con la población cargada 2. El usuario pulsa la flecha izquierda que aparece en al subventana “Scenario tree”. 3. El sistema cambia la subventana “Scenario tree” por la de “Parameters”. 4. El usuario vuelve a pulsar la flecha izquierda que aparece en la subventana “Parameters”. 5. El sistema cambia la subventana “Parameters” por la de “Run options”. 6. EL usuario modifica los parámetros de “Run options” 7. El usuario arranca la simulación 8. El sistema hace la simulación con los parámetros elegidos por el usuario.
Flujo Alternativo: <ol style="list-style-type: none"> 2.1. El usuario arranca la simulación sin tocar los parámetros de la simulación. <ol style="list-style-type: none"> 2.1.1 El sistema simula con los parámetros por defecto.
PostCondiciones: La simulación esta arrancada.

UC3: Ejecutar la simulación
Descripción: El sistema hace la simulación. La interface de simulación es propia de Repast Simphony Java 2.0.
Actores: Usuario
PreCondiciones: El usuario ya ha seleccionado las poblaciones y ha salido de la ventana de selección de poblaciones.
Flujo Básico: <ol style="list-style-type: none"> 1. El sistema muestra la simulación con la población cargada 2. El usuario pulsa el boton “Start run”. 3. El sistema arranca la simulación sin parar. (a no se que haya modificado en el “Run Options” que se pare en un determinado tick.
Flujo Alternativo: <ol style="list-style-type: none"> 2.1. El usuario pulsa el botón “Step”. <ol style="list-style-type: none"> 2.1.1 El sistema simula el siguiente tick. 2.1.2 El usuario pulsa “Stop” <ol style="list-style-type: none"> 2.1.2.1 El sistema para la simulación.
PostCondiciones: La simulación ya se ha efectuado.

3.3 Estructura del proyecto

El proyecto esta hecho en Java. La siguiente imagen muestra la estructura del proyecto:

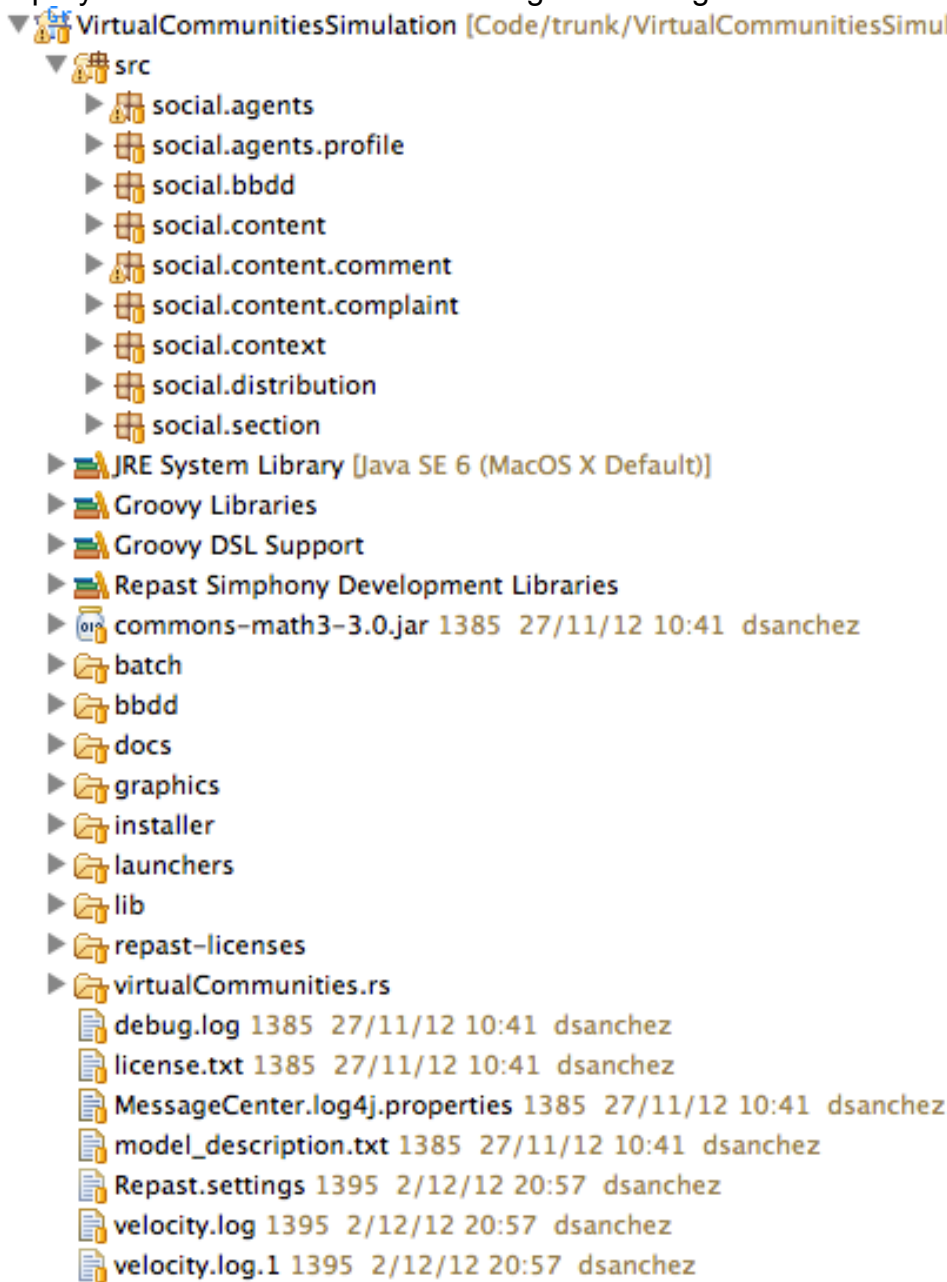


Figura 26: Imagen donde se muestra la estructura de ficheros del proyecto.

La estructura de ficheros del proyecto sigue el esquema propio que te da Repast Simphony Java 2.0. Por un lado tienes la carpeta src donde hay todas las clases que se han implementado para la simulación. Luego como todo framework, Repast tiene muchas librerías propias y también tiene las de Groovy (otro lenguaje con el cual se puede implementar los agentes aparte del de Java). Luego las carpetas batch, installer, launchers, repast-licencias y virtualCommunities.rs son carpetas que Repast genera automáticamente cuando creas un nuevo proyecto. En ellas hay parte de configuración del simulador con el proyecto que vas a desarrollar.

Por otro lado la carpeta bbdd se ha creado para almacenar los scripts y la de graphics para guardar los iconos que nuestra simulación utiliza.

3.4. Diseño de agentes en JAVA

Un agente es una clase Java que tiene diferentes atributos y métodos. Todos los agentes heredan de una clase java llamada Agent que tiene definido elementos comunes de los diferentes tipos de agentes. Cada agente tiene su propia clase java que hereda de Agent y un icono asociado (que es el que se mostrará en la simulación).

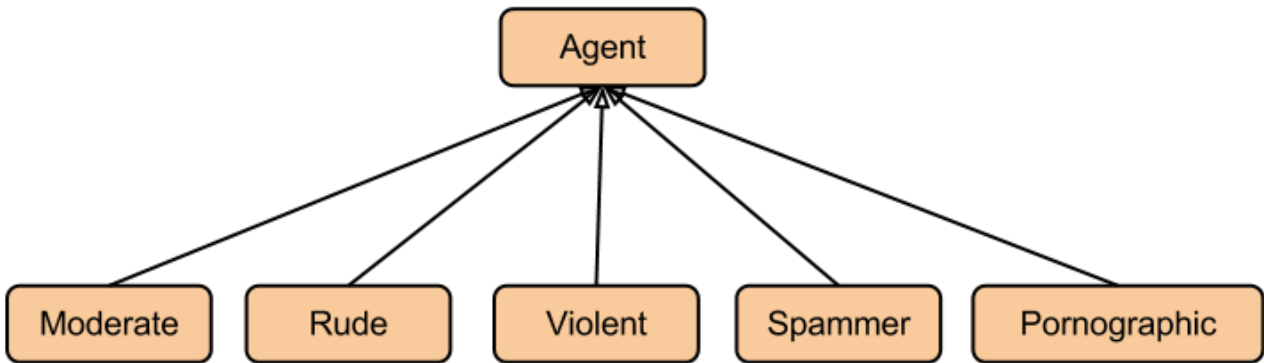


Figura 27: Imagen donde se muestra de forma visual la herencia.

3.5. Diseño de las acciones de los agentes en JAVA

Aunque haya diferentes tipos de agentes todos heredan de Agent. Todos los agentes pueden hacer las mismas acciones, lo único que cambia el momento en que las hacen que depende de los parámetros que tenga en su perfil de agente. A continuación se explicará como los agentes hacen la acción de subir contenidos, visitar contenido o el de quejarse de los contenidos.

3.5.1 Subir contenido (uploadContent)

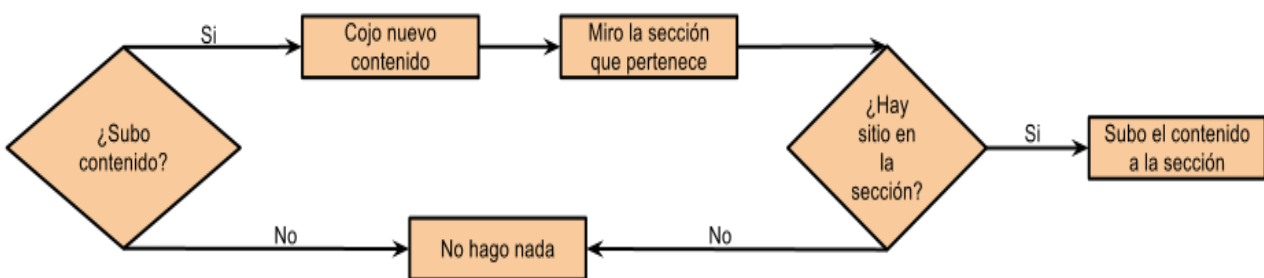


Figura 28: Diagrama de flujo de la subida de contenido.

En el diagrama anterior se muestra el proceso que sigue un agente para subir contenido.

La primera etapa es decidir si tiene que subir contenido o no. Esto se hace en función del parámetro de uploadfrequency del UploadProfile; que determina el porcentaje de contenido que tiene que subir. Si tiene que subir contenido, coge el contenido; que previamente se ha cargado de la base de datos. Luego mira la sección que pertenece el contenido y si la sección esta llena de contenido no deja subir nuevo contenido. E En cambio, si hay sitio el contenido se sube sin ningún problema.

Por otro lado si no le toca subir contenido, el agente no hace la acción de subir contenido y en consecuencia no hace ninguna acción.

3.5.2 Visitar y quejarse de contenidos (viewContent and complaintContent)

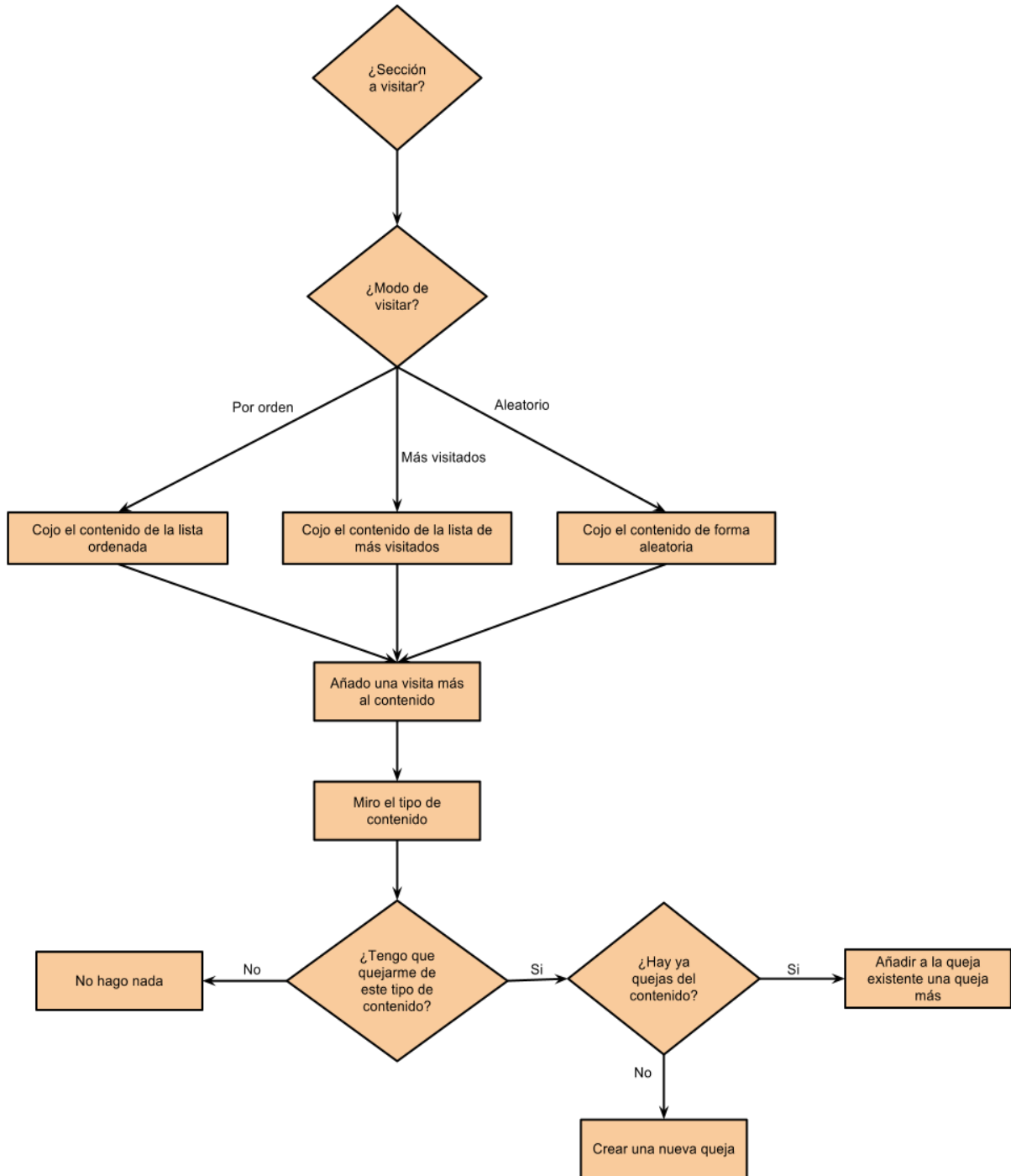


Figura 29: Diagrama de flujo de visitar y de quejarse de contenidos.

En el diagrama anterior se muestra el flujo que todo agente sigue para visitar y quejarse de contenido. El viewProfile o perfil de visita tiene parámetros que determinan el modo de visitar y el porcentaje de visita de cada acción. El agente inicialmente mira en que sección tiene que visitar y con que modo. Una vez hemos hecho una visita a un contenido el agente mira si tiene que quejarse o no de este tipo de contenido. Si se tiene que quejar luego mira si hay ya iniciada una queja o no. Sino esta iniciada la queja la crea sino añade una queja más a la queja iniciada.

3.6. Diseño del Context builder

Todo sistema MAS tiene un contexto de la simulación. Repast tiene un clase para construir el contexto llamada ContextBuilder. Ahí se define que tipo de espacio tendrá la simulación (si GRID o un SPACE).

El espacio GRID es de forma como de tabla donde hay filas, columnas y donde las cosas dentro de este espacio ocupan una casilla determinada. Por otro lado hay el espacio de tipo SPACE. Este tipo es cuando los objetos de tu simulación ocupan "coordenadas" dentro de un sitio. Se usa este tipo para definir espacios continuos donde no se considera que hay limites.

En nuestra simulación usamos ambos tipos de espacios y colocamos los agentes, sus contenidos y quejas en el mismo punto(mismas coordenadas) tanto en el GRID como en el SPACE. Luego visualmente el simulador solo muestra el espacio GRID ya que nos sirve para ver la simulación en forma de tabla y donde los objetos ocupan cierta casillas.

En este contexto es donde los agentes, contenidos y quejas se añaden. Luego con los espacios antes mencionados se pueden hacer que los agentes se muevan de una determinada posición a otra.

4. Desarrollo

4.1. Entornos de trabajo usados

A continuación se explicará y detallará los diferentes entornos que se han usado para desarrollar la simulación.

Repast Simphony Java 2.0



Figura 30: Logo del Repast Simphony Java 2.0.

Repast Simphony Java 2.0 es un fork del Eclipse que añade las librerías de Repast y plugins necesarios para crear simulaciones de MAS. Al ser un clone del Eclipse todo el dominio que puedas tener en Eclipse es el mismo en esta versión. Hay que mencionar que se pueden instalar también todos los plugins propios de Eclipse.

Las librerías y plugins que añade esta versión hacen que la parte del simulador ya este implementada. Por lo tanto, el sistema de ejecución paso a paso, el de reiniciar la simulación o de arrancar la simulación ya esta implementado. De este modo como usuario que usa Repast Simphony de lo que te tienes que centrar principalmente es el de diseñar los agentes y el propio MAS.

Otra característica que facilita dicha implementación es que pueda desarrollar y implementar los agentes en el lenguaje JAVA y no como sucede en muchos casos con lenguajes específicos y propios de algunos sistemas MAS.

Svn



Figura 31: Logo del subversión (svn).

Se ha utilizado un repositorio de subversión para ir subiendo las diferentes versiones del proyecto a la red del IIIA-CSIC. Dado que el proyecto se ha iniciado por mi parte pero posteriormente el estudiante losu Mendizabal seguirá con él, se ha ido subiendo en el repositorio para que losu pudiera ver y entender las diferentes evoluciones del software y de la simulación.

Para acceder al repositorio es necesario que se tenga un usuario creado en el SVN del IIIA-CSIC.

Link del repositorio:

<https://svn.iiia.csic.es/normGeneration/Code/trunk/VirtualCommunities/>



Figura 32: Logo del Xampp.

Se ha utilizado la aplicación XAMPP para simplificar la configuración de una BBDD de MySQL en un servidor local. XAAMP instala una distribución de apache configurada con MySQL, PHP y Perl. Es multiplataforma (Windows, Linux, Mac y Solaris) y es complemente gratuito.

La aplicación facilita la tarea ya que dispone del PhpMyAdmin; que es una interface web para tratar BBDD. Gracias a estás herramientas ha sido más sencillo la puesta en marcha de la BBDD en un servidor local y de poder importar de forma rápida y sencilla nuestra estructura BBDD con todos sus datos.

4.2. Funcionamiento, configuración y curiosidades de Repast Simphony Java 2.0

Insertar y mover objetos en el context de la simulación

Para añadir objetos en la simulación de Repast y colocarlos en una determinada posición se hace con la siguiente instrucción:

```
context.add(objeto);
space.moveTo(objeto, posX, posY);
grid.moveTo(objeto, posX, posY);
```

Scheduler de métodos de los agentes.

El simulador de Repast funciona con un Schedule. Hay dos métodos que puedes hacer que un método se añada al shedule del simulador. Una manera es mediante @notations. Tu añades un tipo de anotación que hace que el simulador incorpore en su Schedule ese método del agente.

```
@ScheduledMethod(start = 1, interval = 1)
```

Cuando tu arrancas el simulador, el mismo Repast coje todos los métodos que tenga una anotación y los añade al Scheduler de la simulación.

El otro modo para decirle que un método del agente se añada al Schedule y por lo tanto se ejecute en el tick que tu has dicho, es hacer que sea dinámico. Al hacer el Schedule dinámico tu en cualquier momento de la simulación puedes añadir un nuevo método de un agente al Scheduler del simulador y no solamente al arrancar la simulación como sucede con las anotaciones.

El código para crear un Scheduler dinámico y que se agregue al Scheduler del simulador es el siguiente:

```
ScheduleParameters scheduleParams;
ISchedule schedule;
schedule = RunEnvironment.getInstance().getCurrentSchedule();
//Create scheduler for upload content
scheduleParams = ScheduleParameters.createRepeating(tickInicial,
intervalo, prioridad);
schedule.schedule(scheduleParams, agent, "nombredelmetodo");}
```

También hay que mencionar que al crear un scheduler dinámico tenemos la posibilidad de asignarle cierta prioridad. La prioridad sirve para que el Scheduler del simulador tenga preferencia a la hora de ejecutar un método o otro según la prioridad que se le ha asignado.

Creación del contexto del simulador.

El contexto de Repast se crea mediante una clase que implemente el `ContextBuilder<Object>`. Ahí se define un identificador para identificar el contexto de la simulación. Eso se hace porque hay determinadas simulaciones que pueden tener más de un contexto asociado. De esta manera con el identificador podemos saber de que contexto concreto queremos y en el que estamos trabajando. También sucede con los espacios que tengamos definidos en el contexto.

```
context.setId("VirtualCommunitiesContext");
```

Una vez se ha definido un identificador para el contexto se tiene que que añadir al fichero `context.xml` dentro de la carpeta `virtualCommunities.rs` las siguientes líneas:

```
<context id="VirtualCommunitiesContext"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="http://repast.org/scenario/context">
    <projection type="continuous space" id="space"/>
    <projection type="grid" id="grid"/>
</context>
```

4.3. Problemas detectados en el desarrollo

1. Problema detectado cuando se define el contexto de la simulación. Sino cargas el Data Loader los datos de tu contexto no se cargan de forma correcta cuando quieres simular-lo. Se tiene que cargar los datos antes de simular y por lo tanto hay que definir bien el Data Loader del Repast.
2. Cuando defines display de la simulación se tiene que apretar el botón de guardar o salvar. Aunque parezca que se guarda bien sino se apriete el botón luego aparecen un montón de errores con el display.

La solución es una vez definido el display apretar el botón de guardar. De esta manera Repast genera guarde el `.xml` asociado al display. Al volver a cargar la simulación, repast leerá ese xml.

3. Problemas al intentar colocar los títulos de las diferentes secciones. Cuando tu defines un objeto en el display y tu lo colocas en la posición por ejemplo 0,0 el internamente hace que el icono dependiendo del tamaño que tenga se centre en esa posición y no que dicho icono o imagen se coloque a partir de dicha posición.

La solución es colocar dicho icono o imagen desplazada para que quede centrado.

4. Cómo se utilizan porcentajes para determinar muchos parámetros como puede el tipo de contenido que se sube a la web hace que muchas veces salga el número de contenidos con precisión decimal. En ese momento habría que decir si redondear al alza o cortar. Se ha decidido cortar ya que de esta manera te aseguras que el número de contenidos que genera un usuario no accede de su límite posible. Dado que si redondeas al alza podría darse el caso que finalmente tuvieras más contenidos de los que el usuario tiene permitido como máximo generar.
5. Problemas con el Scheduler de Repast. Si se usa el método de las anotaciones para decir que un método del agente se tiene que ejecutar en la simulación puede darnos problemas cuando tenemos más de un agente. El problema de este método por anotaciones esta en que el Scheduler de Repast no siempre coge con el mismo orden los métodos que tienen anotaciones. Esto hace que aunque tengas definido los agentes y todo de cierta manera, la ejecución cada vez sea diferente ya que el orden en que actúan los agentes varía.

La solución ha sido hacer los scheduler dinámicos. Gracias a esto se ha podido establecer prioridades entre el mismo método de diferentes agentes (como puede ser el de subir contenido) y entre diferentes acciones o métodos de los agentes como es el de subir contenido y visitar o quejarse. Se ha hecho que lo que tenga más prioridad entre los agentes es lo de subir nuevo contenido.

5. Resultados

Agent Type:

Agent Name:

Upload Profile

Upload Content:

Correct Content:

Wrong Placement:

Insult Content:

Spam Content:

Violent Content:

Porn Content:

View Profile

Forum View:

The Reporter View:

Multimedia View:

Complaint Profile

Wrong Placement Complaint:

Insult Complaint:

Spam Complaint:

Violent Complaint:

Porn Complaint:

Number of Agents:

Actual Population

Moderate:

Pornographic:

Rude:

Spammer:

Troll:

Violent:

Figura 33: Imagen del selector de poblaciones

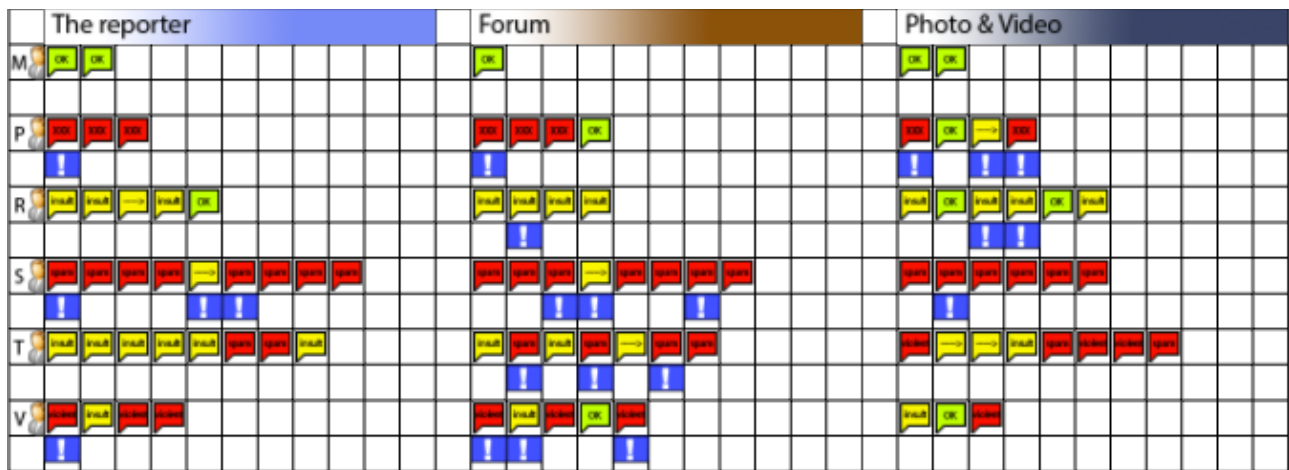


Figura 34: Imagen de la simulación

Los resultados que aparecen en la simulación varían según como esta configurados los agentes y más concretamente sus perfiles. Esta imagen que aparece podría tratarse de un contexto de comunidad virtual de seguidores de fútbol donde hubiera un agente de cada tipo.

Como la mayoría de los parámetros de configuración de las diferentes partes del perfil de agente son porcentajes;: que oscilan entre 0%-100%, eso hace que del total de posibles comentarios que pueda tener la red, los agentes suban un determinados porcentaje de comentario de cada tipo. Por otra parte es recomendable que los agentes se quejen cuando sea necesario y que no sean propensos a quejarse, dado que en el proyecto general se usaran las quejas para generar de forma automática las normas.

6. Conclusiones y trabajo futuro

Después de unos meses de trabajo se ha logrado alcanzar los objetivos que inicialmente me había propuesto.

El camino no ha sido sencillo ya que el framework de Repast Simphony carece de mucho soporte y es difícil tener ayuda de terceros cuando tienes un determinado problema. Por otro lado y siguiendo la misma dirección, tampoco hay muchos manuales o tutoriales aportados por el mismo Repast Simphony que te den información detallada de la API. Otro punto que tampoco ayuda es que la mayoría de los ejemplos que incorpora el propio Repast Simphony están en un lenguaje llamado Groovy en vez de Java.

Hay pocos ejemplos de Java y la mayoría son ejemplos sencillos donde se usan pocas cosas de la API de Repast Simphony. Eso hace que no sea sencilla la tarea de empezar desde cero a crear una simulación cuando no has trabajado nunca con esta API.

Aún así, poco a poco he podido ir implementando nuevas funcionalidades a base de ir mirando ejemplos, repasándome la API; si había algo que pudiera aprovechar o utilizar y a base de implementar y testear (el típico prueba/error).

Pese a todos estos impedimentos se ha logrado alcanzar los objetivos propuestos. Se ha podido crear una simulación que visualmente de mucha información, que esté muy cargada de contenido y en que se simule una comunidad virtual de seguidores de fútbol.

Se han implementado diferentes agentes y sus interacciones con la comunidad virtual, como también se han creado contenidos. También se ha clasificado el contenido de la comunidad en diferentes secciones

Los iconos se ha creado o personalizado para la simulación, así que en todo momento se ha mirado de cuidar mucho la estética visual de la simulación, aparte de su funcionamiento.

Por otro lado, ha ayudado bastante en este TFG trabajar con la herramienta del XAMPP. Gracias a este programa he podido prescindir de horas para configurar un servidor Apache con MySQL. También me ha facilitado la tarea para importar scripts SQL y para ver en todo momento como estaba la base de datos.

Mencionar que como se ha ido comentando durante la memoria, este proyecto es una parte de un proyecto más general. Otro compañero de nombre losu Mendizabal seguirá donde yo he acabado. Él cogerá el simulador que he creado y mirará de integrar el método del Javier Morales para generar normas (IRON). losu se encargará en su proyecto de generar normas a partir de los conflictos que se vayan produciendo en el simulador; recordar que los conflictos en el proyecto se asocian a las quejas que tienen los contenidos. Una vez se tengan las normas creadas, otra etapa será la de crear el otro tipo de rol de usuario: el moderador. Este tipo de usuario se encargará de aplicar castigos a los agentes que dejen de cumplir las normas que vaya apareciendo. El proyecto general no solo quiere generar las normas sino que con ellas se pueda hacer una moderación automática de la comunidad virtual.

Como trabajo futuro serian interesante también disponer de histogramas y gráficas; para visualizar de esta manera, si los usuarios se quejan mucho por ejemplo o el número de quejas de un contenido respecto al total de las quejas. Otra cosa que seria interesante

también serían disponer de logs con datos del comportamiento de los agentes o de la misma simulación. Cosas como histogramas, gráficas o logs ayudan mucho en la tarea de analizar resultados y también en la tarea de hacer comparaciones entre diferentes experimentos.

También mencionar que creo que sería también interesante tener más de un display. Creo que iría bien tener más de un display donde en cada uno podría enfocarse la visualización de la comunidad virtual desde una perspectiva concreta (enfocarla desde los usuarios, desde las quejas, desde las secciones, desde las visitas de los contenidos, desde los tipos de contenidos que sube la gente).

Por todos estos motivos antes mencionados, estoy orgulloso del trabajo que se ha realizado y estoy impaciente de ver los resultados del proyecto general una vez que ya se tiene la simulación de una comunidad de seguidores de fútbol.

7. Referencia bibliográficas

- [1] Artículo de [IJCAI11-061]. Using the experience to generate new regulations
- [2] Reporte final de Rodrigo Gómez. Sistema en línea para la simulación de redes
- [3] Maite López y Javier Morales. Generating norms from experience
- [4] Maite López-Sánchez, Javier Morales Matamoros, Juan Antonio Rodriguez Aguilar. Building self-regulating online communities
- [5] Javier Morales. Using the experience to generate new regulations
<http://campusvirtual.ub.edu/file.php/38535/curs12-13/IADistribuida.pdf>
- [6] Repast, página oficial
<http://repast.sourceforge.net/>
- [7] Api de Repast, página oficial
http://repast.sourceforge.net/docs/api/repast_simphony/index.html
- [8] Tutorial de Repast para programar en Java. RepastJavaGettingStarted
- [9] Creación de Scheduler en Repast, páginas oficiales
http://repast.sourceforge.net/repast_3/how-to/scheduler.html
<http://repast.sourceforge.net/docs/reference/SIM/Working%20with%20the%20scheduler.html>
- [10] Ejemplos de code google de proyectos creados en Repast.
<http://code.google.com/p/cscs-repast-demos/>
- [11] Ejemplo de un contexto en Repast.
<http://socialnetworksimulation.googlecode.com/svn!svn/bc/61/branches/CommunityStructure/src/communitystructure/StudentContext.java>
- [12] Xaamp, página oficial.
<http://clientes.hostingperu.com.pe/knowledgebase/44/iComo-Importar-Scripts-MySQL-desde-el-PhpMyAdmin.html>
- [13] Creación de tablas con MySql, página oficial.
<http://dev.mysql.com/doc/refman/5.0/es/create-table.html>
- [14] Video de creación de proyecto Java con conexión a base de datos
http://www.youtube.com/watch?v=E30_-pQGQXs
- [15] JDBC
<http://www.kitebird.com/articles/jdbc.html>
- [16] Tipos de moderación, Blog de keepcon.
<http://blog.keepcon.com/?p=363>
- [17] La comunicación como herramienta de dirección, Escuela de Organización Industrial.
http://www.edukanda.es/mediatecaweb/data/zip/1187/page_22.htm

[18] Iconos base utilizados para la creación de los iconos del simulador.
<http://www.gettyicons.com/free-icon/112/must-have-icon-set/free-user-icon-png/>
<http://blog.picol.org/uberhangmandate/>
http://openiconlibrary.sourceforge.net/gallery2/?./Icons/others/light_bulb.png

[19] Semillas y random en Java.
http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=240:generacion-de-numeros-aleatorios-en-java-clase-random-ejemplos-y-ejercicios-resueltos-cu00906c&catid=58:curso-lenguaje-programacion-java-nivel-avanzado-i&Itemid=180

[19] Imagen de facebook
<http://techbeat.com/2012/11/kiwi-company-sues-facebook-for-removing-profile/>

[20] Keepcon, página oficial
<http://keepcon.com/>

[21] Daedalus, página oficial
<http://www.daedalus.es/blog-y-recursos/sobre-stilus/stilus-forum-moderacion-automatizada-de-foros-de-participacion/>

[22] Sourtech
<http://www.sourtech.com/quienes-somos/>
<http://www.slideshare.net/sourtech/servicio-de-moderacin-sourtech>
<http://www.moderacion.net/metodo-sourtech/nuestro-software-sourpanel/ç>

[23] Definiciones de Wikipedia
http://es.wikipedia.org/wiki/Sistema_multi-agente
<http://es.wikipedia.org/wiki/Blogger>
http://commons.wikimedia.org/wiki/File:Arpanet_logical_map_march_1977.png
<http://es.wikipedia.org/wiki/Internet>
<http://es.wikipedia.org/wiki/Imprenta>
<http://es.wikipedia.org/wiki/Escritura>
<http://es.wikipedia.org/wiki/ARPANET>
<http://es.wikipedia.org/wiki/Facebook>
[http://es.wikipedia.org/wiki/Moderador_\(Internet\)](http://es.wikipedia.org/wiki/Moderador_(Internet))
http://es.wikipedia.org/wiki/Moderaci%C3%B3n_autom%C3%A1tica_de_contenidos
http://es.wikipedia.org/wiki/Comunidad_virtual
<http://es.wikipedia.org/wiki/Internet>
<http://es.wikipedia.org/wiki/Informaci%C3%B3n>

8. Anexos

8.1. Manual de uso de la aplicación

Este es el manual de uso de la aplicación/simulador para poder hacer un correcto funcionamiento de la aplicación y para detallar los diferentes pasos.

1. Inicialmente tenemos que tener el servidor XAMPP arrancado correctamente.

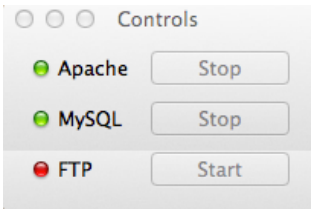


Figura 35: Imagen de los controles de Xampp con apache y mysql encendidos

2. Arrancamos la aplicación y nos aparecerá la siguiente pantalla.

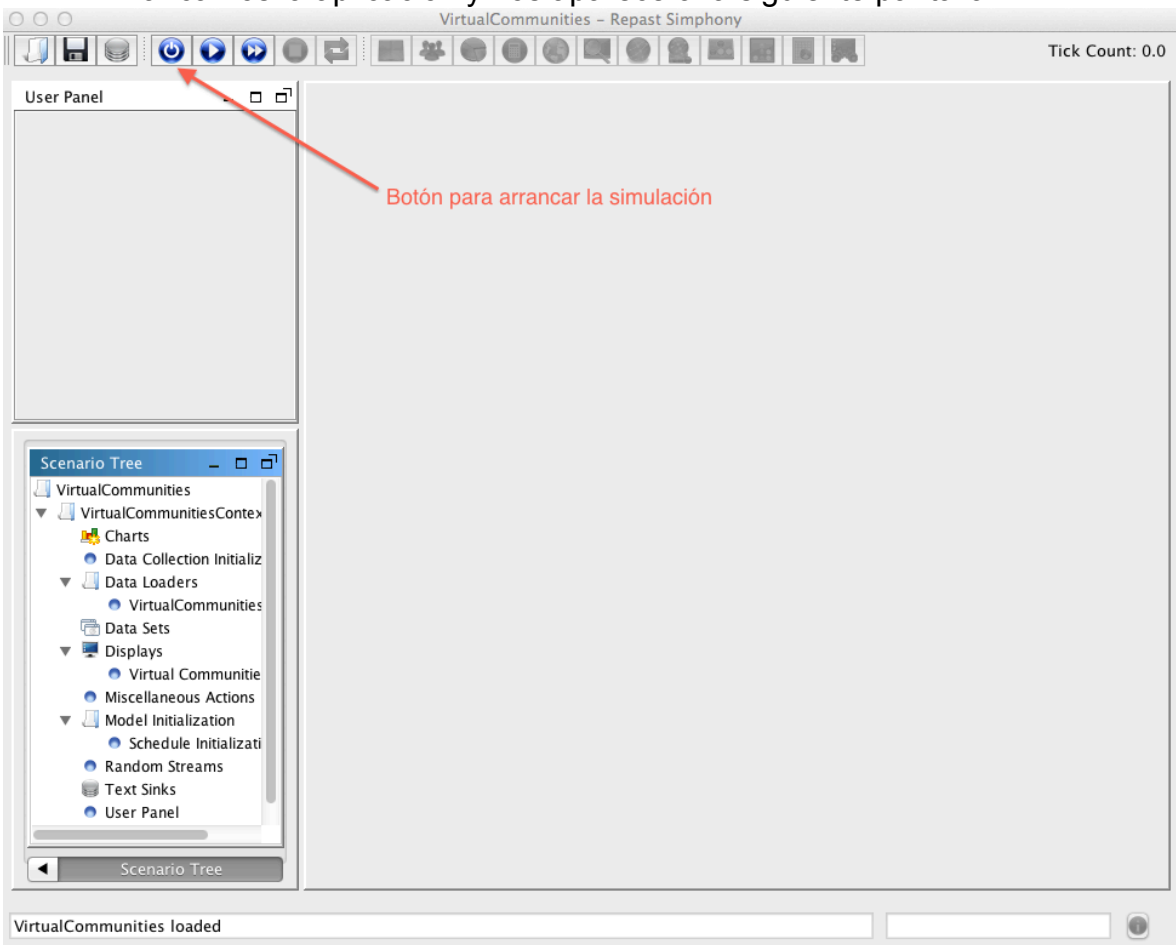


Figura 36: Pantalla que se muestra al cargar la aplicación.

Como se muestra en a la imagen superior hay un botón de encendido para encender la simulación.

3. El siguiente paso es el de arrancar la simulación con el botón de encendido. Una vez realizado eso veremos la siguiente pantalla.

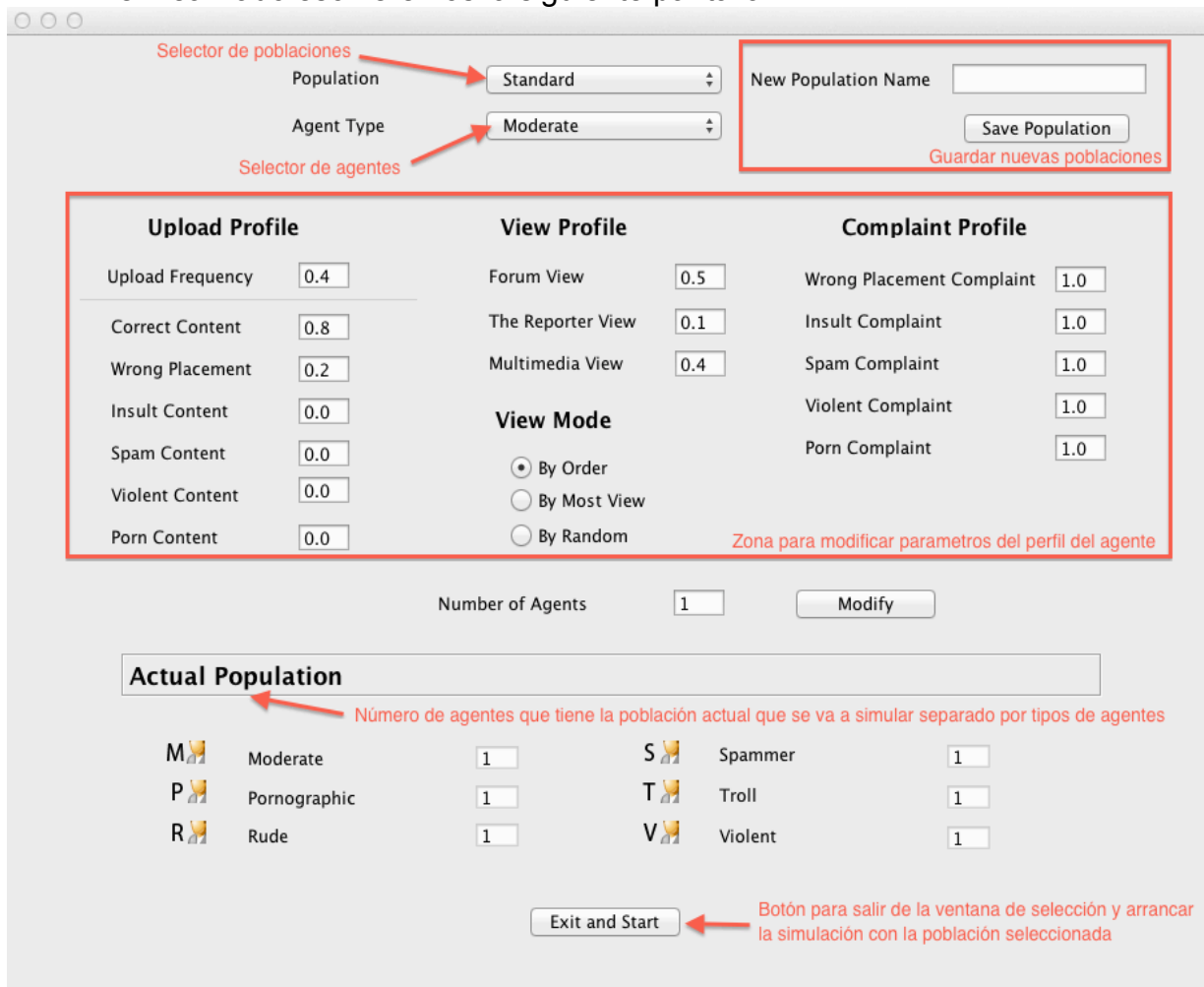


Figura 37: Pantalla de selección de las poblaciones.

Esta es la pantalla de selección de la población de agentes. Podemos seleccionar una que ya exista o podemos modificar o crear una nueva. Como se ve en la imagen todos los parámetros del perfil de un agente puede ser modificados. En la parte inferior de la pantalla se muestra el número de agentes que tiene la población actual; separados por los diferentes tipos de agentes que hay.

El botón de Exit and Start sirve para salir de la ventana de selección de población e ir a la simulación con la población seleccionada.

4. Apretamos el botón Exit and Start y nos aparecerá la siguiente pantalla:

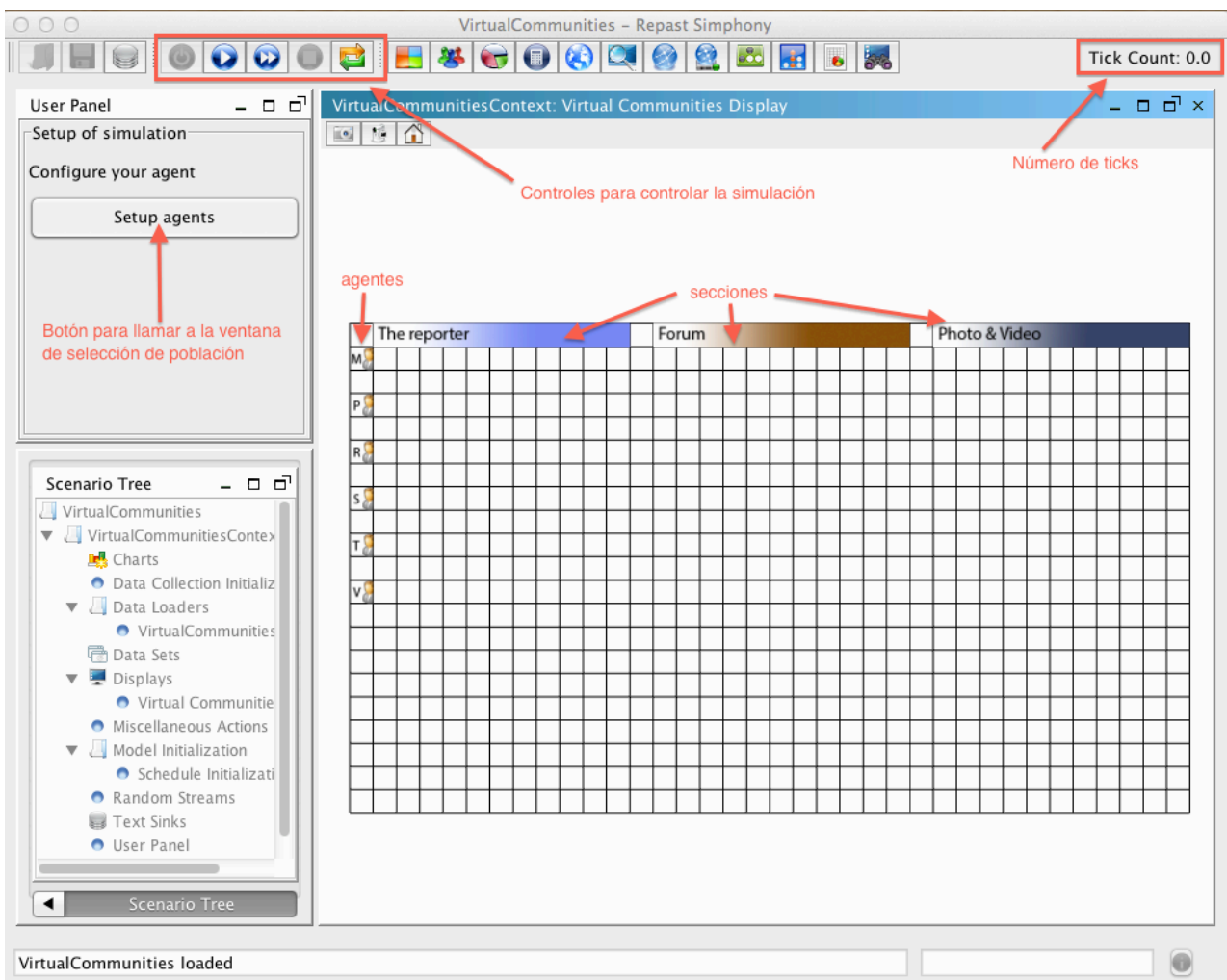


Figura 38: Pantalla con las poblaciones cargadas y la simulación preparada para arrancar.

Como se puede ver de la imagen, en la parte de arriba hay unos controles para manejar la simulación.

El primer control es el que hemos visto que era el de Initialize Eun(o botón de encendido). Luego tenemos el de Start que lo que hace es ejecutar de golpe la simulación. Luego tenemos el de Step run que es para ir paso a paso, tick a tick en la simulación. De esta manera se puede ver de forma clara instantes en los que se encuentra la simulación. Los dos últimos controles son los de Parar la simulación y Reiniciarla.

En la parte izquierda hay un botón para poder volver a la ventana de selección de población sin tener que para y arrancar de nuevo la aplicación.

En la parte central de la imagen es la propia de la simulación de la comunidad virtual. Cómo se puede comprobar mirando la imagen, la primera columna hace referencia a un agente. Los agentes pondrán los comentarios en su fila únicamente pero el espacio de la fila donde pueden poner comentarios esta dividido en diferentes secciones. De esta manera visualmente se puede ver cuantos contenidos ha hecho un determinado agente, de que tipo son, y en que sección los ha hecho.

5. Arrancamos con Start o Step run la simulación

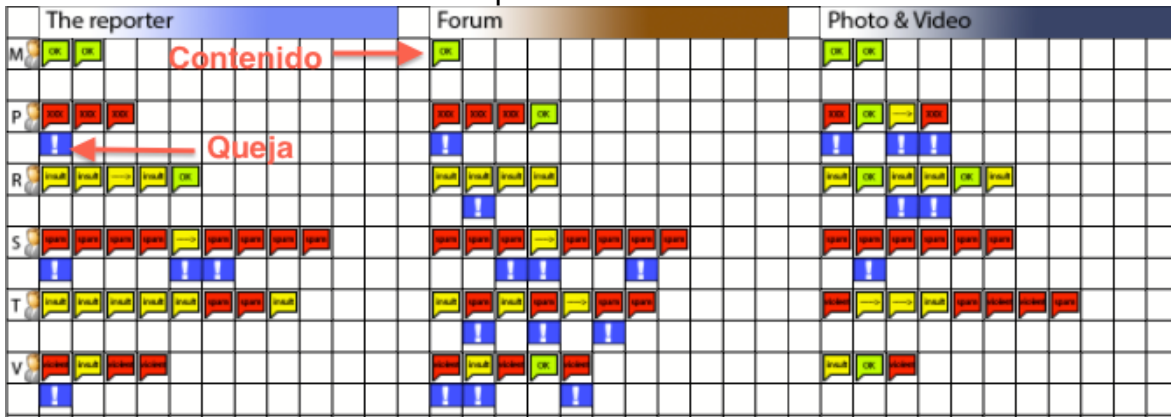


Figura 39: Simulación arrancada.

La pantalla de simulación una vez ya haya arrancado tendrá un estilo similar al mostrado en la imagen de arriba. Los contenidos tienen diferentes colores dependiendo de si son buenos, regulares o malos. En cada uno pone del tipo que se trata de contenido. Por otro lado también hay quejas que tienen el icono de color azul.

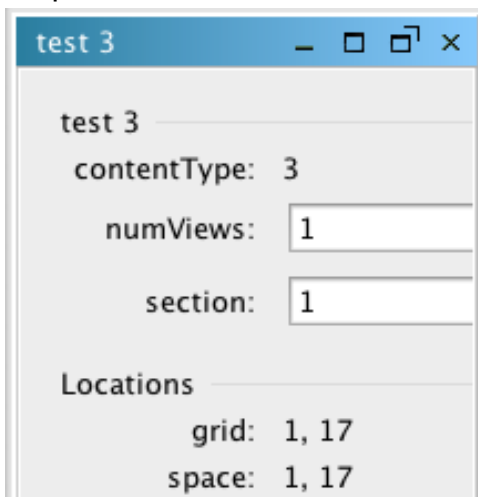


Figura 40: Propiedades de un contenido.

Si pulsamos dos veces en un contenido no saldrá la siguiente pestaña:

Esta pestaña muestra detalles del contenido que pueden ser interesantes como puede ser el número de visitas que tiene.

Por otro lado también existe detalles de las quejas:

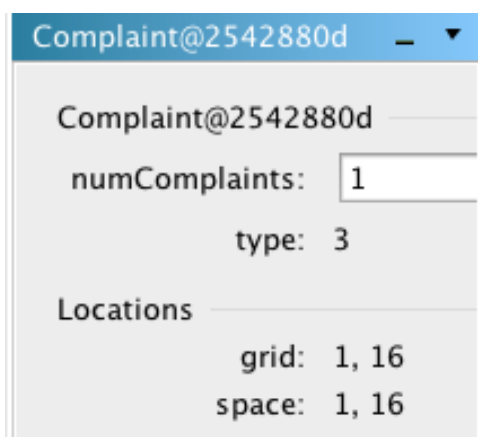
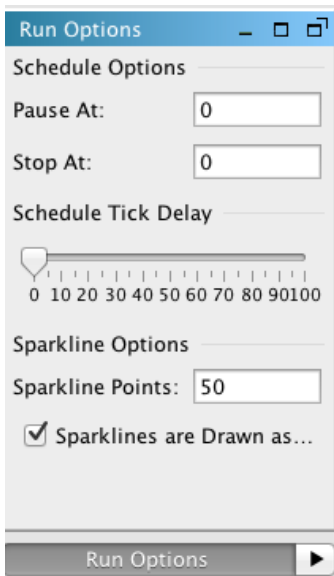


Figura 41: Propiedades de una queja.

6. Por otro lado si queremos modificar parámetros del simulador



El Run Options es una pestaña que hay en Scenario tree o Parameters. Para acceder a él hay que apretar la flecha hacia un lado.

Aquí se puede configurar determinados parámetros del simulador como puede ser el tiempo que hay de espera entre ticks o cuando se quiere pausar o incluso cuando se quiere terminar la simulación.

Figura 42: Pantalla de run Options

8.2. Importar scripts MySQL con XAMPP

El proceso para crear y cargar los datos necesarios de la base de datos se hace mediante la importación de dos ficheros .sql. Se ha hecho de esta manera para que se pueda modificar fácilmente la base de datos o los datos que contiene y a su vez que sea fácil poderlo instalar en cualquier ordenador.

Para poder importar estos ficheros .sql tendremos que seguir estos pasos que se detallan a continuación:

1. Arrancar la aplicación XAMPP Control. Inicialmente veremos que están en rojo.

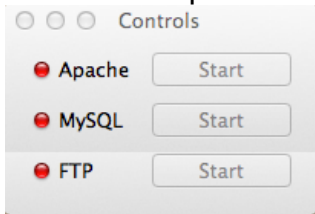


Figura 43: Imagen de los controles de Xampp.

Ejecutamos Apache y MySQL que son realmente lo necesarios.

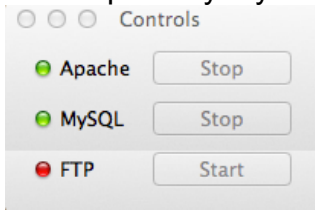


Figura 44: Imagen de los controles de Xampp con apache y mysql encendidos.

2. Una vez arrancados vamos a nuestro explorador web y escribimos la siguiente dirección:
<http://localhost/phpmyadmin/>
3. Veremos que en la opciones de menú de la parte de arriba hay una opción que se llama IMPORTAR.
4. A continuación elegimos primero el fichero que crea las base de datos y sus relaciones virtualCommunitiesDatabase.sql.
5. Por último volvemos hacer el punto 3 y 4 pero con el fichero de datos para nuestra base de datos cuyo nombre es virtualCommunitiesData.sql.
6. Finalmente veremos que se ha creador correctamente la base de datos:



Figura 45: Base de datos disponibles en nuestro servidor local.

8.3. Cambiar iconos de los agentes, secciones, comentarios o quejas en la aplicación.

El propio simulador de Repast Simphony Java 2.0 te da la posibilidad de cambiar los iconos que utilizas en la visualización de la simulación. Para realizar esta tarea se tiene que ir a la pestaña de “Escenario tree”.

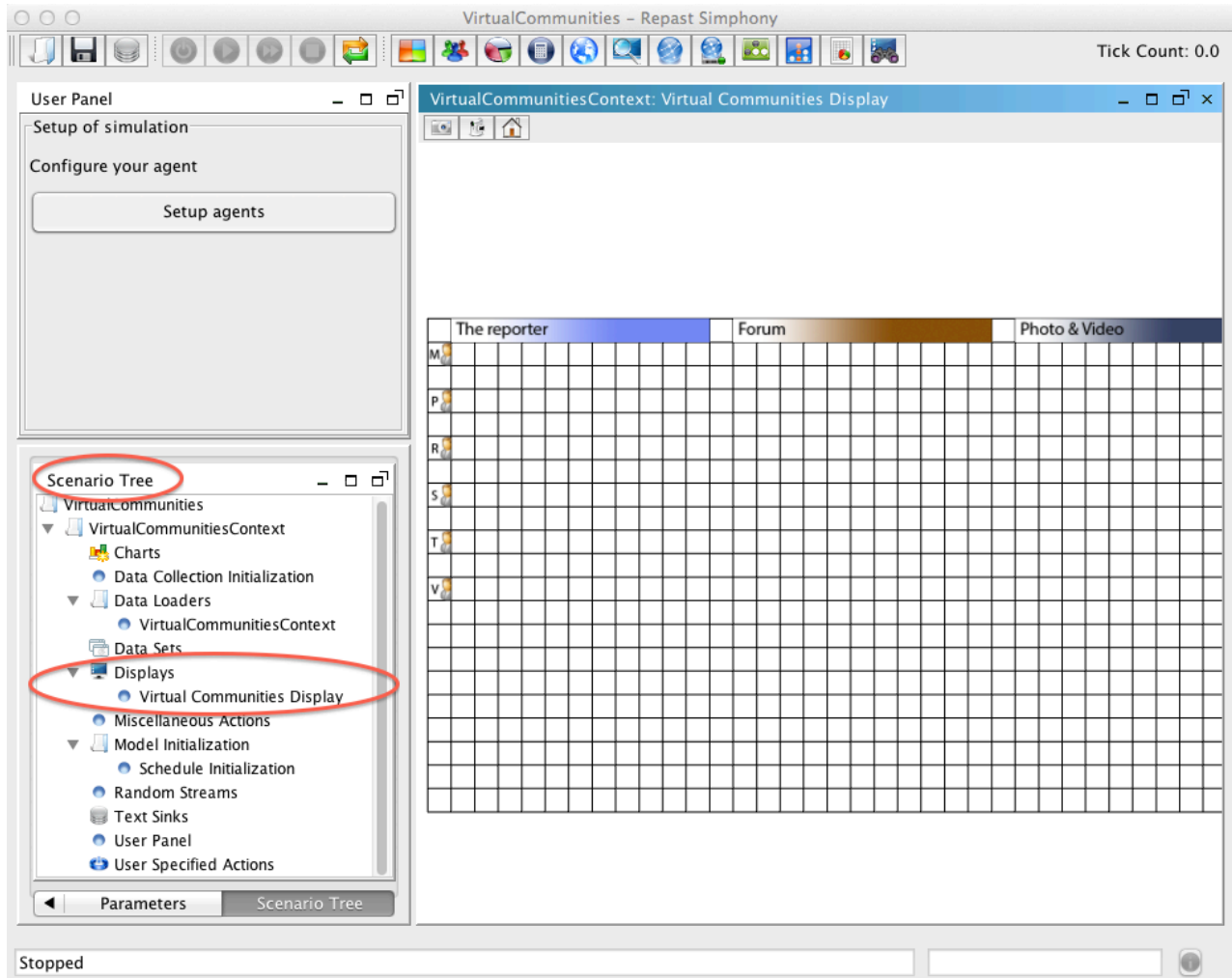


Figura 46: Pantalla donde se muestra donde se halla el display de la simulación.

En la pestaña de “Escenario tree” se puede ver que hay un apartado de Display. Ahí es donde se define el display de nuestra simulación y los iconos de los agentes, comentarios secciones. Si clicamos dos veces nos aparecerá la siguiente pantalla:

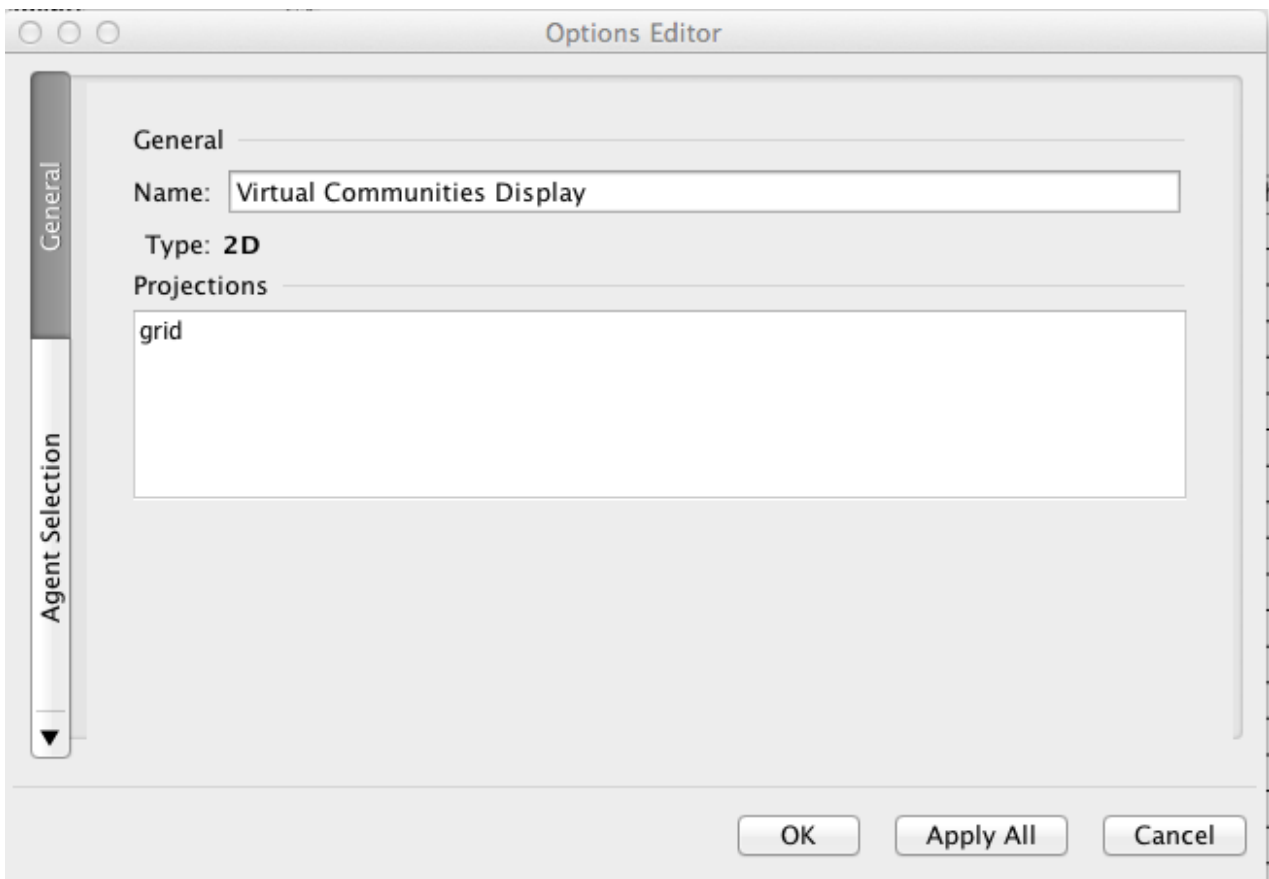


Figura 47: Pantalla que muestra la pantalla inicial del editor del display.

En esta pantalla se muestra el nombre del display y que tipo de escenario usa, en nuestra aplicación como se ha dicho anteriormente se visualiza un escenario de tipo GRID. Si vamos a la opción "agent selection" nos aparecerá la siguiente pantalla:

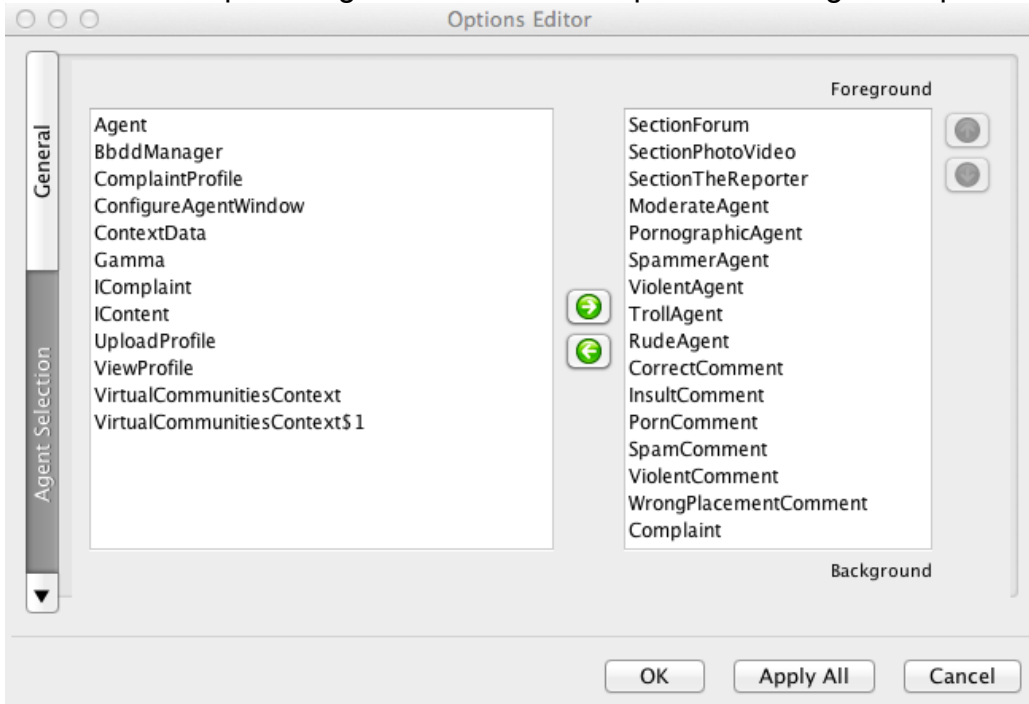


Figura 48: Pantalla de selección de los elementos que queremos que aparezcan en la simulación.

En la anterior imagen se muestran las cosas se visualizarán en el simulador. Todos los elementos que estén al lado del Foreground serán visualizados por el simulador. Si

apretamos la flecha de hacia abajo de “Agent selection” no aparecerá la pantalla para definir la visualización de los elementos que hemos seleccionado.

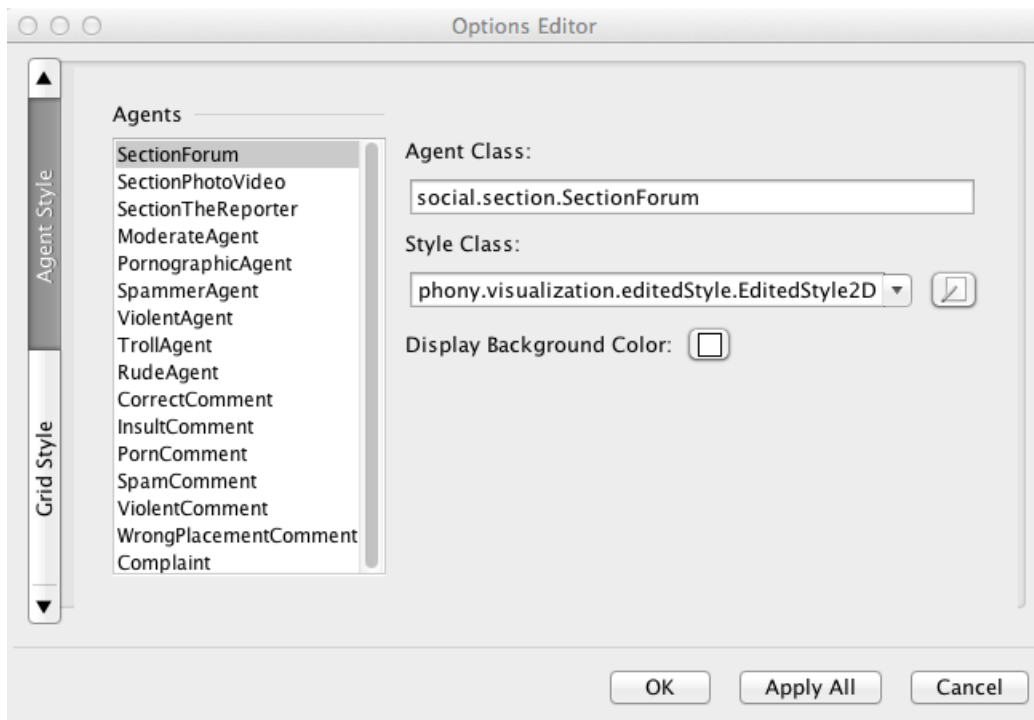


Figura 49: Pantalla donde se muestra el editor de la visualización de los elementos del simulador.

En la imagen anterior se puede ver el listado de agentes que visualizaremos y luego a la derecha hay la parte para cambiar iconos, cambiar el color de fondo del icono...

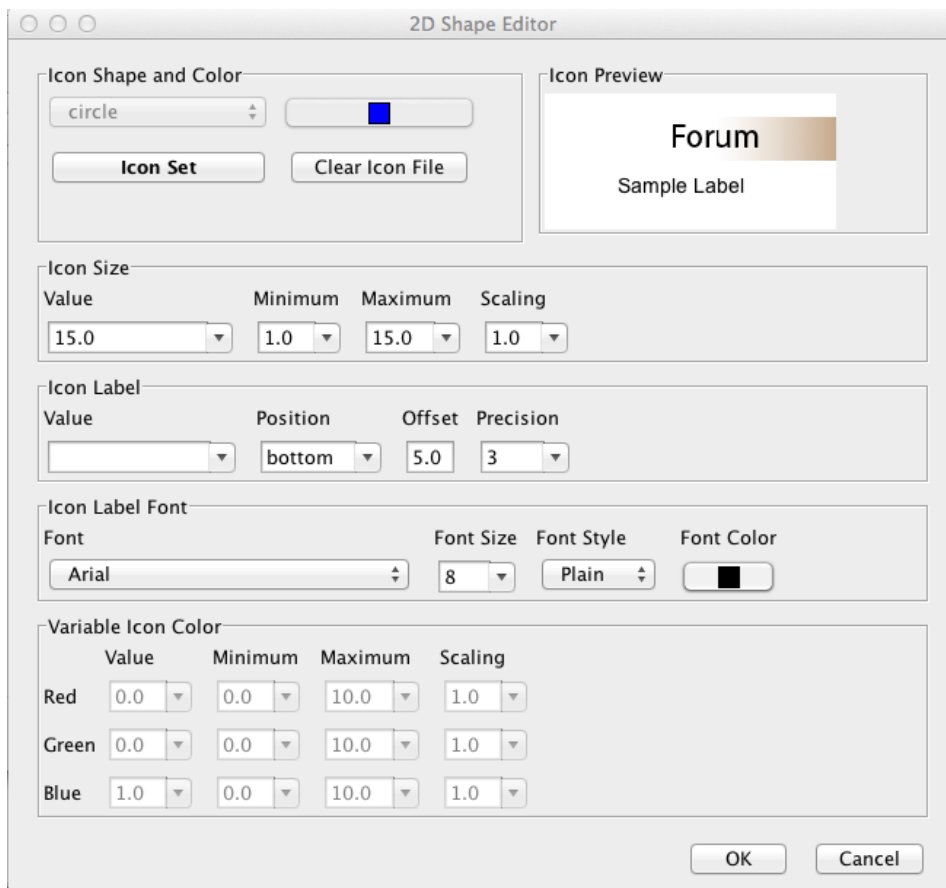


Figura 50: Pantalla que muestra como podemos editar el icono de un elemento de la simulación.