ASA: SISTEMA AUTOMÁTICO DE PERSONALIZACIÓN APLICADO A PERSONAJES DE VIDEOJUEGOS

Francisco Domenech Marín

Codirigido por Héctor Gómez Gauchía y Pedro A. González Calero

Máster en Investigación en Informática, Facultad de Informática,
Universidad Complutense de Madrid

Trabajo Fin de Máster en Sistemas Inteligentes

CURSO 2010-2011

Convocatoria Septiembre
Calificación: 7

INDICE

1 Autorización

El/la abajo firmante, matriculado/a en el Máster en Investigación en Informática de la Facultad de Informática, autoriza a la Universidad Complutense de Madrid (UCM) a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente a su autor el presente Trabajo Fin de Máster: "ASA: Sistema automático de personalización aplicado a personajes de videojuegos", realizado durante el curso académico 2010-2011 bajo la dirección de Pedro A. González Calero [y con la colaboración externa de dirección de Héctor Gómez Gauchía] en el Departamento de Ingeniería del Software e Inteligencia Artificial, y a la Biblioteca de la UCM a depositarlo en el Archivo Institucional E-Prints Complutense con el objeto de incrementar la difusión, uso e impacto del trabajo en Internet y garantizar su preservación y acceso a largo plazo

2 Abstract

ASA is a Automatic System Adaptation of applications to the user's characteristics, this adaption is independent of the domain, which gives reusability to be applied to all applications. For knowledge representation has been used ontologies, which makes the system scalable, and makes use of CBR which gives us "jColibri" to obtain similar profiles, existing in the ontologies. The user's characteristics used to perform the customization are the HCC Issues, which are divided into physical(gender, age, hair color, left or right handed), cultural(computer literacy level, country, cultural symbols, language, profession, religion), emotional(emotions, moods, states of mind) and personal(habits, tastes, temperament). The temperaments are an important part to the determinate the personality, therefore provides three versions of customizers, depending on how use these temperaments: Absolute, which only takes into account one of the temperaments, Combined, takes into account their level of membership of each temperament, and Keirsey, applying the theory of Keirsey, that the people change a temperament or other, in the proportion to the levels temperaments that we have. For temperament, I offer a modified test of Keirsey, applying fuzzy set theory, which gives the level of membership of each temperament.

As a case study was used the adaptation of the behaviour of the Non-Player Character(NPCs) to the user's characteristics. We used the game “CUBE II saurbraten ”, that is free software implemented in C++. For the customization have been used NPCs attributes such as speed, gun, health, aggression, and the shoot frequency.

3 Resumen

ASA es un sistema de adaptación automática de aplicaciones a las características del usuario, esa adaptación es independiente del dominio, lo que le da reusabilidad para ser aplicado en todo tipo de aplicaciones. Para la representación del conocimiento se han utilizado ontologías, lo cual hace al sistema escalable, y se hace uso del CBR que nos proporciona "jColibri" para la obtención de perfiles similares, ya existentes en las ontologías. Las características utilizadas para realizar la personalización son los HCC Issues, que están divididas en físicas(edad, sexo, color de pelo, mano preferida), culturales(conocimientos informáticos, país, símbolos culturales, lenguaje, profesión religión), emocionales(emociones, estado de ánimo, estado mental) y personales(hábitos, gustos, temperamento). Los temperamentos son una parte muy importante para determinar la personalidad, por ello proporciono tres versiones de personificadores en función de como se utilizan dichos temperamentos: Absoluto, que solo tiene en cuenta uno de los temperamentos, Combinado, que tiene en cuenta su grado de pertenencia de cada temperamento, y Keirsey, que aplica la teoría de Keirsey, de que las personas vamos cambiando un temperamento u otro, de manera proporcional a los niveles de temperamentos que tengamos. Para obtener el temperamento, ofrezco una modificación del cuestionario de keirsey, aplicándole la teoría de conjuntos borrosos, del cual se obtiene el grado de pertenencia a cada temperamento.

Como caso de estudio se ha utilizado la adaptación del comportamiento de los personajes de videojuegos (NPCs) a las características de usuario. Se ha utilizado el juego “CUBE II saurbraten ”, que es un shooter de software libre realizado en C++. Para realizar la personalización se han utilizado atributos de los NPCs como son la velocidad, el arma, la salud, la agresividad y la frecuencia de disparo.

4 Listado de palabras clave(Inglés)

Adapted Games
Adapted Application
Application Customization
Human-Centered Computing
Non-Player Character
Human Computer Interaction
User Centered Architecture
Automatic Adaptation to User
Case Based Reasoning

5 Listado de palabras clave

Adaptación de juegos
Adaptación de aplicaciones
Personalización de la aplicación
Informática centrada en humanos
Personajes de videojuegos
Interacción Persona-Ordenador
Arquitectura centrada en el usuario
Adaptación automática a usuario
Razonamiento Basado en Casos

6 Introducción

Yo en este trabajo presento un modelo de personalización independiente del dominio, centrándome en el estudio del comportamiento de los personajes de videojuegos(NPCs), para demostrar la necesidad del uso de la personalización, ya que creo que como cada persona tiene unas preferencias diferentes, es necesaria la adaptación de manera automática de la aplicación según los HCC(Human Centered Computing) issues y otras características más particulares del usuario. La adaptación puede ser por ejemplo la apariencia visual(Colores y formas), el sonido, la narrativa y el comportamiento de los componentes de la aplicación. En mi caso de estudio me voy a centrar el comportamiento de los personajes de videojuegos.

Además de los videojuegos hay numerosas aplicaciones donde se pude aplicar la personalización, como puede ser en aplicaciones educativas ya que cada persona tiene unas capacidades y necesidades particulares y también se puede utilizar para facilitar el acceso a aplicaciones para los que tienen discapacidades.

En el caso de los videojuegos yo me voy a centrar en el comportamiento de los personajes de videojuegos no controlados por el usuario NPCs, aunque hay otras partes de los videojuegos en los que se aprecia antes las adaptaciones como puede ser la parte visual, ya que lo primero que percibimos es lo que nos entra por los ojos, y también los efectos de sonido. Quizás el comportamiento de los NPCs al principio no se note mucho ya que los videojuegos están hechos para entretener, pero sí se aprecia con el tiempo, pues se podría producir cierta pasividad o aburrimiento por un comportamiento monótono de los NPCs, o frustración e impotencia por no cumplir los objetivos por una excesiva dificultad del juego. Está claro que a los usuarios que no les guste jugar a videojuegos va a ser muy complicado que con la adaptación les guste jugar, pero si se puede conseguir que a gente que les produzca miedo el que les ataque con tanta agresividad, modificando los parámetros de los NPCs para un comportamiento menos agresivo, se sientan más seguros y vayan animándose a jugar, quizás para este tipo de usuarios sería mejor adaptar la parte visual, para que se encuentre más a gusto con los videojuegos, ya que por ejemplo en los shooters suelen tener colores muy oscuros y tenebrosos, quizás con colores más alegres se conseguiría un entorno más agradable para el usuario. También se podría realizar un estudio en un futuro de la inclusión de música en las aplicaciones para buscar mayor excitación o relax en función de las preferencias del usuario y de las circunstancias en que se encuentre durante la ejecución de la aplicación.

Este proyecto se ha desarrollado a partir de la idea de COBBER[3] que consiste en un sistema conversacional para mantener un buen estado de ánimo del usuario, teniendo en cuenta su temperamento, hace uso de ontologías para representar el conocimiento y CBR para la recuperación y adaptación de casos, para la personalización de la aplicación e introducido más características de usuario como son los HCC issues, en donde están englobados los temperamentos, y otras características mas particulares del usuario (comportamiento ,habilidades y experiencia del usuario), independientemente del dominio, centrándome en la personalización del comportamiento de los personajes de video-juegos.

Para representar el conocimiento se hace uso de ontologías. El término ontología en informática hace referencia a la formulación de un exhaustivo y riguroso esquema contextual dentro de uno o varios dominios dados; con la finalidad de facilitar la comunicación y el intercambio de información entre diferentes sistemas y entidades. Hago uso de ontologías ya que se pueden reutilizar, lo que facilita que el sistema sea escalable. He creado una ontología HCCIonto.owl que contiene las características del usuario(HCC issues, comportamiento, habilidades y experiencia) independientemente de cuál sea el dominio, para que luego sea reutilizada por otras ontologías que definan las variaciones en un determinado dominio en función de las características definidas en

HCCIonto.owl. Existe otras dos ontologías que importan HCCIonto.owl, la ontología VARIonto.owl que se encarga de representar las variaciones posibles del videojuego del que realizo la personalización y relacionar esas variaciones con las características independientes del dominio, el estado del jugador y del NPC al que se realiza la variación. Mientras que FEEDonto.owl se encarga de determinar que emociones(propiedad de HCCIonto.owl) experimenta el usuario a partir de unas estadísticas recogidas por el FeedBack. Tanto VARIonto.owl como FEEDonto.owl constan al principio con una serie de casos bases iniciales creados a partir de fórmulas y reglas, que serán descritas en el apartado de las Ontologías (8.4.1), y mediante el uso de CBR se busca el caso más similar, aprendiendo un nuevo caso si produce buenas emociones al usuario, ya sea a partir de un caso recopilado o de la adaptación manual del usuario, esta parte se explicará más en detalle en el apartado CBR(8.4.2). En el siguiente esquema de la Figura 6.1 se puede ver la arquitectura del sistema.

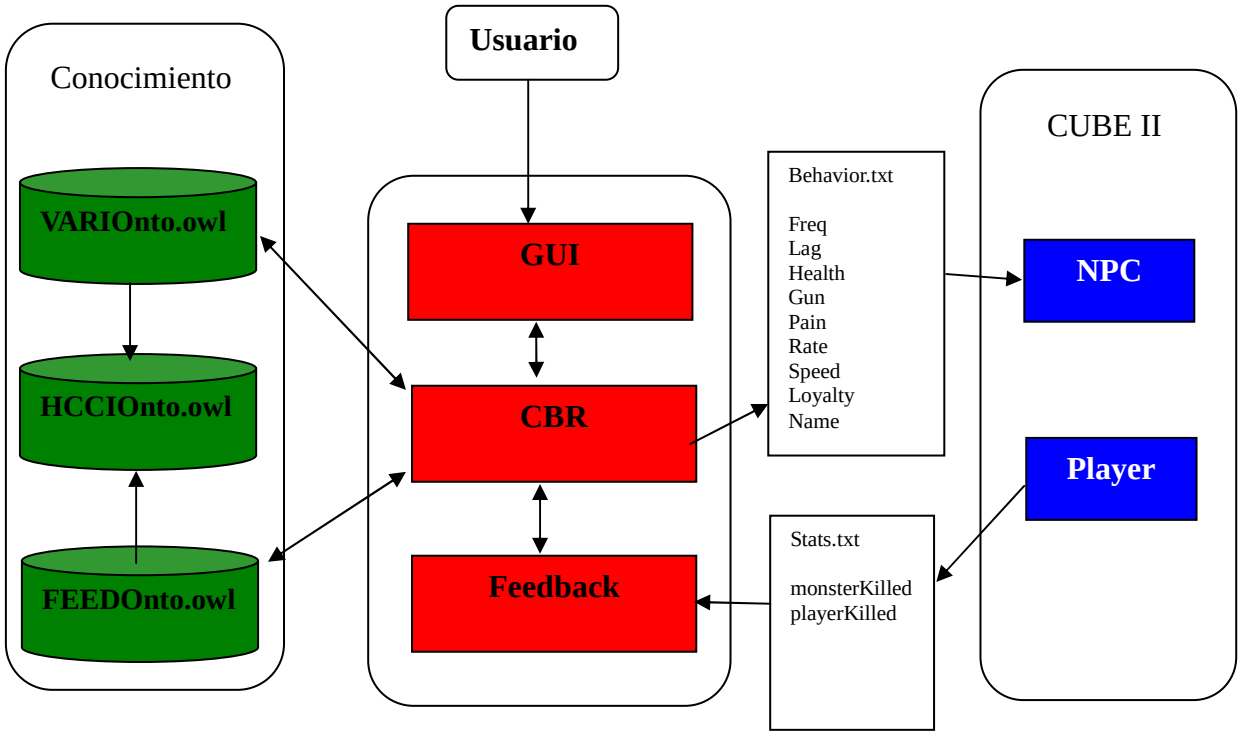


Figura 6.1: Arquitectura ASA

El sistema está integrado en jColibri porque me da funcionalidades para la conexión con la ontologías, la recopilación de casos y el aprendizaje de casos. La primera versión de JColibri fue un completo desarrollo de la arquitectura CBR apoyado con muchas características como interfaces gráficas, descripciones lógicas y ontologías. Sin embargo, por problemas de escalabilidad y facilidad de desarrollo surgió JColibi2.

El objetivo es conseguir un modelo de personalización independiente del dominio, como caso de estudio se aplicará el modelo al comportamiento de los personajes de videojuegos. Para la personalización del dominio en función al temperamento del usuario, se ofrece un test de Keirsey, el cual sufrirá una modificación con respecto a los habituales cuestionarios, ya que el usuario en vez de seleccionar una de las cuatro respuestas que ofrece el test, indicara el grado de pertenencia que tiene con cada una de las respuestas para obtener un temperamento mas preciso del usuario. También se le preguntará al usuario sobre los HCC issues y otras características del usuario, todo esto personalizará el dominio de manera estática y será guardado en un registro para que el usuario no tenga que volver a realizar el cuestionario. También se proporcionará un feedback que recogerá

de manera dinámica las emociones y la experiencia del usuario en función de las estadísticas del juego.

Para el temperamento se ha utilizado la teoría del temperamento de Keirsey. El temperamento describe nuestra personalidad y se obtiene mediante un test. Hay 4 tipos de temperamento principales, los cuales tienen 4 subtipos, por lo que hay 16 tipos que definen nuestra personalidad:
Guardian SJs: Supervisors(ESTJ), Inspectors(ISTJ), Providers(ESFJ), Protectors(ISFJ).
Artisans SPs: Promoters(ESTP), Crafters(ISTP), Performers(ESFP), Composers(ISFP).
Idealists NFs: Teachers(ENFJ), Counselors(INFJ), Champions(ENFP), Healers(INFP).
Rationals NTs: Field Marshals(ENTJ), Masterminds(INTJ), Inventors(ENTP), Architects(INTP).

Los artesanos son observadores y pragmáticos. Buscan la estimulación y el virtuosismo. Su punto fuerte es la táctica. Sobresalen en la solución de problemas, la agilidad y la manipulación de herramientas, instrumentos y equipos. Son animados.

Los guardianes son atentos y cooperativos. Buscan la seguridad y la pertenencia, que tienen que ver con la responsabilidad y el deber. Su mayor fortaleza es la logística. Sobresalen en la organización, facilitando control y apoyo. Son pesimistas.

Idealistas son introspectivos y cooperativos. Buscan el significado, están preocupados con el crecimiento personal y encontrar su propia identidad. Su mayor fortaleza es la diplomacia. Tienden a aclarar, individualizar, unificar e inspirar. Son entusiastas.

Racionales son introspectivos y pragmáticos. Buscan el dominio y auto-control, están preocupados con sus propios conocimientos y competencias. Su mayor fortaleza es la estrategia. Ellos son excelentes en cualquier tipo de investigación lógica, como la ingeniería, la conceptualización, la teorización, y la coordinación. Son tranquilos.

La investigación en HCC (Human Centered Computing) persigue la comprensión de los seres humanos. Para ello se tienen en cuenta varias cuestiones:
-Las cuestiones culturales: se incluyen los términos en muy diversos niveles de abstracción, de los ámbitos generales como lengua, país, religión, a ámbitos más estrechos como profesión, conocimientos informáticos y estudios técnicos.
-Aspectos físicos: entre ellos, la edad, el sexo, raza, si es zurdo.
-Las cuestiones personales: se incluyen términos tales como temperamento, gustos y hábitos.
-Las cuestiones emocionales o afectivas: engloban las emociones, el estado de ánimo y el estado mental.

De todos los “HCC issues” para mi caso de estudio que es el comportamiento de los personajes de video-juegos se han cogido la edad, sexo, temperamento, emociones, estado de ánimo y otras característica como la experiencia del usuario con la aplicación. Aunque en un futuro se podrían coger otras características como los gustos que puede tener el usuario por una determinada música o colores por ejemplo, o hábitos en nuestro caso de estudio, los hábitos del usuario podrían ser que siempre ataca, o siempre elige ir a la derecha, para modificar el comportamiento de los NPCs y así cambiar su estrategia para hacer más entretenido el juego al usuario.

El estado de ánimo viene influido por las emociones experimentadas por el usuario, éste contiene un buen estado de ánimo cuando el usuario se encuentra feliz o satisfecho, mientras que es malo cuando las emociones son negativas como la frustración, impotencia, aburrimiento, apatía, pasividad, pereza, inactividad, tristeza, la insatisfacción, decepción, ira, miedo y ansiedad.

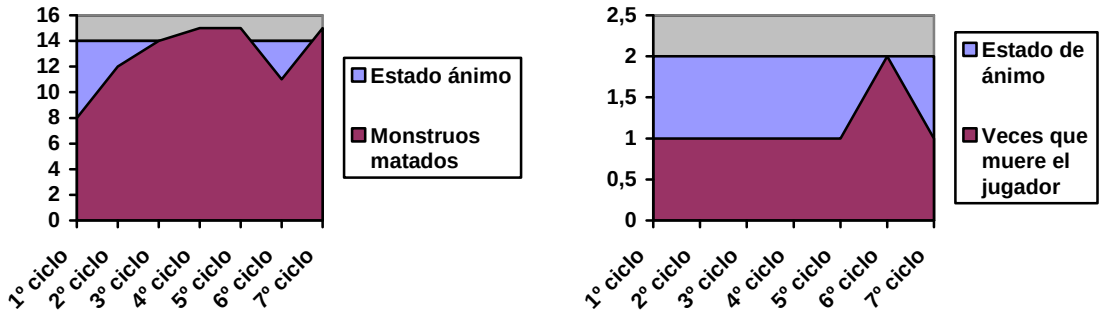
Para llevar a cabo el proyecto, se ha investigado todo lo relacionado con los aspectos humanos

relacionados con los ordenadores, temperamentos, HCC, customización o personalización del dominio, NPC y como adaptar el comportamiento de un personaje de videojuegos a las características del usuario. El proyecto está integrado en JColibri[6], y tiene tres partes principales: Cuestionario, el feedback y el CBR. También se hará uso de ontologías HCCIOnto (HCC issues) independiente del dominio, VARIOnto(variaciones del comportamiento de los personajes de videojuegos) y FEEDOnto(Recoge los resultados del usuario y evalúa las emociones y estado de ánimo).

Existen tres versiones del sistema según como se trate el temperamento. En la primera versión solo se coge el temperamento más característico del usuario y el resto de características para la recuperación de casos y luego se hace uso del feedback para recoger si el estado de ánimo es el adecuado y también si evoluciona la experiencia del usuario. En la segunda versión se hace una mezcla de todos los temperamentos del usuario teniendo en cuanto el grado de pertenencia que tiene el usuario con cada temperamento y junto al resto de características de usuario se hace la recuperación de casos, luego durante la ejecución se hace uso del feedback para recoger el estado de ánimo de la persona y también si evoluciona la experiencia del usuario. En la tercera se tiene un planificador que irá cogiendo de forma aleatoria el temperamento, dando la probabilidad de tener un temperamento u otro al grado de pertenencia a dicho temperamento, para la recuperación de casos también se tendrán en cuanta las otras características del usuario, en esta versión el feedback se utilizará para obtener el estado de ánimo y la experiencia del usuario durante el desarrollo de la aplicación. Estas tres versiones se explicarán de forma más detallada en el punto de análisis del modelo.

El videojuego del que se hace el estudio es el "CUBE II saurbraten "[20], que es un shooter, de software libre realizado en C++, para que el usuario tenga durante las partidas un estado de ánimo adecuado, se modificaran varios atributos de los monstruos(Personajes del video-juego) como son la velocidad, el arma, la salud, la agresividad y la frecuencia de disparo.

A continuación se ve la evolución en una partida de una persona con temperamento artesano.



De estos gráficos explicar que en cada ciclo se evalúa el estado de ánimo del jugador, mientras que la experiencia del jugador se recoge cada tres(6 minutos) ciclos. En la primera gráfica cuando el usuario mata menos de 14 monstruos se considera que se está aburriendo, mientras que en la segunda gráfica cuando a un usuario le matan más de 1 vez se considera que el jugador se siente frustrado. Como se puede ver en la primera gráfica. El que haya muerto 2 veces en el 6º ciclo es debido a que como cada 3 ciclos se recoge la experiencia como en esos 3 ciclos ha matado a más de 42 monstruos se aumenta la experiencia lo que conlleva a que haya muerto 2 veces, lo que llevaría a otra adaptación, todo esto se explicará en el tercer apartado.

7 Estado del arte

Personalización automática del HCI usando Temperamentos [1], este articulo describe un modelo de personalización automática teniendo en cuenta la interacción del usuario con el sistema para obtener el estado de ánimo del usuario y el temperamento del usuario, el modelo es independiente del dominio y reutilizable, se centra en la personalización del sistema operativo Linux. Hacen uso de CBR y ontologías para representar el conocimiento.

Personalización automática de los NPCs al Temperamento [2], este artículo describe un modelo de personalización automática del comportamiento de los NPCs al temperamento del usuario, hace uso de ontologías para representar el conocimiento y de CBR.

COBBER [3], describe un modelo conversacional que intenta conseguir que el usuario mantenga un buen estado de ánimo mientras interactúa con el sistema, a partir del uso de ontologías para representar el conocimiento y de CBR para la recuperación y adaptación de los casos. Esta adaptación se hace teniendo en cuenta el temperamento y el estado de ánimo del usuario.

Mi sistema se basa en este modelo, al que se le han añadido características como el reto de "HCC issues", ya que en COBBER solo tiene en cuenta el temperamento y estado de ánimo del usuario, y se han añadido funcionalidades nuevas, como es la personalización del comportamiento de los NPCs.

Ontología para el comportamiento de los NPCs [4], Habla que dado que los personajes de los videojuegos suelen tener un comportamiento similar, habría que tener una ontología con los comportamientos de los personajes de videojuegos y con el uso de CBR se obtenga el caso más aproximado para adaptar el comportamiento del personaje de videojuego.

PERSIA [5], Proyecto de fin de carrera del curso 2005/2006 que consiste en la personalización del sistema operativo Linux basado en las preferencias y temperamentos de los usuarios. Hacen uso de ontologías para la representación del conocimiento y de CBR.

Para la adaptación del sistema solo tiene en cuenta el temperamento del usuario, y no considera las emociones del usuario, que posiblemente sea la característica más importante a tener en cuenta.

Taller Human-Centered Computing [7] .aplican el uso de HCC centrado en la multimedia en conferencias, para hacer que las conferencias sean más interesantes. Realizaron para ello trabajo de investigación y mejoras con la experiencia.

Uso de HCC multimedia para discapacidades [8], En este artículo centra el uso de la multimedia para adaptar las aplicaciones a las discapacidades del usuario. El articulo habla de uso del HCC para ayudar a las personas con discapacidades y habla de algunas discapacidades. Se centra en el Autismo, Parkinson y Alzheimer.

Uso del HCC para aplicaciones bursátiles [9], este artículo se centra en aplicar HCC en aplicaciones bursátiles para evaluar como se va a comportar la bolsa.

HCC: Toward a Human Revolution [10], este articulo habla de que la tecnología ha avanzado mucho, pero hay que intentar hacerla más accesible. Para ello intenta convencer de que el HCC es una tecnología emergente, dada la necesidad de personalizar las aplicaciones a las necesidades particulares de las personas, el artículo se centra en explicar el HCC en sí.

Ampliación del HCC para discapacidades [11], este artículo trata de la ampliación de HCC para facilitarle las cosas a la gente con discapacidad. Para ello habla de la necesidad de investigar a la gente individualmente y a la sociedad, además de la diversidad que existe entre ellos (HCC issues). Es un taller de investigación, que utiliza la investigación cruzada de distintos objetivos (Necesidad y visión, objetivos de la investigación y la ciencia fundacional).

Evaluación de los tiempos de respuesta [12], Se centra en el estudio de la percepción del usuario en función del tiempo de respuesta del sistema.

Personalización de videojuegos basado en aprendizaje [13], este artículo habla de tratar de personalizar juegos al perfil del usuario. Para ello usa la Interacción persona – ordenador, la respuesta afectiva del usuario dependerá de su personalidad y del estimulo del entorno. La personalidad del usuario se define en un conjunto de atributos: preferencia, actitud, carácter, conocimientos y habilidad. El perfil de usuario se construye a través de pruebas psicológicas y el reconocimiento de la expresión facial. Utilizaron para las prueba el videojuego QUAKE III que es software libre.

Usa solo unas pocas de las características del ser humano(preferencia, actitud, carácter, conocimientos y habilidad), el uso del reconocedor facial para recoger las emociones es interesante.

Evaluación y estudio de ELEKTRA [14] y [15], ELEKTRA es un proyecto subvencionado por la Unión Europea para la personalización de juegos educativos. el entorno gráfico del juego (GE) del motor de aprendizaje (LE), que es el encargado de personalizar la herramienta educativa. El LE se realiza en 4 etapas: la inferencia, la acumulación de contexto, la restricción de la adaptación, y la selección de adaptación. Representa el conocimiento en ontologías. El sistema realiza la adaptación teniendo en cuenta la ciencia cognitiva, la neurociencia, la pedagogía, el diseño del juego, y el desarrollo del juego.

Implementación del comportamiento emocional en sistemas multiagentes usando lógica borrosa y temperamento [16], este artículo describe un modelo para evaluar el estado emocional en sistemas multiagentes mediante algoritmos de toma de decisiones en función al temperamento y lógica borrosa.

Solo utiliza el temperamento y el uso de reglas, hace al sistema difícilmente escalable.

Juegos adaptados emocionalmente [21] y [22], en estos artículos se presenta un sistema de personalización psicológica que implica la personalización de la forma de presentar la información. Para llevar a cabo la personalización se centra en las emociones de las personas, consideran que la narrativa del juego, la presentación visual del juego, la cantidad y ritmo de movimiento de las imágenes, efectos de sonido, música de fondo y el nivel de interactividad que ofrece al jugador son importantes desde el punto de vista de las emociones. En la experiencia con los videojuegos distingue entre dos tipos de experiencias a tener en cuenta, si la experiencia es agradable o no, y si la experiencia crea gran excitación o calma. Distingue entre tres tipos de emociones: positivas(felicidad, satisfacción), negativas(tristeza, insatisfacción, decepción, ira, miedo, ansiedad) y neutras(inactividad, pereza, pasividad, cansancio, aburrimiento, apatía, impotencia), durante la partida hay que intentar prevenir que no surjan ni emociones negativas ni neutras. Sobre la arquitectura del sistema está el videojuego y luego una aplicación de personalización automatizado que interactúa con el videojuego. La aplicación de personalización incluye el modelado de los individuos, grupos y comunidades para crear perfiles psicológicos y otros perfiles más personalizados. Para la personalización se basa en un conjunto de reglas. Hace uso de un feedback que interactúa con el videojuego, en la figura 2.1 se muestra como interactúa en cada una de las

capas del videojuego.

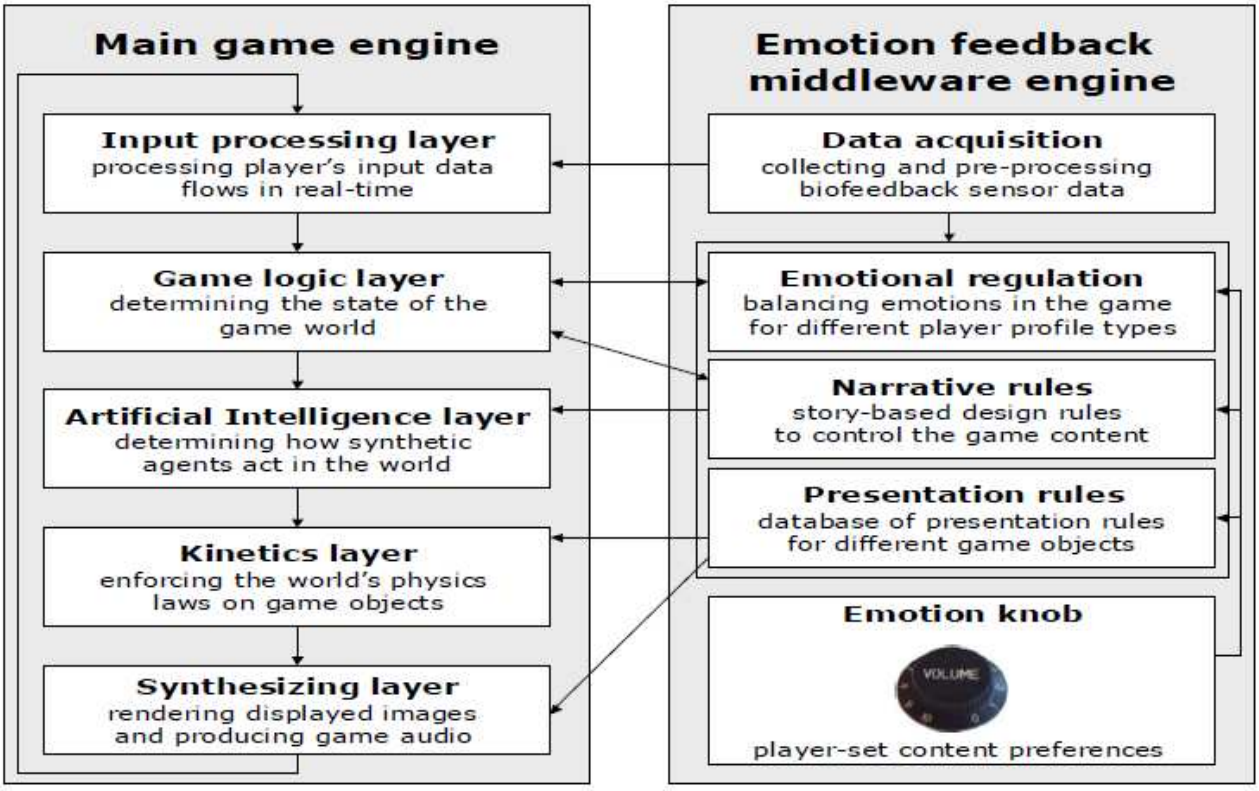


Figura 7.1: Figura extraída del artículo

No es independiente del dominio, se centra solo en las parte afectiva del usuario, y el uso de reglas hace al sistema difícilmente escalable.

Adaptador de escenario [23], este artículo intenta solucionar el problema de que cada alumno tiene unas habilidades, necesidades u objetivos de aprendizaje particulares, para ello presenta una metodología para adaptar automáticamente un escenario de un juego que mejor se adapte a nuestros requisitos. Ellos describen un "Adaptador de escenario" que emplea algoritmos de planificación que añaden y eliminan elementos de un escenario que están relacionados con el objetivo del aprendizaje. Este artículo se centra en escenarios basados en scripts. En la figura 2.2 establecen un gráfico entre la zona de la confusión y el aburrimiento.

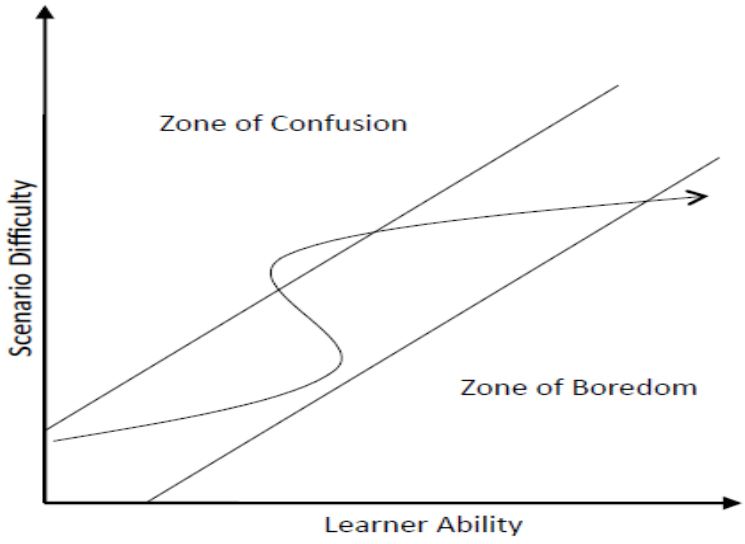


Figura 7.2:Figura extraída del artículo.

Lo ideal es que el alumno con el tiempo no se desvié ni a la zona de confusión, ni a la de aburrimiento. Para evitar esto definen operaciones de eliminación, adición y sustitución de los objetos de aprendizaje en un escenario, presentan una extensión de la planificación para la adaptación parcial de un escenario. Ellos toman de los scripts, la narrativa y los escenarios como descriptores de los acontecimientos o secuencias de acontecimientos. Para el proceso de re-escritura es necesarios de dos procesos externos. El primero que se encargue de la actualización de un modelo de aprendizaje, y la elección del escenario de entrenamiento. El segundo un motor de juego que forma los escenarios con el estudiante como participante interactivo. Además también es necesario un plan de generación de escenarios y un conjuntos de objetos de aprendizaje. Por lo tanto se consigue que los estudiantes puedan dedicar menos tiempo a tareas que son demasiado fáciles o difíciles.

Solo se centra en algunas características como son habilidades, necesidades u objetivos de un aprendizaje del alumno, no es independiente del dominio, ya que se centra en la adaptación de videojuegos a las necesidades del alumno, y se necesita de un experto que diseñe la secuencia de acontecimientos y la relación entre dichos acontecimientos.

Adaptador de la línea argumental del juego [24], en este artículo se trata la personalización de la trama, es decir modificar la historia del juego en función de las preferencias del jugador. Los autores se centran en la generación de contenido automático en los juegos de rol. Además de las preferencias iniciales del jugador, también tienen en cuenta que dichas preferencias pueden cambiar con el tiempo. Ellos ofrecen un sistema que resuelve el problema de adaptación de tramas: dada la historia completa, la trama que consiste en una secuencia de misiones, una biblioteca de misiones, y una serie de requisitos para optimizar el juego para determinado usuario en un momento determinado, intentando mantener gran parte de la trama original cuando sea posible. Utiliza el razonamiento basado en casos para aprender las preferencias de los jugadores. Por lo cual su sistema es un proceso de re-escritura de una trama en función del modelo del jugador antes de su ejecución. Es un sistema estático, es decir, no interactúa con el usuario, y una vez planificada la trama y cuando esta empieza, no hacen cambios. Aunque si se puede combinar con otros sistemas que sí interactúen con el usuario. Lo que ellos llaman modelo de jugador consiste en un conjunto de preferencias del jugador y algo que ellos llaman modelo de la novedad para que la historia no sea aburrida, aunque para que no se produzcan sorpresas desagradables hacen una combinación de las preferencias y la novedad. Su representación del sistema se basa en lo que ellos llaman planes parciales de orden, un plan parcial de orden consiste en eventos y relaciones causales, los eventos codifican las condiciones previas y las relaciones causales denotadas como e1 e2 $\rightarrow$ c, indican que los efectos del evento e1 establecen un estado c que es necesario para el evento e2, es decir que para que se cumpla e2, antes tiene que haberse ejecutado el evento e1. Las reglas de descomposición deben ser escritas a priori.

No es independiente del dominio, ya que se centra solo en la adaptación de la trama del videojuego, aunque hace uso de CBR para recuperar modelos de usuario parecidos, hace uso de reglas para la elección de planes, por consiguiente es un sistema difícilmente escalable.

Estudio sobre la extroversión de una persona en los videojuegos [25], se presenta un estudio experimental que tiene como objetivo realizar la adaptación del videojuego en función del perfil psicológico del jugador, el estudio sólo se centra en uno de los factores del perfil psicológico, la extroversión. El experimento lo realizaron 40 participantes, separados en dos grupos de 20 participantes, en el primer grupo(grupo A) jugaron a una versión del juego que se adaptaba a su nivel de extroversión, mientras que el segundo grupo(grupo B) jugaron a una versión del juego que no se adaptaba a su nivel de extroversión. luego los participantes rellenaron un cuestionario SAM para finalizar el experimento. SAM es un conjunto de maniquíes gráficos diseñados para evaluar: la valencia, la excitación, el dominio. De estos cuestionarios se detectó que los jugadores eran más

extrovertidos que introvertidos, no se detectaron diferencias sustanciales entre las puntuaciones del grupo A y el grupo B, aunque las diferencias no eran significativas, parece que los participantes de la versión A disfrutaron más. El que no haya diferencias significativas viene dado porque la gente extrovertida suele disfrutar en general con los videojuegos, al final concluye que es necesario una mayor investigación.

Adaptación Educacional con sistemas hipermedia [26], en este artículo se hace eco de la utilización de los sistemas hipermedia adaptativos educativos (AEHS) para mejorar el aprendizaje de los alumnos, tratando de personalizar la experiencia de aprendizaje, estas experiencias de aprendizaje educativas se deben adaptar a las necesidades particulares de cada alumno. Esta adaptación se basa en varias características del alumno, incluyendo el nivel de conocimiento, los objetivos o la motivación. Su propósito es maximizar la satisfacción del alumno, la velocidad de aprendizaje y la eficacia de la educación. A los efectos de este trabajo, un sistema de adaptación se refiere a un sistema que adapta su producción, utilizando las inferencias implícitas basadas en la interacción con el usuario. Un sistema adaptativo de hipermedia (AHS) se refiere a un sistema hipertexto o hipermedia que refleja algunas características del usuario en un modelo y aplica este modelo para adaptar varios aspectos visibles del sistema. Uno de los problemas es la falta de recursos que hay actualmente.

Además de la falta de recursos, no es independiente del dominio, ya que está centrado únicamente al la personalización de entornos educativos.

TAP [27] y [28], En estos artículos se muestra un técnica que consiste en un módulo de adaptación de la personalidad encapsulado en una sistema de aprendizaje por refuerzo. Cada NPC es un agente con una personalidad definida, por lo que cada agente tomará acciones de acuerdo a las restricciones impuestas en su personalidad. Ellos crean un equipo (sistema multiagente) que se adapta al jugador, que funciona mejor que un equipo con el comportamiento de secuencias de comandos. En su estudio se centra en los NPCs aliados, es decir personajes de videojuegos que colaboran y se adaptan al jugador. Cada agente se ajusta automáticamente para estilos y estrategias individuales, con el fin de alcanzar objetivos comunes con los jugadores. Ellos crean agentes que se adaptan tanto a la personalidad del jugador, como con la personalidad del resto de NPCs del equipo.

Aunque el uso de la teoría de agentes es interesante, su implementación es costosa, ya que habría que cambiar la inteligencia artificial del juego, para que pudieran cambiar su estrategia ya sea para colaborar con el usuario o para atacarle. No es independiente del domino.

Personalización probabilística [29], Se presenta un enfoque inductivo de la predicción de los objetivos del usuario por modelos de reconocimiento probabilístico, presentan dos modelos, n-gramas y red bayesiana. Toman en cuenta los objetivos del usuario para la construcción de la trama e intentan garantizar en la historia un constante progreso. Para un funcionamiento eficiente debe inferir con precisión los objetivos de los usuarios en tiempo real. No preguntan al usuario sobre sus objetivos, sino que lo consiguen a partir de sus acciones. Para el reconocimiento probabilístico hacen uso del estado de la narración, las acciones del usuario y la ubicación del usuario. Ellos definen el reconocimiento narrativo de la siguiente manera: Dada una secuencia de n acciones del usuario a1,a2,....an, en un entorno de la narrativa, de sus estados asociados de la narrativa n1,n2,....nn, y las ubicaciones del usuario l1,l2,...ln, identifica el objetivo G más probable de un conjunto de objetivos candidatos g1,g2,...gn.

Además de que no es independiente del dominio, dado que solo se centran en la adaptación de las tramas, aunque la idea se podría aplicar a otros dominios, es difícil de implementar a no ser que el sistema a adaptar ofrezca facilidades para la descomposición de la trama.

Director de agentes [30], Presentan un framework para la creación de narrativas interactivas para entretenimiento educación y formación. El "director de experiencia"(Director de agentes) es un agente inteligente que manipula el mundo virtual para obligar que el usuario cumpla un conjunto de propiedades. Su sistema se va adaptando a las acciones del usuario en el mundo virtual. El director de agentes envía comandos para que sean ejecutados por los agentes. Aunque también los agentes tienen libertad para actuar de manera natural para involucrar a los jugadores.

8 Análisis del modelo

Antes de nada hay que explicar que los temperamentos se miden en un rango de valores de 0 a 100 según su grado de pertenencia con el temperamento, ya que una persona puede ser muy de todos los temperamentos o poco. El temperamento se obtiene a partir de un cuestionario de Keirsey, pero a diferencia de la teoría de Keirsey, no se obtiene en términos absolutos, sino que se aplica la teoría de conjuntos borrosos, es decir al realizar el cuestionario en vez de indicar de mayor a menor las respuestas, debes indicar con cada respuesta como te sientes identificado, se indicará de 0 a 5, según se identifique con la respuesta, siendo 0 nada identificado y 5 totalmente de acuerdo, a partir de todas las respuestas se obtendrán cada uno de los temperamentos como una medía ponderada de cada una de las respuestas que pertenezcan al temperamento. El cuestionario consta de 16 preguntas, cada una con 4 respuestas, cada respuesta está asignada a un temperamento. El grado de pertenencia al temperamento se obtiene de la siguiente manera:

para p=1 hasta 16, tT = tT+rT[p], gT = tT/(16*5)

Donde:

p: Indica la pregunta.
tT: Es el acumulado de todas las respuestas del temperamento T.
rT[p]: Es el grado de pertenencia del usuario a la respuesta relacionada con el temperamento T de la pregunta p.
gT: Es el grado de pertenencia del usuario al temperamento T.
T={Artesano, Guardián, Idealista, Racional}

En función de como se utilice el temperamento para adaptar la aplicación existen 3 versiones, en la primera versión llamada Personalizador Absoluto, solo se tiene en cuenta el temperamento más característico del usuario; en la segunda versión llamada Personalizador Combinado, se tienen en cuenta todos los valores del temperamento del usuario; en la tercera versión, Personalizador Keirsey, se van cogiendo cada uno de los temperamentos del usuario de manera absoluta en proporción al grado de pertenencia que tenga al temperamento con respecto al grado de pertenencia con el resto de temperamentos. Para la adaptación también se tiene en cuenta la edad, el sexo, las emociones y la experiencia del usuario. A continuación se muestra un ejemplo para que se entienda mejor, y después se explica con más detalle cada una de las versiones.

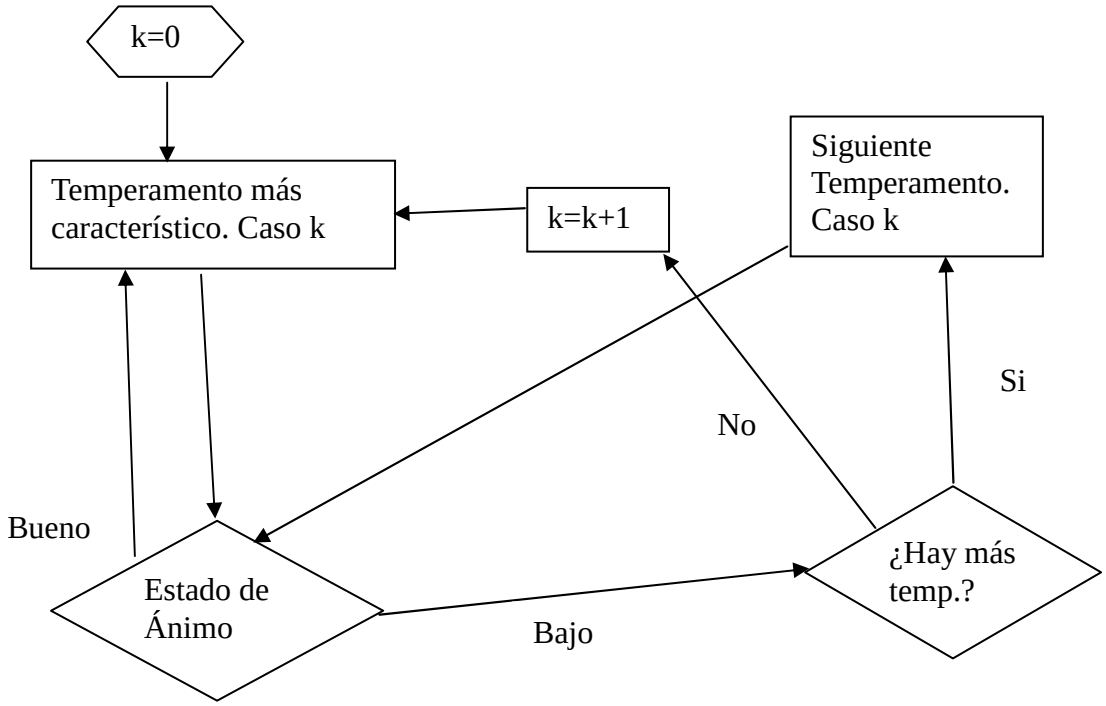
En el siguiente ejemplo, se muestran las diferencias de funcionamiento de las tres versiones, hay que tener en cuenta que durante los diez ciclos el estado de ánimo es bueno, en caso contrario la adaptación se haría de manera distinta en cada una de las versiones, estas adaptaciones se explican más detalladamente en cada una de las versiones. Como se puede ver en el ejemplo tanto en el Personalizador Absoluto, como en el Combinado la adaptación no cambia, mientras que en el pesonalizador Keirsey si varía, cogiendo de manera aleatoria uno de los temperamentos de manera absoluta, como cada uno de los temperamentos se tiene que hacer de manera proporcional al grado de pertenencia al temperamento con respecto al resto de temperamentos, en los diez ciclos, será Artesano 4 veces, Guardián 2 veces, Idealista 2 veces y Racional 2 veces. El cómo se realiza la personalización en la tercera versión se explica después del ejemplo.

USUARIO
Artesano: 70
Guardián:30
Idealista:30
Racional:40

CICLO	ABSOLUTO	COMBINADO	KEIRSEY
1	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=70;G=0; I=0;R=0}
2	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=0;G=0; I=0;R=70}
3	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=0;G=0; I=70;R=0}
4	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=70;G=0; I=0;R=0}
5	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=0;G=70; I=0;R=0}
6	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=70;G=0; I=0;R=0}
7	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=0;G=0; I=70;R=0}
8	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=0;G=70; I=0;R=0}
9	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=70;G=0; I=0;R=0}
10	{A=70;G=0; I=0;R=0}	{A=70;G=30; I=30;R=40}	{A=0;G=0; I=0;R=70}

8.1. Personalizador Absoluto

En la primera versión solo se coge el temperamento más característico del usuario y el resto de características para la recuperación de casos y luego se hace uso del feedback para recoger si el estado de ánimo es el adecuado y también si evoluciona la experiencia del usuario. De los rasgos del temperamento solo se cogen el "intelecto", "ser" y "buscan" de cada uno de los temperamentos y se recogen como rango de valores de 0 a 100, cuanto mayor sea, más pertenece a ese rasgo, en esta versión al descartar el resto de temperamentos, solo se tendrá en cuenta los rasgos de ese temperamento, esos rasgos tomarán el valor del temperamento y el resto de rasgos del resto de temperamentos tomarán el valor 0. Un ejemplo, si una persona es Artesana con un grado de pertenencia de 60, Racional 30, Guardián 30 y Idealista 20, se tendrá solo en cuenta los rasgos del temperamento artesano dándole a táctico(Intelecto) como valor 60, emocionado(Ser) 60 y estímulo(Busca) 60 y al resto de rangos del resto de temperamentos el valor 0. Durante la ejecución si se recoge con el feedback que ninguno de las características del usuario varía y se tiene un estado de ánimo bajo se coge como temperamento el siguiente más característico del usuario, si tampoco se consigue mejorar se coge el siguiente así sucesivamente hasta que se cojan todos los temperamentos, si aún así no se mejora se vuelve a coger el más característico, pero en la recuperación de casos se coge el segundo caso más semejante, si tampoco se consigue mejora, se realizaría el mismo paso, pero con su segundo temperamento más característico, si tampoco se consigue cambio se trata el siguiente temperamento, así sucesivamente hasta encontrar un estado de ánimo adecuado. En caso de que se tenga el mismo nivel en dos temperamentos se cogerá el siguiente temperamento más agresivo o menos agresivo, los temperamentos están ordenados de menor a mayor agresividad de la siguiente manera: Guardián, Idealista, Racional, Artesano. Si está aburrido se coge el siguiente más agresivo, y si está frustrado se coge el siguiente menos agresivo de los que tengan el mismo grado de pertenencia con el temperamento que se está comparando. Una vez que el estado de ánimo sea bueno, se sigue con ese caso recuperado que nos ha dado buenas respuestas hasta que el estado de ánimo sea malo, en tal caso se vuelve a hacer una recuperación de casos como al principio, es decir cogiendo su temperamento más característico y resto de características del usuario. A continuación se puede ver el algoritmo de selección de casos, siempre y cuando no se produzca un cambio de características manualmente.



En la figura 8.1.1 se muestra un ejemplo usando el personalizador absoluto de una persona que tiene un grado de pertenencia al temperamento artesano de 70, al guardián de 40, al idealista de 30 y al racional de 80.

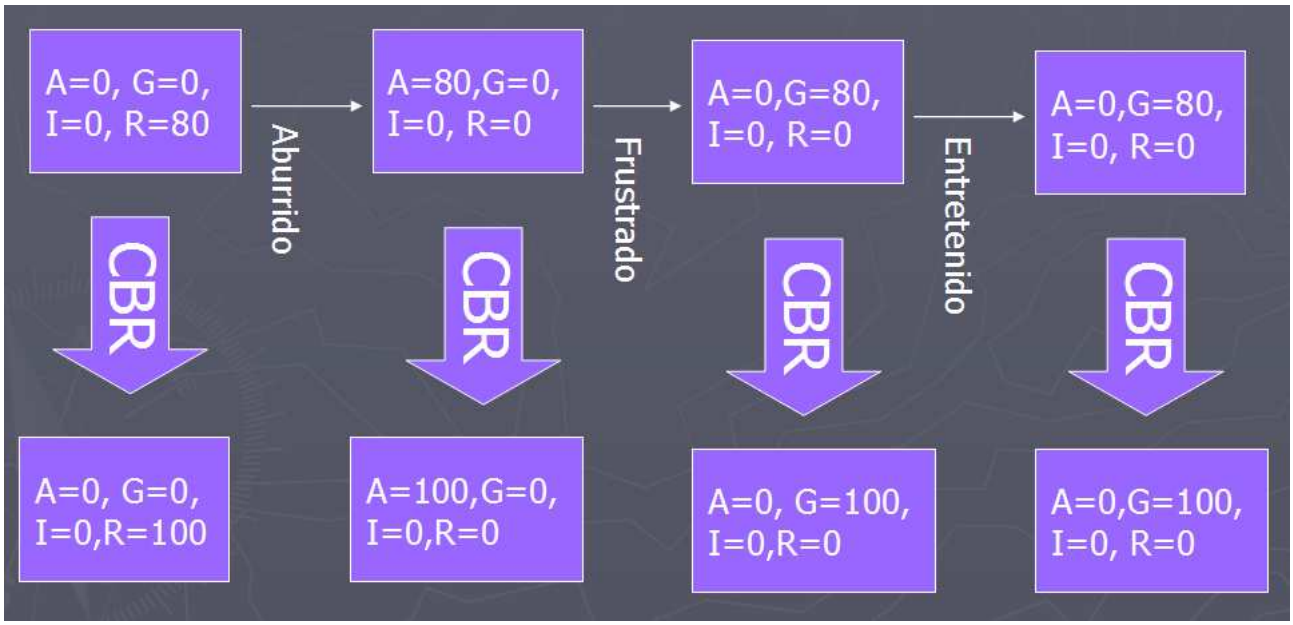


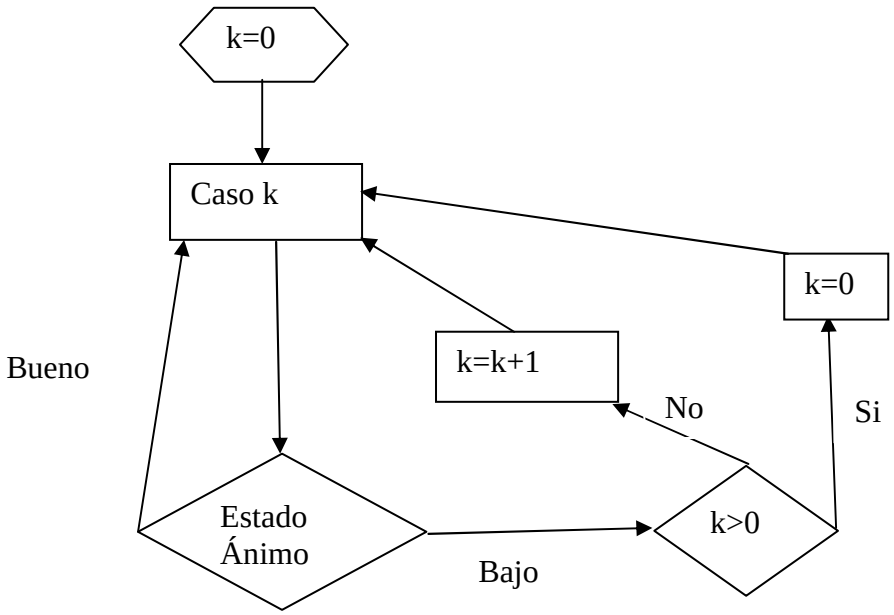
Figura 8.1.1: Ejemplo de personalización absoluta.

Durante el ejemplo se sigue la siguiente ejecución:

1. Como el temperamento más característico de este usuario es el racional, se tiene en cuenta solo este temperamento, dándole al resto el valor 0, y se obtiene en la base de conocimiento como perfil más similar [Artesano=0; Guardián=0, Idealista=0, Racional=100] y se aplica la adaptación asociada a ese perfil.
2. El Feedback recoge que el usuario está aburrido, por lo que se utiliza el siguiente temperamento más característico al actual, que es el artesano, dándole al resto el valor 0, y al artesano el valor del temperamento más característico que es 80, se obtiene en la base de conocimiento como perfil más similar [Artesano=100; Guardián=0, Idealista=0, Racional=0] y se aplica la adaptación asociada a ese perfil.
3. El Feedback recoge que el usuario está frustrado, por lo que se utiliza el siguiente temperamento más característico al actual, que es el guardián, dándole al resto el valor 0, y al guardián el valor del temperamento más característico que es 80, se obtiene en la base de conocimiento como perfil más similar [Artesano=0; Guardián=100, Idealista=0, Racional=0] y se aplica la adaptación asociada a ese perfil.
4. El Feedback recoge que el usuario está entretenido, por lo que se sigue con la ultima adaptación utilizada que nos ha dado un buen resultado, y se aprende este nuevo caso, como se explicará en el apartado del módulo CBR.

8.2. Personalizador Combinado

En la segunda versión se hace una mezcla de todos los temperamentos del usuario teniendo en cuenta el grado de pertenencia que tiene el usuario con cada temperamento y junto al resto de características de usuario se hace la recuperación de casos, luego durante la ejecución se hace uso del feedback para recoger el estado de ánimo de la persona y también si evoluciona la experiencia del usuario. En esta versión se tendrán en cuenta todos los temperamentos del usuario, donde todos los rasgos tomarán el valor que tenga el usuario como grado de pertenencia con dicho temperamento, es decir si una persona es Artesana con un grado de pertenencia de 60, Racional 30, Guardián 30 y Idealista 20, tendrá en los rasgos de artesano 60, es decir táctico(Intelecto) como valor 60, emocionado(Ser) 60 y estímulo(Busca) 60, en los de racional 30, estratégico(Intelecto) 30, tranquilo(Ser) 30 y conocimiento(Busca) 30, en los de guardián 30, logístico(Intelecto) 30, interesado(Ser) 30 y seguridad(Busca) 30, en los de idealista 20, diplomático(Intelecto) 20, entusiasta(Ser) 20, identidad(Busca) 20. Durante la ejecución si se recoge con el feedback que ninguno de las características del usuario varía y se tiene un estado de ánimo bajo se coge el siguiente caso más semejante de los recuperados, si tampoco se consigue mejora se busca el siguiente así hasta encontrar un caso que de buenos resultados, este caso se mantiene hasta que el estado de ánimo baje o se produzca algún cambio en las características del usuario, en tal caso se volvería a coger el primer caso más semejante a las características actuales del usuario. En el siguiente diagrama se muestra el algoritmo de selección de casos sin tener en cuenta los cambios manuales del usuario.



En la figura 8.2.1 se muestra un ejemplo usando el personalizador combinado de una persona que tiene un grado de pertenencia al temperamento artesano de 70, al guardián de 40, al idealista de 30 y al racional de 80.

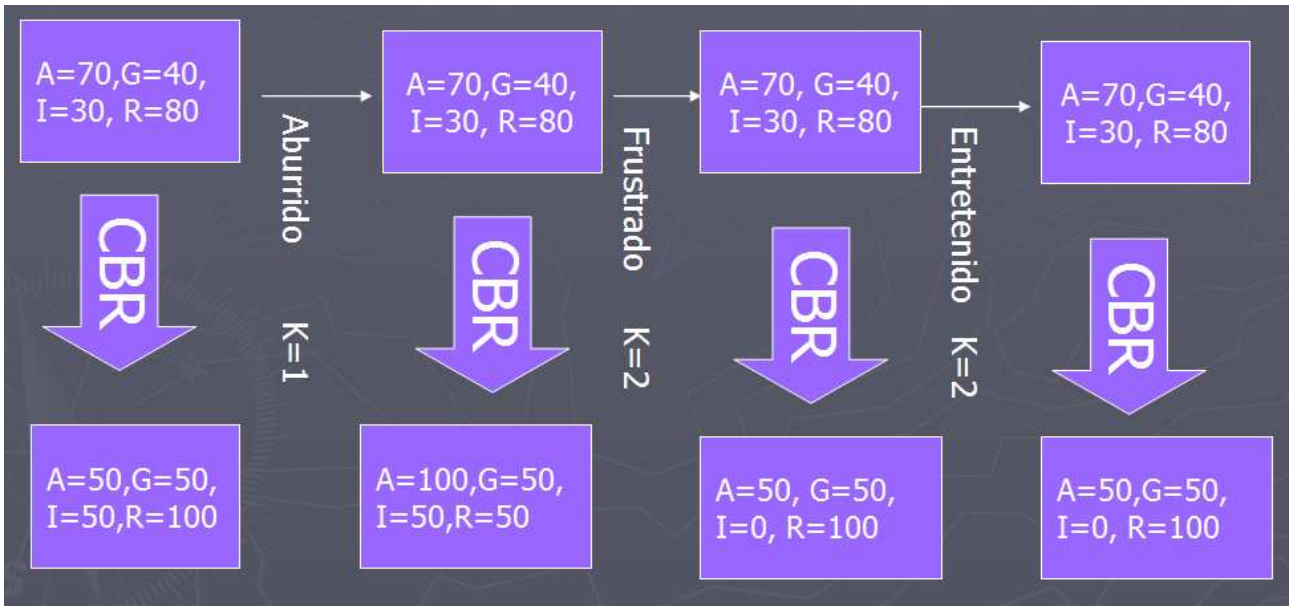


Figura 8.2.1: Ejemplo de personalización combinada.

Durante el ejemplo se sigue la siguiente ejecución:

1. Se busca el perfil más similar (k=0) en la base de conocimiento, y se aplica la adaptación asociada a ese perfil.
2. El Feedback informa de que el usuario está aburrido, por lo que se busca el siguiente perfil más similar (k=1) y se aplica la adaptación asociada a ese perfil.
3. El Feedback informa de que el usuario está frustrado, por lo que se busca el siguiente perfil más similar (k=2) y se aplica la adaptación asociada a ese perfil.
4. El Feedback recoge que el usuario está entretenido, por lo que se sigue con la ultima adaptación utilizada que nos ha dado un buen resultado, y se aprende este nuevo caso, como se explicará en el apartado del módulo CBR.

8.3. Personalizador Keirsey

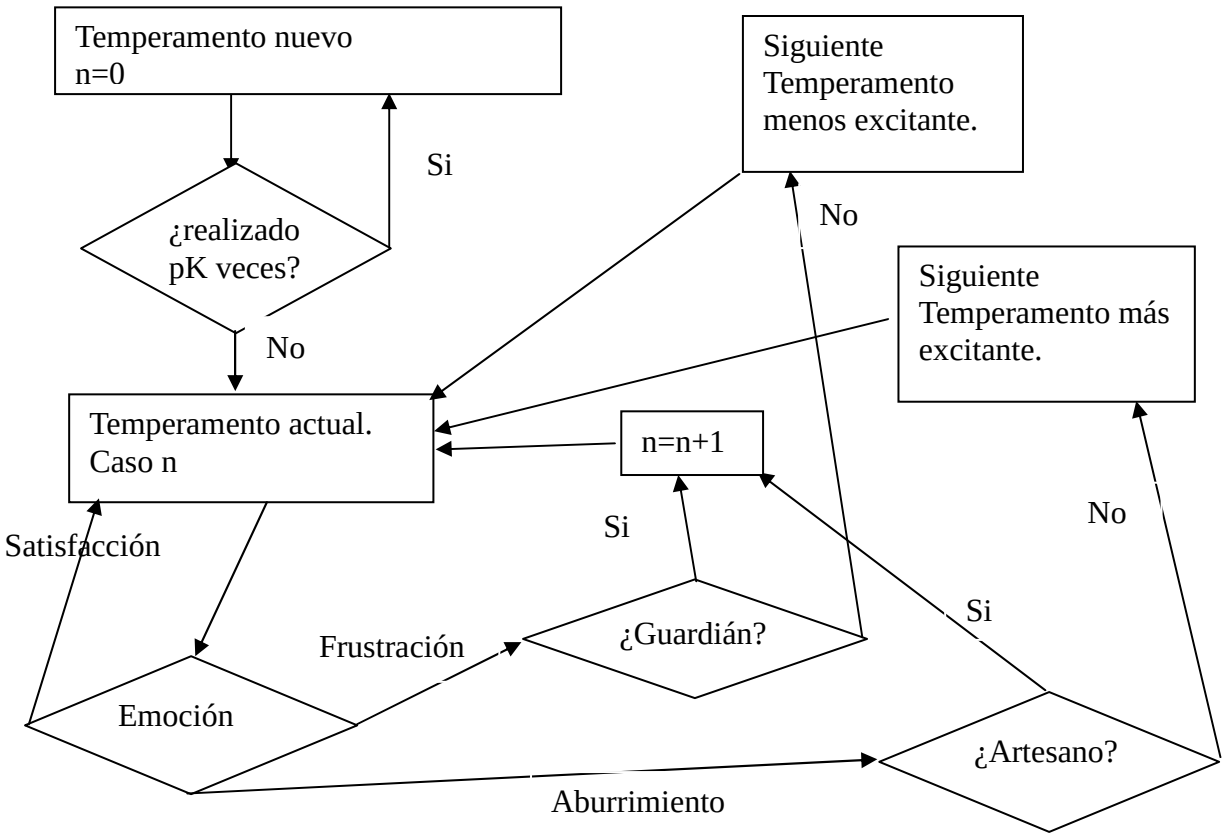
En la tercera versión se tiene un planificador que irá cogiendo de forma aleatoria el temperamento dando la probabilidad de tener un temperamento u otro, al grado de pertenencia a dicho temperamento, para la recuperación de casos también se tendrán en cuanta las otras características del usuario, el feedback se utiliza para obtener las emociones y la experiencia del usuario durante el desarrollo de la aplicación. En esta versión se tienen en cuenta todos los temperamentos para realizar la personalización, el temperamento que se utiliza se elige de manera aleatoria, dándole una probabilidad en función al grado de pertenencia con dicho temperamento de la siguiente manera:

pK= (tK)/(tA+tG+tI+tR)

Donde:

pk: Es la probabilidad que se le da al temperamento.
tK: Es el grado de pertenencia al temperamento del que se quiere la probabilidad.
tA: Es el grado de pertenencia al temperamento Artesano.
tG: Es el grado de pertenencia al temperamento Guardián.
tI: Es el grado de pertenencia al temperamento Idealista.
tR: Es el grado de pertenencia al temperamento Racional.

Para garantizar que se toman todos los temperamentos durante la ejecución en función de su grado de pertenencia a cada temperamento, cada 10 ciclos, cada uno de los temperamentos se realizaran pK*10 veces, y un temperamento en esos 10 ciclos no puede ser elegido más d pK*10 veces, en ese caso se vuelve a recalcular las probabilidades quitando ese temperamento temporalmente, este proceso continúa a no ser que el usuario cambie manualmente su grado de pertenencia a algún temperamento, en tal caso se volvería a repetir el proceso. Si se detectara un estado de ánimo bajo, se coge un temperamento u otro dependiendo si ese estado de ánimo es bajo por aburrimiento o por frustración. Los temperamentos los ordeno en función del nivel de excitación de menor a mayor de la siguiente manera: primero Guardián ya que es una persona que busca la seguridad, segundo Idealista ya que aunque es una persona entusiasta, es diplomática, tercero Racional, ya que aunque es una persona tranquila es estratégica y busca conocer más y conseguir más logros, y en cuarto lugar artesano ya que es una persona que busca el estímulo. Si se detectara que el estado de ánimo es bajo debido a una emoción de aburrimiento, se cogerá como temperamento más característico el siguiente considerado más excitante al actual, a no ser que fuera Artesano, en tal caso se buscaría el siguiente caso más semejante cogiendo como temperamento más característico el Artesano. En caso que la emoción recogida fuera de frustración se coge el siguiente temperamento considerado menos excitante, a no ser que fuese Guardián, en tal caso se cogería el siguiente caso más similar cogiendo como temperamento más característico Guardián. Una vez conseguido que el estado de ánimo sea bueno, se iniciará el proceso de que en cada 10 ciclos se realice cada uno de los temperamentos pK*10 veces. En cada temperamento se cogerá el nivel del temperamento más característico, debido a que se va cambiando de temperamento. Más adelante se verán ejemplos de uso de las tres versiones donde se comprenderán mejor los distintos algoritmos. A continuación se ve un diagrama del funcionamiento del algoritmo sin tener en cuenta los cambios manuales que haga el usuario de su perfil.



Sobre las características del usuario hay dos características que en nuestro caso de estudio van cambiando, como es la experiencia y las emociones del usuario. Las emociones se van recogiendo con el feedback y con ellas se obtiene el estado de ánimo del usuario, si son emociones negativas como la frustración y la impotencia, o neutras como son la pasividad, la inanición y el aburrimiento, si se recogieran algunas de estas emociones, se considera un estado de ánimo bajo. Las emociones del usuario se recogen en cada ciclo del feedback. Mientras que la experiencia del usuario se va evaluando cada 3 ciclos que realiza el feedback, la experiencia va cambiando en función de los objetivos que consiga el usuario, si se considera que el usuario ha cumplido los objetivos se subirá el de nivel de experiencia del usuario, mientras que si se considera que al usuario se le tiene que bajar de nivel de experiencia debido a la frustración, se le bajará, la experiencia se mide en un rango de valores de 0 a 10, siendo 0 el nivel más bajo y 10 el más alto, y se va aumentado y disminuyendo de en uno en uno. También puede ir cambiando el resto de características del usuario, realizando los cambios el usuario manualmente, ya que se proporciona un editor para que el usuario cambie alguna características suyas, como puede ser la edad.

En la figura 8.3.1 se muestra un ejemplo usando el personalizador Keirsey de una persona que tiene un grado de pertenencia al temperamento artesano de 70, al guardián de 40, al idealista de 30 y al racional de 80.

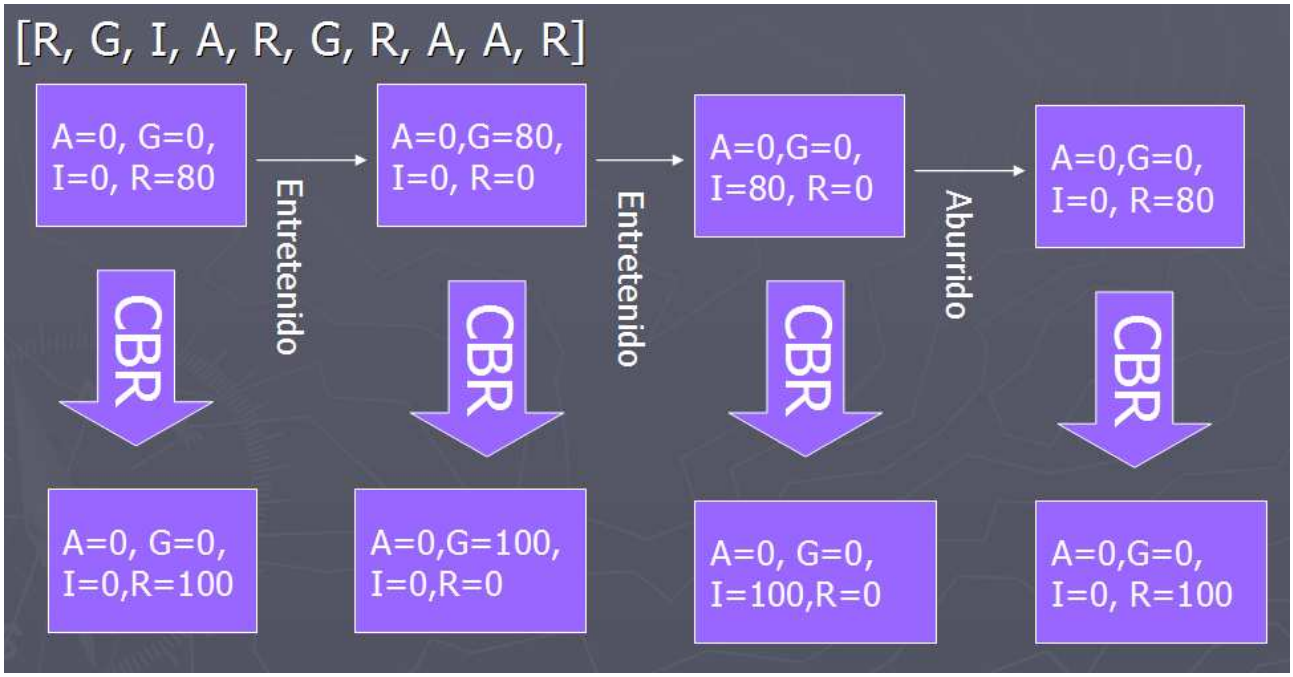


Figura 8.3.1: Ejemplo de personalización Keirsey.

Durante el ejemplo se sigue la siguiente ejecución:

1. Se planifica como se va a personalizar los siguientes 10 ciclos, y se obtiene que aleatoriamente en función al grado de pertenencia a cada temperamento, que el sistema va a ir variando para: racional, guardián, idealista, artesano, racional, guardián, racional, artesano, artesano, racional [R,G,I,A,R,G,R,A,A,R].
2. Se coge de manera absoluta el temperamento artesano, se busca el perfil más similar y se realiza la adaptación para dicho perfil.
3. El Feedback informa de que el usuario está entretenido, por lo que se coge el siguiente temperamento planificado de manera absoluta, es decir se realiza la personalización utilizando el temperamento guardián de manera absoluta, se busca el perfil más similar y se realiza la adaptación del sistema, y se aprende este nuevo caso.
4. El Feedback informa de que el usuario está entretenido, por lo que se coge el siguiente temperamento planificado de manera absoluta, es decir se realiza la personalización utilizando el temperamento idealista de manera absoluta, se busca el perfil más similar y se realiza la adaptación del sistema, y se aprende este nuevo caso.
5. El Feedback informa de que el usuario esta aburrido por lo que se utiliza de manera absoluta el temperamento más excitante al actual, es decir se realiza la personalización absoluta para el temperamento racional, se busca el perfil más similar y se realiza la adaptación.

8.4. Arquitectura

El sistema hace uso de ontologías para representar el conocimiento, una GUI que nos proporciona la interfaz gráfica para realizar el cuestionario, modificar datos, registrarse..., un recuperador de casos CBR, que es el encargado de obtener de las ontologías los casos bases más apropiados para realizar la adaptación, y almacenar nuevos casos que den buenos resultados, un feedback que es el encargado de interactuar con el videojuego para obtener las emociones del usuario.

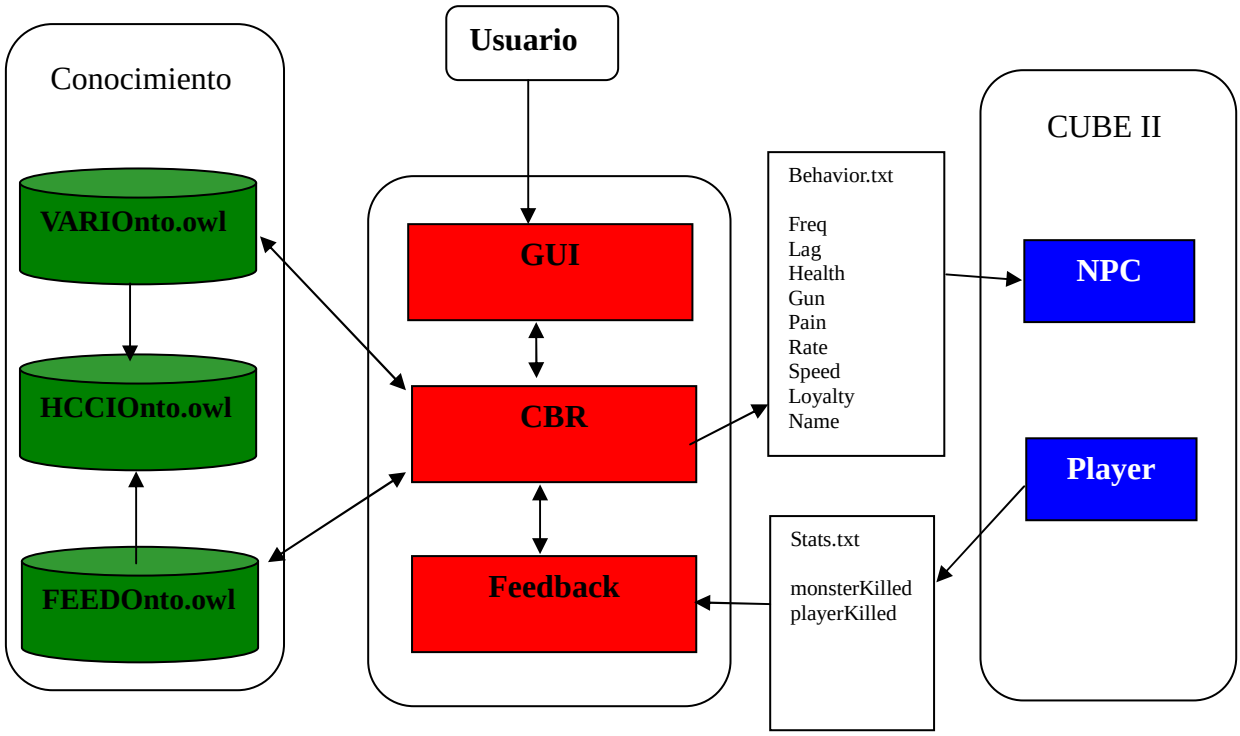


Figura 8.4.1: Arquitectura ASA.

8.4.1. Ontologías

El conocimiento de la aplicación está en tres ontologías, una primera HCCIonto.owl que contiene las características del usuario independientemente de cuál sea el dominio, contiene los HCC issues, otras características más particulares del usuario. La segunda ontología VARIonto.owl que importa HCCIonto.owl y define las variaciones d el comportamiento de los personajes del video-juegos “CUBE II” en función de las características del usuario, la ontología contiene las posibles variaciones del comportamiento de los personajes de los video-juegos y casos base. La tercera ontología FEEDonto.owl importa HCCIonto.owl, se centra en representar la información para el feedback, donde contiene las posibles emociones en función de los datos recogidos por el feedback. También contiene casos base.

En las tres versiones se hace uso de las tres ontologías, y el ciclo CBR es el mismo.

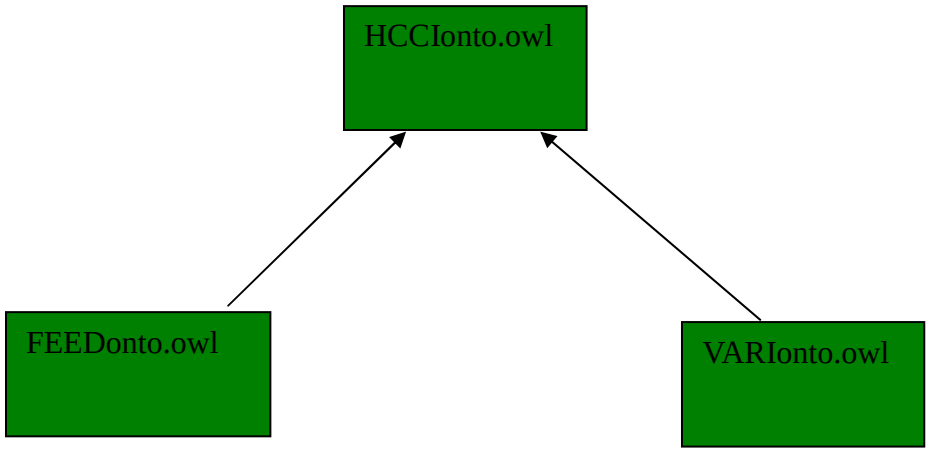


Figura 8.4.1.1: Arquitectura de las ontologías

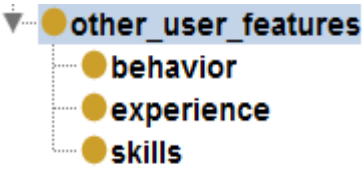
- **HCCIonto.owl**
  - **HCC_issues**

    Contiene los "HCC issues" del usuario, explicados ya en la introducción.

    - Gender: Indica si es masculino o femenino.
    - Age: Edad, y toma un rango de valores de 1 a 100.
    - Moods: Estado de ánimo y puede ser animado o bajo.
    - Temperament: Cada temperamento es un rango de valores de 0 a 100, siendo 0 el menor nivel de pertenencia y 100 el máximo. También se guardan los valores de sus diferentes rasgos(Intelecto, Ser, Buscar): Artesano (Táctico, Emocionado, Estímulo), Guardián (Logístico, Interesado, Seguridad), Idealista (Diplomático, Entusiasta, Identidad) y Racional(Estratégico, Tranquilo, Conocimiento)
    - Emotions: Contiene las emociones, en nuestro caso de aburrimiento o inactividad, frustración o impotencia y satisfacción.

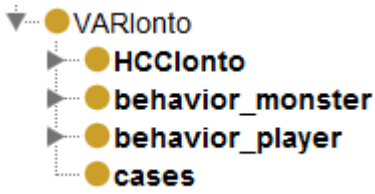
- **other_user_features**

  Contiene otras características mas particulares del usuario.
  experience: Indica el nivel de experiencia del usuario toma un rango de valores de 0 a 10.



- **VARIonto.owl**

  Contiene las variaciones del comportamiento de los personajes de los videojuegos, importa HCCIonto.owl.



  Las variaciones del comportamiento las realiza en función de la agresividad del usuario(treatment), los rasgos del temperamento, las emociones, el estado del jugador y el estado del monstruo.

  - **temperament_trait:** Rasgos, aspectos y valores de los temperamentos. De los cuales se tienen en cuenta el intelecto, el ser y lo que busca.

Comunicación	**Concreta**		**Abstracta**	
Implementación	**Utilitaria**	**Cooperativa**	**Cooperativa**	**Utilitaria**
Carácter	**Artesano**	**Guardián**	**Idealista**	**Racional**
Lenguaje	**Armonioso**	**Asociativo**	**Inductivo**	**Deductivo**
Referencial	**Indicativo**	Imperativo	Interpretativo	Categórico
Sintáctico	**Descriptivo**	Comparativo	Metafórico	Subjuntivo
Retórico	**Heterodoxo**	Ortodoxo	Hiperbólico	Técnico
Intelecto	**Táctico**	**Logístico**	**Diplomático**	**Estratégico**
Papel directivo	**Operador**	Administrador	Mentor	Coordinador
Papel expresivo	**-Promotor**	- Supervisor	- Maestro	- Mariscal
Papel reservado	**-Artífice**	- Inspector	- Consejero	- Organizador
Papel informativo	**Artista**	Conservador	Abogado	Ingeniero
Papel expresivo	**-Ejecutante**	- Sustentador	- Defensor	- Inventor
Papel reservado	**-Compositor**	- Protector	- Sanador	- Arquitecto
Interés				
Educación	**Artesanías**	Comercio	Humanidades	Ciencias
Preocupación	**Técnicas**	Moralidad	Moral	Tecnología
Vocación	**Equipo**	Materiales	Personal	Sistemas
Orientación				
Presente	**Prácticos**	Concienzudos	Altruistas	Pragmáticos
Futuro	**Optimistas**	Pesimistas	Crédulos	Escépticos
Pasado	**Cínicos**	Estoicos	Místicos	Relativistas
Lugar	**Aquí**	Entradas	Caminos	Intersecciones
Tiempo	**Ahora**	Ayer	Mañana	Intervalo
Autoimagen				
Autoestima	**Artísticos**	Confiables	Empáticos	Ingeniosos
Respeto a uno mismo	**Audaces**	Caritativos	Benévolos	Autónomos
Confianza en uno mismo	**Adaptables**	Respetables	Auténticos	Resueltos
Valor				
Ser	**Emocionado**	Interesado	Entusiasta	Tranquilo
Confiar	**Impulso**	Autoridad	Intuición	Razón
Anhelar	**Impacto**	Pertenencia	Romance	Logro
Buscar	**Estímulo**	Seguridad	Identidad	Conocimiento
Valorar	**Generosidad**	Gratitud	Reconocimiento	Deferencia
Aspirar	**Virtuosismo**	Ejecutivo	Sabio	Mago
Papel social				
Pareja	**Compañero de Juegos**	Asistente	Alma Gemela	Mente Afín
Paternidad	**Liberador**	Socializador	Armonizador	Individuador
Liderazgo	**Negociador**	Estabilizador	Catalizador	Visionario

Figura 8.4.1.2: Tabla extraida de COBBER[31]

- **Treatment:** Es el trato agresivo o delicado, digamos que es el nivel de excitación que se quiere dar al usuario, va en función a la edad y sexo del usuario. Para decidir que trato tomar con el usuario se tiene en cuenta un 60% la edad y un 40% el sexo. Su valor va de 0 a 100, de menor a mayor nivel de agresividad o excitación, siendo 0 el nivel más bajo y 100 el más alto. Como se desea que el nivel de excitación sea mas grande cuando la edad ronda los 20 años, la fórmula para conseguir el nivel de excitación es la siguiente.

$$\text{Treatment} = \begin{cases} \text{round}(100*((\log(age)/\log(20))*0.7+gender*0.3)) & \text{si } age<20 \\ 100 & \text{si } age==20 \\ \text{round}(100*(1-((\log(age-19)/\log(81))*0.7+gender*0.3))) & \text{si } age>20 \end{cases}$$

Donde:

gender: 1 si es hombre, 0 mujer
age: Entero que va de 1 a 100.

Los logaritmos los utilizo para crear una mayor nivel de agresividad o excitación a las edades que rondan los 20 años

- **behavior_player:**

Contiene los comportamientos del personaje de video-juego que maneja el usuario a tener en cuenta para la modificación del comportamiento de los personajes del video-juego(monster) manejados por la inteligencia artificial del juego.

  - ability_damage: Es la capacidad de hacer daño del jugador, va en función de las armas que tenga y la munición de cada arma.
  - gun_player: Arma actual que esta utilizando el jugador.
  - health_player: Salud del jugador.
  - state_player: El estado del jugador, vivo, muerto o atacando.

- **behavior_monster:**

  Contiene los comportamientos de los monstruos en el juego.

  - **freq:** Frecuencia relativa de generación de monstruos.

    Tiene un valor mínimo 1 y máximo de 10, y se tiene en cuenta las emociones(emotions), la experiencia(experience) y el estímulo que busca el usuario.

$$freq = \begin{cases} x+1 & \text{aburrido y } freq<10 \\ x & \text{satisfecho} \\ x-1 & \text{frustrado y } freq>0 \end{cases}$$

$$x = FE+(0.001*t+0.005*est+0.05*exp-0.003*tr-0.005*seg-0.004*dip)*FE$$

    FE: Freq estándar del monstruo.
    exp: Es el nivel de experiencia del usuario.
    est: El nivel de estímulo que busca el usuario.

  - **gun:** El arma actual del monstruo. Para obtener el nueva arma(selGunMonster) se tiene en cuenta el arma actual del monstruo(gunMonster), las emociones(emotions) y la experiencia(experience) del usuario. Durante la partida si el usuario está aburrido y la experiencia es mayor que 5, entonces se elegirá el arma siguiente con más capacidad de hacer daño, por el contrario si el usuario está frustrado se cogerá el arma siguiente con menos capacidad de hacer

daño.

Gun	Damage
GUN_SG	10
GUN_FIREBALL	20
GUN_CG	30
GUN_SLIMEBALL	30
GUN_ICEBALL	40
GUN_BITE	50
GUN_GL	75
GUN_RIFLE	100
GUN_RL	120

- **health:** La salud del monstruo. Como con el temperamento artesano y los que tienen mayor nivel de agresividad de usuario, en el resto de atributos toma unos valores que hace que sea más difícil superar los objetivos, este atributo es menor en cuanto se busca estimulo, porque lo que busca es matar cuantos más monstruos mejor. Con las características de usuario que hace más fácil el juego aumenta la salud, y el que más influye es la experiencia. Su rango de valores es de 10..1000. Para su modificación se tiene en cuenta estímulo, seguridad, entusiasta, diplomático, tranquilo, emociones, experiencia, agresividad del usuario y el health estándar del monstruo. Su valor se obtiene de la siguiente manera:

health= { x satisfecho ; x-50 frustrado

x=HE+(0.1*exp-0.003*est-0.002*t)*HE

HE: health estándar.
est: Nivel de estímulo que busca.
exp: Experiencia.
estr: Nivel de estratégico que tiene.
ent: Nivel de entusiasta que tiene.
t: Nivel de agresividad o excitación del usuario.

- **lag:** Representa la agresividad cuando menor es mayor es la agresividad del monstruo, si es 0 van siempre hacia a ti, cuando mayor es, mas tiempo se quedan esperando, toma un rango de valores de 0 a 200. Para

su modificación se tiene en cuenta estímulo, seguridad, entusiasta, diplomático, tranquilo, emociones, experiencia, agresividad del usuario y el lag estándar del monstruo. Su valor se obtiene de la siguiente manera:

lag = { x-50 aburrido; x satisfecho; x+50 frustrado

x=LE+(0.005*seg+0.005*dip+0.005*tr-0.0025*ent-0.005*est-0.005*t-0.05*exp)*LE

LE: Lag estándar.
est: Nivel de estímulo que busca.
exp: Experiencia.
seg: Nivel de seguridad que busca.
dip: Nivel del intelecto diplomático que tiene.
tr: Nivel de tranquilidad que tiene.
ent: Nivel de entusiasta que tiene.
t: Nivel de agresividad o excitación del usuario del usuario.

- **loyalty:** Indica la frecuencia con que tiene que ser golpeado otro monstruo antes de entrar a luchar, toma un rango de valores de 1 a 10. Para su modificación se tiene en cuenta estímulo, seguridad, entusiasta, diplomático, tranquilo, emociones, experiencia, agresividad del usuario y el loyalty estándar del juego. Su valor se obtiene de la siguiente manera:

loyalty = { x-1 aburrido; x satisfecho; x+1 frustrado

x=LE+(0.003*seg+0.003*dip+0.005*estr-0.001*ent-0.005*est)*LE

LE: Loyalty estándar.
est: Nivel de estímulo que busca.
seg: Nivel de seguridad que busca.
dip: Nivel del intelecto diplomático que tiene.

estr: Nivel de estratégico que tiene.
ent: Nivel de entusiasta que tiene.

- **name:** Es el nombre del monstruo, puede ser:

- **pain:** Indica la intención de hacer daño del monstruo, es muy parecido a lag, cuando menor es, el monstruo se comporta de forma mas agresiva. Toma un rango de valores de 100 a 1200. Para su modificación se tiene en cuenta estímulo, seguridad, entusiasta, diplomático, tranquilo, emociones, experiencia, agresividad del usuario y el pain estándar del monstruo. Su valor se obtiene de la siguiente manera:

$$\text{pain} = \begin{cases} x-100 & \text{aburrido} \\ x & \text{satisfecho} \\ x+100 & \text{frustrado} \end{cases}$$

x=(PE+0.005*seg+0.005*dip+0.002*tr-0.001*ent-0.008*est-0.002*t-0.05*exp)*PE

PE: Pain estándar.
est: Nivel de estímulo que busca.
exp: Experiencia.
seg: Nivel de seguridad que busca.
dip: Nivel del intelecto diplomático que tiene.
tr: Nivel de tranquilidad que tiene.
ent: Nivel de entusiasta que tiene.
t: Nivel de agresividad o excitación del usuario.

- **rate:** Indica la frecuencia con la que es disparado el jugador cuando esta en el punto de mira, cuanto menor, hay mayor cantidad de

disparos. Depende de la habilidad. Toma un rango de valores de 1 a 400. Para su modificación se tiene en cuenta estímulo, seguridad, estratégico, diplomático, tranquilo, emociones, experiencia, agresividad del usuario y el rate estándar del monstruo. Su valor se obtiene de la siguiente manera:

rate = { x-30 aburrido; x satisfecho; x+30 frustrado

x=RE+(0.005*seg+0.005dip-0.007*est-0.003*t-0.003*estr-0.05*exp)*RE

RE: Rate estándar.
est: Nivel de estímulo que busca.
estr: Nivel de estratégico que es el usuario.
exp: Experiencia.
seg: Nivel de seguridad que busca.
dip: Nivel del intelecto diplomático que tiene.
t: Nivel de agresividad o excitación del usuario.

- **speed:** Es la velocidad del monstruo. Toma un rango de valores de 5 a 50. Para su modificación se tiene en cuenta la experiencia, la agresividad del usuario y el speed estándar del monstruo. Su valor se obtiene de la siguiente manera:

speed = { x+5 aburrido; x satisfecho; x-5 frustrado

x=0.5*SE+(0.0075*t+0.075*exp)*SE

SE: Speed estándar.
t: Nivel de agresividad o excitación del usuario del usuario.

- **state:** Indica el estado del monstruo, puede ser:

Este apartado se expone solo de manera teórica, debido a las dificultades encontradas de modificarlas en el videojuego.

Para su modificación se tiene en cuenta intellect, be, confide, search, moods, experience del usuario, gunPlayer, healthPlayer del jugador, gunMonster, healthMonster y stateMonster(estado actual del monstruo) del monstruo. Su nuevo estado(newStateMonster) se obtiene de la siguiente manera:

stateMonster:
HOME:
Si moods adecuado:

Artesano --> if healthPlayer<=healthMonster
newStateMonster=SEARCH(busca=estimulo)
else
newStateMOnster=HOME(intelecto=táctico)

Guardián --> newStateMonster=HOME
(busca==seguridad)
Idealista --> newStateMonster=HOME
(intelecto==diplomático)
Racional --> if (healthPlayer<healthMonster)&&
(gunPlayer.damage<gunMonster.damage)
newStateMonster=SEARCH(estratégico)
else
newStateMOnster=HOME(tranquilo)

si moods adecuado y experience==proffessional o
expert
si (healthPlayer<healthMonster)&&
(gunPlayer.damage<gunMonster.damage)
newStateMonster=SEARCH
sino
newStateMOnster=HOME

Si está fustrado: newStateMonster=HOME

Si está aburrido: newStateMonster=SEARCH

SEARCH:

Si moods adecuado: newStateMonster=SEARCH

Si está fustrado: newStateMonster=HOME

Si está aburrido: newStateMonster=SEARCH

ATTACKING:

newStateMonster=ATTACKING

PAIN:

Si moods adecuado:

Artesano --> newStateMonster=ATTACKING (anhela=impulso)

Guardián --> si healthPlayer<=healthMonster newStateMonster=SEARCH(es interesado no le interesa morir)

sino newStateMOnster=ATTACKING (porque sino se puede llegar a aburrir)

Idealista --> newStateMonster=ATTACKING (aunque intelecto==diplomático si ataca es que quiere lucha)

Racional --> newStateMonster=ATTACKING

Si está fustrado: newStateMonster=SEARCH

Si está aburrido: newStateMonster=ATTACKING

SLEEP:

Si moods adecuado:

Artesano --> si healthPlayer<=healthMonster newStateMonster=SEARCH (busca=estimulo)

sino newStateMOnster=SLEEP (intelecto=táctico)

Guardián --> newStateMonster=SLEEP (busca==seguridad)

Idealista --> newStateMonster=SLEEP
(intelecto==diplomático)
Racional --> si (healthPlayer<healthMonster)&&
(gunPlayer.damage<gunMonster.damage)
newStateMonster=SEARCH
(intecto==estratégico)
sino
newStateMonster=SLEEP (es
tranquilo)

si moods adecuado y experience==proffessional o
expert
si (healthPlayer<healthMonster)&&
(gunPlayer.damage<gunMonster.damage)
newStateMonster=SEARCH
sino
newStateMOnster=SLEEP

Si está fustrado: newStateMonster=SLEEP

Si está aburrido: newStateMonster=SEARCH

AIMING(APUNTADO):
Si moods adecuado:

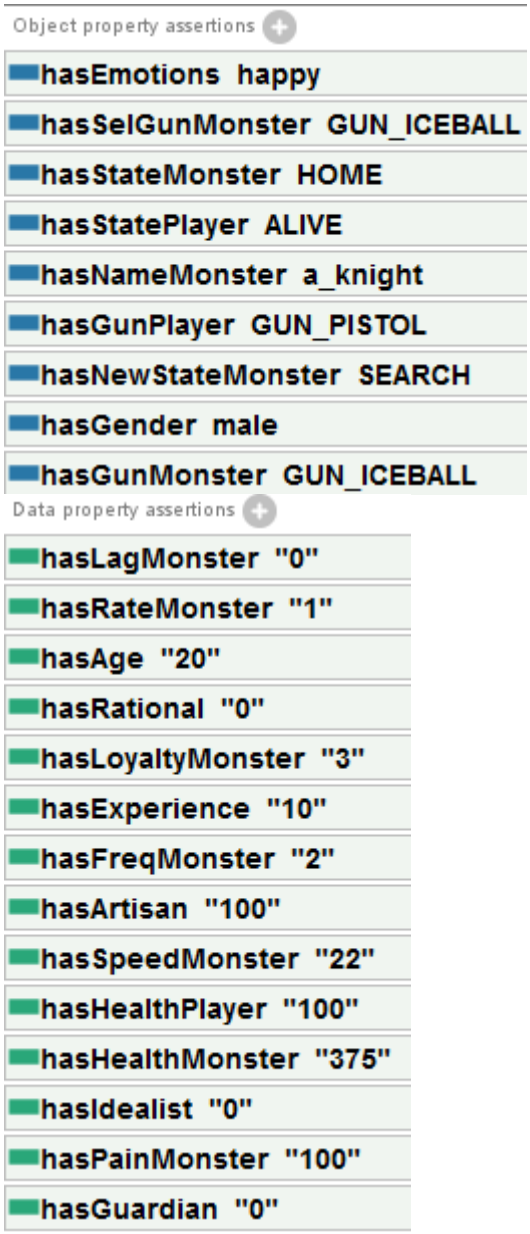
Artesano --> newStateMonster=ATTACKING
(anhela=impulso)
Guardián --> newStateMonster=AIMING (es
interesado no le interesa morir)
Idealista --> newStateMonster=AIMING
(intelecto==diplomático)
Racional --> newStateMonster=ATTACKING

Si está fustrado: newStateMonster=AIMING

Si está aburrido: newStateMonster=ATTACKING

- **cases:**

  Contiene los casos base. Cada caso base contiene como parámetros de entrada Treatment, experience, moods, intellect, be, search, gunPlayer, healthPlayer, gunMonster, healthMonster, stateMonster y como parámetros de salida newStateMonster, selGunMonster, freqMonster, lagMonster, loyaltyMonster, painMonster, rateMonster, speedMonster.



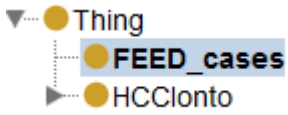
Se han dado unos pesos de importancia a las características del usuario en función del atributo del NPC, estos pesos se pueden ir cambiando en un futuro.

Tabla de valores standard del juego:

name	speed	health	freq	lag	rate	pain	loyalty
an ogro	15	100	3	0	100	800	1
a rhino	18	70	2	70	10	400	2
ratamahatta	13	120	1	100	300	400	4
a slith	14	200	1	80	400	300	4
bauul	12	500	1	0	200	200	6
a hellpig	24	50	3	0	100	400	1
a knight	11	250	1	0	10	400	6
a goblin	15	100	1	0	200	400	2
a spider	22	50	1	0	200	400	1

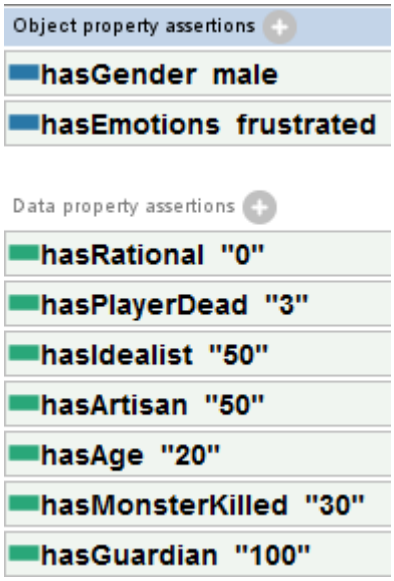
- **FEEDOnto.owl**

Contiene el mapeo entre las emociones y las estadísticas del usuario, importa la ontología HCCIOnto.owl



  - **FEED_cases**

    Contiene los casos base del feedback, en los que se tiene en cuenta el temperamento del usuario(searchStimulus, searchStimulus, beCalm, searchSecurity, intellectDiplomacy), el nivel de agresividad o excitación del jugador(treatment) y sus estadísticas(monsterKilled, playerDead).



8.4.2. CBR

Esta parte se encarga de la recopilación de casos base por similitud de los valores enteros y por exactitud del resto de atributos, aprende aquellos casos que nos dan buenos resultados, en la figura 8.4.2.1 se puede ver el funcionamiento de la metodología CBR.

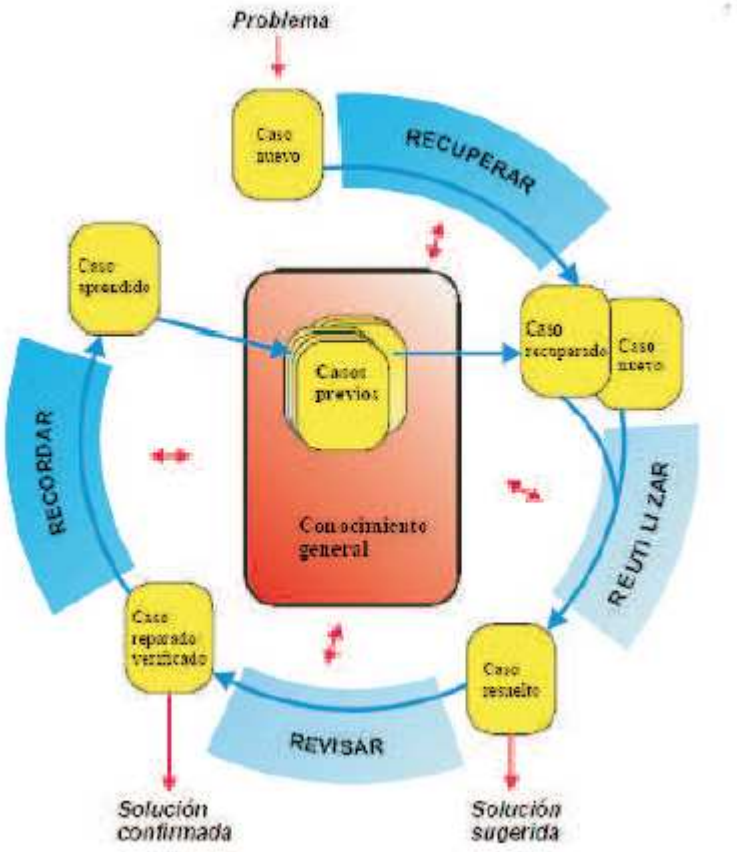


Figura 8.4.2.1: Funcionamiento CBR extraída de COBBER[31]

- **CBRVARI:** Se encarga de conseguir la personalización del comportamiento de los personajes de video-juegos, a partir las características del usuario y con los datos del juego, como son el estado, arma y salud del jugador y de los monstruos nos devuelve la variación de los parámetros del juego. Para la recuperación de casos se le da más peso a las emociones y experiencia del usuario y el nombre del monstruo, luego al temperamento, edad y sexo del usuario y después al resto de atributos de los NPCs. Busca por similitud todos los atributos enteros y el resto de atributos por exactitud. En la figura 8.4.2.2 se puede ver de los parámetros que componen un caso. La descripción del casos contiene las características del usuario, el estado actual del jugador(salud, arma y estado), y el estado actual del monstruo(salud, arma y estado). La solución del caso contiene la variación del monstruo(gun, name, lag, rate, speed, loyalty, pain, health, state, freq).

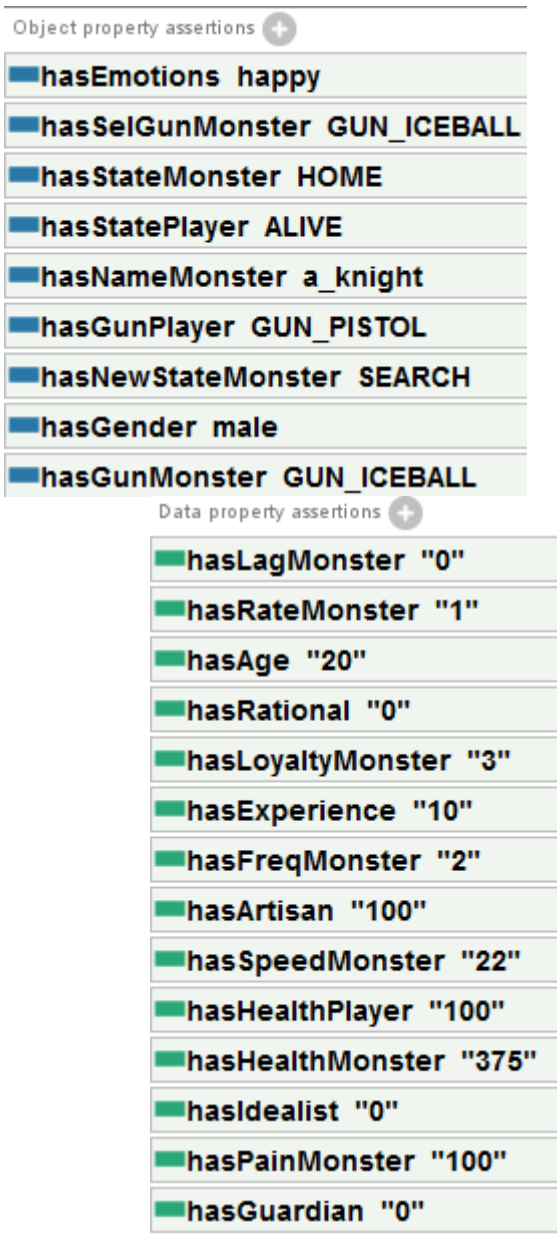


Figura 8.4.2.2: Caso Base para CBRVARI

Del caso recuperado hay que verificar que es del monstruo deseado, sino es así se coge el siguiente caso más similar hasta encontrar el caso deseado. Al final del proceso se comprueba si el caso reconocido nos ha dado un buen resultado, es decir ya sea por el feedback automático y porque el usuario lo indique manualmente, se recibe que usuario esta teniendo emociones positivas, en tal caso se aprende un nuevo caso para las características actuales del usuario.

- **CBRFEED:** Se encarga de obtener a partir de las estadísticas del jugador sus emociones, para luego utilizarlas en CBRVAI. Busca todos los atributos por similitud, dándole más pesos a las estadísticas(monstruos matados, número de veces que muere el jugador). En la figura 8.4.2.3 se puede ver los parámetros que componen un caso para CBRFEED. La descripción del caso contiene algunas características del usuario y las estadísticas del juego. La solución del caso contiene la emoción.

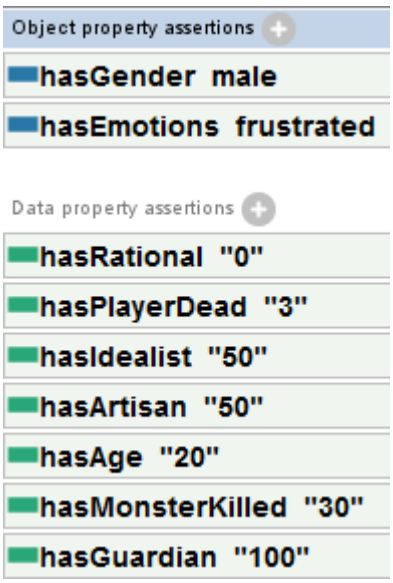


Figura 8.4.2.3: Caso Base para CBRFEED

Se aprende un caso nuevo, cuando el usuario indica manualmente su emoción, guardando la emoción indicada por el usuario para las características y estadísticas actuales del jugador.

En el anexo se puede ver el mapeo entre las ontologías y el CBR.

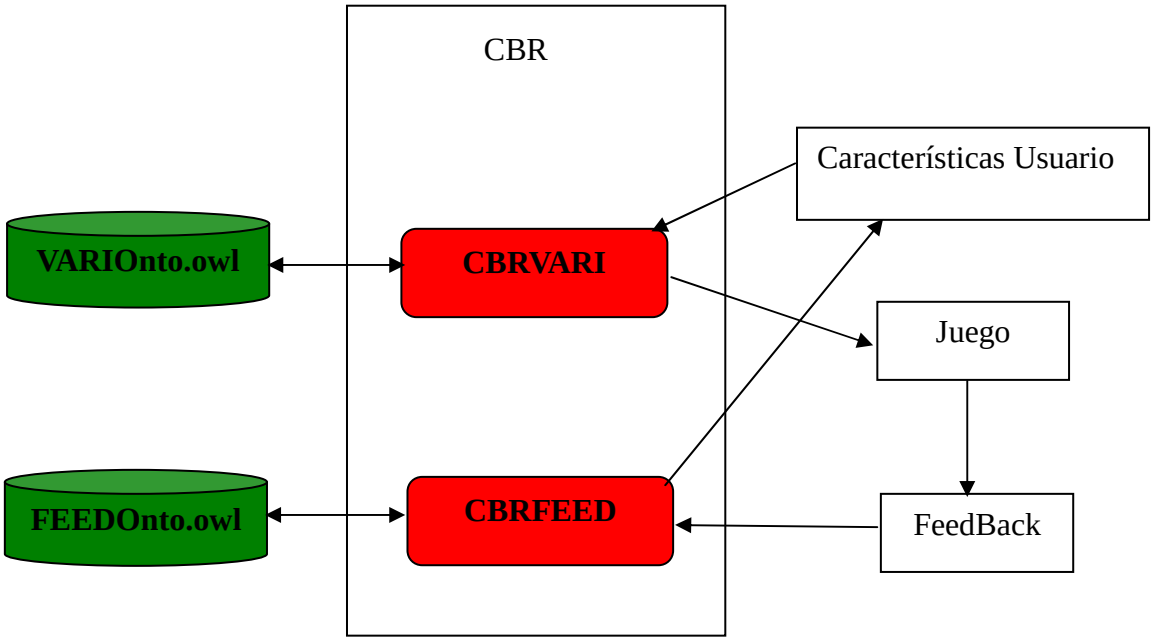


Figura 8.4.2.4: Funcionamiento CBR

- **Ejemplo de ejecución de CBR:**

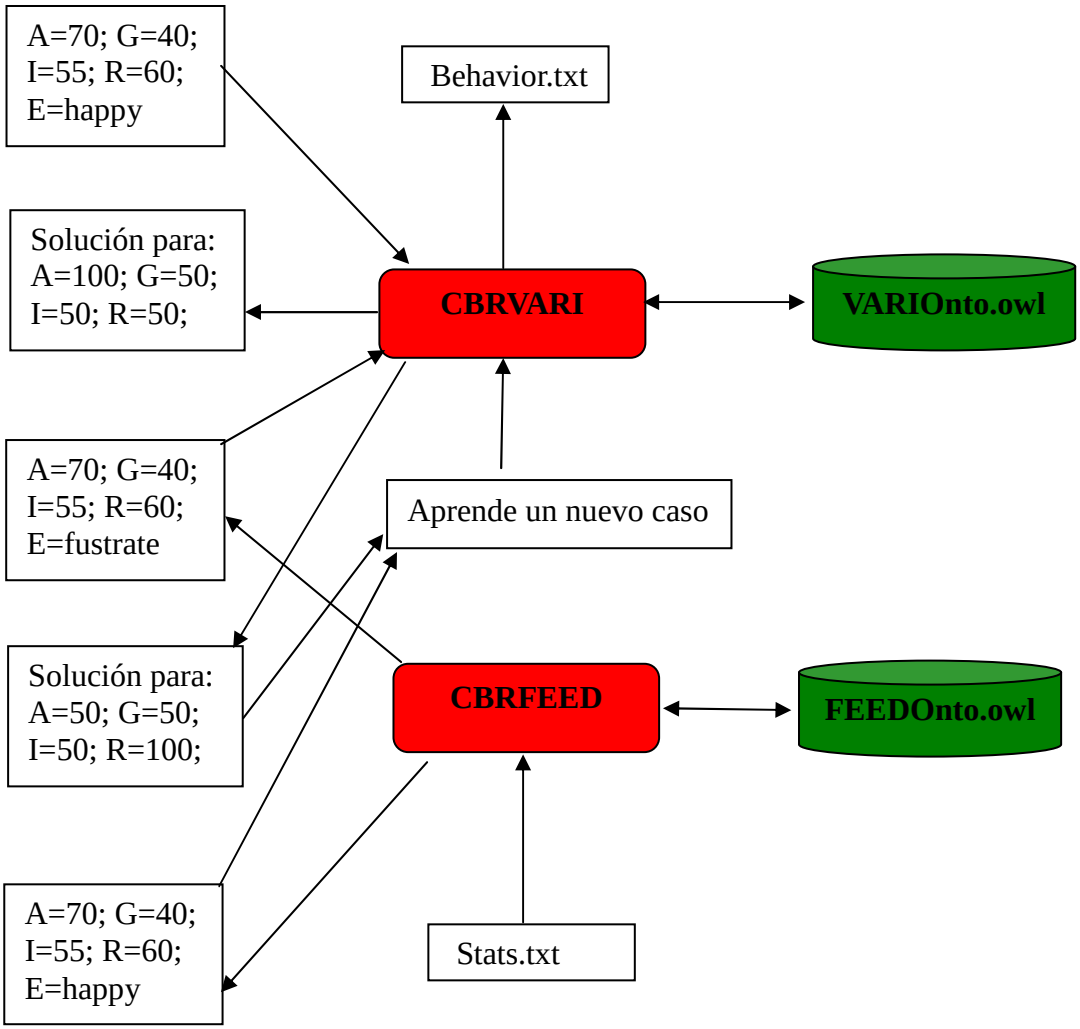


Figura 8.4.2.5: Ejemplo CBR

Al principio tenemos un usuario Artesano 70, Guardián 50, Idealista 50 y Racional 60, el personalizador realizado es el combinado. CBRVARI varía el fichero de comportamiento de los NPCs, y CBRFEED detecta frustración por parte del usuario, por lo que se busca otro caso, el cuál si nos da una solución satisfactoria, por lo que se almacena un nuevo caso en VARIonto.owl, con la descripción del usuario, es decir Artesano 70, Guardián 50, Idealista 50 y Racional 60 y con la solución para un Artesano 50, Guardián 50, Idealista 50, Racional 100.

8.4.3. GUI

El sistema nos proporciona una interfaz que interactúa con el usuario, el CBR y el juego, la GUI consta de cinco subpaneles:

- **Login:** Proporciona la funcionalidad de identificar un usuario, y de elegir la versión de personalizador que quiere realizar el usuario.

- **Registro:** Proporciona un cuestionario, donde pide un identificador, la edad, sexo y un pequeño cuestionario Keirsey para la obtención de temperamentos, aunque estos también se podrán meter manualmente.

- **Comportamiento NPC:** Da la posibilidad al usuario de que modifique los atributos de comportamiento de los personajes del videojuego de forma manual.

- **Usuario:** En este apartado nos proporciona la posibilidad de modificar las características del usuario identificado.

- **FeedBack:** Da la posibilidad de habilitar o deshabilitar el feedback automático del juego, además de proporcionar un feedback manual, donde el usuario podrá mostrar su emoción actual de manera manual.

Esta sección se ve de manera gráfica en la sección 7.

8.4.4. Feedback

Es el encargado de recoger las emociones que experimenta el usuario con el juego de manera automática o manual. Es utilizado por CBRFEED para a partir de las estadísticas del usuario recogidas por el feedback obtener la emoción que está experimentando el usuario, aunque esa emoción también se puede recoger de manera manual, indicándola manualmente el usuario.

Para obtener las emociones se hace de la siguiente manera:

emoción =
- aburrido — no frustrado y mk<=(13+0.05*t+0.1*est-0.03*tr-0.05*seg-0.04*dip)
- feliz — no aburrido y no frustrado
- frustrado — pd>=(3-0.01*tr-0.01*seg-0.01*dip)

Donde:

mk(monsterKilled): Número de monstruos matados en el ciclo.
pd(playerDead): Veces que muere el jugador en el ciclo.
t(treatment): Nivel de agresividad o excitación del jugador.
est(searchStimulus): Nivel de estímulo que busca el jugador.
tr(beCalm): Nivel de tranquilo que es el jugador.
seg(searchSecurity): Nivel de seguridad que busca el jugador.
dip(intellectDiplomacy): Nivel de intelecto diplomático que tiene el jugador.

El atributo experiencia(Experience) del conjunto de características, aunque no se trata en la ontología, viene relacionado de la siguiente manera, teniendo en cuenta que la experiencia varía de -5 a 10.

experiencia =
- experiencia+1 — si durante 3 ciclos seguidos emoción==feliz
- experiencia-1 — si emoción==frustrado
- experience — en otro caso

Las características del usuario que se cogen son las iniciales, excepto la experiencia que es la experiencia que tenga el usuario en ese momento.

8.4.5. Juego

Del juego tengo el código fuente el cual lo he modificado para que las características de cada tipo de monstruo las lea del fichero "behavior.txt" obtenido de la etapa CBRVARI. Las estadísticas del jugador se actualizan en el fichero "stats.txt", las cuales las recopila el feedback para proporcionárselas a CBRFEED y así poder obtener la emoción que está experimentado el usuario.

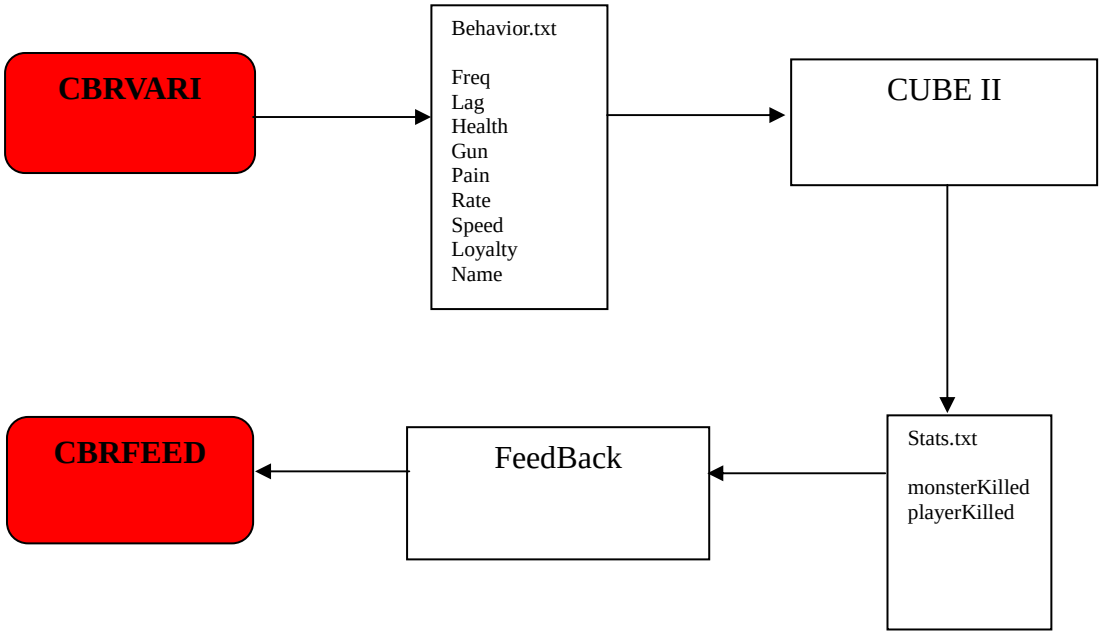


Figura 8.4.5.1

Figura 8.4.5.2: Cube II

8.5. Casos de uso

CASO DE USO Nº 1

Nombre: Registro o modificación de usuario.
Objetivos: Guardar en el registro un nuevo usuario.
Entrada: Usuario de la aplicación
Precondiciones: El usuario no existe en el registro o desea modificar alguna de sus datos.
Salida: Registro del usuario con sus HCC issues.
Postcondiciones si éxito: Registro del usuario con sus HCC issues.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Usuario que se conecta.
Secuencia de ejecución normal:

1. Crear un formulario que contenga preguntas para obtener el temperamento y los HCC issues del usuario.
2. Guardar en el registro al nuevo usuario.
3. Realizar el ciclo CBR.

Secuencia alternativa:
• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.
• El usuario también puede cerrar la aplicación.

CASO DE USO Nº 2

Nombre: Inicio sesión de usuario ya registrado.
Objetivos: Obtener la última variación registrada para el usuario.
Entrada: Usuario de la aplicación
Precondiciones: El usuario existe en el registro.
Salida: Última variación asociada al usuario.
Postcondiciones si éxito: Última variación asociada al usuario.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Usuario que se conecta.
Secuencia de ejecución normal:

1. Crear un GUI para que el usuario se identifique.
2. Obtener los HCC issues y la última variación positiva asociada al usuario en el registro.

Secuencia alternativa:
• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

CASO DE USO Nº 3

Nombre: Ciclo CBR para el comportamiento de los NPCs.
Objetivos: Obtener variación a aplicar en el comportamiento de los NPCs a partir de las características de usuario independientemente de cuál sea el dominio.
Entrada: Las características actuales del usuario.
Precondiciones: Se tiene la variación independiente del dominio.
Salida: La variación del comportamiento de los NPCs.
Postcondiciones si éxito: La variación del comportamiento de los NPCs.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Sistema.

Secuencia de ejecución normal:

1. Conecta con la ontología VARIonto.owl.
2. Se obtienen todos los casos base de la ontología.
3. Se busca caso base mas similar.
4. Se comprueba que el caso base es para el monstruo deseado.
5. Si se produce una emoción positiva se almacena el caso, teniendo como descripción las características actuales del usuario y como solución, el comportamiento actual del monstruo.

Secuencia alternativa:

• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

CASO DE USO Nº 4

Nombre: Ciclo CBR para tratar las estadísticas recogidas por el feedback.
Objetivos: Obtener la emoción del usuario en función de las estadísticas recogidas por el usuario y las propias características del usuario.
Precondiciones: Se tienen las estadísticas del juego.
Salida: Se obtiene la emoción del usuario en ese momento.
Postcondiciones si éxito: Se obtiene la emoción del usuario en ese momento.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Sistema.
Secuencia de ejecución normal:

1. Conecta con la ontología FEEDonto.owl.
2. Se obtienen todos los casos base de la ontología.
3. Se busca caso base mas similar.
4. Si el usuario indica manualmente la emoción que está experimentando, se almacena un nuevo caso en FEEDonto.owl, con las estadísticas del usuario actuales y sus características.

Secuencia alternativa:

• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

CASO DE USO Nº 5

Nombre: Adaptación automática.
Objetivos: Variar el comportamiento de los NPCs en función de los “HCC issues” del usuario.
Entrada: Los “HCC issues” del usuario.
Precondiciones: Se tiene la información relativa a los “HCC issues” del usuario.
Salida: La variación del comportamiento de los NPCs.
Postcondiciones si éxito: La variación del comportamiento de los NPCs.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Sistema.
Secuencia de ejecución normal:

1. Se realiza el ciclo CBR independiente del dominio, a partir de los HCC issues del usuario obtenidos a partir del cuestionario y del feedback.

2. Con el caso base obtenido en el paso 1, se realiza el ciclo CBR para la variación del comportamiento de los NPCs.
3. Se aplica la variación del comportamiento de los NPCs obtenida del paso 2.

Secuencia alternativa:
• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

CASO DE USO Nº 6

Nombre: Adaptación manual del juego.
Objetivos: Variar manualmente el comportamiento de los NPCs por si el usuario no está de acuerdo con la adaptación automática.
Entrada: Parámetros modificables del juego.
Precondiciones: Parámetros modificables del juego.
Salida: La variación del comportamiento de los NPCs.
Postcondiciones si éxito: La variación del comportamiento de los NPCs.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Sistema.
Secuencia de ejecución normal:

1. Se ofrece una interfaz gráfica con los atributos modificables de los NPCs como puede ser la velocidad, la frecuencia y velocidad de disparo en caso de un shooter.
2. Aplicar las variaciones confirmadas por el usuario.

Secuencia alternativa:
• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

CASO DE USO Nº 7

Nombre: Cuestionario inicial.
Objetivos: Obtener los “HCC issues” del usuario a partir de un cuestionario.
Entrada: Cuestionario.
Precondiciones:
Salida: Los “HCC issues” del usuario.
Postcondiciones si éxito: Se guardara los “HCC issues” en el registro.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Sistema.
Secuencia de ejecución normal:

1. Se ofrece una interfaz gráfica con el cuestionario en el que se preguntará a los usuario por los “HCC issues” como son la edad, sexo, temperamento… Para el temperamento el usuario tendrá que realizar un test de Keirsey. El formato del test de Keirsey será del modo pregunta y se le ofrecerán varias respuestas, indicando el usuario el grado de pertenencia a cada una de ellas, del test de Keirsey se obtendrá el grado de pertenencia a cada uno de los 4 temperamentos principales (Artesano, Guardián, Racional e Idealista).
2. Guardar en el registro los “HCC issues” del usuario.

Secuencia alternativa:
• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

CASO DE USO Nº 8

Nombre: Feedback.
Objetivos: Obtener las emociones del usuario en tiempo real.
Precondiciones: El usuario ha empezado a jugar.
Salida: La variación del comportamiento de los NPCs.
Postcondiciones si éxito: Variación del comportamiento de los NPCs.
Postcondiciones si fracaso: Mensaje informativo del problema que ha ocurrido.
Actor: Sistema.
Secuencia de ejecución normal:

1. Se recogen las emociones del usuario ya sea por el tiempo de respuesta o por su respuesta a las preguntas que les haga el sistema, dichas preguntas se realizaran cuando se vea inanición por parte del jugador y también de manera periódica.
2. Se realiza del ciclo CBR para obtener la nueva variación del comportamiento de los NPCs con los nuevos datos recogidos mas los datos que ya tenía el sistema del usuario.
3. Se realiza la variación en el comportamiento de los NPCs.

Secuencia alternativa:
• Si se produce algún error en el proceso se sacará un mensaje de error explicativo de lo que ha ocurrido.

8.6. Ejemplo Personalizador Absoluto

Al principio se tiene un temperamento{A=80;G=30;I=30;R=40}

CICLO		1	2	3	4	5	6	7	8	9	10
USUARIO	**Edad**	20	20	20	20	20	20	20	20	20	20
	Sexo	M	M	M	M	M	M	M	M	M	M
	Emoción	F	I	I	A	F	F	F	I	F	F
	Experiencia	0	0	0	0	0	0	1	0	0	0
	Artesano	100	0	0	0	100	100	100	100	100	100
	Guardián	0	0	0	100	0	0	0	0	0	0
	Idealista	0	0	100	0	0	0	0	0	0	0
	Racional	0	100	0	0	0	0	0	0	0	0
NPC	**Name**	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro
	Gun	SG	SG	SG	SG	SG	SG	SG	SG	SG	SG
	Freq	4	2	2	4	4	4	5	3	4	4
	Lag	0	87	87	0	0	0	0	50	0	0
	Loyalty	1	2	2	1	1	1	1	2	1	1
	Pain	200	575	575	400	200	200	190	300	200	200
	Rate	1	105	180	145	1	1	1	31	1	1
	Speed	30	25	25	35	30	30	32	25	30	30
	Health	50	35	30	135	50	50	60	10	50	50
STATS	**M. Killed**	10	9	12	14	16	15	10	16	16	
	P. Dead	3	2	1	1	1	0	2	0	0	

M: Masculino.
F: Femenino.
F: Feliz.
I: Impotencia.
A: Aburrido.
M. Killed: Monstruos matados
P. Dead: Veces que muere el jugador en el ciclo.

8.7. Ejemplo Personalizador Combinado

Al principio se tiene un temperamento{A=80;G=30;I=30;R=40}

CICLO		1	2	3	4	5	6	7	8	9	10
USUARIO	**Edad**	20	20	20	20	20	20	20	20	20	20
	Sexo	M	M	M	M	M	M	M	M	M	M
	Emoción	F	A	I	F	A	F	F	F	F	I
	Experiencia	0	0	0	0	0	0	0	1	1	0
	Artesano	100	100	50	50	100	100	100	100	100	100
	Guardián	50	0	50	50	50	50	50	50	50	50
	Idealista	50	0	50	50	50	50	50	50	50	50
	Racional	50	0	100	100	50	50	50	50	50	50
NPC	**Name**	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro
	Gun	SG	SG	SG	SG	SG	SG	SG	SG	SG	SG
	Freq	4	5	2	3	5	4	4	5	5	3
	Lag	0	0	84	34	0	0	0	0	0	0
	Loyalty	1	1	3	2	1	1	1	1	1	2
	Pain	325	100	562	462	325	325	325	315	315	425
	Rate	100	1	142	112	70	100	100	95	95	70
	Speed	30	35	25	30	35	30	30	32	32	25
	Health	60	100	27	77	110	60	60	70	70	10
STATS	**M. Killed**	12	14	15	12	14	15	16	18	16	
	P. Dead	1	3	1	1	0	1	0	1	3	

M: Masculino.
F: Femenino.
F: Feliz.
I: Impotencia.
A: Aburrido.
M. Killed: Monstruos matados
P. Dead: Veces que muere el jugador en el ciclo.

8.8. Ejemplo Personalizador Keirsey

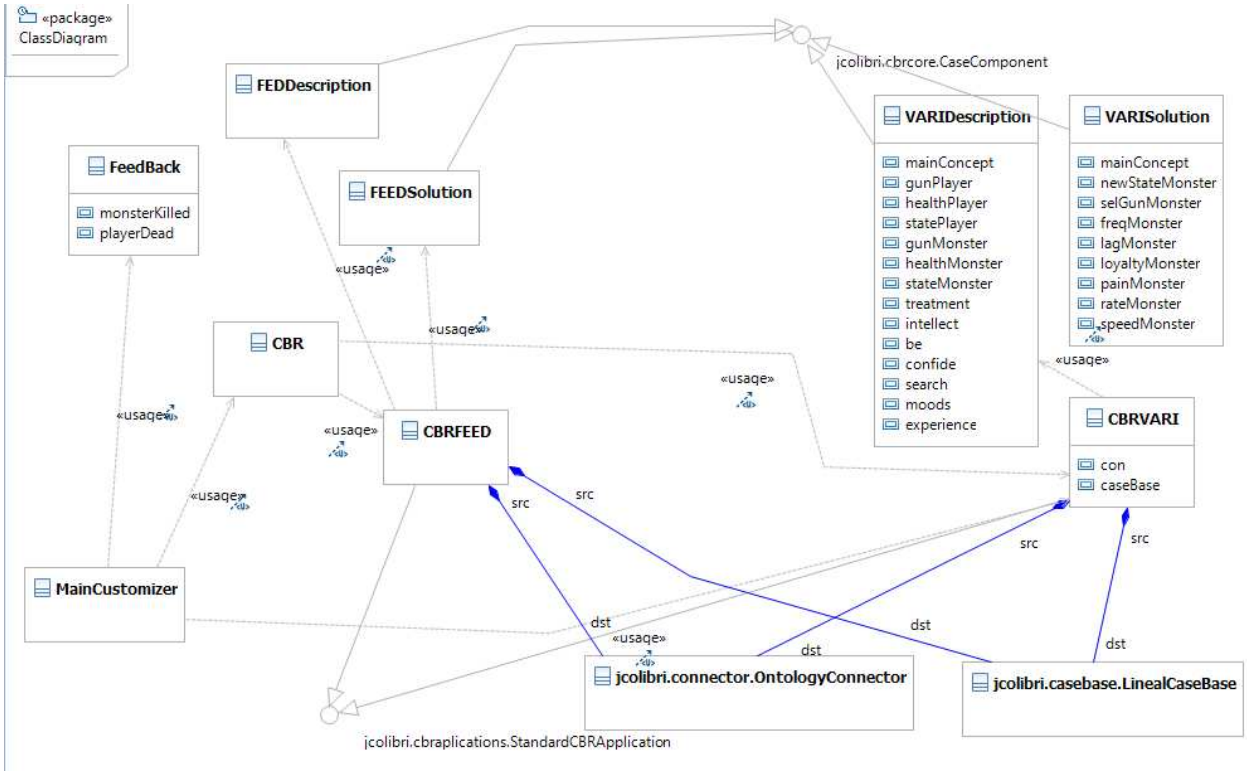
Al principio se tiene un temperamento{A=80;G=30;I=30;R=40}

CICLO		1	2	3	4	5	6	7	8	9	10
USUARIO	**Edad**	20	20	20	20	20	20	20	20	20	20
	Sexo	M	M	M	M	M	M	M	M	M	M
	Emoción	F	A	A	A	F	I	F	F	F	I
	Experiencia	0	0	0	0	0	0	0	0	1	0
	Artesano	0	0	0	100	100	0	100	100	100	100
	Guardián	100	0	0	0	0	0	0	0	0	0
	Idealista	0	100	0	0	0	0	0	0	0	0
	Racional	0	0	100	0	0	100	0	0	0	0
NPC	**Name**	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro	Ogro
	Gun	SG	SG	SG	SG	SG	SG	SG	SG	SG	SG
	Freq	3	4	4	5	4	2	4	4	5	3
	Lag	50	0	0	0	0	87	0	0	0	50
	Loyalty	4	1	1	1	1	2	1	1	1	2
	Pain	500	375	375	100	200	575	200	200	190	300
	Rate	175	120	45	1	1	105	1	1	1	31
	Speed	30	35	35	35	30	25	30	30	32	25
	Health	85	130	135	100	50	35	50	50	60	10
STATS	**M. Killed**	12	12	13	14	14	15	16	15	15	
	P. Dead	0	1	1	1	3	0	0	0	2	

M: Masculino.
F: Femenino.
F: Feliz.
I: Impotencia.
A: Aburrido.
M. Killed: Monstruos matados
P. Dead: Veces que muere el jugador en el ciclo.

9 Diseño e Implementación

- **Arquitectura software**
  - **Diagrama de clases**



- **VARIDescription**

  Implementa la interfaz CaseComponent, es la descripción de las consultas que se van a realizar sobre la ontología VARIonto.owl (Ontología de personalización del comportamiento de los monstruos de Cube 2) para obtener los casos base.

- **VARISolution**

  Implementa la interfaz CaseComponent, describe los atributos que van a tener las respuestas de las consultas que se van a realizar sobre la ontología VARIonto.owl (Ontología de personalización del comportamiento de los monstruos de Cube 2).

- **FEEDDescription**

  Implementa la interfaz CaseComponent, es la descripción de las consultas que se van a realizar sobre la ontología FEEDonto.owl (Ontología para evaluar las emociones en función de las estadísticas y el temperamento) para obtener los casos base.

- **FEEDSolution**

  Implementa la interfaz CaseComponent, describe los atributos que van a tener las respuestas de las consultas que se van a realizar sobre la ontología FEEDonto.owl (Ontología para evaluar las emociones en función de las estadísticas y el temperamento).

- **CBRVARI**

  Implementa la interfaz StandardCBRApplication. Se encarga de hacer las consultas sobre VARIonto.owl (Ontología de personalización del comportamiento de los monstruos de Cube 2), para obtener los casos base mas similares hace uso de las descripción VARIDescription.

  - Funciones:
    - configure: Configuras el conector iniciándolo a partir de "variconfig.xml", donde le asigno como ontología principal "VARIonto.owl", la subontología "HCCIonto.owl", como clase descriptora "VARIDescription.java". como clase que nos da la solución "VARISolution.java" y haciendo los mappings correspondientes entre los atributos de las clases y los atributos de la ontología. En el anexo se puede ver el fichero "variconfig.xml", donde se realiza el mapeo.
    - preCycle: Inicializo los casos base a partir del conector.
    - cycle: Donde asigno como voy a buscar cada atributo. siendo por similitud los de tipo entero, y el resto por exactitud. También es donde asigno los pesos a los atributos, le asigno mas peso a el estado de ánimo y al nombre del NPC, y luego el mismo peso al resto de atributos. Solo se obtiene el caso base más similar, la función getCase se encarga de devolver el caso base más idóneo.
    - postCycle: Cierro los casos base.
    - getCase: Me devuelve el caso base solución más similar encontrado, le entra como parámetro VARIDescription.

- **CBRFEED**

  Implementa la interfaz StandardCBRApplication. Se encarga de hacer las consultas sobre FEEDonto.owl (Ontología para evaluar las emociones en función de las estadísticas y el temperamento), para obtener los casos base mas similares hace uso de las descripción FEEDDescription.

- Funciones:
  - configure: Configuras el conector iniciándolo a partir de "feedconfig.xml", donde le asigno como ontología principal "FEEDonto.owl", la subontología "HCCIonto.owl", como clase descriptora "FEEDDescription.java". como clase que nos da la solución "FEEDSolution.java" y haciendo los mappings correspondientes entre los atributos de las clases y los atributos de la ontología. En el anexo se puede ver el fichero "feedconfig.xml", donde se realiza el mapeo.
  - preCycle: Inicializo los casos base a partir del conector.
  - cycle: Donde asigno como voy a buscar cada atributo. siendo todos por similitud. También es donde asigno los pesos a los atributos, le asigno mas peso a las estadísticas, y luego el mismo peso al resto de atributos. Se almacena el caso base mas similar, la función getCase se encarga de devolver el caso base más idóneo.
  - postCycle: Cierro los casos base.
  - getCase: Me devuelve el caso base solución, le entra como parámetro FEEDDescription.

- **CBR**

  En esta clase es donde se implementan los tres tipos de personalizadores, hace uso de los CBRs CBRHCC, CBRVARI y CBRFEED.

  - Funciones:
    - configureVARI: Se configura y se realiza el pre-ciclo del CBRVARI.
    - configureFEED: Se configura y se realiza el pre-ciclo del CBRFEED.
    - closeVARI: Realiza el post-ciclo de CBRVARI.
    - closeFEED: Realiza el post-ciclo de CBRFEED.
    - getAdaptionAbsolute: A partir del dato de entrada del tipo VARIDescription, se obtiene la adaptación por el algoritmo del personalizador Absoluto.
    - getAdaptionCombined: A partir del dato de entrada del tipo VARIDescription, se obtiene la adaptación por el algoritmo del personalizador Combinado.
    - getAdaptionKeirsey: A partir del dato de entrada del tipo VARIDescription, se obtiene la adaptación por el algoritmo del personalizador Keirsey.
    - getAdaptionVARI: A partir del dato de entrada del tipo VARIDescription, se obtiene la adaptación del juego.
    - getAdaptionFEED: A partir del dato de entrada del tipo FEEDDescription, se obtiene la emoción del usuario.

- **FeedBack**

  Es la clase encargada de recoger el estado de ánimo, la experiencia del usuario, el arma del jugador, la salud del jugador y el arma. salud y estado de cada monstruo.

  - Funciones:

    - execute: Cada 2 minutos lee el fichero "stats.txt" que es escrito por el juego y se actualiza siempre que se mata un monstruo o muere el jugador, en dicho fichero contiene los monstruos matados durante la sesión, y las veces que muere el jugador. En cada ciclo(2 minutos) se recogerán las emociones del usuario de la siguiente manera:

      Formato de "stats.txt":

      ```
      monsterKilled:20
      playerDead:2
      ```

      stats.txt

10 Diseño de Experimentos y Experimentos

Las pruebas deberían realizarse a varios grupos con las mismas características para evaluar que características son mas determinantes. Cada grupo tiene características distintas a otros grupos. Primero se hará jugar a la persona al juego original, y luego se le pondrá a jugar después de realizar la personalización. Como existen tres versiones de personalizadores, se le hará pruebas con las tres versiones, para evaluar cuál funciona mejor o peor. Al finalizar las pruebas, al usuario se le hará un pequeño cuestionario, para sacar conclusiones, en dicho cuestionario se le preguntará si nota diferencia con el juego original, y de cada una de las versiones de personalizadores se le pedirá que la evalúe de 0 a 10, indicando con 0 nada satisfecho con las modificaciones, y 10 totalmente satisfecho con las modificaciones. De todas maneras para que la evaluación sea correcta deben tenerse partidas muy largas, ya que los juegos están hechos para entretener y emociones como el aburrimiento, suceden cuando se lleva mucho tiempo jugando. Para la evaluación de los tres personalizadores se les irá alternando aleatoriamente, ya que con el tiempo el usuario va adquiriendo experiencia con el juego, por lo que sus características van cambiando.

Además del cuestionario, como se puede tener acceso a las estadísticas de NPCs muertos y las veces que muere el jugador, nos sirven para sacar conclusiones sobre como afectan las modificaciones en función las características del usuario y de cada una de las versiones de personalizadores. Una de las cosas para determinar que personalizador es mejor es evaluar el tiempo de las partidas, cuanto mayor sea ese tiempo, se considera que el sistema de adaptación es mejor, debido a que si juegas más tiempo es porque te entretiene más.

La experimentación ha sido realizada en 9 personas, obteniendo los siguientes resultados:

Evaluación				
	Original	Absoluto	Combinado	Keirsey
Todos(9)	4.2	6.6	8.1	6.4
Artesano(2)	4	7.5	9	9
Guardián(4)	3.75	5	6.75	4.75
Idealista(2)	4	7.5	9	6
Racional(4)	4.25	7.5	8.75	6.5
Mujer(4)	3.25	6.25	7.75	4.5
Hombre(5)	5	7	8.4	8
<30 años(7)	4.85	7.57	9	7.14
>50 años(2)	2	3.5	5	4
Si domina(3)	6.67	7.3	8.67	8.67
No domina(6)	3	6.3	7.8	5.3

Estadísticas (5 min.)					
Absoluto		Combinado		Keirsey	
MM	VM	MM	VM	MM	VM
47.8	2.8	56.8	1.5	56.9	2.5
61.5	1.5	67.5	0	69	1
34	4.7	41	3	39.2	4.2
50	2.5	67.5	2	66.5	3
49.2	2.5	59.2	1.5	57.7	2.5
42.5	3	53	1	53	2.2
52.2	2.4	59.8	1.6	58.2	2.8
65.4	2.3	65.1	1.1	65	2
21.5	4.5	27.5	3	24	4.5
60.3	2	66.7	1.3	65.3	2
53.6	3.2	51.8	1.7	51.2	2.8

Figura 10.1: Tabla de resultados de los experimentos.

MM: Monstruos Matados.
VM.: Número de veces que muere el jugador

Los resultados no son muy concluyentes ya que solo se ha experimentado con nueve personas y las partidas eran solo de cinco minutos. Pero si se puede observar que se consigue una mejoría con el uso de la personalización, que el personalizador mejor valorado es el combinado, esto pienso que sucede porque las partidas son muy cortas y entonces no se puede ver la verdadera funcionalidad del personalizador Keirsey. A los que más le gustan este tipo de videojuegos son a los artesanos, donde se consigue una mayor mejora es en la gente que no tiene experiencia con este tipo de videojuegos, por lo que la experiencia junto con las emociones son las características más a tener en cuenta en este caso de estudio. La gente que tiene dificultad con el uso de los terminales(ratón y teclado), se les hace muy difícil jugar. Para las partidas primero se utilizó el personalizador Absoluto, luego el Combinado y luego el de Keirsey, como se puede observar en las estadísticas la experiencia de los jugadores mejora con el tiempo. Mirando los resultados se puede ver que la valoración del usuario dependió en gran medida de las veces que le mataron, por lo que la emoción de frustración es una emoción a tratar, antes que la de aburrimiento.

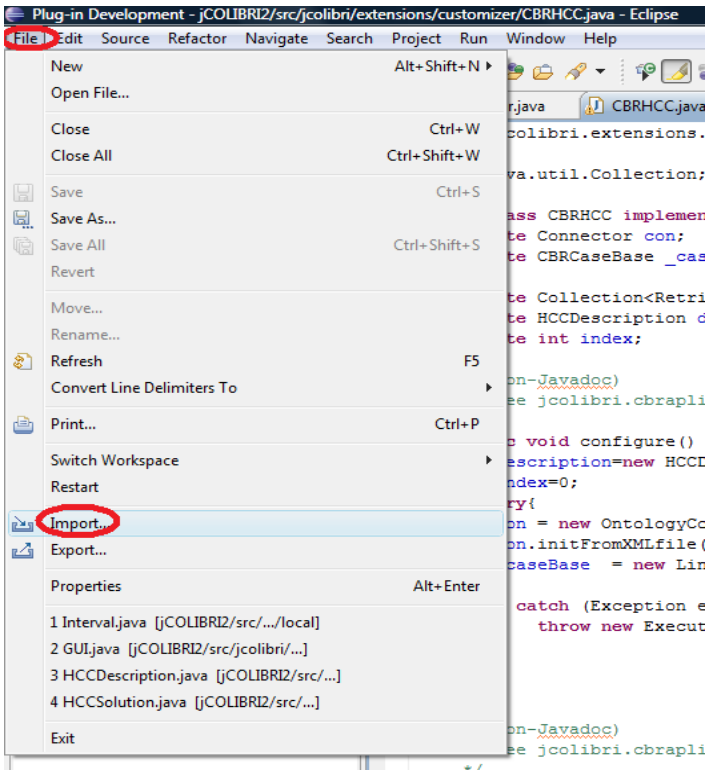
11 Conclusiones y Trabajo Futuro

En este proyecto se ha realizado un modelo de personalización independiente del dominio usando el HCC issues y otras características del usuario. Se hace uso de CBR y ontologías que lo hace fácilmente escalable. Este modelo ha sido aplicado para adaptar el comportamiento de los personajes de videojuegos a las características de usuario, más específicamente en CUBE II. Existen otros modelos de personalización pero algunos dependen del dominio, otros son difícilmente escalables, mientras que otros se centran en algunas de las características de los HCC Issues como el temperamento o las emociones. Aunque para el caso de estudio las adaptaciones se han sacado a partir de fórmulas, lo ideal sería usar perfiles reales a partir de la experimentación, de la cuál también ajustar mejor los parámetros del feedback, añadiendo más atributos, como puede ser la duración de las partidas. En el estudio me he dado cuenta que hay gente que el principal problema que tiene, es de no haber utilizado mucho el ratón y el teclado, una falta de habilidad necesaria para estos juegos. Este problema se podría solucionar suavizando los cambios bruscos, cuando sean detectados durante el juego.

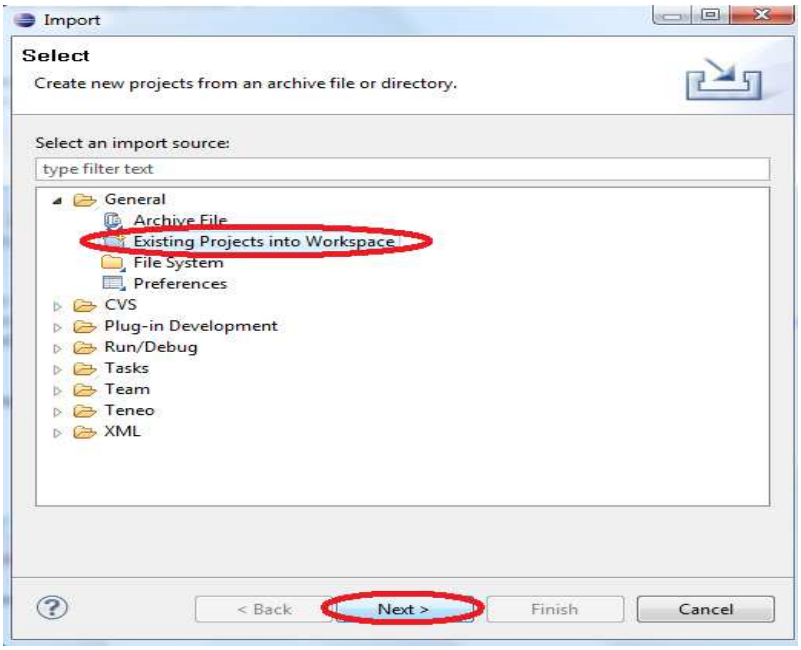
En cuanto al trabajo futuro se podría estudiar la personalización en más campos como puede ser las discapacidades de las personas, entornos gráficos, juegos educativos, sistemas operativos, cualquier tipo de aplicación. Dentro del mismo juego(Cube 2) permite la modificación de los gráficos de monstruos, en el foro de http://sauerbraten.org/ aparecen monstruos creados por otras personas. Añadir más funcionalidades al feedback, como un reconocedor facial y un sistema conversacional con la máquina para determinar el estado del ánimo del usuario. Ampliar las ontologías, con más características de usuario, además de incluir más posibles variaciones del sistema. Hacer un estudio en personas, del cuál obtener más casos base y así tener una base de conocimiento más precisa y amplia. Además también se puede tratar a cada personaje de videojuego NPC como un agente, creando un sistema multiagente, donde cada uno de los monstruos tuviera un comportamiento distinto, lo que daría mayor entretenimiento al jugador, y mayor dificultad en el futuro ya que se podría hacer un sistema colaborativo de NPCs. También se podría proporcionar aprendizaje a los NPCs para que aprendieran del jugador y así hacerle cambiar de táctica para evitar la inactivad y pasividad del jugador. En un futuro se podría hacer que el juego se comunicara con la aplicación de personalización por medio de sockets, haciendo el personalizador de servidor y el juego de cliente. Recoger más información para determinar las emociones que experimenta el usuario, como puede ser el tiempo que duran las sesiones.

12 Documentación del sistema implementado

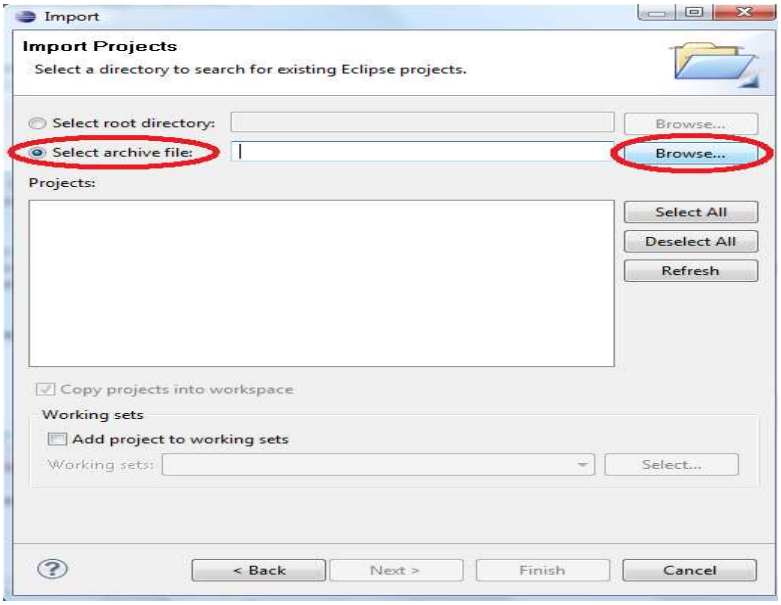
El sistema ha sido desarrollado en eclipse, por lo que se recomienda ejecutarlo en eclipse, para lo cual se importa el proyecto "jColibri2.zip" que contiene el "Personalizador" y otras clases de las que hace uso dicho "Personalizador". Pero primero insertamos el archivo "jColibri2.zip" en la carpeta que tengamos asignada como workspace. Para importar el proyecto, se pincha en "file" y luego "import".



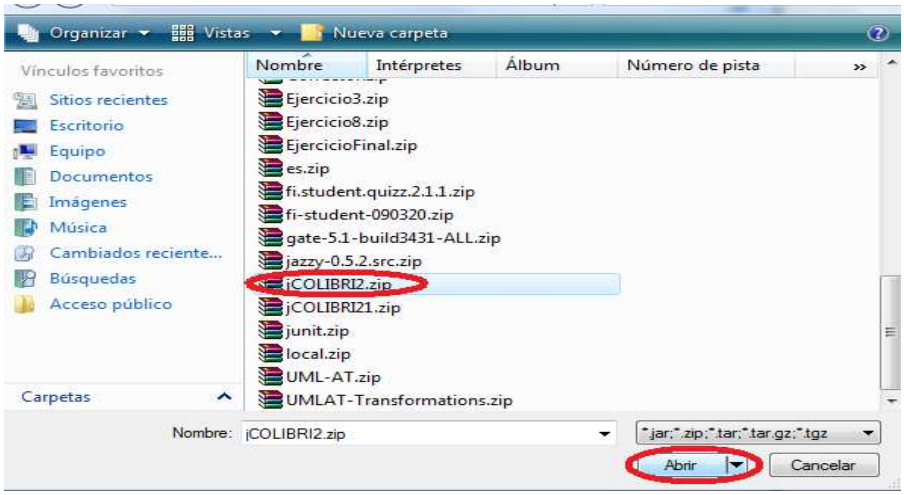
Luego seleccionamos "Existing Project in Workspace" y hacemos clic en "Next".



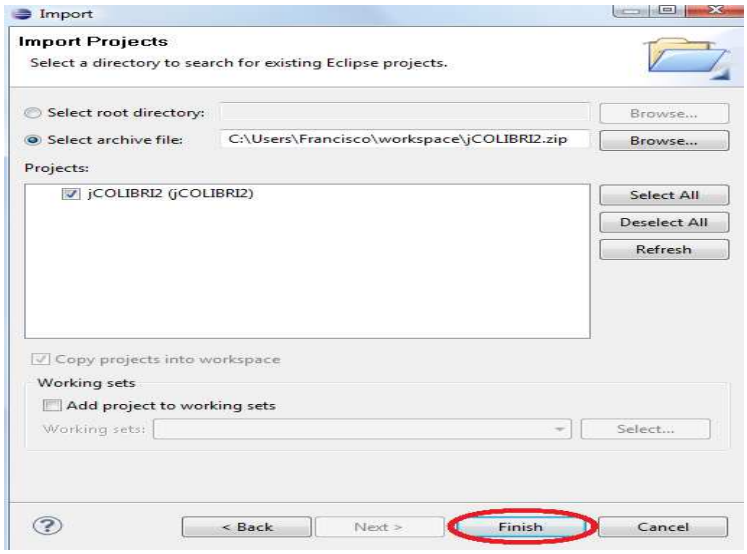
Más adelante seleccionamos "Select archive file" y pinchamos en "Browse..."



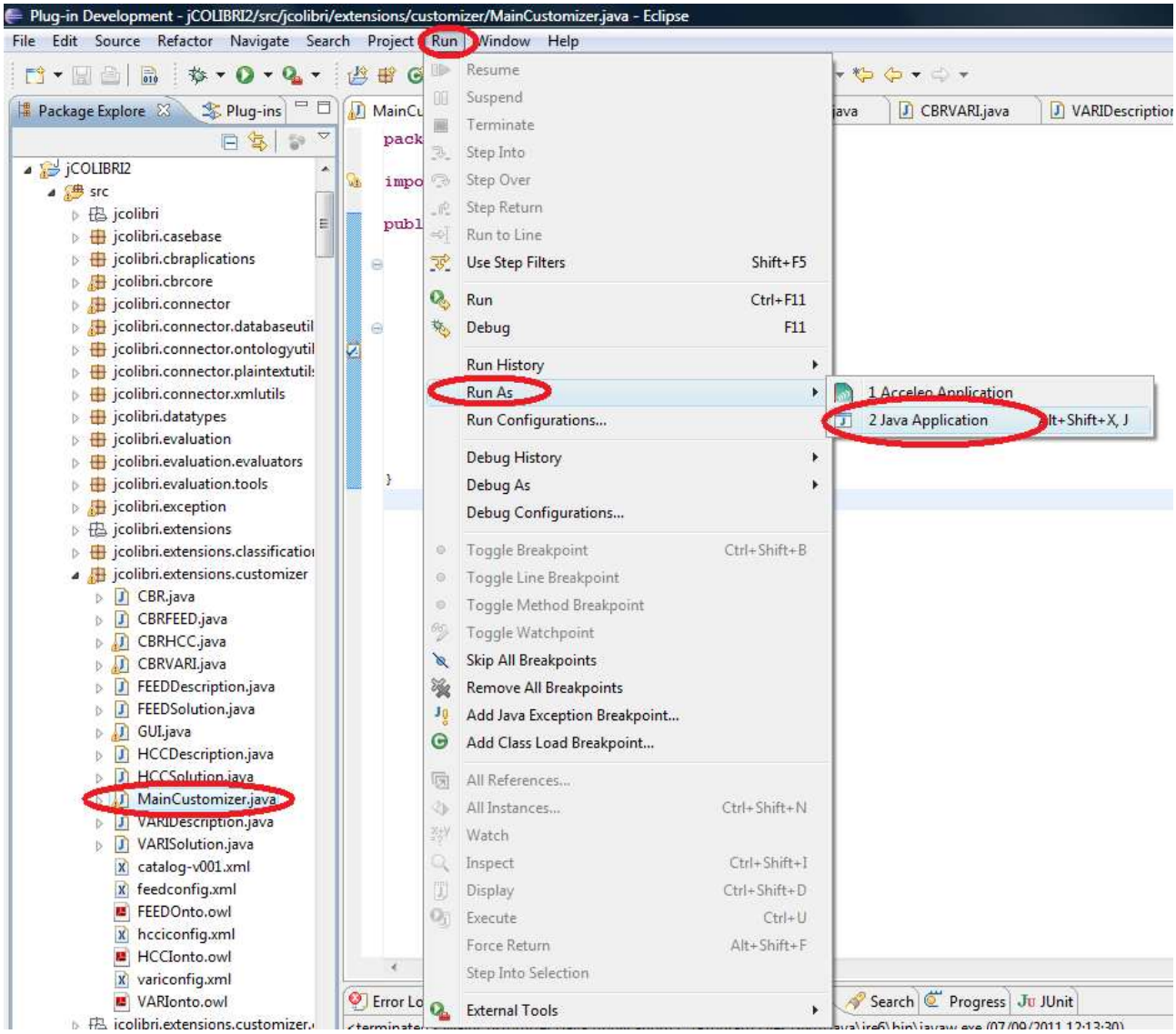
Luego elegimos el archivo "jColibri2.zip" y pinchamos "Abrir".



Para terminar pinchamos en "Finish".

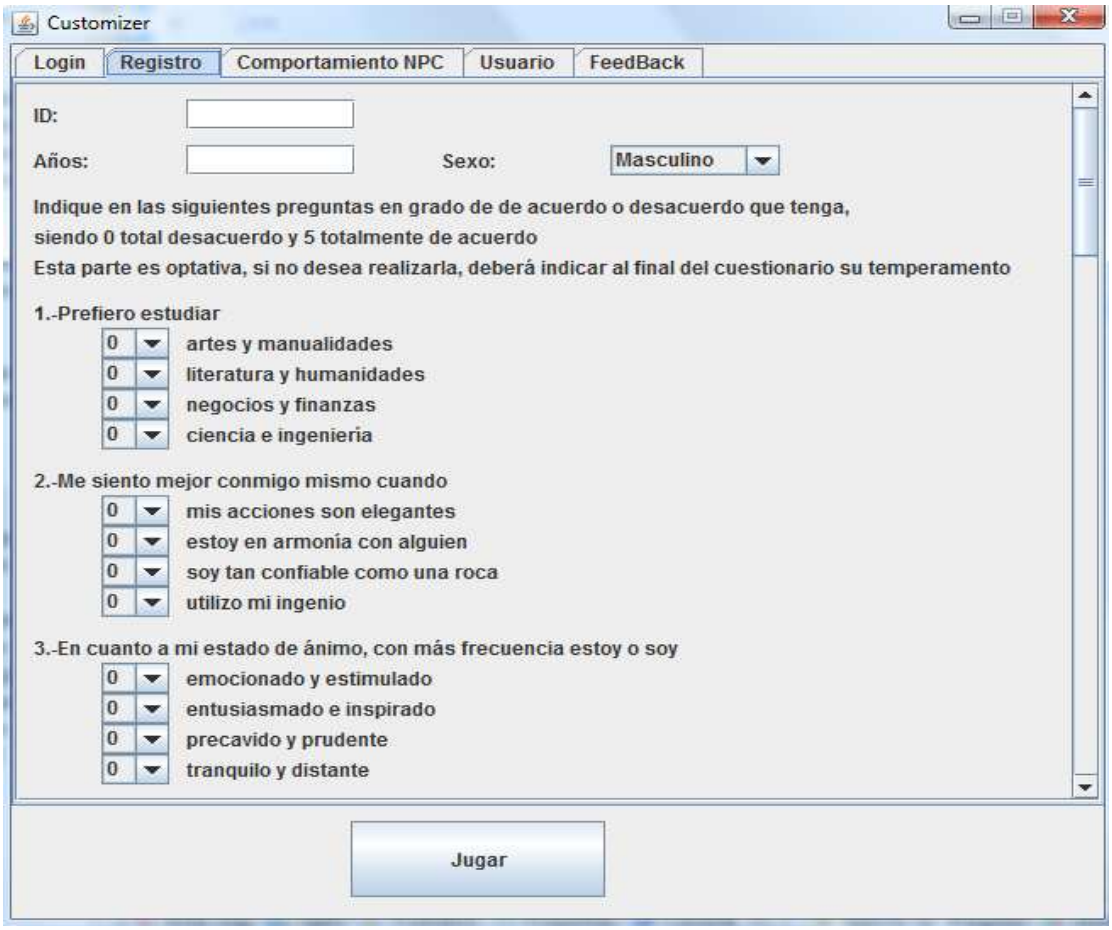


Una vez importado el proyecto, la aplicación se ejecuta en jColibri2/src/jcolibri/extensions/customizer/MainCustomizer.java. Una ver abierto MainCustomizer.java, para ejecutarlo se pincha en Run/Run As/Java Application.

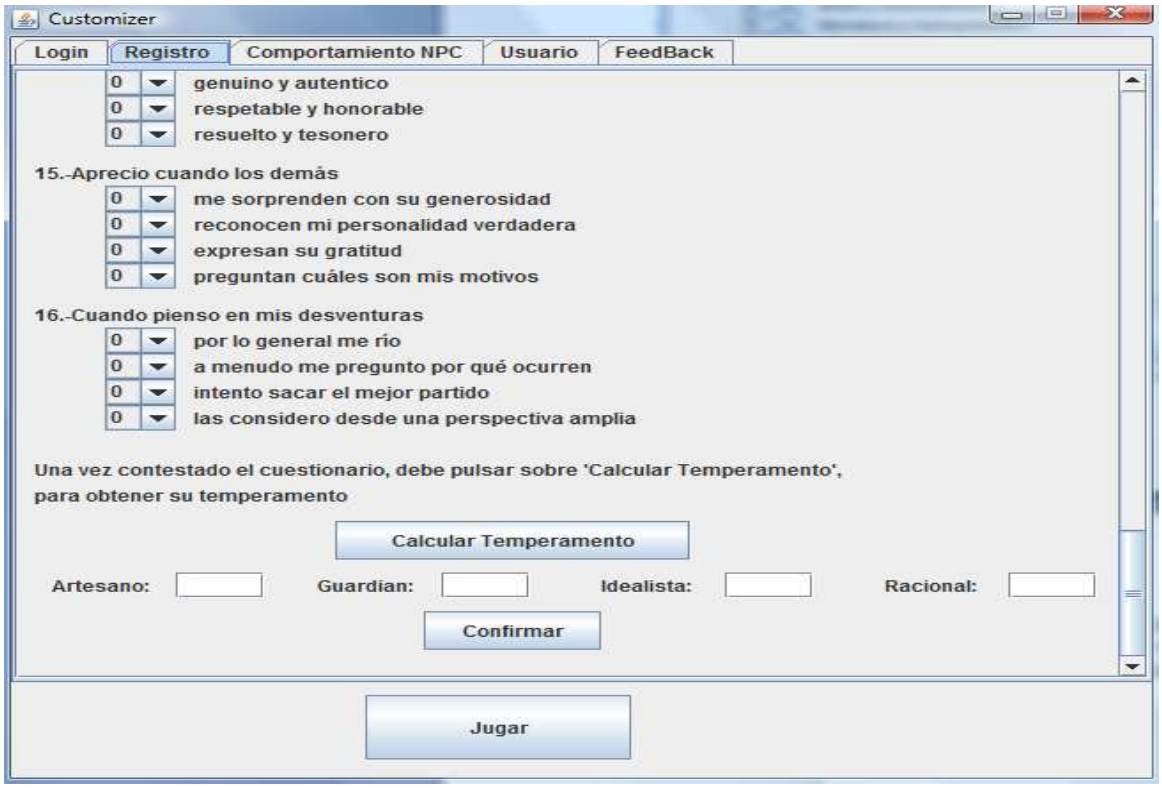


Una vez ya ejecutado, aparece una ventana con cinco pestañas y un botón común que es el de "Jugar". El botón "Jugar" es para ejecutar el juego. Primero vamos a enseñar la primera pestaña, que es la de identificación del usuario y elección de personalizador que queremos realizar. Para identificarse un usuario primero tiene que estar registrado en el sistema.

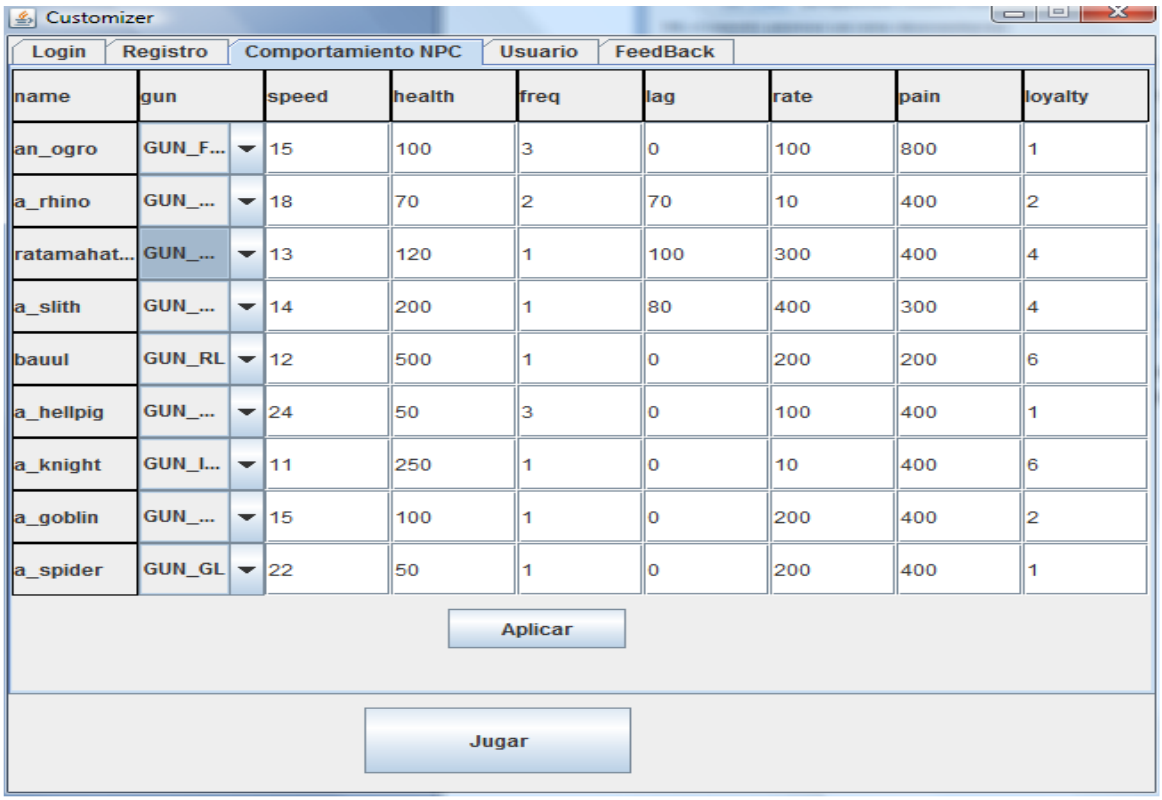
La segunda pestaña nos da la funcionalidad de registrarnos, la cual nos pide el identificador, la edad y el sexo del usuario. Además de proporcionarnos un cuestionario para obtener el temperamento. Para responder el cuestionario se seleccionará en cada una de las respuestas lo identificado que se siente con dicha respuesta, siendo 0 nada identificado y 5 totalmente identificado.



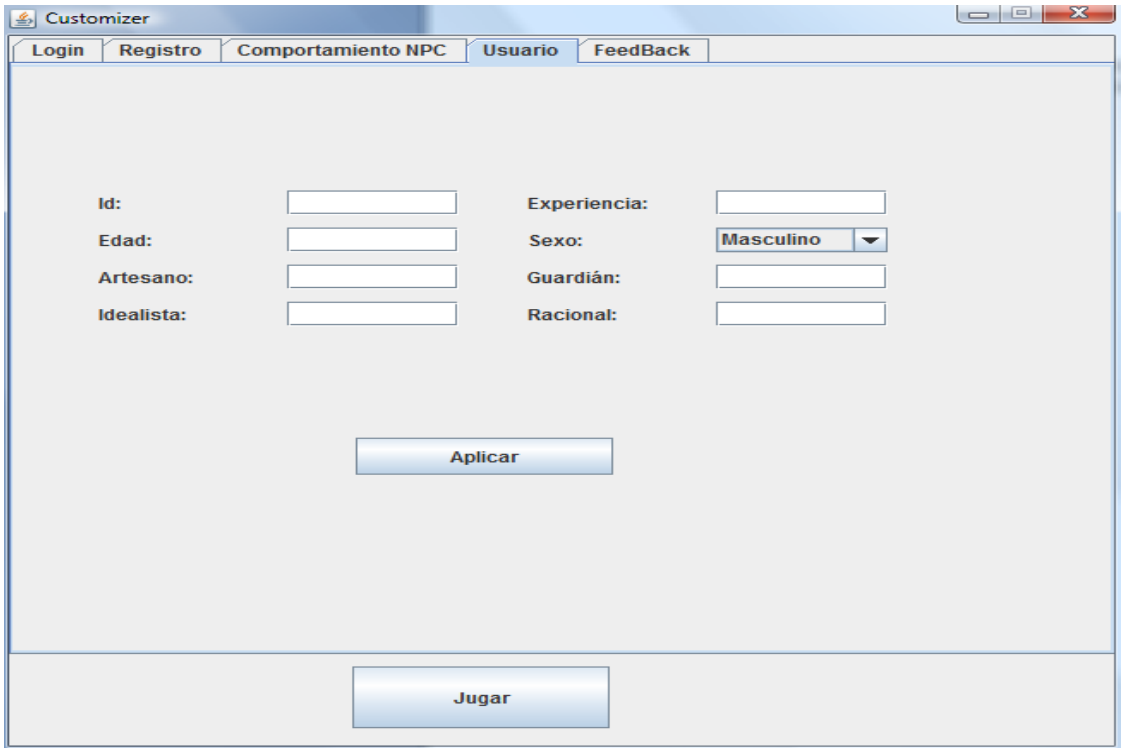
Para calcular el temperamento el usuario tiene que pulsar el botón "Calcular el temperamento" antes de confirmar, y si el usuario no está conforme dicho calculo podrá meter los niveles de cada temperamento manualmente. Para confirmar el registro, el usuario tiene que pinchar el botón "Confirmar".



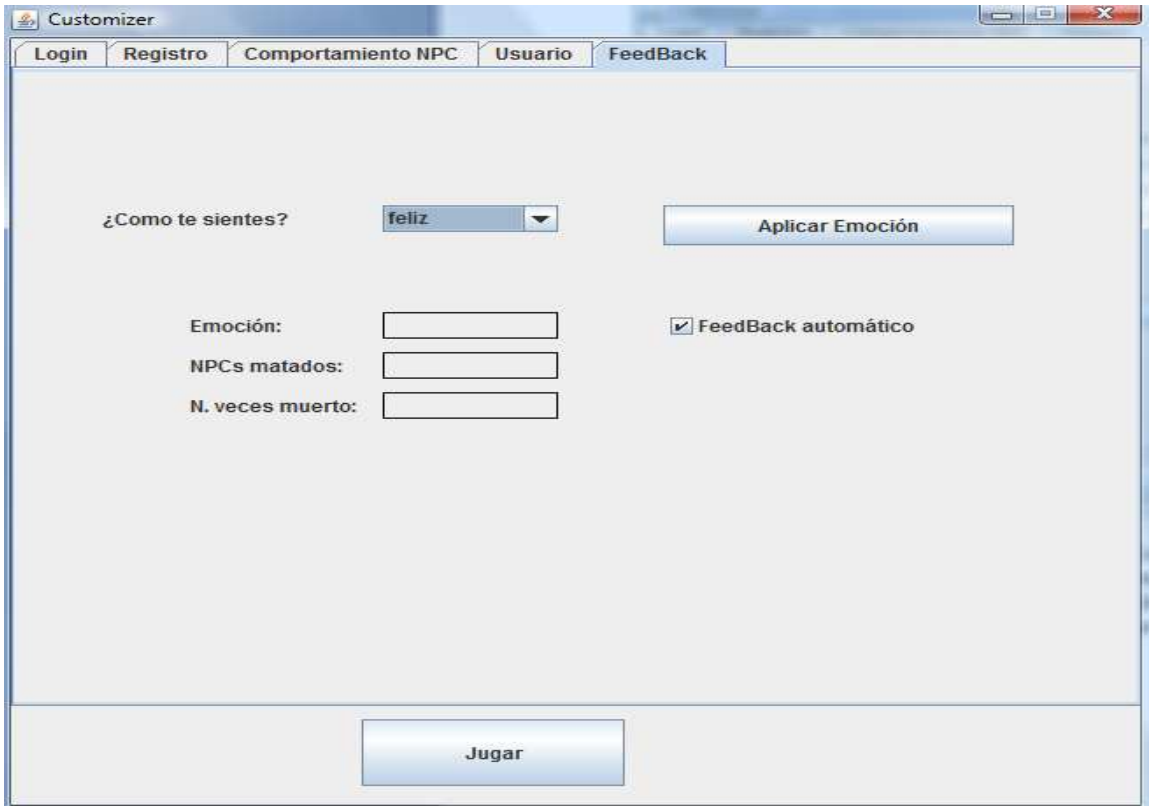
En las pestaña de "Comportamiento NPC" nos muestra una tabla con todos los NPCs y sus atributos, los cuales pueden cambiarse manualmente, para aplicar dicho cambio manual se debe pulsar el botón "Aplicar".



La pestaña "Usuario", nos ofrece una serie de campos para modificar los datos del usuario recogidos en el Registro. Para aplicar los cambios tiene que pinchar en el botón "Aplicar".



Por último la pestaña "FeedBack" nos da la posibilidad de expresar nuestras emociones de manera manual. También nos da posibilidad de habilitar/deshabilitar el Feedback automático, además de proporcionarnos la emoción que está siendo recogida por el Feedback automático y las estadísticas de los monstruos que hemos matado y las veces que hemos muerto.



13 Anexo

- **Cuestionario corto de Keirsey:**

1. Prefiero estudiar
   (a) artes y manualidades
   (b) literatura y humanidades
   (c) negocios y finanzas
   (d) ciencia e ingeniería

2. Me siento mejor conmigo mismo cuando
   (a) mis acciones son elegantes
   (b) estoy en armonía con alguien
   (c) soy tan confiable como una roca
   (d) utilizo mi ingenio

3. En cuanto a mi estado de ánimo, con más frecuencia estoy o soy
   (a) emocionado y estimulado
   (b) entusiasmado e inspirado
   (c) precavido y prudente
   (d) tranquilo y distante

4. Siempre insisto en
   (a) perfeccionar mis habilidades
   (b) ayudar a que otros se afirmen
   (c) ayudar a que otros hagan lo correcto
   (d) explicarme cómo funcionan las cosas

5. Cuando se trata de hacer algo soy
   (a) práctico y oportunista
   (b) compasivo y altruista
   (c) concienzudo y diligente
   (d) eficiente y pragmático

6. Me respeto más por
   (a) ser audaz y con espíritu de aventura
   (b) ser bondadoso y con buena voluntad
   (c) realizar buenas obras
   (d) ser autónomo e independiente

7. Me inclino más a confiar en
   (a) impulsos y caprichos
   (b) intuiciones e insinuaciones
   (c) costumbres y tradiciones
   (d) la razón pura y la lógica formal

8. A veces anhelo
   (a) causar una gran impresión y tener impacto
   (b) perderme en mis sueños románticos
   (c) ser un miembro legítimo y valorado
   (d) hacer un gran descubrimiento

9. En mi vida busco
   (a) aventuras y emociones
   (b) conocimiento de mi mismo
   (c) protección y seguridad
   (d) métodos de operación eficaces

10. Al mirar al futuro
   (a) aseguro que ocurrirá algo bueno
   (b) creo en la bondad innata de la gente
   (c) hay que ser sumamente cuidadoso
   (d) es mejor ser cauteloso

11. Si fuera posible, me gustaría convertirme en
   (a) un artista virtuoso
   (b) un visionario sabio
   (c) un ejecutivo de alto rango
   (d) un genio de la tecnología

12. Funcionaría mejor en un trabajo si me ocupara de
   (a) herramientas y equipo
   (b) desarrollo de recursos humanos
   (c) materiales y servicios
   (d) sistemas y estructuras

13. Como dirigente de acciones, sobre todo miro
   (a) las ventajas inmediatas
   (b) las posibilidades futuras
   (c) la experiencia pasada
   (d) las condiciones necesarias y suficientes

14. Confío más en mí mismo cuando soy
   (a) flexible y adaptable
   (b) genuino y autentico
   (c) respetable y honorable
   (d) resuelto y tesonero

15. Aprecio cuando los demás
   (a) me sorprenden con su generosidad
   (b) reconocen mi personalidad verdadera
   (c) expresan su gratitud
   (d) preguntan cuáles son mis motivos

16. Cuando pienso en mis desventuras
    (a) por lo general me río
    (b) a menudo me pregunto por qué ocurren
    (c) intento sacar el mejor partido
    (d) las considero desde una perspectiva amplia

- **variconfig.xml**

MainOntology	
CaseMainConcept	VARIonto
DescriptionClassName	jcolibri.extensions.customizer.VARIDescription

Ontología principal

SubOntology	
URL	http://www.owl-ontologies.com/HCCIonto.owl
LocalCopy	jcolibri/extensions/customizer/HCCIonto.owl

SubOntología

DescriptionClassName	jcolibri.extensions.customizer.VARIDescription
DescriptionMappings	
Map	
Property	hasGunPlayer
Concept	GUNPLAYER
Attribute	gunPlayer
Map	
Property	hasHealthPlayer
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	healthPlayer
Map	
Property	hasStatePlayer
Concept	STATEPLAYER
Attribute	statePlayer
Map	
Property	hasNameMonster
Concept	NAMEMONSTER
Attribute	nameMonster
Map	
Property	hasGunMonster
Concept	GUNMONSTER
Attribute	gunMonster
Map	
Property	hasStateMonster
Concept	STATEMONSTER
Attribute	stateMonster
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasAge
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	age
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasEmotions
Concept	EMOTIONS
Attribute	emotions
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasExperience
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	experience
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasArtisan
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	artisan
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasGuardian
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	guardian

Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasIdealist
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	idealist
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasRational
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	rational
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasGender
Concept	GENDER
Attribute	gender

Clase Descriptora y mapping

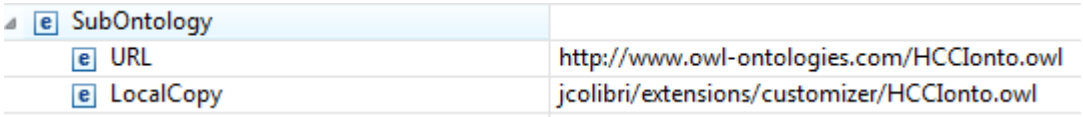
SolutionClassName	jcolibri.extensions.customizer.VARISolution
SolutionMappings	
Map	
Property	hasNewStateMonster
Concept	NEWSTATEMONSTER
Attribute	newStateMonster
Map	
Property	hasNameMonster
Concept	NAMEMONSTER
Attribute	nameMonster
Map	
Property	hasHealthMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	healthMonster
Map	
Property	hasSelGunMonster
Concept	SELGUNMONSTER
Attribute	selGunMonster
Map	
Property	hasFreqMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	freqMonster
Map	
Property	hasLagMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	lagMonster
Map	
Property	hasLoyaltyMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	loyaltyMonster
Map	
Property	hasPainMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	painMonster
Map	
Property	hasRateMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	rateMonster
Map	
Property	hasSpeedMonster
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	speedMonster

Clase Solución y mapping

- **feedconfig.xml**

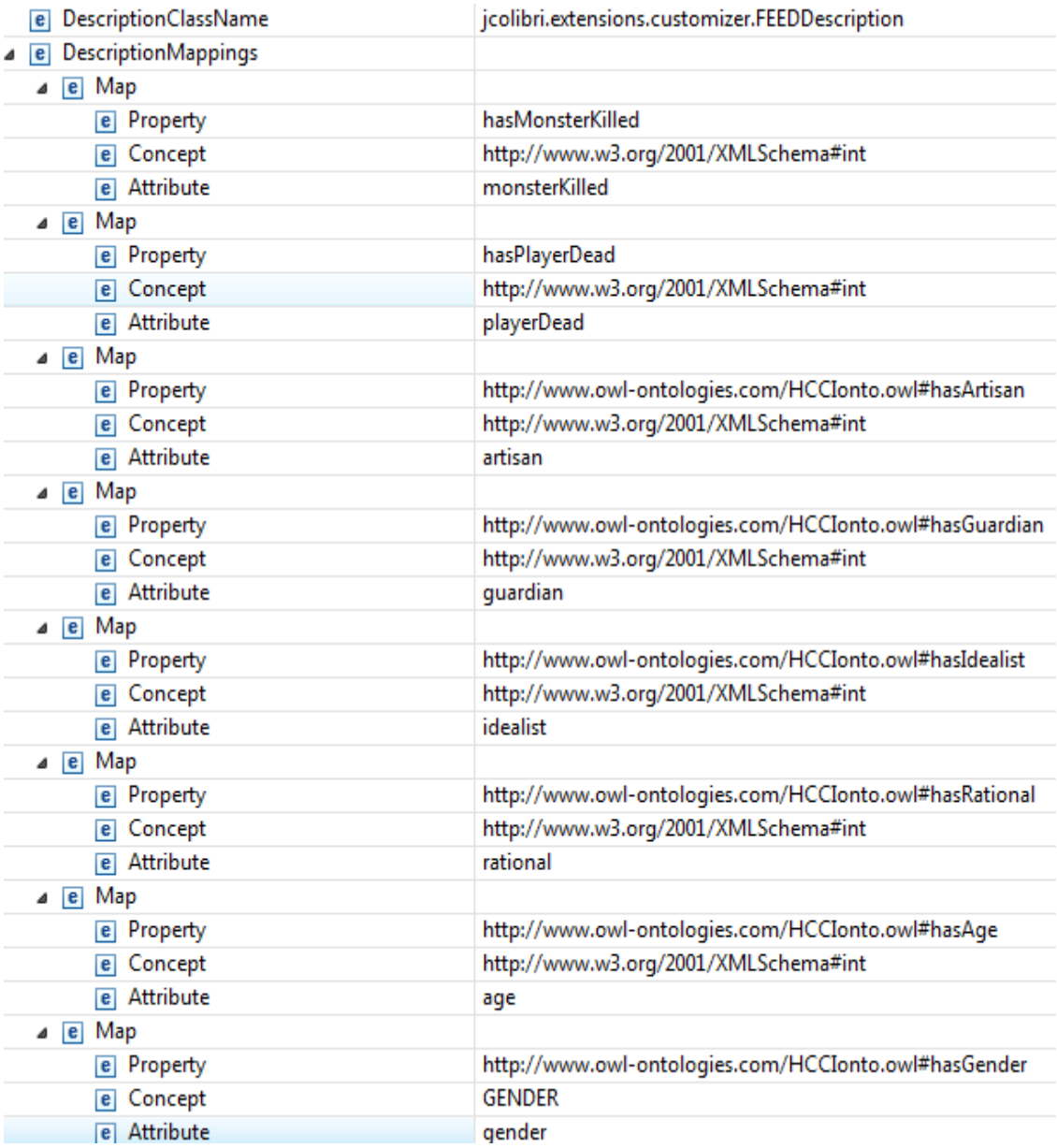
MainOntology	
URL	http://www.owl-ontologies.com/FEEDonto.owl
LocalCopy	jcolibri/extensions/customizer/FEEDonto.owl

Ontología principal



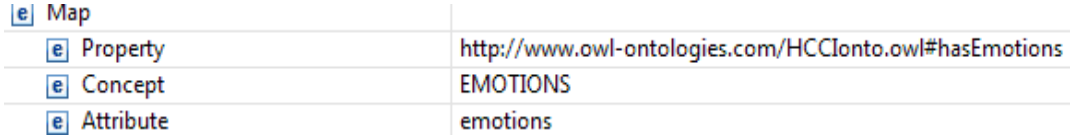
SubOntology	
URL	http://www.owl-ontologies.com/HCCIonto.owl
LocalCopy	jcolibri/extensions/customizer/HCCIonto.owl

SubOntología



DescriptionClassName	jcolibri.extensions.customizer.FEEDDescription
DescriptionMappings	
Map	
Property	hasMonsterKilled
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	monsterKilled
Map	
Property	hasPlayerDead
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	playerDead
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasArtisan
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	artisan
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasGuardian
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	guardian
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasIdealist
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	idealist
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasRational
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	rational
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasAge
Concept	http://www.w3.org/2001/XMLSchema#int
Attribute	age
Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasGender
Concept	GENDER
Attribute	gender

Clase Descriptora y mapping



Map	
Property	http://www.owl-ontologies.com/HCCIonto.owl#hasEmotions
Concept	EMOTIONS
Attribute	emotions

Clase Solución y mapping

14 Bibliografía

1. Hector Gómez-Gauchía, Belén Díaz-Agudo and Pedro A. González-Calero. Automatic Personalization of the Human Computer Interaction Using Temperaments. FLAIRS Conference 2006:352-357.
2. Hector Gómez-Gauchía and Federico Peinado. Automatic Customization of Non-Player Characters Using Players Temperament. TIDSE 2006:241-252.
3. Hector Gómez-Gauchía, Belén Díaz-Agudo and Pedro A. González-Calero. COBBER: Ontology Based Model for Human Centered Computing.
4. Gonzalo Flórez-Puga, Marco Antonio Gómez-Martín, Pedro Pablo Gómez-Martín, Belén Díaz-Agudo, and Pedro Antonio González-Calero. Query-Enabled Behavior Trees. IEEE Trans. Comput. Intellig. and AI in Games (TCIAIG) 1(4):298-308 (2009).
5. Julio Alberto Martínez Real, Alberto Pedrera Castela y Guillermo Ortiz Fernández. Personalización inteligente del sistema operativo Linux basado en las preferencias y temperamentos de los usuarios.
6. http://gaia.fdi.ucm.es/projects/jcolibri/
7. Alejandro Jaimes, Daniela Nicklas and Nicu Sebe. 3rd International Workshop on Human-Centered Computing (HCC '08). ACM Multimedia 2008:1151-1152
8. Sethuraman Panchanathan, Narayanan C Krishnan, Sreekar Krishna, Troy McDaniel and Vineeth Nallure Balasubramanian. Enriched Human-Centered Multimedia Computing through Inspirations from Disabilities and Deficit-Centered Computing Solutions. HCC 2008:35-42
9. Nithya Sambasivan,, Melissa Ho, Matthew Kam, Neesha Kodagoda, Susan Dray, John C. Thomas, Ann Light, Kentaro Toyama. Human-centered Computing in International Development. CHI Extended Abstracts 2009:4745-4750
10. Alejandro Jaimes Daniel Gatica-Perez, Nicu Sebe and Thomas S. Huang. Human-Centered Computing: Toward a Human Revolution. IEEE Computer (COMPUTER) 40(5):30-34 (2007)
11. Richard A. Foulds and Arthur W. Joyce. Ill. Expanded Interactions: Broadening Human-Centered Computing. International ACM Conference on Assistive Technologies 1998:49-50
12. Gerd Waloszek and Ulrich Kreichgauer. User-Centered Evaluation of the Responsiveness of Applications. INTERACT 2009:239-242.
13. Changneng Zhou, Xueli Yu, Yujie Dong, Jing Tian, Qian Cui, Lingzi Hu. Affective Computation Driven Personalization Modeling in Game-Based Learning. Web Intelligence/IAT Workshops 2007:87-90.
14. Owen Conlan1, Cormac Hampson1, Neil Peirce1, Michael Kickmeier-Rust2. Realtime Knowledge Space Skill Assessment for Personalized Digital Educational Games. ICALT 2009:538-542.
15. http://www.elektra-project.org/
16. Daria Barteneva, Nuno Lau, Luis Paulo Reis. Implementation of Emotional Behaviours in Multi-agent System Using Fuzzy Logic and Temperamental Decision Mechanism. EUMAS 2006.
17. http://en.wikipedia.org/wiki/Keirsey_Temperament_Sorter.
18. http://gaia.fdi.ucm.es/
19. http://www.keirsey.com/
20. http://sauerbraten.org/
21. Timo Saari, Niklas Ravaja, Jari Laarni, Marco Turpeinen. Towards Emotionally Adapted Games based on User Controlled Emotion Knobs. DIGRA Conf. 2005.
22. Timo Saari, Niklas Ravaja, Jari Laarni, Marco Turpeinen. Towards Emotionally Adapted Games.
23. James Niehaus, Mark Riedl. Scenario Adaption: An Approach to Customizing Computer-Based Training Games and Simulations. School of Interactive Computing, Georgia Institute of Technology.
24. Boyang Li, Mark Riedl. An Offline Planning Approach to Game Plotline Adaptation. AIIDE 2010.
25. Alexandra Guzga. Automatic Game Adaption Using Player Profiling to Increase Entertainment.
26. Catherine Mulwa, Seamus Lawless, Mary Sharp, Inmaculada Arnedillo-Sanchez, Vincent Wade. Adaptive Educational Hypermedia Systems in Technology Enhanced Learning: A Literature Review.
27. Chek Tien Tan, Ho-lun Cheng. Personality-based Adaption for Teamwork in Game Agents. AIIDE 2007:37-42.
28. Chek Tien Tan, Ho-lun Cheng. TAP: An Effective Personality Representation for Inter-Agent Adaption in Games. AIIDE 2008.
29. Bradford Mott, Sunyoung Lee, James Lester. Probabilistic Goal Recognition in Interactive Narrative Environments. In Proceedings of the Twenty-First National Conference on Artificial Intelligence, pages 187–192, Boston, Massachusetts, 2006.
30. Mark O. Riedl, Andrew Stern, Don Dini, and Jason Alderman. Dynamic experience management in virtual worlds for entertainment, education, and training. International Transactions on Systems Science and Applications, Special Issue on Agent Based Systems for Human Learning, 4(2), 2008.
31. Héctor Gómez Gauchía. Un Enfoque Sistémico, Afectivo y Ontológico para el Razonamiento Basado en Casos Conversacional. Tesis Doctoral, 2008.